Received April 7, 2019, accepted May 7, 2019, date of publication May 24, 2019, date of current version June 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2918210

Addressing the Cold-Start Problem in Personalized Flight Ticket Recommendation

QI GU[1,2], JIAN CAO[1], (Member, IEEE), YAFENG ZHAO[1], AND YUDONG TAN[3]
[1]Department of Computer Science and Engineering, School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
[2]Department of Computer Science, School of Information Science and Technology, Nantong University, Nantong 226019, China
[3]Ctrip.com International, Ltd., Shanghai 200335, China

Corresponding author: Jian Cao (cao-jian@sjtu.edu.cn)

This work was supported in part by the National Key Research and Development Plan under Grant 2018YFB1003800, and in part by the China National Science Foundation under Grant 61472253.

ABSTRACT With the rapid development of the tourist industry, an increasing number of passengers book flight tickets through online travel agencies. When searching for a flight ticket online, users are overwhelmed by the choice on offer. Even though flight ticket recommendation has been widely investigated, the current approaches are unable to recommend flight tickets that meet an individual's preference efficiently because of the severe cold-start problem. This paper provides strategies to address the cold-start problem for flight ticket recommendation. We conduct an exploratory study over the real-world flight ticket recommendation scenario and classify the cold-start problem of flight ticket recommendation into two categories, namely route cold-start and user cold-start. We propose methods based on route similarity and social relationships between passengers to improve user models. Finally, we map an enhanced user preference model and flight features to latent factor spaces to generate the recommendation results. The experimental results on a real-world data set demonstrate the effectiveness of the proposed approaches.

INDEX TERMS Flight ticket recommendation, cold-start problem, latent factor.

I. INTRODUCTION

With the popularity of ecommerce and the tourist industry, an increasing number of travel agencies (e.g., Ctrip,[1] Bookairfare,[2] AirFareExperts[3]) provide services on the Internet. For example, Ctrip, which is currently the largest online travel agency (OTA) in China, provides various travel services (e.g., accommodation reservation, transportation ticketing, packaged tours) online. A widely adopted travel service, online flight booking provides users with an efficient and convenient experience. Once a user inputs their travel information (e.g., *departure city*, *arrival city*, and *takeoff date*), the OTA presents flight ticket information from different airlines, which includes the suitable flight tickets as well as preliminary information (e.g., *takeoff time*, *airline*, *class*, and *price*) and the relevant service information (e.g., *punctuality rate*, *rescheduling and cancellation policy*). However, for passengers, making the better choice from the hundreds of candidate flight tickets available is a non-trivial task.

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

[1]www.ctrip.com
[2]www.bookairfare.com
[3]www.airfareexperts.com

In general, OTAs help passengers by providing sorting and filtering functions, e.g., sorting by *takeoff time* or *price*, filtering by *departure time* or *airline*. Booking a flight is a decision-making process which is associated with the passenger's preferences and travel motivations [1]. For example, price-sensitive passengers tend to choose flight tickets with lower prices and passengers travelling on business tend to choose flights in terms of arrival time and the airline for which he is a VIP passenger. Therefore, it is of great importance to provide personalized flight ticket recommendation.

Nowadays, recommender systems [2], [3] are becoming popular in many fields. In contrast with traditional products (e.g., books and movies), the features of flight tickets are dynamic. For example, the price of a flight ticket fluctuates from time to time [4]. Moreover, the features of available flight tickets are not equally distributed for different routes, which makes it difficult to directly apply traditional recommendation methods such as the collaborative filtering method [5], [6] to flight ticket recommendation.

The cold-start problem [7] is a significant challenge in recommendation. The problem becomes even worse in flight ticket recommendation since most people take flights only a few times each year [8]. As a matter of fact, there are no historical records for the majority of passengers on most routes due to the large number of routes.

To improve the effectiveness of recommendation systems, many efforts have been made to solve the cold-start problem [9]. For a new user or a user with very little historical information, a simple strategy is to recommend popular products. However, each flight has only about two hundred tickets, resulting in only a small difference between popular flights and unpopular flights. Another strategy is to recommend products based on the user's profile. Unfortunately, user profiles are not always available. Recently, cross-domain recommendation [10] has been applied to the cold-start problem, which leverages knowledge learned from other domains to the target domain. Since the attributes of products in different domains are heterogeneous, the knowledge that can be transferred is very limited [11].

In this paper, recommendation approaches are specifically designed for flight tickets. User models learned from historical orders are enhanced based on route similarity and social relationships between passengers in order to alleviate the cold-start problem. Moreover, we map the user's preferences and flight features to latent factor spaces to generate the recommendation results. The contributions of this paper are summarized as follows:

- We select the flight ticket features that influence users' purchasing decisions by analyzing real-world flight ticket orders. We model a user's preference by calculating the entropy of each flight ticket feature based on his historical orders.
- We provide strategies to address the cold-start problem for flight ticket recommendation and classify the problem into two categories, namely route cold-start and user cold-start. We propose methods based on route similarity and social relationships between passengers to enhance the user models.
- We employ a conversion matrix to map the enhanced user preferences and flight features to latent spaces to generate the recommendation results.
- We conduct extensive experiments on a real-world data set, showing the effectiveness of the proposed approaches.

The rest of this paper is organized as follows. Section 2 summarizes the related work. We describe the user model for flight ticket recommendation in Section 3. Section 4 describes how to enhance the user models to address the cold-start problem. Section 5 combines explicit and latent factors for flight ticket recommendation. Section 6 presents the experiment results. Finally, we conclude the paper in Section 7.

II. RELATED WORK

The GroupLens system [3] marked the birth of the recommender system, which recommends products according to user's preferences and the features of products. Typically, recommendation approaches are divided into collaborative filtering recommendation [12], [13], content-based recommendation [14] and hybrid filtering recommendation [15], [16]. Users' preferences are often learned from their behavior data, which can be divided into explicit feedback [17], [18] and implicit feedback [19], [20]. Explicit feedback can clearly indicate how much the user likes or dislikes an item, but requires a user's additional efforts. On the contrary, implicit feedback, such as when a user clicks on an item or adds an item into their shopping cart, is more abundant and does not impose an extra burden on the user. However, implicit feedback can only indicate positive attitudes. For the flight ticket recommendation problem, only implicit feedback is available [21].

In recent years, the flight ticket recommendation problem has been studied in both academia and industry. A personal travel assistant (PTA) [22] uses case-based reasoning (CBR) [23], [24] instead of learning user preferences to make recommendations, where a user's historical orders are stored as cases in the PTA. When a flight ticket needs to be recommended to a user, all the candidate tickets are compared with the stored case and the most similar ones are recommended. The main function of the flight recommender engine (FRE) [25], developed by Amadeus, is to recommend suitable flights to the user. FRE extracts 26 dimensions of flight attributes and uses weighted Euclidean distance to measure flight similarity. The system takes the user's departure, arrival city and time as the conditions, and selects flights with the minimum distance from the user's conditions as the recommendation results. Obviously, these systems do not model users' preferences sufficiently nor do they consider the heterogeneous nature of different routes.

The cold-start problem is a significant challenge for practical recommender systems since to learning reliable models relies on having a sufficient amount of data [26]. For a totally new user, the system can simply recommend the most popular items [27], which may not meet their personal requirements. When a user visits a system for the first time, a series of preference options are presented for selection, so as to analyze the user's preferences. For example, the Jester [28] joke recommendation system requires users to rate selected jokes in terms of how humorous they find them, but this requires the users' cooperation. In addition, according to the users' social information [29], users who are socially connected are assumed to have similar preferences. The hidden community groups with similar preferences can be clustered [30] to solve the cold-start problem. Although these general solutions to the cold-start problem are good references, the characteristics of flight ticket recommendation problems should be fully considered if we want to design efficient approaches to this problem.

III. USER MODELING FOR FLIGHT TICKET RECOMMENDATION

In this section, we formulate the passengers' preferences models based on their historical orders.

A. PREPROCESSING OF FLIGHT TICKET INFORMATION

Flight tickets include continuous features (e.g., *takeoff time*, *price*) and discrete features (e.g., *class*, *airline*). For ease of modeling, we discretize the continuous features. The discretization of departure time is intuitive and we divide the departure time into four sessions, namely, morning, afternoon, evening, and night sessions. The discretization of flight price is more complicated since different flights have very different prices. We use price index P_{KPI} to normalize the price information:

$$P_{KPI} = \frac{P_{full} - P_{cur}}{P_{full} - P_{low}}, \tag{1}$$

where P_{full} is the full price of the economy class ticket, which is the standard price set by the airline and the frequency of change is small (usually in years); P_{cur} is the current available price; P_{low} is the lowest price in the search results. For a ticket bought by a passenger, the larger the price index, the more price-sensitive the passenger is and vice versa. Furthermore, based on the domain knowledge, we divide the price index into several representative intervals.

Moreover, additional key features, such as *destination port*, *origin port*, *airline*, *class* are also parts of the user model. In general, each feature of a flight ticket can be represented as a vector, where each dimension represents one possible value (interval) of this feature. We use d_f to represent the user's preference for feature f.

$$d_f = [x_1, x_2, \ldots], \tag{2}$$

where x_i is the appearance ratio of the corresponding value in all of this user's orders and the sum of the values of all dimensions of a feature for the user is 1.

B. USER PREFERENCE MODELING

A flight ticket t is represented as a set of feature values.

$$t = [f_0, f_1, \ldots], \tag{3}$$

where each column vector f denotes a feature of the flight ticket, and only the dimension corresponding to the feature value of the ticket is set to 1, and the remaining dimensions are set to 0, e.g., $f = [0, \ldots, 1, \ldots, 0]$.

Suppose flight ticket t consists of $|t|$ features, the historical ticket information of user u can be summarized by:

$$D_u = [d_{f_0}, d_{f_1}, \ldots], \tag{4}$$

where each element of D_u describes user choices in a feature. Moreover, for a feature, some users may show strong preferences for some selections while others may have weak preferences. We introduce entropy $S(f)$ to measure a user's selective degree on feature f:

$$S(f) = E[-ln(P(f))] = -\sum_{i=1}^{|f|} P(x_i) lnP(x_i), \tag{5}$$

where x_i denotes the value of each dimension and $|f|$ is the number of dimensions. The more selective a user is, the smaller the entropy. When the entropy reaches 0, it indicates that for this feature, the user always has the fix selection. We normalize the entropy values of different features and regard them as the weight of each feature:

$$W(f) = \frac{ln|f| - S(f)}{\sum_{f \in F}(ln|f| - S(f))}, \tag{6}$$

where F is the feature set of the flight ticket and $W(f)$ is the weight value of the target user on feature f. The user's preference on each feature constitute a complete user model, which is shown in Algorithm 1.

Algorithm 1 User Model Construction

```
Input: $\mathcal{O}$: historical orders of the route,
       $F$: the set of ticket features.
Output: User ticket choice information model $D$.
 1: Function ModelConstruction{O,F}
 2: $D \leftarrow \phi$;
 3: for $t \in O$ do
 4:     for $f \in F$ do
 5:         $D[f][t_f] += 1$;
 6:     end for
 7: end for
 8: $W \leftarrow \phi$;
 9: for $f \in F$ do
10:     $W[f] \leftarrow \frac{\ln|f| - S(f)}{\sum_{f \in F}(\ln|f| - S(f))}$;
11: end for
12: return $D, W$.
13: End Function
```

The user's model (including ticket choice information and feature weight) can be calculated offline.

C. FLIGHT TICKET RECOMMENDATION BASED ON USER MODELS

The flight ticket recommendation is equivalent to ranking all the candidate flight tickets according to the matching score between each ticket and the user's preference. Therefore, the calculation of the matching score is as follows:

$$Rating_t = \sum_{f \in F} W(f) \times t_f^T \times d_f, \tag{7}$$

where $W(f)$ is the weight of each feature f, t_f is the value of the ticket of this feature, and d_f is the choice distribution vector of this feature of the user. We rank the candidates in the search results according to the matching score and get the personalized recommendation results for the user. The recommendation algorithm is shown in Algorithm 2.

Suppose $|C|$ is the number of searched flights and $|O|$ represents the number of a user's historical orders, thus, the algorithm complexity is $O(|O| * |C| * \log |C|)$. Of these, $O(|C| * \log |C|)$ is the time complexity of calculating the score and sorting. In practice, as the user's model can be computed offline, the time complexity decreases to $O(|C| * \log |C|)$, which is only associated with the number of candidate flight tickets.

Algorithm 2 Flight Ticket Recommendation Algorithm

```
Input: $\mathcal{O}$: historical orders of the route,
       $F$: the feature set of ticket,
       $C$: the candidate ticket list.
Output: A list of ranked tickets $R$.
 1: $D, W \leftarrow$ ModelConstruction(O,F);
 2: $R \leftarrow \phi$;
 3: for $t \in C$ do
 4:     $G_t \leftarrow 0$;
 5:     for $f \in F$ do
 6:         $G_t += \frac{D[f][t_f]}{Sum(D[f])} \times W[f]$;
 7:     end for
 8:     Append $G_t$ to $R$;
 9: end for
10: Sort $R$ by descending;
11: return $R$.
```

IV. USER MODEL ENHANCEMENT TO ADDRESS THE COLD-START PROBLEM

In flight ticket recommendation, the cold-start problem can be further classified into two categories, i.e., the route cold-start problem and the user cold-start problem.

A. SOLUTIONS TO THE ROUTE COLD-START PROBLEM

Obviously, there are significant differences in the features of flight tickets on different routes. Moreover, each of the flight tickets' features may play different roles in the decision making process of flight ticket choices. In order to show this phenomenon, we analyze the correlations between the user model and the feature distribution of the route. The data set consists of real ticket orders of four routes collected from January 2013 to July 2015 (further details are given in the experiment section).

We use the Pearson coefficient to calculate their correlations:

$$r(X,Y) = \frac{\sum(X-(\bar{X}))(Y-(\bar{Y}))}{\sqrt{\sum(X_i-(\bar{X}))^2}\sqrt{\sum(Y_i-(\bar{Y}))^2}} = \frac{n\sum(X_iY_i)-\sum X * \sum Y}{\sqrt{n\sum X_i^2-(\sum X_i)^2}\sqrt{n\sum Y_i^2-(\sum Y_i)^2}}, \quad (8)$$

where X and Y are two vectors with the same number of dimensions, and the Pearson coefficient is between $[-1, 1]$. When X and Y are positively correlated, the increase and decrease of X and Y are the same. When X and Y are negatively correlated, the increase and decrease of X is opposite to that of Y, and the Pearson coefficient ranges from $[1, 0]$. The stronger the correlation between X and Y, the closer the absolute value of the correlation coefficient is to 1.

We establish a user feature distribution model on route *BJS-XIY* and calculate the average correlation coefficient of the user model with four routes in three dimensions: *Price* (Pr), *Airline* (Air), and *Aircraft type* (Type). It can be seen in Fig. 1 that the user model on route *BJS-XIY* has the highest correlation with the feature distribution of this route. Among the three dimensions, the one with the smallest difference in correlation degree is *Pr*, because the proportion of economy class and low price ticket users is the highest on any route, the most significance difference in correlation degree is *Air*, which may be related to the operational strategy of different airlines on different routes. The user model is negatively correlated with the feature distribution on route *BJS-SZX*.

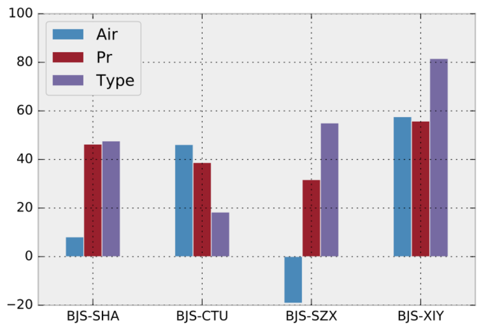

FIGURE 1. Correlation analysis between user model on BJS-XIY and feature distribution over all routes.

The heterogeneity of different routes means that user models learned from one route do not fit to other routes. The route cold-start problem means that the user does not have enough historical orders to build a route-specific user model on the target route that needs ticket recommendation.

1) SIMILARITY MEASUREMENT BETWEEN ROUTES

To solve the route cold-start problem, we need to transfer the models learned from other routes. But since flight tickets of different routes have different feature distributions, we should select the routes carefully.

We use a similarity function to quantify the flight ticket differences between two routes r_a and r_b, which is defined as:

$$Sim(r_a, r_b) = \sum_{f \in F} W(f) \times \sigma(f^{r_a}, f^{r_b}), \quad (9)$$

where $W(f)$ represents the weight of each feature f, f^a and f^b denotes a feature on route r_a and r_b respectively, and $\sigma(f^{r_a}, f^{r_b})$ is the local feature similarity measurement. Each route is represented by a multi-dimensional vector. The key features of the route are consistent with the features of the flight tickets, including *price*, *takeoff time*, *aircraft type*, *class*, *airline*, and *rescheduling and cancellation policy*. However, unlike user models, the value of each feature of the route is based on the orders from all users. We use Euclidean distance to measure the distribution difference between routes over a feature:

$$Dist(f^{r_a}, f^{r_b}) = \sqrt{\sum_{i=1}^{|f|}(x_i^{r_a} - x_i^{r_b})^2}. \quad (10)$$

Consequently, the similarity measurement is defined as:

$$Sim(f^{r_a}, f^{r_b}) = \frac{1}{Dist(f^{r_a}, f^{r_b})}. \quad (11)$$

In addition, even if two routes have similar flight ticket distributions, the buy decisions may be affected by very complex latent relationships between features or additional factors that have not yet been modeled. Therefore, we also consider users' choice similarity between these two routes. In order to measure user behavior similarity, we uniformly divide the normalized entropy value into multiple intervals and treat each interval as one dimension. Similar to the calculation of the user model, the overall user model is defined as the normalized entropy of the features and is denoted as $\overline{S}(f^{r_a})$. The weight of each feature is defined as the sum of the normalized entropy of the feature on the two routes.

$$W(f) = \frac{\overline{S}(f^{r_a}) + \overline{S}(f^{r_b})}{2}. \tag{12}$$

Finally, the differences between the routes from two aspects, i.e., feature distribution similarity and user behavior similarity is as follows:

$$\sigma(f^{r_a}, f^{r_b}) = \alpha Sim(f^{r_a}, f^{r_b}) + (1-\alpha) Sim(S(f^{r_a}), S(f^{r_b})), \tag{13}$$

where α is the tuning parameter between the two terms.

2) USER MODEL TRANSFER FROM SIMILAR ROUTES

In order to make use of the data from similar routes to enhance the user models, the simplest method is to select the route with the smallest distance from the target route. However, there are still two issues with this method: (1) this user is a cold-start user on some of the similar routes and his user models on these routes are not reliable; (2) even though the number of orders on the similar routes is sufficient, simply copying these user models to the target route is not reasonable.

To solve the first problem, the number of orders of this user on the route is regarded as an incentive factor:

$$\begin{aligned} ReSim(r_a, r_b) &= \log |O_b| \times Sim(r_a, r_b) \\ &= \log |O_b| \times \sum_{f \in F} W(f) \times \sigma(f^{r_a}, f^{r_b}), \end{aligned} \tag{14}$$

where r_a represents the target route, r_b represents a similar route, and O_b represents the number of orders on a similar route. We rank all routes according to their weighted similarities and select the one that is most similar for model enhancement.

To solve the second problem, the original user model is modified as:

$$D'_u = [\frac{d_{f_0}}{f_0^{r_b}}, \frac{d_{f_1}}{f_1^{r_b}}, \ldots], \tag{15}$$

where d_f represents the user's original model on feature f, and f^{r_b} is the overall distribution of all users on this feature on route r_b. By dividing the user feature distribution by the route feature distribution, a route-free user model can be obtained. For example, suppose the original distributions of the user of a feature is [20%, 30%, 30%, 20%], and the overall distribution of the feature on this route is [10%, 40%, 20%, 30%]. Then, this user's route-free value distribution of the feature is [2, 0.75, 1.5, 0.67] and it can be further normalized.

After obtaining the user's model from the most similar route, we integrate this model with the user's model obtained directly from the target route. If the user has no historical order on the target route, the model from the most similar route is used. Otherwise, these two models are linearly combined in terms of their order numbers. Algorithm 3 shows the construction process of the route transfer enhancement model.

Algorithm 3 User Model Transfer from Similar Routes

```
Input: $O$: the historical orders of all the routes,
       $F$: the set of ticket features,
       $r_a$: the recommended route,
       $M$: the overall feature distribution over different routes.
Output: The Enhanced User Model $D_{mix}$.
1: $P \leftarrow routePartition(O, r_a)$;
2: for $r \in P$ do
3:     $D[r] \leftarrow extractDistribution(O_r, F)$;
4:     $W[r] \leftarrow getWeight(O_r, F)$;
5: end for
6: $r_b \leftarrow getOptimalRoute(P)$
7: $D'[r_b] \leftarrow D[r_b]/M[r_b]$
8: $D_{mix} = Normalize(|O_{r_a}| \times D[r_a] + |O_{r_b}| \times D'[r_b])$
9: return $D_{mix}$.
```

B. USER COLD-START RECOMMENDATION

The user cold-start problem means that the number of orders of a user on all routes is too small to learn a reliable user model. Therefore, we cannot use the user models from any other routes to solve the user cold-start problem. A simple strategy is to make use of orders to build a general user model. However, this suffers from the problem of the heterogeneity of different routes. In order to provide a better solution, we use the social relationships between users to enhance the user model.

1) SOCIAL RELATIONSHIPS BETWEEN USERS

In real life, social relationships are used to describe the social connections between people, such as family, colleagues, friends, etc. In the flight ticket recommendation scenario, we are unable to ascertain the specific relationship between users in reality, and it is difficult to measure the degree of closeness of social relationships between users. However, we can mine the relationship between users through the ticket order data set. Although this indicates the connection of users in the flight ticket domain and does not reflect the actual social relationship between users, this relationship can be used to solve the user cold-start problem to a certain extent.

a: ORDER-SHARED RELATIONSHIPS

A ticket order may include multiple tickets for a group of passengers and their identity information is included in this order.

The relationships between the passengers who have appeared in an order is called an order-shared relationship. Although we do not know their actual social relationships, it is believed they may share similar preferences regarding flight ticket choices.

b: FLIGHT-SHARED RELATIONSHIPS

In addition to the order-shared relationship, if two users have taken the same flight more than once, we can assume that these two users have a flight-shared relationship. Broadly speaking, users who have a flight-shared relationship may share similar preferences.

After defining the social relationships among users, we define the measurement criteria to calculate the degree of social closeness. When a cold-start user has social relationships with multiple users, we use this metric to select the closest user for his user model enhancement. Suppose u_1 is the cold-start user whose user model should be improved, and u_2 is a candidate user who has a reliable user model. Their closeness is measured by:

$$Rela(u_1, u_2) = \log |O_{u_2}| \frac{|P(u_1, u_2)| + |Q(u_1, u_2)|}{\sqrt{|Z(u_1)| \times |Z(u_2)|}}, \tag{16}$$

where $|O_{u_2}|$ is the number of orders of user u_2, $|P(u_1, u_2)|$ represents the times that two users appear in the same order, $|Q(u_1, u_2)|$ represents the number of common flight-shared passengers of user u_1 and u_2, $|Z(u_1)|$ and $|Z(u_2)|$ represent the numbers of socially related passengers of user u_1 and u_2 respectively.

2) USER MODEL ENHANCEMENT BASED ON SOCIAL RELATIONSHIPS

Based on the (14), users that have a social relationships with the target user are sorted in descending order. The algorithm takes the top ranked user to enhance the user model of the target user. If the candidate user is not a cold-start user on the target route, the candidate user's model is linearly added to the target user's model; otherwise, a strategy similar to the one adopted for the route cold-start problem is applied, i.e., his model on the most similar route is added to the model of the target user. Algorithm 4 describes the process of model enhancement model based on the social relationships for the user cold-start problem. Steps 1 to 5 result in a collection of users who have social relationships with the user u_a. The candidate users who are the closest to the target user are selected according to (14) in Step 6. From Steps 7 to Step 10, the user model is enhanced.

V. TICKET RECOMMENDATION BY COMBINING EXPLICIT AND LATENT FACTORS

Although flight ticket recommendation can be based on the features of flight tickets and a user's preference model, the decision process relating to a user's choice is very complex and there are always additional factors which are not considered in the model. Therefore, latent factors should also be considered for recommendation. When a user selects a ticket, it means this ticket can bring a higher utility than other tickets. This is equivalent to modeling a utility function between user u and ticket t. Suppose $V(u, t)$ is a pseudo rating that user u assigns to ticket t and this pseudo rating matrix can be factorized into two vectors ϕ_u and θ_t of dimensions $W \times 1$, respectively. In other words, rating $V(u, t)$ is approximated by a dot product of ϕ_u and θ_t in a joint latent factor space of dimensionality W:

$$V(u, t) = \phi_u^T \times \theta_t, \tag{17}$$

where ϕ_u ($\phi_u = W_u \times K_u$) denotes the preference of user u, and θ_t ($\theta_t = M_t \times F_t$) denotes the features of ticket t. In this model, with the use of transformation matrix W_u, we convert the feature distribution model K_u from explicit space to implicit space. Simultaneously, with the use of transformation matrix M_t, we convert feature F_t from explicit space to implicit space. As users' preferences change over time, takeoff time is involved in K_u and F_t as a flight feature. The definition of each notation is listed in Table 1.

We use $P(t_i, t_j)$ to denote the preference probability between ticket t_i and ticket t_j. Moreover, a logistic sigmoid function is used to model the preference probability.

$$\begin{aligned} P(t_i, t_j) &= P(V(u, t_i) > V(u, t_j)) \\ &= \frac{1}{1 + e^{-(V(u,t_i) - V(u,t_j))}}. \end{aligned} \tag{18}$$

We can establish a preferential pair-wise relationship between the chosen ticket and the other candidate tickets in a choice. Equation (18) is used as an objective function whose

Algorithm 4 User Model Enhancement Based on Social Relationships

```
Input: O: the historical orders of the route,
       F: the set of ticket features.
Output: social relationship enhanced model $D_{amp}$.
1:  $U \leftarrow getRelatedUsers(u_a)$;
2:  rank_list $\leftarrow \phi$;
3:  for $u \in U$ do
4:      rank_list.append(Rela($u_a$,u));
5:  end for
6:  $u_b \leftarrow Max(rank\_list)$;
7:  $r_b \leftarrow getOptimalRoute(u_b)$;
8:  $D'[r_b] \leftarrow D[r_b]/M[r_b]$;
9:  $D_{amp} = Normalize(D[r_a] + D'[r_b])$;
10: return $D_{amp}$.
```

TABLE 1. Notations.

Notation	Definition
$V(u,t)$	the preference of user u on flight t
ϕ_u	user preference in implicit space, $\lvert W \times 1\rvert$
θ_t	flight feature in implicit space, $\lvert W \times 1\rvert$
W_u	the transformation matrix of user preference, $\lvert W \times K\rvert$
K_u	user preference in explicit space, $\lvert K \times 1\rvert$
M_t	the transformation matrix of flight feature, $\lvert W \times F\rvert$
F_t	flight feature in explicit space, $\lvert F \times 1\rvert$

parameters should be learned.

$$\arg\min_{W,M} : -logP(t_i, t_j) + \frac{\lambda}{2} \times (\sum_{t_i,t_j \in W} W^2_{t_i,t_j} + \sum_{t_i,t_j \in M} M^2_{t_i,t_j}), \tag{19}$$

where W and M are the two transformation matrices to be learned. To simplify the training process, we take the logarithm over $P(t_i, t_j)$. Also, we add the regular terms to prevent overfitting. Here we apply stochastic gradient descent (SGD) to estimate the parameter values, yielding:

$$W_u = W_u + \alpha(\frac{\partial logP(t_i, t_j)}{W_u} - \lambda \times W_u), \tag{20}$$

$$M_{t_i} = M_{t_i} + \alpha(\frac{\partial logP(t_i, t_j)}{M_{t_i}} - \lambda \times M_{t_i}). \tag{21}$$

Here α is the learning rate. At each iteration, we calculate the gradient of W_u and M_{t_i} with respect to the objective function and it iterates to the gradient direction. To ensure the chosen ticket has the maximum utility, we set up the relationship between the selected ticket and any other ticket in the candidate list. If each order is considered in the training process, it results in a tremendous computing load. In reality, we collect data from a random sample of the candidate list and form the training pair and update the parameter matrix during the training. We combine the results of the explicit model and implicit model.

$$R(t) = R(K^T_u \times F_t) + R(\phi^T_u \times \theta_t), \tag{22}$$

where the first item denotes the ranking based on the explicit model, and the second item denotes the ranking based on the implicit model. We recommend flight tickets to the user based on the final ranking $R(t)$. Algorithm 5 presents the process of recommendation.

For each training data, we need to obtain the search results from which a ticket is chosen. We collect these flights in terms of flight number, the same takeoff date, and the same class. The time complexity of our method is $O(WN)$, where W is the number of features in the implicit space and N is the upper limit of the number of iterations.

Algorithm 5 Ticket Recommendation by Combining Explicit and Latent Factors

```
Input: $O$: the historical orders of the route,
       $F$: the set of ticket features,
       $C$: the search result list.
Output: A ranked tickets list $R$.
 1: $W_u \leftarrow 0$;
 2: $M_{t_i} \leftarrow 0$;
 3: for $o \in O$ do
 4:     $C_o \leftarrow getSearchResult(o)$;
 5:     for $t_o \in C_o.sample()$ do
 6:         $W_u = W_u + \alpha(\frac{\partial logP(t_i,t_j)}{W_u} - \lambda \times W_u)$;
 7:         $M_{t_i} = M_{t_i} + \alpha(\frac{\partial logP(t_i,t_j)}{M_{t_i}} - \lambda \times M_{t_i})$;
 8:     end for
 9:     If reach iteration limit, break;
10: end for
11: for $t \in C$ do
12:     $R_t = R(K^T_u \times F_t) + R(\phi^T_u \times \theta_t)$;
13:     Append $R_t$ to $R$;
14: end for
15: Sort $R$ by ascending;
16: return $R$.
```

VI. EXPERIMENTS AND RESULTS

A. DATA SET

We use a real-world data set provided by an online travel agency to perform the experiments. The fields of each flight ticket order record include but is not limited to *departure city, arrival city, takeoff date, takeoff time, airline, departure port, arrival port, class, price, ticket policy, craft size, passenger ID*. Among them, *departure city* and *arrival city* indicate the route to which the flight belongs; *departure port* and *arrival port* represent the departure and arrival airports of flights as some cities may have two or more airports; *class* can be first class, business class or economy class; *craft size* represents the size of the aircraft. We collect domestic ticket orders from January 2013 to July 2015, comprising direct flight ticket orders for all Chinese domestic routes. Some routes are more popular, carrying a large number of passengers traveling every day and there are many flights on these routes, e.g., the route from *Beijing* to *Shanghai*. However, most routes do not have a large number of flights.

We choose four routes as our testing routes, namely *BJS-SHA* (the route from Beijing to Shanghai), *BJS-CTU* (the route from Beijing to Chengdu), *BJS-SZX* (the route from Beijing to Shenzhen) and *BJS-XIY* (the route from Beijing to Xi'an). The ticket order quantity distribution over the four testing routes is shown in Fig. 2(a) and the user quantity distributions over the four testing routes is shown in Fig. 2(b). It can be seen that, *BJS-SHA* is the most popular route and for routes *BJS-SHA* and *BJS-SZX*, the ratio between the number of orders and the number of users being about 2 to 1. The ratio between the number of orders and the number of users is about 1.6 to 1 for route *BJS-CTU* and *BJS-XIY*, which indicates that the latter routes have more new users. This, in part, reflects the nature of different cities, i.e., passengers of *BJS-SHA* and *BJS-SZX* are more likely to travel for business purposes, thus resulting in multiple round-trips; whereas the passengers of *BJS-CTU* and *BJS-XIY* are more likely to travel for leisure purposes.

Detailed data distribution statistics for route *BJS-SHA* are shown in Fig. 3. The user number distribution over the number of ticket orders is shown in Fig. 3(a). It can be seen that the number of users with only one order accounts for 61% of the total number of users, the number of users with orders not exceeding 2 accounts for 80% of the total, and more than 99% of user orders are below 10. The percentages of total orders of users who have placed a certain number of ticket orders is shown in Fig. 3(b). Although the number of users with only one order accounts for 61% of the total number of users, their

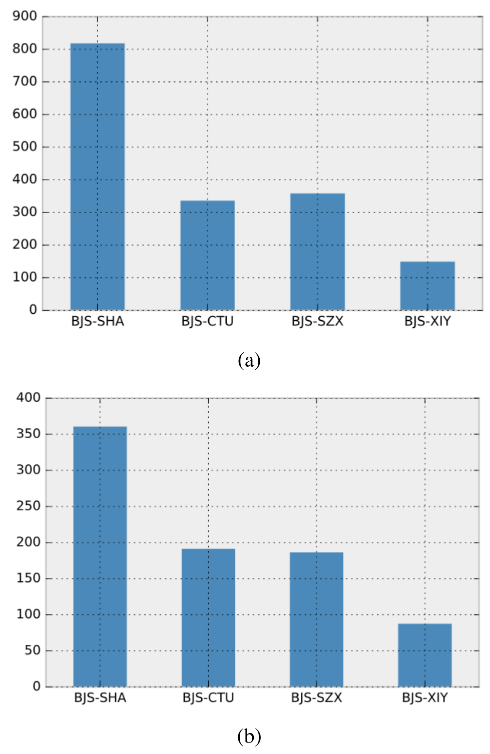

FIGURE 2. Data statistics of the four testing routes. (a) The number of orders. (b) The number of users.

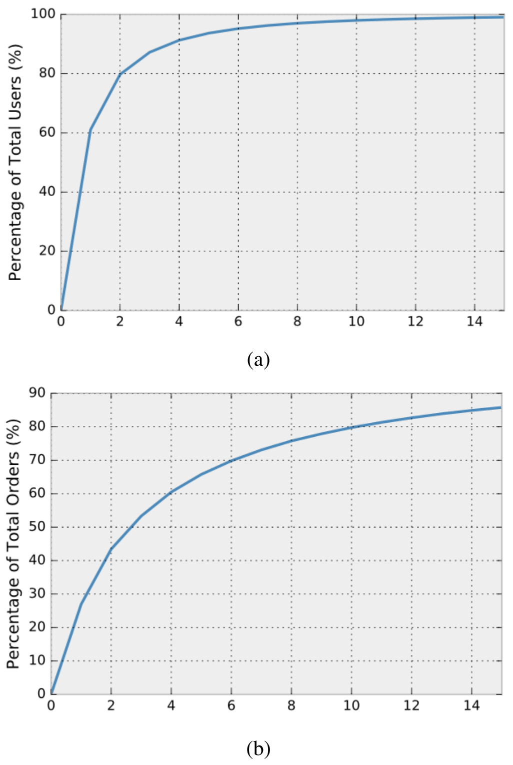

FIGURE 3. Data statistics for route *BJS-SHA*. (a) The number of orders. (b) The number of users.

orders only account for 25% of the total orders and 15% of users place 85% of orders. It can be seen that the cold-start problem is very serious for flight ticket recommendation.

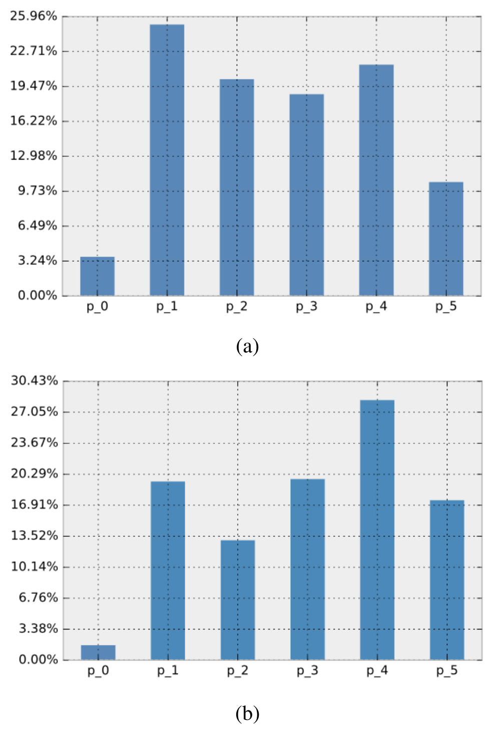

FIGURE 4. Price index distribution over different routes. (a) BJS-SHA. (b) BJS-XIY.

Fig. 4 shows the price index distribution on routes *BJS-SHA* and *BJS-XIY*. The price index (as defined in (1)) is divided into 6 levels (from *p_0* to *p_5* in ascending order). The horizontal axis represents the different price index levels. The vertical axis represents the percentage of ticket orders in each price index level. As shown in Fig. 4, route *BJS-SHA* and route *BJS-XIY* are similar in that most of their ticket orders fell into a median price index level, however on route *BJS-SHA*, a larger proportion of orders were in the high price level (e.g., *p*_1 represents 25% of ticket orders), whereas on route *BJS-XIY*, there were more lower price tickets (e.g., *p*_4 represents 27% of ticket orders). This is consistent with the differences in the travel purposes between the routes and indicates that the feature distributions of different routes are different.

Fig. 5 shows the user number distribution along with the standard entropy of airline choices of route *BJS-SHA* and route *BJS-XIY* respectively. The horizontal axis represents the standard entropy, which is obtained by normalizing the entropy value calculated according to (5). The degree of users' airline preference concentration increases as the entropy value declines. As shown in Fig. 5, both routes have the largest number of users with standard entropy less than 0.02. On *BJS-SHA*, a large proportion of users have a standard entropy less than 0.3, which indicates that users' preferences for airlines on this route are more similar. On *BJS-XIY*, the standard entropy of most users centers around 0.6 and users' preferences for airlines on this route are more diversified.

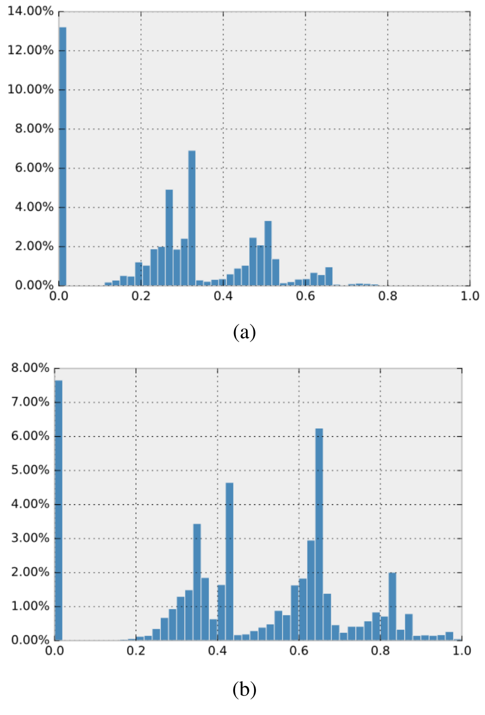

FIGURE 5. Standard entropy distribution of airline choices over different routes. (a) BJS-SHA. (b) BJS-XIY.

TABLE 2. Similarities between the routes.

Route 1	Route 2	Similarity
BJS-SHA	*BJS-CTU*	55.4%
BJS-SHA	*BJS-SZX*	50.5%
BJS-SHA	*BJS-XIY*	47.6%
BJS-CTU	*BJS-SZX*	61.8%
BJS-CTU	*BJS-XIY*	49.0%
BJS-SZX	*BJS-XIY*	49.6%

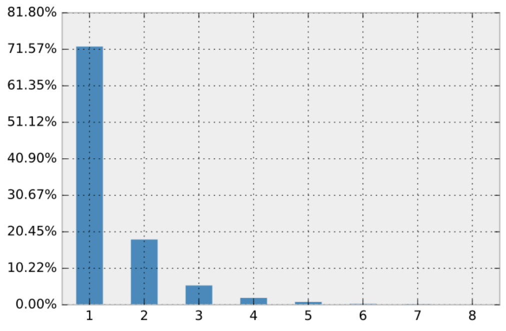

FIGURE 6. Co-passenger relationship analysis.

The similarities between the different routes are shown in Table 2. Of these four routes, the similarities between route *BJS-CTU* and the route from *Beijing* to the other three cities are high, while the similarities between route *BJS-XIY* and the other routes are low.

Fig. 6 shows the distribution of users who establish social relations with others on the same flights. The horizontal axis represents the number of users who are co-passengers with one other user. It can be seen 70% of users take the same flight with only one other user, 18% of users take the same flight with two other users and 10% of users take the same flight with more than three other users.

B. EVALUATION METRICS

For a user to buy a flight ticket, which is flight ticket t in the actual order, we recommend a list of candidate flight tickets C to the user. Therefore, the accuracy of the recommendation can be computed by the following equation:

$$A_{rec}(t) = \begin{cases} 1 & |C| = 1 \\ 1 - \dfrac{Tindex - 1}{|C| - 1} & |C| > 1, \end{cases} \tag{23}$$

where t_{index} is the index of flight t in the recommending list ($t_{index} \in [1, |C|]$). The higher the actual position, the smaller T_{index} is. For example, when t is ranked first, $A_{rec} = 1$; when T is listed at the bottom, $A_{rec} = 0$. Moreover, accuracy is also related to the number of flight tickets in the candidate lists, which is in line with the requirements of a real business scenario. In addition, if there are other candidate tickets in the collection that have the same score as the testing ticket, we uniformly place the testing tickets at the end of these tickets. We can also compute the mean accuracy over the entire data set, i.e., the average of the recommended accuracy of each flight in the data set.

$$MA = \frac{\sum_{i=1}^{|M|} A_{rec}(M_i)}{|M|}, \tag{24}$$

where M stands for the testing data set. Equation (23) defines the recommendation effect over a testing ticket, and (24) defines the mean accuracy over the entire testing data set, i.e., the average of the recommendation accuracy for each testing order in the testing set.

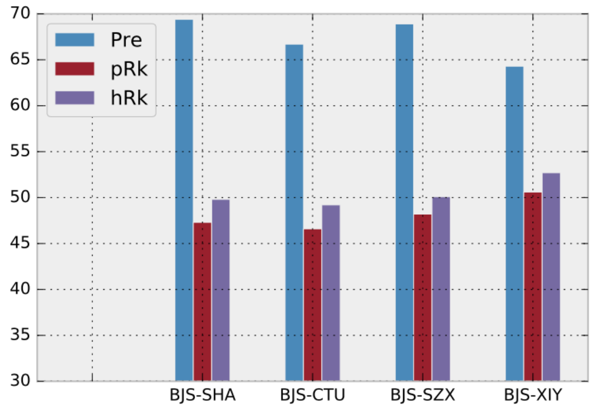

FIGURE 7. Comparisons of mean accuracy between different approaches (*Pre*, *pRk*, and *hRk*).

C. RECOMMENDATION PERFORMANCE

We conduct experiments on the four testing routes, including *BJS-SHA*, *BJS-CTU*, *BJS-SZX* and *BJS-XIY*. Fig. 7–Fig. 10 shows the recommendation performance of the different methods over the four testing routes.

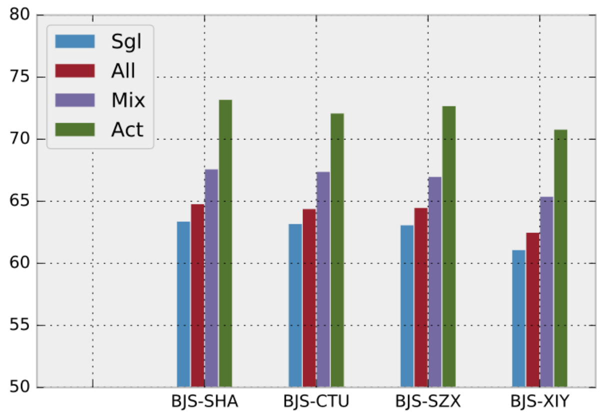

FIGURE 8. **Comparisons of mean accuracy between different approaches (*Sgl*, *All*, *Mix*, and *Act*).**

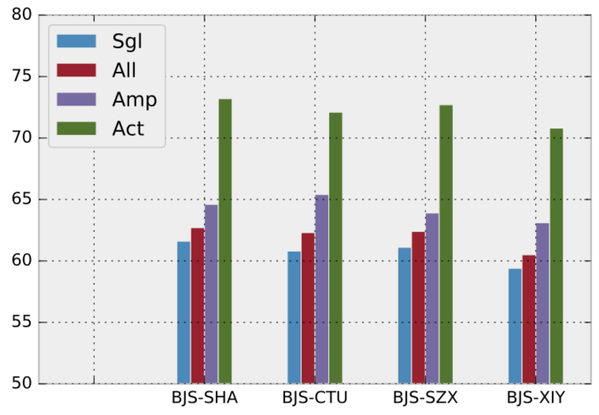

FIGURE 9. **Comparisons of mean accuracy between different approaches (*Sgl*, *All*, *Amp*, and *Act*).**

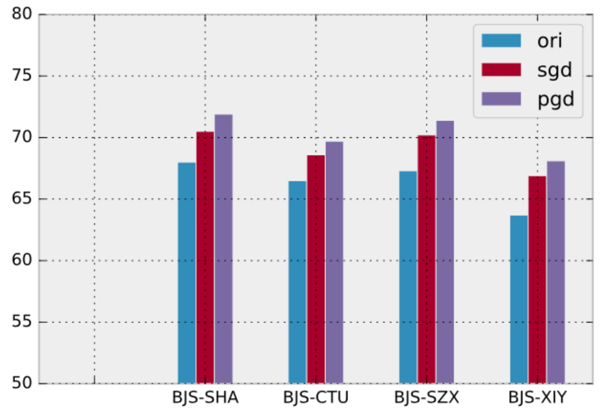

FIGURE 10. **Comparisons of mean accuracy between different approaches (*ori*, *sgd*, and *pgd*).**

1) EVALUATION OF THE BASELINE USER FEATURE DISTRIBUTION MODEL

We conduct experiments on the historical orders of each user independently. The user's last order is used as the testing data, and all the previous orders are used as the training data to calculate the user feature distribution model. We compare the proposed basic user modeling method *Pre* against two competitors:

(1) *pRk*. *pRk* is the strategy that ranks flights by price, i.e., candidate flights are presented in ascending order by price.

(2) *hRk*. *hRk* is the strategy that ranks flights by popularity. In the *hRk* strategy, flight tickets with the same *class*, the same *flight* or within the same *price* range are treated as the same items. We analyze the tickets purchased on all four routes for the past two weeks from the departure date of the testing data, using order quantity to indicate the popularity of the ticket, and sort the candidate flights in descending order of popularity.

As can be seen from Fig. 7, the method based on the feature distribution model has a higher mean accuracy rate, which demonstrates that the candidate tickets that meet the user's preference are located in a better position compared to the others. Two sorting strategies *pRk* and *hRk* have lower mean accuracy. There is a big gap between the two basic sorting strategies and the proposed recommendation algorithm based on the user feature distribution model. The reason for this can be attributed to the following: (1) these two sorting strategies are based only on statistical analysis and do not fully consider the users' personalized preferences; (2) for the *pRk* strategy, price is not the only factor that affects a user's purchasing decision, and for the *hRk* strategy, since the number of tickets is limited by the number of seats on the flight, it is impossible to accurately ascertain its popularity, which has an inevitable impact on the recommendation performance.

2) EVALUATION OF THE ENHANCED USER MODEL BY TRANSFERRING USER MODELS OF SIMILAR ROUTES

To deal with the route cold-start scenario, we select users who have no more than two orders on the target route and have historical orders whose number is no less than three on other routes. Even if the user only has a training order on the target route, the recommendation effect is much better than the *pRk* and *hRk* as shown in Fig. 7. Therefore, these two models of ticket recommendation based on basic business strategies are no longer tested here. We test the following four methods:

(1) *Act* represents the single route model recommendation for active users of the route.

(2) *Sgl* represents the recommended result of using the single route model for the route's cold-start users;

(3) *All* represents the recommended result of using the whole route model for the route's cold-start users;

(4) *Mix* represents the recommended results for the route's cold-start users using the enhanced model by transferring the user models of similar routes.

The recommendation performance of *Mix* and the other models (*Act*, *Sgl*, and *All*) over the four testing routes is shown in Fig. 8. It can be seen that on each route, active users have a higher recommendation accuracy than that of inactive users. We conclude that the recommendation accuracy of using *All* for such users is higher than using only *Sgl*. *Mix* has the highest accuracy, but there is still a gap between it with those of active users.

3) EVALUATION OF THE USER MODEL ENHANCEMENT BASED ON SOCIAL RELATIONSHIPS

We test the recommendation performance of the four models *Act*, *Sgl*, *All*, and *Amp*. The first three models were introduced in the previous section. *Amp* is the enhanced model for cold-start users based on the social relationship enhancement model. As shown in Fig. 9, the flight recommendation results of *Amp* are better than models *Sgl* and *All* using only the user's own data. This shows that the models of users with social relationships can be migrated and applied to enhance the user models of cold-start users and finally improve the accuracy of flight ticket recommendation.

4) EVALUATION OF FLIGHT TICKET RECOMMENDATION COMBINED WITH LATENT FACTORS

We study whether the model combined with latent factors can further improve the recommendation performance. Fig. 10 shows the flight ticket recommendation results on the four routes, using the explicit user model (*ori*), random sampling model (*sgd*) and preference ranking sampling model (*pgd*).

The difference between *sgd* and *pgd* is that the former uses a random sampled paired sample for model training, whereas the latter uses a paired sample ordered according to explicit preference for model training. It can be seen that the accuracy of ticket recommendation combined with latent factors is improved compared to using only explicit features. Compared with the *sgd* model, accuracy is slightly improved using *pgd*. It can be concluded that the user model combined with latent factors can mine the correlation between the explicit features of flight tickets to a certain extent.

VII. CONCLUSION

This paper studied how to build a user's preference model for personal flight ticket recommendation. Specifically, the user model is enhanced to deal with the cold-start problems, which is very serious in this application. In order to formulate each user's preferences, this paper proposes a user feature distribution model. To address the cold-start problems, including the route cold-start problem and the user cold-start problem in flight ticket recommendation, we present methods based on route similarity and social relationships between passengers to improve the user models. In order to further improve the recommendation performance, we map the user models and flight ticket features into a latent space. The experiment results on a real-world data set demonstrate that the enhanced user model together with our recommendation models can improve recommendation performance significantly, especially for cold-start users.

ACKNOWLEDGMENT

The authors would like to thank the editor and the anonymous reviewers for their insightful and constructive comments and suggestions on improving this paper.

REFERENCES

[1] Y. Ge, Q. Liu, H. Xiong, A. Tuzhilin, and J. Chen, "Cost-aware travel tour recommendation," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2011, pp. 983–991.

[2] J. K. Schafer, J. Ben, and J. Riedl, "Recommender systems in E-commerce," in *Proc. 1st ACM Conf. Electron. Commerce*, Nov. 1999, pp. 158–166.

[3] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "GroupLens: Applying collaborative filtering to usenet news," *Commun. ACM*, vol. 40, no. 3, pp. 77–87, 1997.

[4] J. Cao, F. Yang, Y. Xu, Y. Tan, and Q. Xiao, "Personalized flight recommendations via paired choice modeling," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 1265–1270.

[5] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 735–749, Jun. 2005.

[6] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2011.

[7] W. Jian, J. He, K. Chen, Y. Zhou, and Z. Tang, "Collaborative filtering and deep learning based recommendation system for cold start items," *Expert Syst. Appl.*, vol. 69, pp. 29–39, Mar. 2017,

[8] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades, "Facing the cold start problem in recommender systems," *Expert Syst. Appl.*, vol. 41, no. 4, pp. 2065–2073, Mar. 2014.

[9] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, "A collaborative filtering approach to mitigate the new user cold start problem," *Knowl.-Based Syst.*, vol. 26, pp. 225–238, Feb. 2012.

[10] P. Cremonesi, A. Tripodi, and R. Turrin, "Cross-domain recommender systems," in *Proc. IEEE 11th Int. Conf. Data Mining Workshops*, Dec. 2011, pp. 496–503.

[11] M. Qi, J. Cao, and Y. Tan, "Cross-domain tourist service recommendation through combinations of explicit and latent features," in *Proc. Asia–Pacific Services Comput. Conf.* Cham, Switzerland: Springer, 2016, pp. 92–105.

[12] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proc. 14th Conf. Uncertainty Artif. Intell.*, Morgan Kaufmann, 1998, pp. 43–52.

[13] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan./Feb. 2003.

[14] M. J. Pazzani, "A framework for collaborative, content-based and demographic filtering," *Artif. Intell. Rev.*, vol. 13, nos. 5–6, pp. 393–408, Dec. 1999.

[15] L. M. D. Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks," *Int. J. Approx. Reasoning*, vol. 51, no. 7, pp. 785–799, Sep. 2010.

[16] K. Yu, A. Schwaighofer, and V. Tresp, "Collaborative ensemble learning: Combining collaborative and content-based information filtering via hierarchical Bayes," in *Proc. 19th Conf. Uncertainty Artif. Intell.*, Morgan Kaufmann, 2002, pp. 616–623.

[17] P. Zigoris and Y. Zhang, "Bayesian adaptive user profiling with explicit & implicit feedback," in *Proc. 15th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2006, pp. 397–404.

[18] G. Jawaheer, M. Szomszor, and P. Kostkova, "Comparison of implicit and explicit feedback from an online music recommendation service," in *Proc. 1st Int. Workshop Inf. Heterogeneity Fusion Recommender Syst.*, Sep. 2010, pp. 47–51.

[19] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2009, pp. 263–272.

[20] Y. T. Lin and S. S. Tseng, "A characterized rating recommend system," in *Proc. Knowl. Discovery Data Mining (PAKDD)*, Hong Kong, Apr. 2001, pp. 41–46.

[21] J. Cao, Y. Xu, H. Ou, Y. Tan, and Q. Xiao, "PFS: A personalized flight recommendation service via cross-domain triadic factorization," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jul. 2018, pp. 249–256.

[22] L. Coyle, "Making personalised flight recommendations using implicit feedback," Univ. Dublin, Trinity College, Dublin, Ireland, Tech. Rep., 2004.

[23] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches," *AI Commun.*, vol. 7, no. 1, pp. 39–59, 1994.

[24] J. Kolodner, *Case-Based Reasoning*. San Mateo, CA, USA: Morgan Kaufmann, 1993.

[25] R. S. Barth, "Design and implementation of a flight recommendation engine," Federal Univ. Rio Grande do Sul, Porto Alegre, Brazil, Tech. Rep., 2014.

[26] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2002, pp. 253–260.

[27] D. T. Sun, T. He, and F. H. Zhang, "Survey of cold-start problem in collaborative filtering recommender system," *Comput. Modernization*, vol. 5, pp. 59–63, May 2012.

[28] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Inf. Retr.*, vol. 4, no. 2, pp. 133–151, Jul. 2001.

[29] S. Sedhain, S. Sanner, D. Braziunas, J. Christensen, and J. Christensen, "Social collaborative filtering for cold-start recommendations," in *Proc. 8th ACM Conf. Recommender Syst.*, Oct. 2014, pp. 345–348.

[30] S. Sahebi and W. W. Cohen, "Community-based recommendations: A solution to the cold start problem," in *Proc. WOODSTOCK*, 1997, pp. 1–5.

QI GU is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. She is also teaching with Nantong University, China. Her research interests include intelligence recommender systems, natural language processing, and service computing.

JIAN CAO received the Ph.D. degree from the Nanjing University of Science and Technology, in 2000. He is currently a Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China, and the Deputy Head of the Department. He is also the Leader of the Laboratory for Collaborative Intelligent Technology. He leads the Research Group of Collaborative Information System. He has published more than 100 research papers in prestigious journals, such as the IEEE Transactions on Mobile Computing, the IEEE Transactions on Services Computing, and the *ACM Transactions on Information Systems*. His research interests include intelligent service computing, co-operative information systems, data analytics, and software engineering. He is a Senior Member of the China Computer Federation.

YAFENG ZHAO received the B.S. degree in computer science from Zhejiang University, China, in 2014, and the M.S. degree from Shanghai Jiao Tong University, China, in 2017. He is currently a Software Engineer with Tencent, Shanghai, China. His research interests include intelligent recommendation systems, data analytics, and software engineering.

YUDONG TAN received the B.S. and M.S. degrees from Xi'an Jiaotong University, in 2000, and the Ph.D. degree in computer engineering from the Georgia Institute of Technology, in 2005. He held numerous senior positions in the technology companies in Silicon Valley, first at Intel and then Nvidia, Yelp, and Microsoft. In 2014, he joined the Flight Business Unit, Ctrip.com International, Ltd., Shanghai, where he is currently the Vice President and the Chief Technology Officer. He leads the engineering, product, operation, and customer service teams.