DeepLoc: A Location Preference Prediction System for Online Lodging Platforms

Yihan Ma[1,2], Hua Sun[1,2], Yang Chen[1,2 (*)], Jiayun Zhang[1,2], Yang Xu[1], Xin Wang[1,2], and Pan Hui[3,4]

[1] School of Computer Science, Fudan University, China
[2] Shanghai Key Lab of Intelligent Information Processing, Fudan University, China
[3] Department of Computer Science, University of Helsinki, Finland
[4] Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong
{mayh18,hsun15,chenyang,jiayunzhang15,xuy,xinw}@fudan.edu.cn
panhui@cs.helsinki.fi

Abstract. Online lodging platforms have become very popular around the world. To make a booking, a user normally needs to select a city first, then browses among prospective options. To improve the user experience, understanding the location preference of a user's booking behavior will be useful. In this paper, we propose DeepLoc, a location preference prediction system, adopting deep learning technologies to predict the location preference of a user's next booking, based on both the descriptive features and the user's historical booking records. Using the real data collected from Airbnb, we can see that DeepLoc can achieve an F1-score of 0.885 for booking apartments in the city of London.

Keywords: Online lodging systems · Location preference · Prediction · Deep learning

1 Introduction

Traveling has been an important part in people's daily life. A satisfactory accommodation in an unfamiliar city can greatly promote people's travel experience. There are a number of online lodging services, such as Airbnb [1] [2] [3], Booking.com [4] and Homestay[1], offering both traditional hotels and residential accommodations for visitors.

When booking accommodations online, users usually have a specific destination, such as London, Paris and New York. In addition, online lodging services such as Airbnb, Booking.com and Homestay offer a set of filters. Users can choose accommodations which meet their requirements by setting these filters. In particular, many users have location preferences for accommodations. When traveling to a new city, some people like to live in prosperous places, for example, the downtown of the city. These places are normally well connected, and have

[1] https://www.homestay.com/, accessed on May 1, 2019.

attractions for tourism and shopping. On the contrary, some people may tend to live in less prosperous places, where they can get rid of the traffic jam and the noisy environment of the downtown. However, users cannot select accommodations based on their location preferences directly on most online lodging platforms.

Given the importance of a user's location preference, we aim to understand a user's location preference and recommend her accommodations accordingly for her next trip. In this paper, we propose a location preference prediction system named DeepLoc. In our system, we utilize users' history booking records to predict the desired location of users' next booking in a given city. Our methodology combines the long short-term memory (LSTM) neural networks [5] [6] and some conventional supervised machine learning algorithms to make use of both dynamic and descriptive characteristics of each user. LSTM is well-known for its ability to process time sequence data and analyze the dynamic patterns.

In this paper, we use Airbnb as a case study. Founded in 2008, Airbnb has quickly grown to be one of the world's most popular online lodging platforms with about 200 million registered users. It has 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[2]. In particular, we choose London as a sample city for our study. London is one of the world's leading tourism destinations, which attracted over 20 million international visitors in 2018 and it has over 80 thousand accommodations in Airbnb by Dec. 2018. By dividing the city of London into central London and non-central London, our DeepLoc system can predict which part a traveler will choose to live. The results show that our system can achieve an F1-score of 0.885, indicating that DeepLoc performs very well when predicting a user's location preference. The contributions of this paper are summarized as follows.

First, we formulate the location preference prediction problem in online lodging platforms and design a system which can obtain the location preference for booking and give users better recommendations.

In addition, we propose a location preference prediction system named DeepLoc to predict the desired location of a user's next booking in a selected city. Our system combines the advantages of LSTM and traditional supervised machine learning algorithms to deal with both dynamic and descriptive features of the dataset.

Last but not least, we evaluate our system using a real dataset collected from Airbnb. The results show that our system can predict the location of user's next booking in London with an F1-score of 0.885.

2 Data Collection and Feature Extraction

In this section, we give an overall introduction of our datasets. We delineate the preprocessing part of our datasets and describe the features we used in the experimental part.

[2] https://press.airbnb.com/about-us/, accessed on May 1, 2019.

2.1 Datasets and Preprocessing

In this paper, we want to design a system which can use the history booking records of a user to predict which part she will live in the next trip in a given city. Also, every individual is special, the personal information of the user might be useful for us to do the prediction task. Thus, our datasets consist of 2 parts, InsideAirbnb dataset from which we obtain the detailed information of the history booking record and user profile dataset where we can get the personal information of each user.

InsideAirbnb Dataset. First, we obtain the accommodation data and review data of all Airbnb accommodations in 84 cities from InsideAirbnb[3]. InsideAirbnb dataset is widely used in studies about Airbnb [2] [3]. For each city, we get two .csv files which store the accommodation data and review data of the city, respectively. The accommodation data includes price, longitude, latitude, type of Airbnb accommodations, amenities of accommodations, demographical information of hosts and so on. The review data is the collection of reviews of all the accommodations in the city. Each line of review data represents an actual visit including the ID of the accommodation, time stamp of the review, ID and name of the guest who lived in this accommodation and wrote this review. The detailed information of InsideAirbnb dataset is summarized in Table 1.

Table 1. Description of the InsideAirbnb Dataset

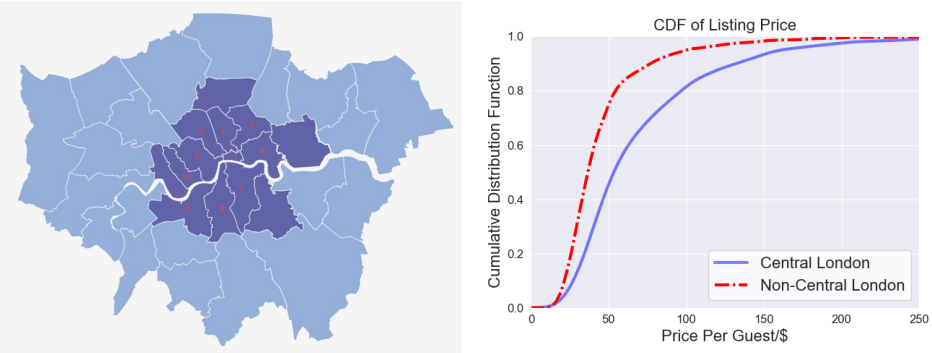
Categories	Features	Description
Accommodation data	Price	The price of accommodation
	Location	The longitude and latitude of accommodation
	Type	The type of accommodation, including villa, apartment...
	Amenities	The amenities of accommodation, including hair dryer, kitchen...
Review data	ID	The ID of reviewed recommendation
	Time	The time when the review is given
	Guest_ID	The ID of the guest
	Guest_name	The name of the guest
	Comment	The detailed comment of the review

Since we want to utilize the history records of a particular user to predict the desired location of her next booking, we extract all the reviews of the same user and build a user-review related database based on MongoDB, which is a cross-platform document-oriented database program. Unlike NoSQL, MongoDB uses JSON-like documents to store data. Because the lengths of history booking

[3] InsideAirbnb (http://insideairbnb.com/, accessed on May 1, 2019) is a website which offers open sourced dataset contains the detailed information of the accommodations and reviews in 84 cities in Airbnb.

records of different users are not the same, it is convenient to store the history booking records with MongoDB. Finally we get a database consisting of more than 20 million users. To make sure we can get enough information from previous booking records, we select 15,442 users who have at least 7 history records. As introduced in previous sections, users usually select city first when booking accommodations in online lodging platforms. We choose London as our target city at first, and get a dataset containing the detailed information of 2,045 users whose latest booking records are in London.

User Profile Dataset. Second, we build a crawler to get the profiles of selected users. 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, government ID and so on. Note that when crawling the profile of the user, some of them maybe not available[4]. Finally, we get 2,004 user profiles. All the profile data was crawled between 10 Mar. 2019 and 11 Mar. 2019.



(a) Division (b) Price Distribution

Fig. 1. The Division and Airbnb Accommodation Price Distribution of London

2.2 Obtaining Ground Truth

In this paper, our main target is to predict which part of a given city a user will choose for her next booking, so that we can obtain the preferences of users for location and improve the quality of users' booking experiences. To evaluate the performance of our system, we use the latest booking record as our ground truth. With our processed InsideAirbnb dataset, we get 2,045 selected users. In our DeepLoc system, we utilize the detailed information of 6 booking records

[4] If an account is deleted by the corresponding user or by the Airbnb platform, the profile page of this user will be unavailable

before the latest one as input data, and the output data is the location of the last booking record.

Since we want to predict whether a user will choose to live in prosperous downtown or quiet outskirts, we need to divide London into two different parts. In this paper, the division criterion is according to *London Plan*[5] which defines the *Central Activities Zone* as a set of 10 Boroughs. This area is described as "a unique cluster of vitally import activities". In the light of this description, we divide London into two parts, central London which includes the 10 boroughs and the other boroughs in Greater London, as shown in Fig. 1(a). Users whose latest booking records are located in central London are labeled as the first class. Users with latest booking record in non-central London are labeled as the second class. The ratio of the number of sample in first class to second class is 1,071:974.

To validate our classification criterion of London, we compare the booking price of accommodations in central London and non-central London in Airbnb. Fig. 1(b) shows the comparison of price distribution of accommodations in central London and non-central London. We can see from the figure that the price of accommodations in central London are higher than that in non-central London on average. In non-central places of London, the price of over 60 percent accommodations per guest per night is less than 50 dollars. As for accommodations in central London, only 30 percent of them have a price under 50 dollars per guest per night. Also, the 90th percentile personal price of all the accommodations in non-central London is under 150 dollars, while in central London, the 90th percentile personal price is over 200 dollars. This figure indicates that there are significant differences between the price of accommodations in central London and other places.

2.3 Feature Extraction

After generating the final dataset and confirming the ground truth of our prediction work, we extract a series of features to be used as input data of our system. There are 2 types of features in this paper.

Historical Booking Features. In our final InsideAirbnb user-review related dataset, we have 2,045 users who have at least 7 history records. As shown before, we utilize the 6 history each user before the latest one to predict which part she will live in London. And we use the location of the latest booking record as our ground truth. At first, we need to extract features of the previous 6 booking records.

For each record, the review data contains time, ID and city of this accommodation and the comments to it. First, we acquire the sentiment of each comment by VADER Sentiment [7]. VADER (Valence Aware Dictionary and sEntiment Reasoner) 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 is sentences and the results of VADER usually contain four items, i.e.

[5] https://www.london.gov.uk/what-we-do/planning/london-plan, accessed on May 1, 2019.

the ratio of positive words, negative words, neural words and a compound score. The compound score is a general measure of sentiment of a given sentence. Then we obtain the detailed information of accommodations through the accommodation csv file of each city. Note that in Airbnb, each accommodation has a fixed ID, so we can look up the detailed information of this accommodation by its ID. From the accommodation file, we can get some information of this accommodation itself, such as the price, longitude and latitude, property type[6], room type[7], accommodates[8], amenities, cleaning fee. Also, we can get some information about the host, including demographic characteristics, the response rate of the host, the total number of accommodations owned by the host, some verified information the host offered.

Also, for each history booking record, we can get the longitude and latitude of the corresponding accommodation. To enrich our feature set, we obtain the POI information within 5 miles of each accommodation via Google Places API[9]. Google Places API is a service that returns information about places. When sending HTTP requests with the longitude and latitude of accommodations to Google Places service, users can choose to return information of certain kinds of Points of Interest (POI), such as shopping malls and museums. It will return a JSON-formatted file which contains the information of all the required categories of POIs. Since Google Places API has very strict access speed restrictions, we choose to get the information of only 5 categories, including subway station, bus station, train station, airport and shopping mall.

Finally, we use the extracted historical booking features to formulate a Historical Matrix $H^{6\times 263}$, which stores the extracted features of 6 history booking records, where each booking record consists of a 263-dimensional feature set.

User Profile Features. The user profile features are extracted from our user profile dataset, which contains the city where the user lives in, the created time of Airbnb account, the total number of reviews from guests[10], the total number of reviews from hosts, the verified information and so on. The feature set of users whose profiles are no longer available are filled by -1.

Finally, we get a dataset including 2,045 samples. Each sample has a 1594-dimensional feature set consisting of information of 6 history booking and profile data of the user. Table 2 is the summarized feature set.

3 System Design

Online lodging platforms usually offer filters like price, amenities to help users choose accommodations, but the filters cannot help users to select accommodations based on their location preferences directly. Thus, we propose a location

[6] The types of Airbnb accommodations, including apartments, villas, tree houses.
[7] There are three types of rooms in Airbnb:Private room, Shared room and Entire home.
[8] It refers to the number of people that this accommodations can host at one time.
[9] https://developers.google.com/places/web-service/intro, accessed on May 1, 2019.
[10] The reviews user received from her guests.

Table 2. Feature set of each user

Category	Features
Historical Booking Features	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 Amenities of accommodation in each booking record Demographic information of host in each booking record Geographical information of accommodation in each booking record
User Profile Features	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

preference prediction system names DeepLoc to predict the location of next booking in a given city for users. DeepLoc is based on a model named DeepScan which was proposed by Gong et el. [8]. To utilize the history booking records, we need to involve an algorithm which is capable to process time series information. There have been a lot of techniques which can process time sequence data. LSTM networks [5] [6] have shown its power in recent studies. So we involve LSTM in our system to acquire the dynamic features of sequential booking records. In this section, we will first give an overall description of our system, and then specifically introduce its workflow.

Our online lodging recommendation system is mainly made of 2 parts: a bidirectional LSTM (BLSTM) module which processes time sequence data and a Decision Maker which utilizes combined conventional features and dynamic features as input data and outputs the classification results of each sample. The architecture of DeepLoc is shown in Fig. 2.

As shown in Fig. 2, we first input extracted features of booking records into a BLSTM layer to get the dynamic information of each user. The historical matrix represents the features of 6 history booking records. After putting the last output of BLSTM into a softmax layer, we can get 2 normalized probabilities. Then we concatenate the probability features, the historical booking features and the user profile features to get the final feature set. In the Decision Maker module, we train the classification model with the final feature set to get classification results of each user.

4 Implementation and Evaluation

In this section, we show the implementation details of our system, and evaluate its prediction performance with real data collected from Airbnb. In the next

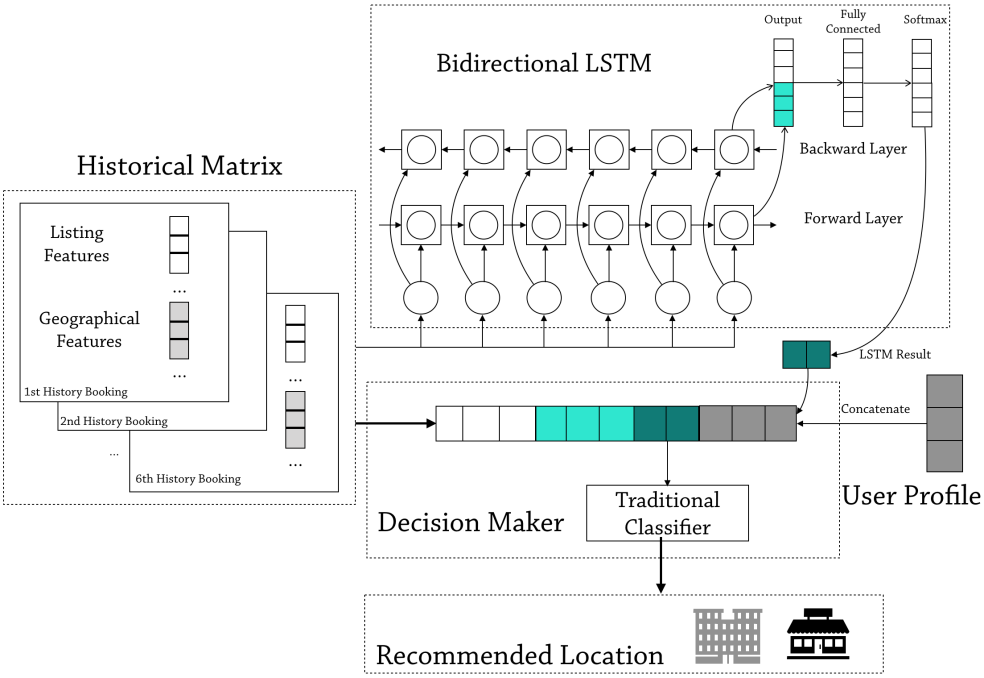


Fig. 2. The Architecture of DeepLoc

subsections, we will introduce the implementation of BLSTM, Decision Maker and the evaluation results in detail.

4.1 Bidirectional LSTM

LSTM units were proposed by Hochreiter et el. [5]. They are designed to process long-term dependency information and commonly used to overcome gradient vanishing problem. Compared to traditional recurrent unit, the main improvement of LSTM is the introduction of *forget gate*, which determines how much information should be kept from previous state. The output h_t for the forward pass of an LSTM unit is computed by the following equations:

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

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

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

$$c_t = f_t c_{t-1} + i_t \sigma_h\left(W_c x_t + U_c h_{t-1} + b_c\right) \tag{4}$$

$$h_t = o_t \sigma_h\left(c_t\right) \tag{5}$$

Where t and $t-1$ represent the information of the current step and the previous step, respectively. σ_g and σ_h are activation functions, representing a sigmoid

function and hyperbolic tangent function named $tanh()$, respectively. x_t is the input vector of each LSTM unit at time t. W, U 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 i_t is the input gate, which defines the degree to which the current input information is added to the memory cell. The output gate o_t of each LSTM unit at time t is computed to get the output memory. c_t is the cell state vector which drops part of the memory based on forget gate and adds new memory. 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.

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 model a lot. Hence, we introduce BLSTM network, 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. And finally, the output h_t of a BLSTM unit of the current step t is computed as follows:

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

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

$$loss = -y_i \log\left(\sigma_g\left(s_i\right)\right) - \left(1 - y_i\right) \log\left(1 - \sigma_g\left(s_i\right)\right) \tag{7}$$

Where y_i is the ground truth label and s_i represents the probability to be the first class. σ_g represents a sigmoid function.

In this paper, our LSTM model is constructed by Keras[11], a high-level neural networks API which is capable of running on top of some deep learning platforms, such as TensorFlow[12]. The LSTM we use is a fully connected Bidirectional LSTM. The learning rate is set as 0.01. We utilize a dropout layer to prevent over-fitting problem, and the dropout ratio is set to 0.1. Also, we use the Adam optimizer to optimize our model in the training process. In the learning process, the last output of the BLSTM unit 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 compute the probability to be samples in the first class and in the second class. Thus, the ultimate output of our BLSTM model is a two-dimensional probability vector.

4.2 Decision Maker

The Decision Maker is made of conventional supervised machine learning algorithms. In this paper, we choose several frequently-used machine learning algo-

[11] https://keras.io/, accessed on May 1, 2019.
[12] https://www.tensorflow.org/, accessed on May 1, 2019.

rithms, including RandomForest [9], Decision Tree [10], XGBoost [11], LightGBM [12] and Catboost [13]. In the training process, GridSearchCV [14] is applied to get the optimal parameters of each model automatically. Given a set of values of each parameter which needs 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.

4.3 Evaluation

In this work, we first randomly select 90 percent of samples in the first class and in the second class as training and validation set, and use the other 10 percent as test set. Note that we use the same training and validation set and test set in BLSTM and Decision Maker. The evaluation metrics we use for Decision Maker are F1-score and AUC [15]. F1-score is a combination and balance of precision and recall. Precision reflects the performance of a model when identifying a sample as positive one. And recall is introduced to measure the ratio of positive samples that have been correctly predicted. AUC is the area under the ROC (receiver operating characteristic) curve, which tells how much the model is capable of distinguishing between classes. All these metrics range from 0 to 1. The larger the value is, the better the performance is.

To better evaluate the performance of our system, we introduce some baselines. We utilize basic BLSTM and several representative supervised machine learning algorithms as baselines. For LSTM algorithm, the implementation details are the same as that in our online lodging recommendation system. The input feature of LSTM is the history booking features of each user. The algorithms we use for machine learning task coincide with our system. For each machine learning algorithm, the input feature includes the historical matrix and user profile features. The classification results of our system and all baselines are summarized in Table 3. In general, we can see that among all the experiments, our system with a Decision Maker of LightGBM performs the best, with an F1-score of 0.885 and an AUC-ROC score of 0.877. Also, when comparing the classification results of our system and machine learning algorithms which coincide with the Decision Maker in our system, a better result can be noticed except for XGBoost. It indicates that in general, our system can utilize the dynamic history information of users better. It can also be noticed that the result of LSTM is almost the worst. The results suggest that for a given city, our system can successfully predict which area a user will live in the city.

5 Related Work

Research on Airbnb On account of the rapid development of Airbnb, there have been some researches about the profiles of Airbnb users, the accommodation in Airbnb and the comparison of Airbnb accommodations and traditional hotels. Fradkin et el. [1] did a field experiment on reviews from Airbnb and found that

Table 3. The performance of our system and baselines

Algorithms	Precision	Recall	F1-score	AUC-ROC
DeepLoc (RF)	0.862	0.859	0.858	0.857
DeepLoc (DT)	0.832	0.824	0.823	0.832
DeepLoc (XGBoost)	0.858	0.854	0.853	0.852
DeepLoc (LightGBM)	0.857	0.914	**0.885**	**0.877**
DeepLoc (Catboost)	0.854	0.849	0.863	0.862
RF	0.852	0.849	0.848	0.847
DT	0.820	0.820	0.819	0.819
XGBoost	0.870	0.868	0.868	0.848
LightGBM	0.870	0.868	0.868	0.867
Catboost	0.857	0.854	0.853	0.852
BLSTM	0.772	0.905	0.824	0.830

reviews were typically informative but negative experiences were under-reported. Ma et el. [2] studied the profile of Airbnb users and got a conclusion that Airbnb hosts who disclosed more information on the profile could gain more trust from guests. Lee analyzed the social features associated with accommodations and found the most significant features for room sale in Airbnb. Quattrone et el. [3] did an cross-ref analysis of Airbnb economy with Foursquare data, census data and hotel data in London. Also, Grbovic et el. [16] gave real-time recommendations for users in Airbnb based on their click data and search history. However, none of previous work studied the location preferences of users when booking in Airbnb. Zhou et al. [17] presented a comprehensive and evolutionary study of Airbnb, using the information of 43.8 million users.

Research on hotel recommendation system Zhang et el. [18] combined collaboration filtering (CF) with content-based (CBF) method to overcome sparsity issue in hotel recommendation. Chu et el. [19] utilized users' browsing information when reading hotel reviews on mobile devices to obtain users' preference to make personal recommendations. Sanchez-Vazquez et el. [20] tried to acquire features to capture the user's price sensitivity, and then constructed a recommendation system which is price sensitive. Most of these approaches were based on the search history or click history. As for our work, we use the history records to obtain the location preferences of users to give them better recommendations.

6 Conclusion and Future Work

In this paper, we study the users' location preferences in booking accommodations on online lodging platforms. To improve the user experience, we propose a deep learning-based location preference prediction system, called DeepLoc, for online lodging platforms. Our system combines BLSTM and traditional machine learning algorithms. It can utilize a user's fine-grained historical booking records

and descriptive characteristics. We implement our system with a real dataset collected from Airbnb using London as the target city. Our evaluation results show that DeepLoc can predict the location preference of a user's next booking in London with an F1-score of 0.885.

In the future, we will give more fine-grained prediction by dividing a given city into multiple parts. We will use the data of other cities to further validate its performance on Airbnb. Also, experiments with datasets from other online lodging platforms like Booking.com will be conducted to evaluate the compatibility of DeepLoc.

Acknowledgment

This work is sponsored by National Natural Science Foundation of China (No. 61602122, No. 71731004), the Research Grants Council of Hong Kong (No.16214817) and the 5GEAR project from the Academy of Finland. Yang Chen is the corresponding author.

References

[1] Fradkin, A., Grewal, E., Holtz, D., Pearson, M.: Bias and reciprocity in online reviews: Evidence from field experiments on airbnb. In: Proc. of EC (2015)
[2] Ma, X., Hancock, J.T., Mingjie, K.L., Naaman, M.: Self-disclosure and perceived trustworthiness of airbnb host profiles. In: Proc. of ACM CSCW (2017)
[3] Quattrone, G., Proserpio, D., Quercia, D., Capra, L., Musolesi, M.: Who benefits from the "sharing" economy of airbnb? In: Proc. of WWW (2016)
[4] Mellinas, J.P., María-Dolores, S.M.M., García, J.J.B.: Booking. com: The unexpected scoring system. Tourism Management **49**, 72–74 (2015)
[5] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997)
[6] Graves, A., Mohamed, A., Hinton, G.E.: Speech recognition with deep recurrent neural networks. In: Proc. of ICASSP (2013)
[7] Hutto, C.J., Gilbert, E.: VADER: A parsimonious rule-based model for sentiment analysis of social media text. In: Proc. of ICWSM (2014)
[8] Gong, Q., Chen, Y., He, X., Zhuang, Z., Wang, T., Huang, H., Wang, X., Fu, X.: Deepscan: Exploiting deep learning for malicious account detection in location-based social networks. IEEE Communications Magazine **56**(11), 21–27 (2018)
[9] Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (2001)
[10] Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
[11] Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proc of ACM SIGKDD (2016)

[12] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.: Lightgbm: A highly efficient gradient boosting decision tree. In: Proc. of NIPS (2017)
[13] Prokhorenkova, L.O., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: Catboost: unbiased boosting with categorical features. In: Proc. of NeurIPS (2018)
[14] Fabian, P., Gaël, V., Alexandre, G., Vincent, M., Bertrand, T., Olivier, G., Mathieu, B., Peter, P., Ron, W., Vincent, D., Jake, V., Alexandre, P., David, C., Matthieu, B., Matthieu, P., Edouard, D.: Scikit-learn: Machine learning in python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
[15] Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters **27**(8), 861–874 (2006)
[16] Grbovic, M., Cheng, H.: Real-time personalization using embeddings for search ranking at airbnb. In: Proc. of ACM SIGKDD (2018)
[17] Zhou, Q., Chen, Y., Ma, C., Li, F., Xiao, Y., Wang, X., Fu, X.: Measurement and analysis of the reviews in airbnb. In: Proc. of IFIP Networking (2018)
[18] Zhang, K., Wang, K., Wang, X., Jin, C., Zhou, A.: Hotel recommendation based on user preference analysis. In: Proc. of ICDE Workshops (2015)
[19] Lin, K., Lai, C., Chen, P., Hwang, S.: Personalized hotel recommendation using text mining and mobile browsing tracking. In: Proc. of IEEE SMC (2015)
[20] Raul, S., Jordan, S., Rodrygo, L.T.S.: Exploiting socio-economic models for lodging recommendation in the sharing economy. In: Proc. of ACM RecSys (2017)