JOURNAL OF SOCIAL COMPUTING
ISSN 2688-5255 05/06 pp52–70
Volume 2, Number 1, March 2021
DOI: 10.23919/JSC.2021.0004

DeepPredict: A Zone Preference Prediction System for Online Lodging Platforms

Yihan Ma, Hua Sun, Yang Chen*, Jiayun Zhang, Yang Xu, Xin Wang, and Pan Hui

Abstract: Online lodging platforms have become more and more popular around the world. To make a booking in these platforms, a user usually needs to select a city first, then browses among all the prospective options. To improve the user experience, understanding the zone preferences of a user's booking behavior will be helpful. In this work, we aim to predict the zone preferences of users when booking accommodations for the next travel. We have two main challenges: (1) The previous works about next information of Points Of Interest (POIs) recommendation are mainly focused on users' historical records in the same city, while in practice, the historical records of a user in the same city would be very sparse. (2) Since each city has its own specific geographical entities, it is hard to extract the structured geographical features of accommodation in different cities. Towards the difficulties, we propose DeepPredict, a zone preference prediction system. To tackle the first challenge, DeepPredict involves users' historical records in all the cities and uses a deep learning based method to process them. For the second challenge, DeepPredict uses HERE places API to get the information of POIs nearby, and processes the information with a unified way to get it. Also, the description of each accommodation might include some useful information, thus we use Sent2Vec, a sentence embedding algorithm, to get the embedding of accommodation description. Using a real-world dataset collected from Airbnb, DeepPredict can predict the zone preferences of users' next bookings with a remarkable performance. DeepPredict outperforms the state-of-the-art algorithms by 60% in macro F1-score.

Key words: online lodging platform; zone preference; prediction; deep learning

1 Introduction

Traveling has been an essential activity in people's daily life. A satisfactory accommodation during a trip can greatly promote people's traveling experience. There are a number of online lodging services, such as Airbnb[1–5], Booking.com[6], and Homestay (https://www.homestay.com/, accessed on May 1, 2019), offering both traditional hotels and residential accommodations for visitors.

When booking accommodations online, users usually have a specified destination, such as London, New York City, and Melbourne. Users will look up the accommodation list of this city to find a desirable place. Since there might be tens of thousands of accommodations available in one city, it always takes a lot of time and energies to search for a satisfactory one. To overcome this problem to some extent, most online lodging services offer a series of filters. Users can choose accommodations which meet their requirements by setting these filters. For example, if a user wants to live in a relatively cheaper place, she can set a price range to search for accommodations that she could afford.

In particular, many users have location preferences for

• Yihan Ma, Hua Sun, Yang Chen, Jiayun Zhang, Yang Xu, and Xin Wang are with the School of Computer Science, Fudan University, Shanghai 200433, China. E-mail: {mayh18, hsun15, chenyang, jiayunzhang15, xuy, xinw}@fudan.edu.cn.
• Pan Hui is with the Department of Computer Science, University of Helsinki, Helsinki 00014, Finland, and also with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China. E-mail: panhui@cs.helsinki.fi.
† A preliminary version of this paper has been published in *Proc. of the 14th CCF Conference on Computer Supported Cooperative Work and Social Computing* (*ChineseCSCW'19*).
* To whom correspondence should be addressed.
Manuscript received: 2020-12-20; accepted: 2021-01-22

accommodations. To study the location preferences of different users from the perspective of the geography, we divide the city into different "zones". After dividing a city into different parts according to grid, each part represents a "zone". When traveling to a new city, some people like to live in prosperous places, such as the Central Business District (CBD) of the city. These places are well connected, and have more attractions for tourisms and shopping malls. On the contrary, some people tend to live in less prosperous places, where they can get rid of the traffic jam and the noisy environment of the downtown. Also, some people enjoy natural attractions, so they may want to live closer to them. However, almost all the filters offered by the current online lodging platforms do not support zone-based accommodation selection. For example, in Airbnb, the only location related filter is the districts/suburbs where the accommodations are located. However, for newcomers travelling to an unfamiliar city, they can get very limited information from the names of districts/suburbs.

Helping people choose accommodations based on their zone preferences is of great importance for ensuring good user experience. Several accommodation recommendation systems have been proposed in previous studies[7–11]. These approaches could predict a user's next zone in one city using the historical records of the user in the same city. However, most Airbnb users do not have many booking records, thus the historical booking records in the same city would be sparse, which makes it challenging to analyze users' zone preferences in the traditional ways. Figure 1 displays the distribution of the number of booking records. We can see that most Airbnb users only have 1 booking record, and Airbnb users who have more than 10 booking records are very rare. To solve this problem, we try to involve the historical booking records in all the cities, and the results show that the performance of zone prediction will be significantly promoted in this way.

We assume that users tend to live in places surrounded by similar geographic entities as the places they used to live before, so we need to obtain the geographical features of each accommodation to achieve a better performance. While each city has its own characteristics, it is hard to extract structured geographical data of accommodations in different cities. To overcome this challenge, we propose our solution from two different aspects. Firstly, we use the Swarm dataset and HERE places API to get the information of Points Of Interests

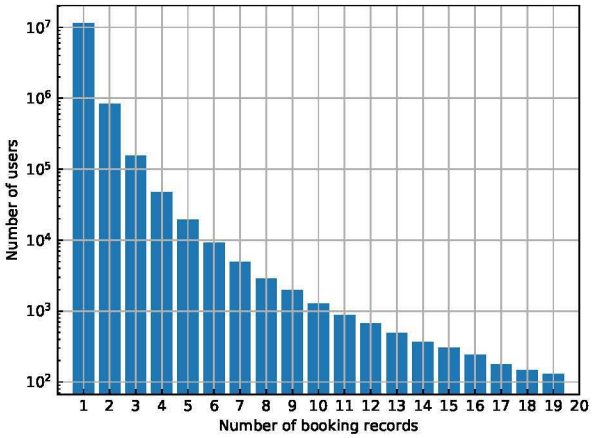


Fig. 1 Distribution of number of Airbnb user booking records.

(POIs) nearby. By processing the information of POIs with a unified standard, we can get the structured geographical data. Secondly, the description of each accommodation always includes the information of public transport and famous places nearby. These textual information can reflect the geographical features to some extent, so we use Sent2Vec[12] to get the embedding of accommodation descriptions.

In this paper, we propose a zone preference prediction system named DeepPredict. Given a city that a user wants to go to, we explore her historical booking records and user profile to predict the desired zone of her next booking. Our system combines the attention-based Bidirectional Long Short-Term Memory (BLSTM) neural networks[13,14] with some conventional supervised machine learning algorithms to make use of both dynamic and descriptive characteristics of each user. We use a real-world dataset collected from Airbnb to evaluate the prediction performance of DeepPredict. Airbnb is one of the world's most popular online lodging platforms which has a history of more than 10 years. In our study, we choose London, New York City, and Melbourne as sample cities. All of these three cities are famous tourist cities and are from different continents. By dividing each city into multiple zones, our DeepPredict system can predict which particular zone a traveler will choose to live in. The results show that DeepPredict performs well when predicting a user's zone preference. Also, the prediction performance of DeepPredict is improved by up to 60% compared with the state-of-the-art algorithms.

The contributions of this paper are summarized as follows:

- Firstly, we perform a comprehensive analysis on

people's booking behavior on Airbnb and find that people may prefer accommodations in a relatively fixed area. Thus, we formulate the zone preference prediction problem in online lodging platforms and perform a series of analysis based on the zone preference of different users.

- Secondly, we propose a zone preference prediction system named DeepPredict to predict the desired zone of a user's next booking. We use attention-based BLSTM to acquire the dynamic features of each user. Then LightGBM is used to do the classification with dynamic and user profile features as input. Our system combines the advantages of attention-based BLSTM and traditional supervised machine learning algorithms to deal with both dynamic and descriptive features of the dataset.
- Thirdly, we evaluate our system using a real-world dataset collected from Airbnb. The results show that under different classification granularities, DeepPredict can achieve an macro AUC of up to 0.933. The experimental performance indicates that our system can predict the zone of a user's next booking successfully and our system can achieve a performance promotion up to 60% when compared with state-of-the-art algorithms.

The rest of this paper is organized as follows. In Section 2, we present our dataset and introduce the cleaning and structuring process. In Section 3, we introduce the methodology of this paper. In Section 4, we demonstrate the experimental results and compare DeepPredict with some state-of-the-art POI recommendation algorithms. In Section 5, we analyze the zone preference difference between different users and discuss the importance of different features. In Section 6, we review the related work. In Section 7, we conclude the paper.

2 Dataset Description and Feature Extraction

In this section, we first introduce the dataset and the fetching method of the original data. Then we give a systematic introduction of the preprocess of feature extraction. We describe the strategy we take to divide the target cities. Last we analyze the zone preferences of different users.

2.1 Dataset description

As one of the world-leading online lodging platforms, Airbnb has about 200 million registered users. It also attracted millions of hosts to rent out their apartments. In Airbnb, users can find over 6 million accommodations in more than 81 000 cities in 191 countries (https://press.airbnb.com/about-us/, accessed on May 1, 2019). Also, millions of deals have been accomplished by Airbnb, which makes it a perfect platform for us to collect data for our study.

In order to capture the accommodation which a user selects in a specific city for her next trip, we collect the historical booking records of each user and the user's personal information from her profile page. Our dataset consists of two parts, i.e., historical booking subset and user profile subset.

(1) Historical booking subset: The historical booking subset contains the reviews to each accommodation that a user has ever booked and the detailed information of accommodation in each review, such as the location, the description, the amenities, the rating, and the price of it. The collection and preprocessing of historical booking subset are described as follows.

Firstly, we collect the accommodation data and review data of all Airbnb accommodations from InsideAirbnb (InsideAirbnb is a website which offers open sourced dataset containing the detailed information of the accommodations and reviews in 84 cities in Airbnb, http://insideairbnb.com/, accessed on May 1, 2019). The InsideAirbnb dataset has been widely used in studies about Airbnb[2, 3]. For each city, we get two files which store the accommodation data and review data of the city, respectively. The accommodation data include price, longitude, latitude, type of Airbnb accommodations, amenities of accommodations, and demographic information of hosts. The review data are the collection of reviews of all the accommodations in the city. Each line of review data represents an actual visit including the Identifier (ID) of the accommodation, time stamp of the review, and ID and name of the guest who lived in this accommodation.

Secondly, we extract all the reviews of the same user and build a user-review related database based on MongoDB (https://www.mongodb.com/, accessed on May 1, 2019). MongoDB is a cross-platform document-oriented database program which uses documents with JavaScript Object Notation (JSON) style to store data.

Finally, we get a subset consisting of more than 20 million users. To make sure that we can get enough information from previous booking records, we select 15 442 users who have at least 7 historical records. In our work, we choose London, New York City, and Melbourne as the target destinations for case study. The

final dataset contains 2045, 1125, and 942 users whose latest booking records are located in London, New York City, and Melbourne, respectively.

(2) User profile subset: A user profile includes name, location, registration time, self description, the total number of reviews, and other verification items, such as work, language, credit card, and government ID. Each registered user in Airbnb has a unique User ID (UID). We can get access to the profile page of users via Uniform Resource Locator (URL): https://www.airbnb.com/users/show/UID. We got the UIDs of the 2045, 1125, and 942 selected users for the three cities, respectively. After that, we design a crawler to get the needed profiles. All the profile data were crawled between March 2019 and July 2019. Note that when crawling the profiles of the users, some of them might not be available. If an account is deleted by the corresponding user or by the Airbnb platform, the profile page of this user will be unavailable.

2.2 Feature extraction

As shown in Table 1, the feature set F_u for each user u consists of 2 parts: the historical booking feature f_{h} and user profile features f_{p}.

(1) Historical booking features: For all of the users, we keep the latest 7 records in temporal order. Then we take the first 6 records to predict the zone of the last booking record. We first need to extract features of the previous 6 booking records.

For each record, the review data contain review ID, time, and city of this accommodation and the comments. Firstly, as in Ref. [15], we use the sentiment score of each comment by Valence Aware Dictionary and sEntiment Reasoner (VADER) Sentiment[16]. VADER is a lexicon and rule based sentiment analysis tool and it is specifically used to detect sentiments expressed in social media. The input data of VADER are a sentence and the output of VADER usually contains four items, i.e., the ratio of positive words, negative words, neural words, and a compound score. The compound sentiment score is a general measure of sentiment of a given sentence. Secondly, we obtain the detailed information of accommodations by looking up the accommodations with their IDs in the accommodation csv files. From the accommodation file, we can get the information of the accommodation, including the price, longitude and latitude, property type*, room type†, accommodates‡, amenities, and cleaning fee. Besides, we can also get the information about the host, including demographic characteristics, the response rate of the host, the total number of accommodations owned by the host, and some verified information that the host offered, such as the government ID and the phone number.

Table 1 Description of feature set F_u.

Category	Feature
Historical booking feature (f_{h})	ID of accommodation in each booking record
	City of accommodation in each booking record
	Time of each booking record
	Sentiment of comments in each booking record
	Amenity of accommodation in each booking record
	Demographic feature of host in each booking record
	Geographical feature (f_{g}) of accommodation in each booking record
User profile feature (f_{p})	ID of user
	City of user
	Created time of user's Airbnb account
	Number of reviews from hosts
	Number of reviews from guests
	Verified information

In addition, we obtain the geographical and semantical information of each accommodation. The geographical information is obtained via Swarm dataset[17] and HERE places API (https://www.here.com/products/location-based-services/poi-tools, accessed on May 1, 2019). Swarm[18] is one of the leading social network websites in the world, users can search for POIs for any city via Swarm. Swarm dataset contains the longitude/latitude, category, and description of each POI. It covers information of 28 892, 61 096, and 7738 POIs from London, New York City, and Melbourne, respectively. Reference [17] described the collection of Swarm dataset and introduced the dataset in detail. To obtain the entity distribution around each accommodation, we calculate the proportions of POIs with different categories within 10 km. The proportions are used as the geographical feature f_{swarm} extracted from the Swarm dataset. HERE places API offers access to POIs in the physical world. It receives Hypertext Transfer Protocol (HTTP) requests with the longitude and latitude of accommodations, and returns the POI information of a certain category, such as shopping mall, airport, or railway station, within 10 km. The API returns up to 20 most popular POIs. The information of each POI includes its latitude and

* The types of Airbnb accommodations, including apartments, villas, and tree houses.

† There are three types of rooms in Airbnb: private room, shared room, and entire home.

‡It refers to the number of people that these accommodations can host at one time.

longitude, the administrative address, the open hour, and the distance between the POI and the accommodation. Due to the access restrictions of HERE places API, we choose to get the geographical information of each accommodation of only 4 categories, including airport, shopping mall, attraction tourism, and transport. Since there will be only one airport at most within 10 km, when the category is set to "airport", we can only get 0 or 1 item through HERE places API. When the category is set to shopping mall, attraction tourism, or transport, we can get up to 20 items. The geographical feature set f_{g} for each accommodation consists of 4 parts: airport feature f_{airport}, shopping mall feature f_{shopping}, tourism attraction feature f_{tourism}, and transport feature $f_{\mathrm{transport}}$.

$f_{\mathrm{airport}} = \{f_1, f_2\}$ includes 2 elements, f_1 will be set to 1 if there is an airport within 10 km. Otherwise, it will be set as 0. If f_1 equals 1, f_2 is the distance between the airport and the accommodation. Otherwise, f_2 will be set as 999 999.

Each of the other three feature sets has 5 elements: $f_{\mathrm{shopping}}, f_{\mathrm{tourism}}, f_{\mathrm{transport}} = \{f'_1, f'_2, f'_3, f'_4, f'_5\}$. f'_1, f'_2, f'_3, and f'_4 represent the number of POIs within 1 km, between 1 km and 2 km, between 2 km and 5 km, as well as more than 5 km, respectively. f'_5 is the average distance of all the POIs.

The semantical information is the embedding feature set of the description of each accommodation. It is extracted via Sent2Vec[12]. Sent2Vec is an unsupervised model which allows to compose sentence embeddings using the word vectors along with n-gram embeddings. The dimension of semantic feature set f_{s} is 700. For each accommodation, the lengths of the descriptive feature set and geographical feature set are 284 and 17, so the dimension of the whole feature set is 1001. We use the extracted historical booking features to formulate a historical feature set $f_{\mathrm{h}}^{6\times 1001} : [h_1, h_2, h_3, h_4, h_5, h_6]$, where h_i represents the extracted feature set of the i-th historical booking record.

(2) User profile features: The user profile feature is extracted from our user profile dataset. It contains the geographical information such as the city that the user lives in, the account information such as the account creation time, the total number of reviews from guests[¶], and the total number of reviews from hosts. It also contains demographic information including the user's education background, work, the government ID, and the other social accounts of users. The feature set of users whose profiles are no longer available is filled by -1.

Table 1 shows the summarized feature set. The dimensions of historical booking feature f_{h} and user profile feature f_{p} are 1806 and 16, respectively.

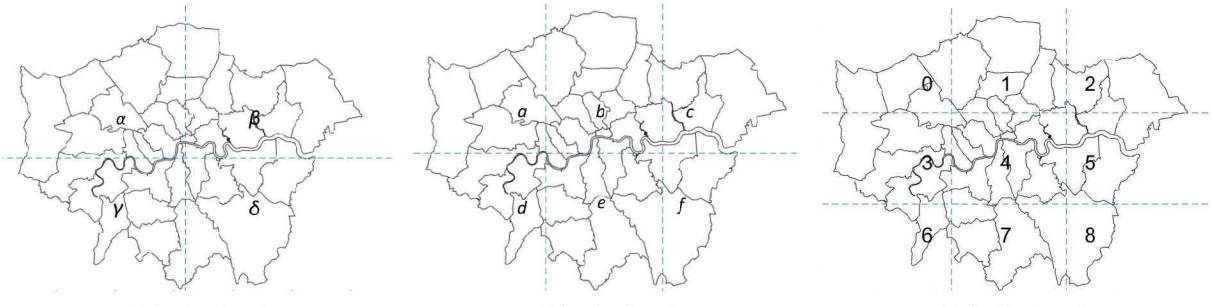
2.3 City partition

The main target of DeepPredict is to predict which part of a given city ϕ that a user u will choose to book when she takes a trip to this city. Thus, we can obtain the zone preferences of users and improve the quality of users' booking experience. To evaluate the prediction performance of our system, we use the latest booking record as the ground truth. We utilize the detailed information of 6 booking records before the latest one as input data, and predict which part of the city that the user chose in her last booking record.

For each city, we divide it into multiple zones according to the grid-based method[19]. We set the number of classes k as 4, 6, and 9. Figure 2 is a division example of London, and the shape file of London is downloaded from https://data.gov.uk/dataset. As shown in Fig. 2, users with the locations of the latest booking record fall in different grids, representing different labels.

Grid-based division relies on the coordinates instead of considering the economic situation of each area. Thus, we analyze the condition of each area in detail and

[¶] The reviews that user received from her guests.



(a) London, $k = 4$ (b) London, $k = 6$ (c) London, $k = 9$

Fig. 2 Division of London.

compare the differences between different zones. The labels of each divided zone are α, β, γ, and δ when $k = 4$; a, b, c, d, e, and f when $k = 6$; 0, 1, 2, 3, 4, 5, 6, 7, and 8 when $k = 9$ as shown in Fig. 2.

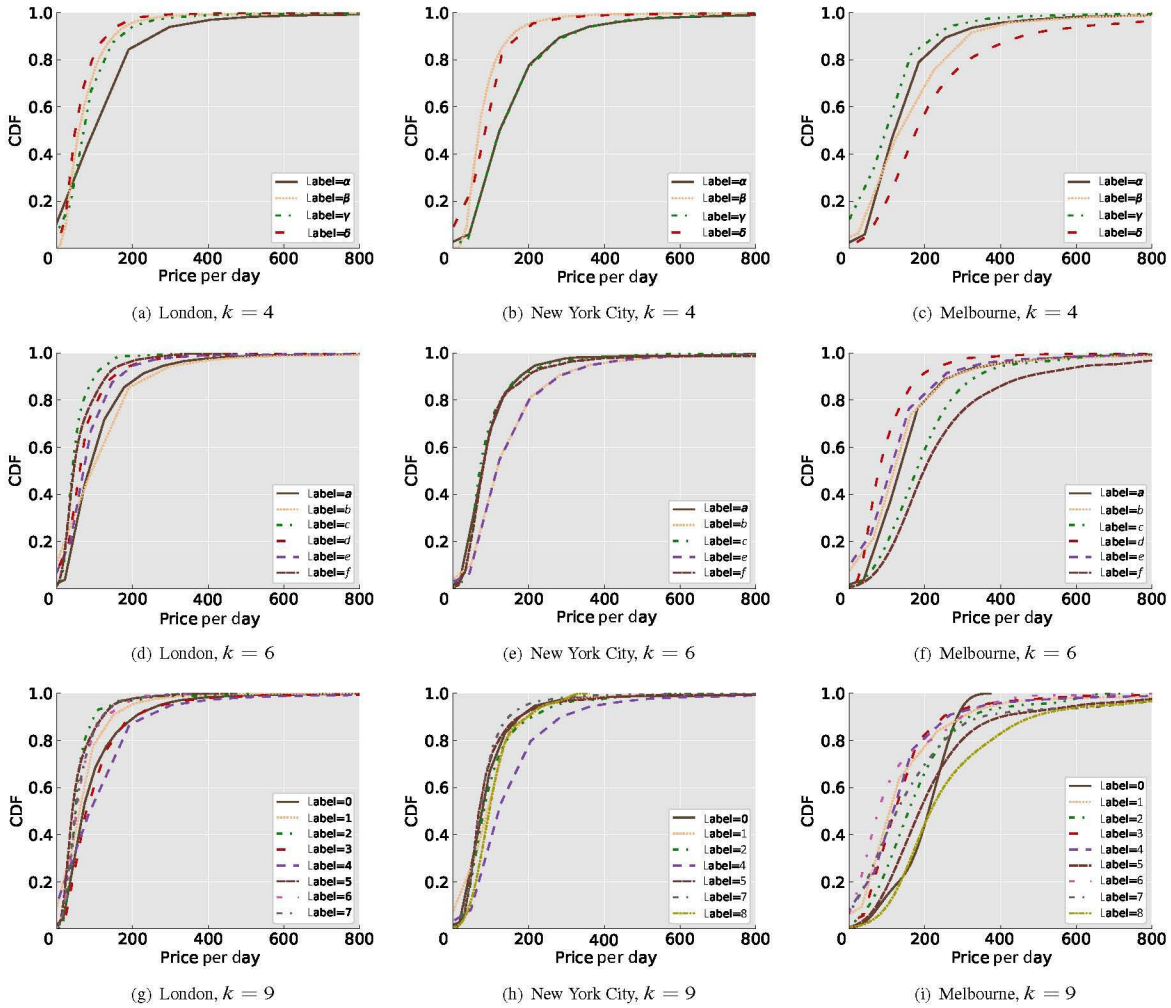
As Ref. [20] shows, the pricing of Airbnb accommodations is determined by the location and neighborhood to some extent. We compare the daily price of Airbnb accommodations in different zones in Fig. 3 to verify whether our division makes sense. We can see from Fig. 3 that in most cases, the daily prices of Airbnb accommodations in different labeled zones have significant differences. When $k = 6$, the 6 zones in London can be divided into 3 classes: a and b; d and e; c and f, which represent the northwest part, the southwest part, and the east part of London, respectively. As for New York City, the prices of Airbnb accommodations have a trend of polarization. The prices of Airbnb accommodations in zones b and c have similar distribution, and are higher than other 4 zones. Yet the price distribution of all different zones in Melbourne has significant differences. As a whole, zone f has the highest price, and the price goes down by step in zones c, a, b, e, and d.

2.4 Airbnb user zone preference analysis

In this part, we verify the existence of user zone preference with collected dataset. Figure 4 shows the distribution of the distances between every 2 booking records of the same user from the same city. We can see from Fig. 4 that the probability of the two bookings of the same user within 10 km is 80%. It indicates that users have zone preferences in most cases. They tend to live in a relatively fixed area when booking accommodations in the same city.

3 Methodology

In this section, we show the detailed information of DeepPredict. Section 3.1 specifies the problem



(a) London, $k = 4$ (b) New York City, $k = 4$ (c) Melbourne, $k = 4$

(d) London, $k = 6$ (e) New York City, $k = 6$ (f) Melbourne, $k = 6$

(g) London, $k = 9$ (h) New York City, $k = 9$ (i) Melbourne, $k = 9$

Fig. 3 Distribution of accommodation price (dollar) per day in different zones. Here, CDF represents cumulative distribution function.

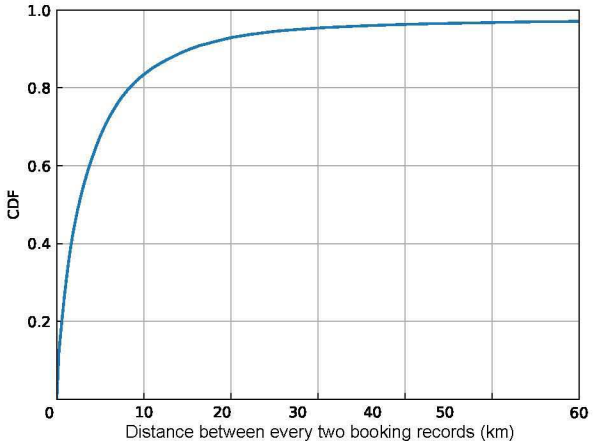


Fig. 4 Distribution of distances between every 2 booking records.

definition. Section 3.2 introduces the framework of DeepPredict.

3.1 Problem statement

In this study, we aim to predict users' zone preferences for a given city according to their profiles and historical booking records. Specifically, our system will take a user's historical booking data, profile, and her target city as the input, then recommend a certain part of the city for her next travel destination in the given city. We choose London, New York City, and Melbourne as case studies, which we denote as C: $\{c_{\mathrm{L}}, c_{\mathrm{N}}, c_{\mathrm{M}}\}$, respectively. U: $\{U_{\mathrm{L}}, U_{\mathrm{N}}, U_{\mathrm{M}}\}$ denote the sample users we extracted from all three cities. The sizes of U_{L}, U_{N}, and U_{M} are 2045, 1125, and 942, respectively.

3.2 System design

To utilize the historical booking records, we need to introduce an algorithm which is capable of processing time series information. There are a lot of techniques to process time sequence data. LSTM networks[13, 14] have shown its power in recent studies[21, 22]. So we involve LSTM networks in our system to synthesis the temporal features of sequential booking records. In this section, we first introduce the framework of DeepPredict and then describe each module in detail. DeepPredict is composed of two parts. The neural network module processes f_{h} and outputs the dynamic feature f_{d}. f_{d} also represents probabilities of users being assigned into each class. The decision maker utilizes combined conventional features f_{h} and f_{p} and dynamic feature f_{d} as input data and outputs the classification results of each sample. The architecture of DeepPredict is shown in Fig. 5.

(1) Neural network module: In the neural network module, we first use Sent2Vec[12] to embed the description of each accommodation. Then we add the embedded features into the corresponding accommodation feature sets, and feed them into neural networks. Finally, we use an attention layer to process the outputs of the fully connected layers to get the final results of the neural network module.

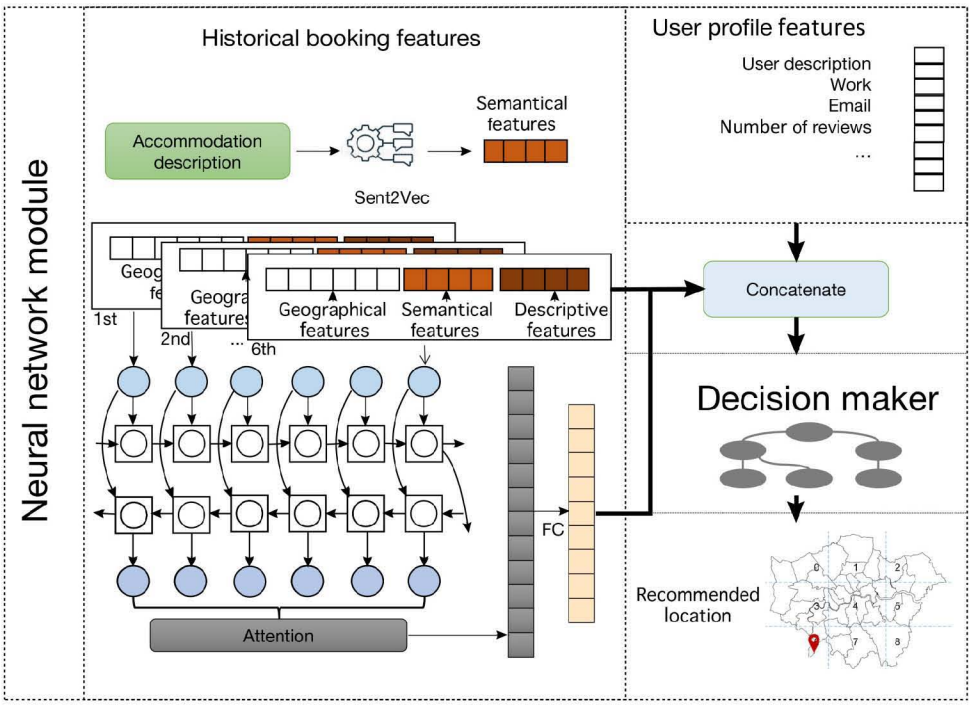


Fig. 5 Architecture of DeepPredict.

In this paper, we try several classic neural networks, such as LSTM[13], BLSTM[23], Gated Recurrent Unit (GRU)[24], Phased LSTM (PLSTM)[25], and ClockWork (CW) RNN[26]. The neural networks process the sequential historical data f_{h} of each user and output a vector containing the probabilities that users will be classified into each possible label.

• **LSTM structure.** LSTM was proposed by Hochreiter and Schmidhuber[13]. It is designed to process long-term dependency information and commonly used to overcome gradient vanishing problems. Compared with traditional recurrent neural networks, the main improvement of LSTM is the introduction of *forget gate*, which determines how much information should be kept from the previous state. The output h_t for the forward pass of an LSTM unit is computed by the following equations:

$$f_t = \sigma\left(W_f x_t + V_f h_{t-1} + b_f\right) \tag{1}$$

$$i_t = \sigma\left(W_i x_t + V_i h_{t-1} + b_i\right) \tag{2}$$

$$o_t = \sigma\left(W_o x_t + V_o h_{t-1} + b_o\right) \tag{3}$$

$$c_t = f_t c_{t-1} + i_t \sigma_{\mathrm{h}}\left(W_c x_t + V_c h_{t-1} + b_c\right) \tag{4}$$

$$h_t = o_t \sigma_{\mathrm{h}}\left(c_t\right) \tag{5}$$

where t represents the time of the current step and $t-1$ represents the previous step. σ and σ_{h} are activation functions, representing a sigmoid function and hyperbolic tangent function named tanh(), respectively. x_t means the input vector of each LSTM unit at time t. W, V, and b are weight metrics and bias parameters which need to be learned during training. f_t is the forget gate, it determines the extent to which the existing memory is forgotten. The input gate's activation vector is i_t, which defines the degree to which the current input information is added to the memory cell. o_t is the output vector of each LSTM unit at time t. c_t is the cell state vector which integrates the information from the forget gate and input gate. Finally, the output h_t is computed based on the output gate. h_t is the hidden state vector and also known as output vector of the LSTM unit at time t.

• **BLSTM structure.** The current cell state and the output of an LSTM unit are generated by previous and current input vectors. However, for some sequence modeling tasks, future information can improve the performance of LSTM models a lot. Hence, we introduce BLSTM network[23], which is an extension to unidirectional LSTM network by adding a backward LSTM layer. BLSTM has a capability to utilize both previous and future input vectors.

$$h_t = [\overrightarrow{h_t} \oplus \overleftarrow{h_t}] \tag{6}$$

• **GRU structure.** GRU is a variation of LSTM and it is proposed by Cho et al.[24]. A GRU unit can capture the dependency of information in different time scales. Similar to LSTM units, GRU units bring in gating unit to modulate the flow of information inside them. However, the GRU structure does not have separate memory cells, which makes it more concise. In a GRU unit, the output h_t is computed by the following equations:

$$z_t = \sigma\left(W_z\left[h_{t-1}, x_t\right]\right) \tag{7}$$

$$r_t = \sigma\left(W_r\left[h_{t-1}, x_t\right]\right) \tag{8}$$

$$\widetilde{h_t} = \sigma_{\mathrm{h}}\left(W\left[r_t h_{t-1}, x_t\right]\right) \tag{9}$$

$$h_t = \left(1 - z_t\right) h_{t-1} + z_t \widetilde{h_t} \tag{10}$$

where z_t is the update gate, which helps the GRU in retaining the cell state as long as it is needed. r_t is the reset gate, it decides how relevant the previous information is. If r_t is close to 0, the GRU unit will forget the previously computed cell state. $\widetilde{h_t}$ represents the candidate gate, which is dependent on the cell state at previous timestamp h_{t-1} and the reset gate r_t. The final output is computed by the update gate z_t and the candidate gate $\widetilde{h_t}$. W_z, W_r, and W are weight metrics which need to be trained in the learning process.

• **PLSTM structure.** PLSTM[25] structure extends LSTM by adding a new gate, k_t. The gate is controlled by an independent rhythmic oscillation specified by three parameters: τ, r_{on}, and s. The updates of the cell state c_t and the hidden state h_t are permitted only when k_t is open. τ controls the real-time period of the oscillation. r_{on} controls the ratio of the "open" phase to the whole period. s controls the phase shift of the oscillation to each PLSTM cell. All the parameters can be learned during the learning process. k_t is calculated by a linearized formulation as follow:

$$\xi_t = \frac{(t-s)\mathrm{mod}\tau}{\tau},$$

$$k_t = \begin{cases} \dfrac{2\tau}{r_{\mathrm{on}}}, & \text{if } \xi_t < \dfrac{1}{2} r_{\mathrm{on}}; \\ 2 - \dfrac{2\xi_t}{r_{\mathrm{on}}}, & \text{if } \dfrac{1}{2} r_{\mathrm{on}} < \xi_t < r_{\mathrm{on}}; \\ \omega \xi_t, & \text{otherwise} \end{cases} \tag{11}$$

Then the update equations for c_t and h_t (Eqs. (4) and (6)) can be rewritten using the proposed cell updates $\widetilde{c_t}$ and $\widetilde{h_t}$ mediated by the time gate k_t. ω is a rate.

$$\widetilde{c_t} = f_t c_{t-1} + i_t \sigma_{\mathrm{h}}\left(W_c x_t + V_c h_{t-1} + b_c\right) \tag{12}$$

$$c_t = k_t \widetilde{c_t} + (1 - k_t)\, c_{t-1} \tag{13}$$

$$\widetilde{h_t} = o_t \sigma_{\mathrm{h}}(c_t) \tag{14}$$

$$h_t = k_t \widetilde{h_t} + (1 - k_t)\, h_{t-1} \tag{15}$$

Attention layer. Attention-based neural networks have demonstrated success in lots of works, ranging from machine translations[27], speech recognition[28], to image captioning[29]. In the attention layer, we use H to denote the output vectors $[h_1, h_2, \ldots, h_6]$ which are produced by the previous network. The output ψ of attention layer is calculated by a weighted sum of these output vector H:

$$M = \tanh(H) \tag{16}$$

$$\lambda = \mathrm{softmax}(\boldsymbol{w}^{\mathrm{T}} M) \tag{17}$$

$$\psi = H\boldsymbol{\lambda}^{\mathrm{T}} \tag{18}$$

where $\boldsymbol{w}$ is a trained parameter vector and $\boldsymbol{w}^{\mathrm{T}}$ is its transpose.

The loss function we use in this paper is called cross-entropy loss, which is commonly used in classification tasks. The equation of cross entropy loss is given:

$$\mathrm{Loss} = -\left(\sum_{i=1}^{k} (Y \log \hat{y})\right) \tag{19}$$

where Y is the ground truth label and $\hat{y}$ represents the probability to be classified as different classes.

In this paper, our neural network models is constructed by Keras (https://keras.io/, accessed on May 1, 2019), a high-level neural network API which is capable of running on top of some representative deep learning platforms, such as TensorFlow (https://www.tensorflow.org/, accessed on May 1, 2019). The learning rate is set as 0.0001. We utilize a dropout layer to prevent the over-fitting problem, and the dropout ratio is set to 0.6. Also, we use the Adam optimizer to optimize our model in the training process. In the learning process, the last output of the neural network model will be sent to a 3-layer Fully Connected layer (FC layer). Then the output of FC layer will go through a softmax layer to get the probability vector.

(2) Decision maker: The decision maker is powered by a conventional supervised machine learning classifier. After processing the historical booking features in neural network module, we concatenate the output vector f_{d} of neural network module, the historical booking features $f_{\mathrm{h}}^{6\times 301}$: $\{h_1, h_2, h_3, h_4, h_5, h_6\}$, and user profile feature f_{p} to get a final feature set F_u. We use F_u as the input of decision maker to get the classification results. In this paper, we choose several frequently-used machine learning algorithms, including Random Forest[30], Decision Tree[31], XGBoost[32], and LightGBM[33]. In the training process, GridSearchCV[34] is applied to get the optimal parameters of each algorithm automatically. Given a set of values of the parameters which need to be tuned, GridSearchCV iterates through each parameter combination and records the parameters which lead to best F1-score. For all the machine learning algorithms, we use a 5-fold cross-validation to avoid over-fitting.

4 Experiment and Result

The prediction performance of DeepPredict is evaluated in this section. Firstly, we show the implementation details of DeepPredict in Section 4.1. In order to get a better performance, we compare the experimental results of different neural networks and decision makers in Sections 4.2 and 4.3. The results show that DeepPredict can achieve the best performance with attention-based BLSTM and LightGBM to implement the neural network module and decision maker, respectively. Also, to evaluate the significance of each feature subset, we compare DeepPredict with some of its variants in Section 4.4. In Section 4.5, we compare the performance of DeepPredict with other state-of-the-art algorithms. Last, we evaluate the robustness of DeepPredict in Section 4.6.

4.1 Implementation details

We first randomly select 80% of samples as the training and validation set, and use the other 20% as the testing set. Note that we use the same training, validation, and testing set in deep neural networks and decision maker. Also, we use Synthetic Minority Oversampling Technique (SMOTE)[35] to balance the dataset. SMOTE is a widely used oversampling method which is proposed to improve random oversampling. This over-sampling method creates synthetic minority class examples, and the experiments show that the SMOTE approach can improve the accuracy of classifiers for a minority class a lot. We use macro F1-score and macro AUC as the evaluation metrics.

- **Macro F1-score.** The average of F1-scores of all the classes. For each class, the F1-score means the harmonic mean of precision and recall of this class. The precision represents the fraction of correctly classified samples to all the samples which are classified as this class. The recall means the fraction of correctly classified samples to all samples which belong to this class.
- **Macro AUC.** The average of AUC of all the

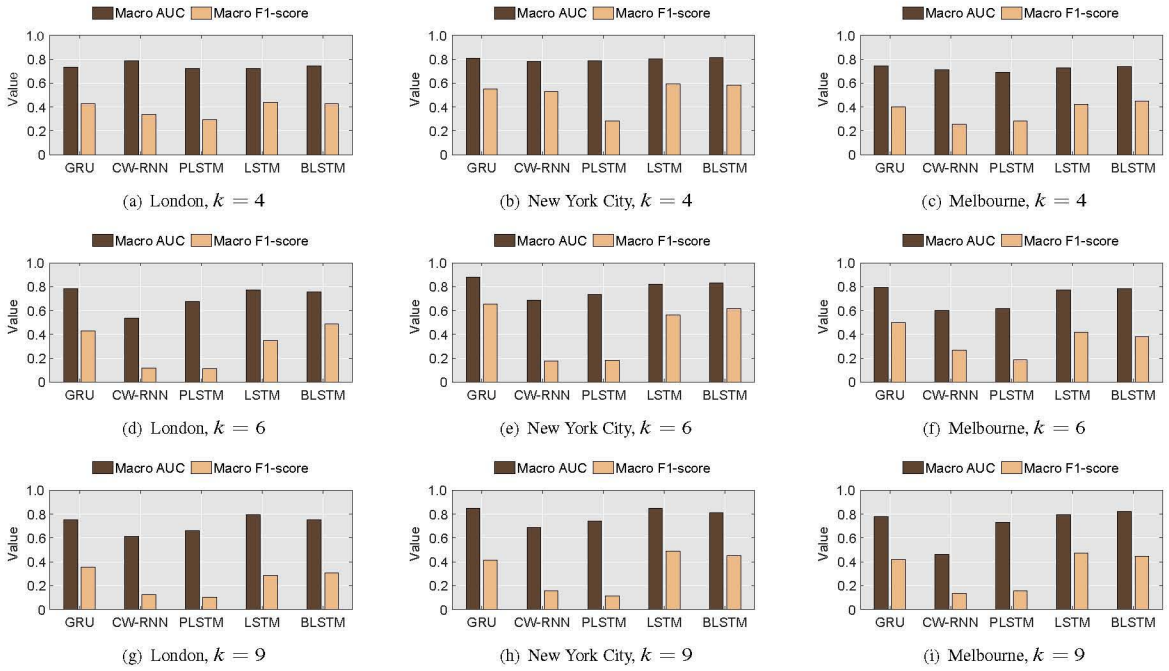
classes. AUC means area under the Receiver Operating Characteristic (ROC) curve. It represents the expectation that a uniformly drawn random positive is ranked before a uniformly drawn random negative.

4.2 Comparison of different neural network models

LSTM and its variations, such as GRU, BLSTM, and PLSTM, have been widely used in classification tasks. In this paper, we evaluate the prediction performance of five algorithms on our dataset. Figure 6 shows the performance of different neural networks. We can see that GRU and BLSTM perform the best in most cases. In DeepPredict, we choose to use BLSTM in the neural network module.

4.3 Comparison of different classifiers

The decision maker in DeepPredict does the final classification. Various machine learning based classifiers can be applied in this stage. We evaluate the performance of different machine learning algorithms, including XGBoost, Random Forest (RF), LightGBM, CatBoost, and Decision Tree (DT). We can see from Table 2 that in most cases, when applying LightGBM as the decision



(a) London, $k = 4$ (b) New York City, $k = 4$ (c) Melbourne, $k = 4$

(d) London, $k = 6$ (e) New York City, $k = 6$ (f) Melbourne, $k = 6$

(g) London, $k = 9$ (h) New York City, $k = 9$ (i) Melbourne, $k = 9$

Fig. 6 Prediction performance of different neural networks.

Table 2 Prediction performance of different classifiers.

City	Classifier	$k = 4$		$k = 6$		$k = 9$	
		Macro F1-score	Macro AUC	Macro F1-score	Macro AUC	Macro F1-score	Macro AUC
London	RF	0.670	0.916	0.740	0.907	0.672	0.910
	DT	0.451	0.730	0.566	0.806	0.577	0.822
	XGBoost	0.649	0.919	0.783	0.918	0.678	0.908
	LightGBM	**0.703**	**0.928**	**0.793**	**0.933**	0.665	**0.921**
	CatBoost	0.654	0.923	0.782	0.926	**0.695**	0.920
New York City	RF	0.642	0.884	0.807	0.910	0.715	0.917
	DT	0.474	0.669	0.578	0.782	0.435	0.802
	XGBoost	0.700	0.921	0.790	0.908	0.731	0.913
	LightGBM	**0.707**	**0.926**	**0.817**	**0.918**	**0.756**	**0.928**
	CatBoost	0.680	0.923	0.816	0.901	0.712	0.896
Melbourne	RF	0.557	0.839	0.728	0.912	0.847	0.919
	DT	0.474	0.646	0.638	0.838	0.648	0.859
	XGBoost	**0.579**	0.841	0.748	0.912	0.887	0.924
	LightGBM	0.447	0.850	**0.770**	**0.928**	**0.865**	**0.927**
	CatBoost	0.566	**0.884**	0.766	0.920	0.838	0.922

maker, DeepPredict can achieve the best performance. Thus, we end up using LightGBM as the decision maker in DeepPredict.

4.4 Comparison with variants of DeepPredict

The final feature set of DeepPredict consists of several parts, such as the semantical and geographical features of each accommodation, the profile features of each user, and the dynamic features obtained via neural network module. To better evaluate the importance of each feature subset, we list several variants of DeepPredict:

- **DeepPredict w/o nn** removes the outputs of the neural network module from the final feature set.
- **DeepPredict w/o sem** removes the semantic features of each accommodation from the final feature set.
- **DeepPredict w/o geo** removes the geographical features of each accommodation from the final feature set.
- **DeepPredict w/o pro** removes the user profile features from the final feature set.
- **DeepPredict w/o att** removes the attention layer and computes the results of the neural network module by the BLSTM algorithm.

Table 3 shows the experimental results of DeepPredict and its variants. We can see from Table 3 that among all the variants, DeepPredict w/o pro performs the best, which indicates that user profile features can barely affect the prediction performance of DeepPredict. DeepPredict w/o nn performs the worst, which implies that the output of the neural network module plays an important role in the prediction performance of DeepPredict. The prediction performance of DeepPredict w/o sem and DeepPredict w/o geo are similar but both are inferior to DeepPredict. The results of DeepPredict w/o att are worse than those of DeepPredict, which implies that the attention layer can enhance the prediction performance of DeepPredict.

4.5 Comparison with state-of-the-art algorithms

In this part, we compare DeepPredict with some state-of-the-art algorithms.

- LCA-LDA[9]: LCA-LDA was proposed to deal with out-of-town POI recommendation problems. This method extracted the user's personal interest characteristics and local personal preferences. Based on these two characteristics, recommendations were made to the POI that the user would visit in other cities.
- PRME[10]: PRME (Personalized Ranking metric embedding Method) used the paired metric embedding algorithm to model the user's POI mobility trajectory and the relationship between user and item (POI). Through analysis, the authors also found that users were inclined to visit a POI that was close to the current POI[10]. Therefore, a function was designed to obtain the influence factor of the user's geographical zone and finally recommend the POI the user would go next.
- FRMC-LR[36]: FRMC-LR adopted the user's historical check-in trajectory and calculated the user's

Table 3 Performance of DeepPredict and its variants.

City	Algorithm	$k = 4$		$k = 6$		$k = 9$	
		Macro F1-score	Macro AUC	Macro F1-score	Macro AUC	Macro F1-score	Macro AUC
London	DeepPredict	0.703	0.928	0.793	0.933	0.665	0.921
	DeepPredict w/o nn	0.582	0.868	0.655	0.879	0.604	0.850
	DeepPredict w/o sem	0.673	0.917	0.783	0.923	0.665	0.892
	DeepPredict w/o geo	0.675	0.915	0.772	0.918	0.663	0.903
	DeepPredict w/o pro	0.694	0.921	0.788	0.930	0.660	0.918
	DeepPredict w/o att	0.672	0.911	0.767	0.924	0.661	0.905
New York City	DeepPredict	0.707	0.926	0.817	0.918	0.756	0.928
	DeepPredict w/o nn	0.605	0.837	0.758	0.884	0.634	0.837
	DeepPredict w/o sem	0.647	0.882	0.813	0.902	0.750	0.903
	DeepPredict w/o geo	0.699	0.907	0.794	0.900	0.733	0.905
	DeepPredict w/o pro	0.700	0.919	0.814	0.919	0.749	0.925
	DeepPredict w/o att	0.698	0.915	0.808	0.903	0.696	0.895
Melbourne	DeepPredict	0.447	0.850	0.770	0.928	0.865	0.927
	DeepPredict w/o nn	0.359	0.804	0.725	0.887	0.698	0.837
	DeepPredict w/o sem	0.447	0.834	0.760	0.917	0.744	0.903
	DeepPredict w/o geo	0.429	0.833	0.755	0.910	0.742	0.886
	DeepPredict w/o pro	0.440	0.848	0.761	0.923	0.857	0.924
	DeepPredict w/o att	0.407	0.830	0.744	0.919	0.739	0.893

Markov Transition Matrix based on the historical check-in trajectory. With this transition matrix, the similarities between every two users were calculated. Then the final recommendation results based on the similarities could be obtained.

The outputs of these state-of-the-art methods were top z (we choose z =100 in this paper) recommended POIs. We transfer the results to the most recommended zone to make them comparable with DeepPredict.

Firstly, we compute the proportion of Airbnb accommodations in each labeled grid to the total accommodations of the whole city:

$$A_i = \frac{n_i}{N} \tag{20}$$

where n_i represents the number of accommodations in grid i, and N means the number of accommodations in the whole city.

Secondly, we compute the proportion of recommended POIs in each labeled grid to all the recommended POIs:

$$R_i = \frac{r_i}{R} \tag{21}$$

where r_i represents the number of recommended accommodations in grid i, and R means the number of recommended accommodations in the whole city.

Then according to Formula (22) and Eq. (23), we use grid i_{target} as the final classification result:

$$\text{Label} = \begin{cases} \alpha, \beta, \gamma, \delta, & k = 4; \\ a, b, c, d, e, f, & k = 6; \\ 0, 1, 2, 3, 4, 5, 6, 7, 8, & k = 9 \end{cases} \tag{22}$$

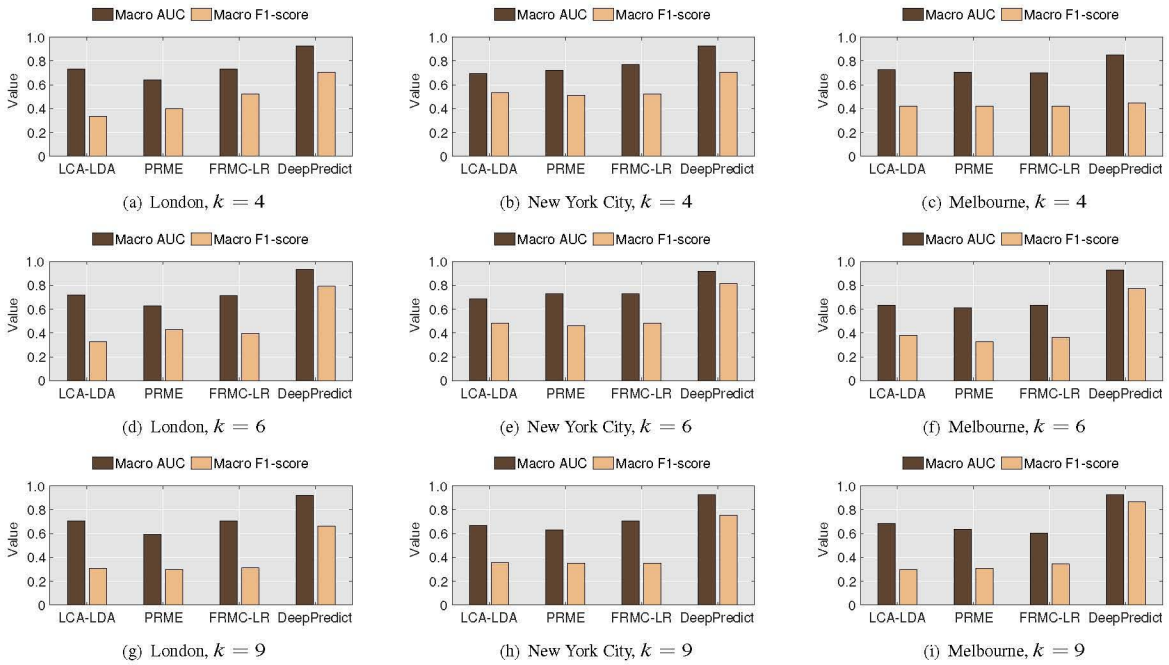
$$i_{\text{target}} = \arg\max_{i \in \text{label}} \left(\frac{R_i}{A_i} - 1 \right) \tag{23}$$

Figure 7 shows the performance of DeepPredict and the state-of-the-art algorithms in all our experiments. It can be observed that DeepPredict significantly outperforms the state-of-the-art algorithms. For example, compared with LCA-LDA, PRME, and FRMC-LR, when the the target city is London, DeepPredict improves macro F1-score by 22.5%, 24.1%, and 40.9% when the number of classes is 4, 35.8%, 43.0%, and 45.8% when the number of classes is 6, and 58.5%, 60.6%, and 61.8% when the number of classes is 9. The observation are summarized as follows:

- DeepPredict outperforms all the state-of-the-art algorithms. On one hand, it combines user personal features (user profile features) and zone preferences (geographical features) together. On the other hand, the deep neural networks and the classic machine learning method solve the data sparsity problem to some extent.
- LCA-LDA outperforms PRME, which indicates that the zone preferences extracted from the historical booking records in other cities of a special user might be useful to predict the desired zone for her next trip.
- FRMC-LR performs not as good as other algorithms. We think the main reason is that deep neural networks and machine learning methods might perform better than Markov transition matrics when dealing with sparse data.

4.6 Robustness analysis of DeepPredict

As aforementioned, the previous experiment selects users with more than six historical records, while 90% users in Airbnb have less than 7 historical records. In that



(a) London, $k = 4$ (b) New York City, $k = 4$ (c) Melbourne, $k = 4$

(d) London, $k = 6$ (e) New York City, $k = 6$ (f) Melbourne, $k = 6$

(g) London, $k = 9$ (h) New York City, $k = 9$ (i) Melbourne, $k = 9$

Fig. 7 Performance of state-of-the-art algorithms and DeepPredict

case, we need to validate that if DeepPredict is useful for most Airbnb users. Thus, we design a robustness experiment to testify the extensibility of DeepPredict. In this experiment, we choose users with two or more than two historical records and use all the historical records except the last one as the input sequence data of the neural network module. Table 4 shows the results of the robustness experiment. It shows that the macro AUC values of most cases are still around 0.9, which indicates that DeepPredict can still predict the zone preference of users with fewer historical records. The results of the robustness experiment further confirmed the extensibility of DeepPredict.

5 Feature Importance Analysis and Discussion

In this section, we want to see how different features contribute in DeepPredict, so we analyze the zone preference difference between people from different continents. Also, we use a representative method named SHAP[37] to do interpretations for the input features. It can estimate the contribution of each feature in a prediction model.

As described before, the feature set of each target user includes the profile information, such as the city and the verified information. In this part, we analyze the zone preferences of users from different continents. We collect the corresponding continents of each user based on their current cities, and Fig. 8 shows the continent distribution of users in the London dataset (U_{L}) of different labels. In general, we can see that most users (over 80%) in the London dataset are from Europe in all 4 areas, which means that the users who book London accommodations and have over 7 historical records are most likely from Europe. Also, we can see that for European users, their labels are most likely β, which means that they tend to choose accommodations in the northeast part of London. While for Asian, North American, and Oceanian users, their labels are most likely γ, which represents the southwest part of London. The findings show that users from different continents may have different zone preferences.

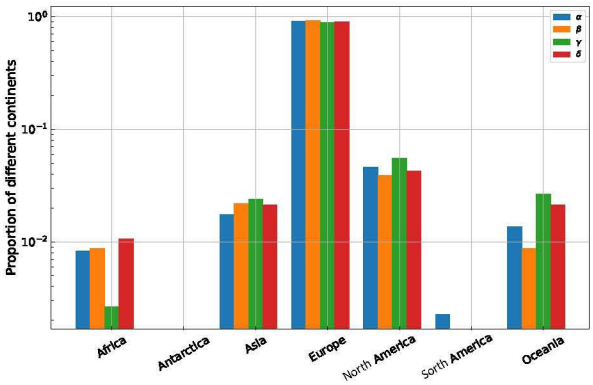


Fig. 8 Distribution of continents of users in London dataset (U_{L}) (k=4).

Machine learning models are mostly deployed as black-box models. Given a set of input data, they return the corresponding output data, while the internal information is not disclosed. In this part, we want to interpret the model to some extent, thus we can learn more about the internal decision making procedure of DeepPredict and evaluate the importance of each feature. SHAP is a well-known method of model interpretation. It can calculate the feature importance by comparing the model outputs with or without each feature.

Table 5 shows the top 20 most important features with highest SHAP values. We can see that the most important feature is the number of reviews per month in the 5th historical record (the 5th historical record represents the second latest booking record over all 6 input booking records). And the 2nd and 3rd most important features are among the dynamic features, which represents the significance of the neural network module. Also, we can see that the SHAP values of some sociological features of hosts, such as the average response time or the response rate, are relatively high. The descriptive features of each accommodation, such as the average score of reviews, the number of reviews or the number of bedrooms, are also important. In addition, the geographical features, such as the average distance of "transport" nearby or the average distance of "shopping" nearby, also contribute a lot to the final prediction. Observing the whole Table 5, we can see that the features of all 6 historical records contribute to the

Table 4 Results of DeepPredict robustness experiment.

City	$k=4$		$k=6$		$k=9$	
	Macro F1-score	Macro AUC	Macro F1-score	Macro AUC	Macro F1-score	Macro AUC
London	0.791	0.858	0.781	0.901	0.728	0.892
New York City	0.655	0.889	0.838	0.902	0.734	0.901
Melbourne	0.582	0.847	0.782	0.907	0.820	0.895

Table 5 SHAP values for top 20 features.

Feature	SHAP value
Number of reviews per month (5th record)	2.73
The second element of dynamic features (neural network output)	2.35
The third element of dynamic features (neural network output)	1.91
Average response time of the host (2nd record)	1.62
Average score of reviews (4th record)	1.55
Time of the host starting to rent houses in Airbnb	1.47
Possibility that host will reply (2nd record)	1.38
Will the accommodation accept extra people? (2nd record, 0 or 1)	1.17
Average score of reviews (3rd record)	0.83
The minimum nights should be booked (2nd record)	0.78
Is the host a superhost? (2nd record, 0 or 1)	0.76
Average distance of "transport" nearby (3rd record)	0.69
Number of reviews (1st record)	0.63
Average distance of "shoppings" nearby (4th record)	0.59
Receptible number of people (6th record)	0.53
Number of bedrooms (4th record)	0.48
Will the accommodation accept extra people? (6th record, 0 or 1)	0.45
Average distance of "tourism" nearby (1st record)	0.43
Average distance of "shoppings" nearby (6th record)	0.41
Average distance of "shoppings" nearby (3rd record)	0.40

final result to a considerable level, which means that the zone preference of each user may not be easily changed over time. DeepPredict can learn the zone preference not only from the recent records, but also from the earlier records.

The SHAP analyzation results of important features also reveal an interesting sociological phenomenon. Table 5 shows that issues like "number of reviews per month", "the average response time of host", "the average score of the reviews", "the time of the host starting to rent houses in Airbnb", "the possibility that host will reply", "will the accommodation accept extra people", "the minimum nights should be booked", and "is the host a superhost" are within the most important features. All these features quantify the service that the host offers from different perspectives. Among the most important 11 features, 9 of them are related to service quality. That is to say, users would value the service quality that host offers more than the accommodation itself. According to the service quality theory[38–40], the service quality and the customer satisfaction are independent but closely related, which means that the increase of service quality is likely to lead to an increase in customer satisfaction. The finding implies that the service quality of the host is more important than the quality of the accommodation itself to some extent, which validates service quality theory.

6 Related Work

6.1 Research on Airbnb

On account of the rapid development of Airbnb, there have been some pieces of research about the profiles of Airbnb users, the accommodation in Airbnb, and the comparison of Airbnb accommodations and traditional hotels. Fradkin et al.[1] did a field experiment on reviews from Airbnb and found that reviews were typically informative but negative experiences were under-reported. Ma et al.[2] studied the profiles of Airbnb users and drew a conclusion that Airbnb hosts who disclosed more information on the profile could gain more trust from guests. Lee et al.[41] analyzed the social features associated with accommodations and found the most significant features for room sale in Airbnb. Quattrone et al.[3] did a cross-ref analysis of Airbnb economy with Foursquare data, census data, and hotel data in London. Also, Grbovic and Cheng[42] gave real-time recommendations for users in Airbnb based on their click data and search history. Zhou et al.[43] presented a comprehensive and evolutionary study of Airbnb, using the information of 43.8 million users.

However, to the best of our knowledge, the previous works are mostly measurement analysis of Airbnb users, comments, or accommodations. None of the previous works studied the zone preferences of users when booking on Airbnb.

6.2 Research on sequenctial POI recommendation

(1) Markov chain based methods: Rendle et al.[44] proposed a recommendation algorithm Factorized Personalized Markov Chains (FPMC) based on Markov chains. He and McAuley[45] introduced a model named Fossil which fused Markov chain and similarity-based model. Yang et al.[36] calculated the similarities between users and constructed a similarity matrix. In the meantime, they used the historical records to get the Markov chains of all the users.

(2) Machine learning based methods: Monreale et

al.[7] trained a T-pattern Tree to learn the trajectory patterns of a certain area and found out the best matching next location in the tree. Baraglia et al.[8] used Gradient Boosted Regression Trees and Ranking SVM to predict the next geographical position given the location history of a certain tourist. Yin et al.[9] proposed Location-Content-Aware Recommender System (LCARS) that offers a particular user a set of venues (e.g., restaurants) or events (e.g., concerts and exhibitions) by giving consideration to both personal interest and local preference. Feng et al.[10] studied the next new POI recommendation problem and proposed a PRME to model personalized check-in sequences. Yang et al.[36] presented a method to provide POI recommendation based on the user's current location and historical information. Chang et al.[11] proposed Content-Aware POI Embedding (CAPE) model which utilized the text content to capture the characteristics of POIs and the check-in sequence to capture the geographical influence of POIs simultaneously.

Besides, neural networks have been widely used in sequential recommendation recently. Hidasi et al.[46] used GRU to do sequential recommendation and could achieve a promotion of over 20% compared with other benchmarks. References [47–52] were all based on Ref. [46]. Reference [47] was another work of Hidasi et al., and it used pair-wise loss function instead of cross-entropy loss. Li et al.[48] proposed Neural Attentive Recommendation Machine (NARM). NARM used attention-based neural network to process the sequential data. Chen et al.[49] tried to use an intuitional, efficient, and dynamic way to model the historical records. They fused matrix factorization and memory network and proposed Memory-Augmented Neural Network (MANN)[49]. Quadrana et al.[50] proposed an algorithm named HRNNs (a model based Hierarchical RNN). Compared with previous work, it had some optimization on the architecture of neural networks. HRNNs added a GRU layer on top of the previous model.

There are some other works which are extension works of Ref. [46]. Lv et al.[51] and Ma et al.[52] introduced neural networks with different architectures. Reference [47, 53–55] added contextual information, which promoted the accuracy of recommendation and enhanced the interpretability of these algorithms to some extent.

(3) Summary: The researches on sequential recommendation are divided into two parts: Markov chain based methods and machine learning based methods. Among them, Markov chain related studies need to integrate all the corresponding POI information in the city, so most of these works use the information of the same city to recommend the POI that the user will visit next in the city. However, most of the users' historical records in our Airbnb dataset are not located in the same city, so this approach is not applicable in the context of the problem we are trying to solve. Machine learning based approaches use similar methods to our work, but there are some differences in feature extraction and model construction. In general, the feature set we used is more comprehensive and more focused on the geographically related feature set. In addition, our model not only uses a variety of commonly used neural network models (LSTM, BLSTM, PLSTM, GRU, and CW-RNN), but also combines the attention mechanism to effectively classify users.

7 Conclusion and Future Work

In this paper, we study the users' zone preferences in booking accommodations on online lodging platforms. We propose a deep learning based zone preference prediction system, called DeepPredict. Our system combines deep learning and traditional machine learning algorithms. It utilizes a user's fine-grained historical booking records and descriptive characteristics. We implement DeepPredict and evaluate it using a real-world dataset collected from Airbnb. We select London, New York City, and Melbourne as our target cities and use a grid-based method to partition the cities into 4, 6, and 9 zones. Our evaluation results show that DeepPredict can predict the zone preference of a user's next booking in London with macro AUC values of 0.928, 0.933, and 0.921 when the numbers of grids are 4, 6, and 9, respectively. We also demonstrate the usefulness of DeepPerdict in New York City and Melbourne.

In future work, we will use the data of more cities to further validate the prediction performance of DeepPredict on Airbnb. Also, we will conduct experiments with the datasets from other online lodging platforms to evaluate the compatibility of DeepPredict.

Acknowledgment

This work was sponsored by the National Natural Science Foundation of China (Nos. 71731004, 62072115, 61602122, and 61971145), Shanghai Pujiang Program (No. 2020PJD005), the Research Grants Council of Hong Kong (No. 16214817), and the 5GEAR Project and FIT Project from the Academy of Finland.

References

[1] A. Fradkin, E. Grewal, D. Holtz, and M. Pearson, Bias and reciprocity in online reviews: Evidence from field experiments on airbnb, in *Proc. 16th ACM Conf. Economics and Computation*, Portland, OR, USA, 2015, p. 641.

[2] X. Ma, J. T. Hancock, K. L. Mingjie, and M. Naaman, Self-disclosure and perceived trustworthiness of airbnb host profiles, in *Proc. 2017 ACM Conf. Computer Supported Cooperative Work and Social Computing*, Portland, OR, USA, 2017, pp. 2397–2409.

[3] G. Quattrone, D. Proserpio, D. Quercia, L. Capra, and M. Musolesi, Who benefits from the "sharing" economy of airbnb?, in *Proc. 25th Int. Conf. World Wide Web*, Montreal, Canada, 2016, pp. 1385–1394.

[4] T. Ikkala and A. Lampinen, Monetizing network hospitality: Hospitality and sociability in the context of airbnb, in *Proc. 18th ACM Conf. Computer Supported Cooperative Work & Social Computing*, Vancouver, Canada, 2015, pp. 1033–1044.

[5] A. Lampinen and C. Cheshire, Hosting via airbnb: Motivations and financial assurances in monetized network hospitality, in *Proc. 2016 CHI Conf. Human Factors in Computing Systems*, San Jose, CA, USA, 2016, pp. 1669–1680.

[6] J. P. Mellinas, S. M. M. María-Dolores, and J. J. B. García, Booking.com: The unexpected scoring system, *Tourism Management*, vol. 49, pp. 72–74, 2015.

[7] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti, WhereNext: A location predictor on trajectory pattern mining, in *Proc. 15th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Paris, France, 2009, pp. 637–646.

[8] R. Baraglia, C. I. Muntean, F. M. Nardini, and F. Silvestri, LearNext: Learning to predict tourists movements, in *Proc. 5th Italian Information Retrieval Workshop*, Roma, Italy, 2014, pp. 75–79.

[9] H. Z. Yin, Y. Z. Sun, B. Cui, Z. T. Hu, and L. Chen, LCARS: A location-content-aware recommender system, in *Proc. 19th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Chicago, IL, USA, 2013, pp. 221–229.

[10] S. S. Feng, X. T. Li, Y. F. Zeng, G. Cong, Y. M. Chee, and Q. Yuan, Personalized ranking metric embedding for next new POI recommendation, in *Proc. 24th Int. Conf. Artificial Intelligence*, Buenos Aires, Argentina, 2015, pp. 2069–2075.

[11] B. Chang, Y. Park, D. Park, S. Kim, and J. Kang, Content-aware hierarchical point-of-interest embedding model for successive POI recommendation, in *Proc. 27th Int. Joint Conf. Artificial Intelligence*, Stockholm, Sweden, pp. 3301–3307.

[12] M. Pagliardini, P. Gupta, and M. Jaggi, Unsupervised learning of sentence embeddings using compositional n-gram features, in *Proc. 2018 Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, New Orleans, LA, USA, 2018, pp. 528–540.

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

[14] A. Graves, A. Mohamed, and G. E. Hinton, Speech recognition with deep recurrent neural networks, in *Proc. 2013 IEEE Int. Conf. Acoustics, Speech and Signal Processing*, Vancouver, Canada, 2013, pp. 6645–6649.

[15] Y. Chen, J. Y. Hu, Y. Xiao, X. Li, and P. Hui, Understanding the user behavior of foursquare: A data-driven study on a global scale, *IEEE Trans. Comput. Soc. Syst.*, vol. 7, no. 4, pp. 1019–1032, 2020.

[16] C. J. Hutto and E. Gilbert, VADER: A parsimonious rule-based model for sentiment analysis of social media text, in *Proc. 8th Int. Conf. Weblogs and Social Media*, Ann Arbor, MI, USA, 2014.

[17] S. H. Lin, R. Xie, Q. Xie, H. Zhao, and Y. Chen, Understanding user activity patterns of the swarm app: A data-driven study, in *Proc. 2017 ACM Int. Joint Conf. Pervasive and Ubiquitous Computing and Proc. 2017 ACM Int. Symp. Wearable Computers*, Maui, HI, USA, 2017, pp. 125–128.

[18] Y. Chen, J. Y. Hu, H. Zhao, Y. Xiao, and P. Hui, Measurement and analysis of the swarm social network with tens of millions of nodes, *IEEE Access*, vol. 6, pp. 4547–4559, 2018.

[19] Y. Z. Wang, N. J. Yuan, D. F. Lian, L. L. Xu, X. Xie, E. H. Chen, and Y. Rui, Regularity and conformity: Location prediction using heterogeneous mobility data, in *Proc. 21th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Sydney, Australia, 2015, pp. 1275–1284.

[20] E. Tang and K. Sangani, Neighborhood and price prediction for San Francisco airbnb listings, http://cs229.stanford.edu/proj2015/236_report.pdf, 2015.

[21] Y. Suhara, Y. Z. Xu, and A. S. Pentland, DeepMood: Forecasting depressed mood based on self-reported histories via recurrent neural networks, in *Proc. 26th Int. Conf. World Wide Web*, Perth, Australia, 2017, pp. 715–724.

[22] Q. Y. Gong, Y. Chen, X. L. He, Z. Zhuang, T. Y. Wang, H. Huang, X. Wang, and X. M. Fu, DeepScan: Exploiting deep learning for malicious account detection in location-based social networks, *IEEE Communications Magazine*, vol. 56, no. 11, pp. 21–27, 2018.

[23] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, A novel connectionist system for unconstrained handwriting recognition, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 855–868, 2009.

[24] K. Cho, B. van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in *Proc. 2014 Conf. Empirical Methods in Natural Language Processing*, Doha, Qatar, 2014, pp. 1724–1734.

[25] D. Neil, M. Pfeiffer, and S. C. Liu, Phased LSTM: Accelerating recurrent network training for long or event-based sequences, in *Proc. 30th Conf. Neural Information Processing Systems*, Barcelona, Spain, 2016, pp. 3882–3890.

[26] J. Koutník, K. Greff, F. J. Gomez, and J. Schmidhuber, A clockwork RNN, in *Proc. 31st Int. Conf. Machine Learning*, Beijing, China, 2014, pp. II-1863–II-1871.

[27] D. Bahdanau, K. Cho, and Y. Bengio, Neural machine translation by jointly learning to align and translate, in *Proc. 3rd Int. Conf. Learning Representations*, San Diego, CA, USA, 2015.

[28] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, Attention-based models for speech recognition, presented at Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, Canada, 2015, pp. 577–585.

[29] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in *Proc. 32nd Int. Conf. Machine Learning*, Lille, France, 2015, pp. 2048–2057.

[30] L. Breiman, Random forests, *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[31] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA, USA: Morgan Kaufmann, 1993.

[32] T. Q. Chen and C. Guestrin, Xgboost: A scalable tree boosting system, in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, San Francisco, CA, USA, 2016, pp. 785–794.

[33] G. L. Ke, Q. Meng, T. Finley, T. F. Wang, W. Chen, W. D. Ma, Q. W. Ye, and T. Y. Liu, LightGBM: A highly efficient gradient boosting decision tree, in *Proc. 31st Int. Conf. Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 3149–3157.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[35] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[36] R. Yang, R. Zhao, and D. Wang, Successive point-of-interest recommendation in intelligent business area, in *Proc. 2015 Int. Conf. Education Technology, Management and Humanities Science (ETMHS 2015)*, Xi'an, China, 2015, pp. 958–961.

[37] S. M. Lundberg and S. I. Lee, A unified approach to interpreting model predictions, in *Proc. 31st Int. Conf. Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 4768–4777.

[38] R. T. Rust and R. L. Oliver, *Service Quality: New Directions in Theory and Practice*. Thousand Oaks, CA, USA: Sage Publications, 1993.

[39] A. Ghobadian, S. Speller, and M. Jones, Service quality: Concepts and models, *International Journal of Quality & Reliability Management*, vol. 11, no. 9, pp. 43–66, 1994.

[40] G. S. Sureshchandar, C. Rajendran, and R. N. Anantharaman, The relationship between service quality and customer satisfaction–A factor specific approach, *Journal of Services Marketing*, vol. 16, no. 4, pp. 363–379, 2002.

[41] D. Lee, W. Hyun, J. Ryu, W. J. Lee, W. Rhee, and B. Suh, An analysis of social features associated with room sales of airbnb, in *Proc. 18th ACM Conf. Companion on Computer Supported Cooperative Work & Social Computing*, Vancouver, Canada, 2015, pp. 219–222.

[42] M. Grbovic and H. B. Cheng, Real-time personalization using embeddings for search ranking at airbnb, in *Proc. 24th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, London, UK, 2018, pp. 311–320.

[43] Q. Zhou, Y. Chen, C. H. Ma, F. Li, Y. Xiao, X. Wang, and X. M. Fu, Measurement and analysis of the reviews in airbnb, in *Proc. 17th Int. IFIP TC6 Networking Conf.*, Zurich, Switzerland, 2018, pp. 82–90.

[44] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, Factorizing personalized Markov chains for next-basket recommendation, in *Proc. 19th Int. Conf. World Wide Web*, Raleigh, NC, USA, 2010, pp. 811–820.

[45] R. N. He and J. McAuley, Fusing similarity models with Markov chains for sparse sequential recommendation, presented at IEEE 16th Int. Conf. Data Mining, Barcelona, Spain, 2016, pp. 191–200.

[46] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, Session-based recommendations with recurrent neural networks, in *Proc. 4th Int. Conf. Learning Representations*, San Juan, Puerto Rico, 2016.

[47] B. Hidasi, M. Quadrana, A. Karatzoglou, and D. Tikk, Parallel recurrent neural network architectures for feature-rich session-based recommendations, in *Proc. 10th ACM Conf. Recommender Systems*, Boston, MA, USA, 2016, pp. 241–248.

[48] J. Li, P. J. Ren, Z. M. Chen, Z. C. Ren, T. Lian, and J. Ma, Neural attentive session-based recommendation, in *Proc. 2017 ACM on Conf. Information and Knowledge Management*, Singapore, 2017, pp. 1419–1428.

[49] X. Chen, H. T. Xu, Y. F. Zhang, J. X. Tang, Y. X. Cao, Z. Qin, and H. Y. Zha, Sequential recommendation with user memory networks, in *Proc. 11th ACM Int. Conf. Web Search and Data Mining*, Marina Del Rey, CA, USA, 2018, pp. 108–116.

[50] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, Personalizing session-based recommendations with hierarchical recurrent neural networks, in *Proc. 11th ACM Conf. Recommender Systems*, Como, Italy, 2017, pp. 130–137.

[51] F. Y. Lv, T. W. Jin, C. L. Yu, F. Sun, Q. Lin, K. P. Yang, and W. Ng, SDM: Sequential deep matching model for online large-scale recommender system, in *Proc. 28th ACM Int. Conf. Information and Knowledge Management*, Beijing, China, 2019, pp. 2635–2643.

[52] C. Ma, P. Kang, and X. Liu, Hierarchical gating networks for sequential recommendation, in *Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, Anchorage, AK, USA, 2019, pp. 825–833.

[53] J. Huang, Z. C. Ren, W. X. Zhao, G. L. He, J. R. Wen, and D. X. Dong, Taxonomy-aware multi-hop reasoning networks for sequential recommendation, in *Proc. 12th ACM Int. Conf. Web Search and Data Mining*, Melbourne, Australia, 2019, pp. 573–581.

[54] J. Huang, W. X. Zhao, H. J. Dou, J. R. Wen, and E. Y. Chang, Improving sequential recommendation with knowledge-enhanced memory networks, in *Proc. 41st Int. ACM SIGIR Conf. Research & Development in Information Retrieval*, Ann Arbor, MI, USA, 2018, pp. 505–514.

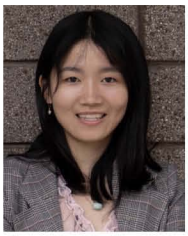
[55] S. Q. Wang, C. P. Li, K. K. Zhao, and H. Chen, Context-aware recommendations with random partition factorization machines, *Data Science and Engineering*, vol. 2, no. 2, pp. 125–135, 2017.

Yihan Ma received the bachelor and master degrees from Fudan University in 2018 and 2021, respectively. She was a student research assistant at the Mobile Systems and Networking (MSN) group from 2017 to 2021. Her research interests include online social networks and data mining.

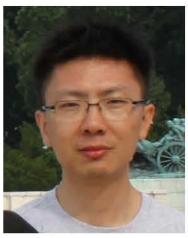
Yang Chen is an associate professor at the School of Computer Science, Fudan University, and leads the Mobile Systems and Networking (MSN) group. From April 2011 to September 2014, he was a postdoctoral associate at the Department of Computer Science, Duke University, USA, where he served as the senior personnel in the NSF MobilityFirst project. From September 2009 to April 2011, he was a research associate and the deputy head of Computer Networks Group, Institute of Computer Science, University of Göttingen, Germany. He received the BS and PhD degrees from Tsinghua University in 2004 and 2009, respectively. He visited Stanford University (in 2007) and Microsoft Research Asia (2006–2008) as a visiting student. He was a Nokia visiting professor at Aalto University in 2019. His research interests include online social networks, Internet architecture, and mobile computing. He serves as an associate editor-in-chief of *Journal of Social Computing*, an associate editor of *IEEE Access*, and an editorial board member of *Transactions on Emerging Telecommunications Technologies* (ETT). He served as a OC/TPC member for many international conferences, including SOSP, SIGCOMM, WWW, IJCAI, AAAI, ECAI, DASFAA, IWQoS, ICCCN, GLOBECOM, and ICC. He is a senior member of the IEEE.

Xin Wang is a professor at School of Computer Science, Fudan University. He received the BS degree in information theory and the MS degree in communication and electronic systems from Xidian University in 1994 and 1997, respectively. He received the PhD degree in computer science from Shizuoka University, Japan in 2002. His research interests include quality of network service, next-generation network architecture, mobile Internet, and network coding.

Hua Sun is a master student at Tandon School of Engineering, New York University. He received the BS degree from Fudan University in 2019. From 2016 to 2019, he worked at the Mobile Systems and Networking (MSN) group as a research assistant. From May to August, 2020, he worked as a software engineer intern at Google. Currently, he works at High Speed Network Lab, New York University. His research interests include machine learning, data mining, and networking.



Jiayun Zhang received the BS degree from Fudan University, in 2020. She is currently pursuing the PhD degree at the Department of Computer Science and Engineering, University of California, San Diego, CA, USA. Her research interests include data mining, social networks, and internet of things.



Yang Xu is the Yaoshihua chair professor at the School of Computer Science, Fudan University. Prior to joining Fudan University, he was a faculty member at the Department of Electrical and Computer Engineering, New York University Tandon School of Engineering. He received the PhD degree in computer science and technology from Tsinghua University in 2007 and the BEng degree from Beijing University of Posts and Telecommunications in 2001. His research interests include software-defined networks, data center networks, distributed machine learning, edge computing, network function virtualization, and network security. He has published about 80 journal and conference papers and holds more than 10 U.S. and international granted patents on various aspects of networking and computing. He served as a TPC member for many international conferences, an editor for *Journal of Network and Computer Applications* (Elsevier), a guest editor for *IEEE Journal on Selected Areas in Communications-Special Series on Network Softwarization & Enablers*, and *Wiley Security and Communication Networks Journal-Special Issue on Network Security and Management in SDN*.

Pan Hui is the Nokia chair in data science and a full professor in computer science at the University of Helsinki. He has also been a faculty member at the Department of Computer Science and Engineering, Hong Kong University of Science and Technology since 2013 and an adjunct professor of social computing and networking at Aalto University Finland since 2012. He received the PhD degree from Computer Laboratory, University of Cambridge, and the MPhil and BEng degrees both from University of Hong Kong in 2007, 2004, and 2002, respectively. He has published over 300 research papers with over 19 000 citations and has around 30 granted/filed European patents. He was an associate editor for *IEEE Transactions on Mobile Computing* and *IEEE Transactions on Cloud Computing*. He is an IEEE Fellow, an ACM distinguished scientist, an International Fellow of the Royal Academy of Engineering (FREng), and a member of the Academia Europaea (MAE).