Personalized Flight Itinerary Ranking at Fliggy

Jinhong Huang, Yang Li, Shan Sun, Bufeng Zhang, Jin Huang
Alibaba Group
Hangzhou, China
{jinhong.hjh,sushu.ly,sunshan.ss,feitong,fengchu.hj}@alibaba-inc.com

ABSTRACT

Flight itinerary ranking is critical for Online Travel Agencies (OTAs) since more and more customers book flights online. Currently, most OTAs still adopt rule-based strategies. However, rule-based methods are not able to model context-aware information and user preferences. To this end, a novel Personalized Flight itinerary Ranking Network (PFRN) is proposed in this paper. In PFRN, a Listwise Feature Encoding (LFE) structure is proposed to capture global context-aware information and mutual influences among inputs. Besides, we utilize behaviors of both individual user and group users sharing the same intention to express user preferences. Then a User Attention Mechanism is proposed to rank flight itineraries based on the user preferences effectively and efficiently. Offline experiments on real-world datasets from Amadeus and Fliggy show the superior performance of the proposed PFRN. Moreover, PFRN has been successfully deployed on online system for searching itineraries at Fliggy and achieved significant improvements.

CCS CONCEPTS

• **Computing methodologies** → **Ranking**; **Neural networks**.

KEYWORDS

flight itinerary ranking; personalization; attention mechanism

ACM Reference Format:
Jinhong Huang, Yang Li, Shan Sun, Bufeng Zhang, Jin Huang. 2020. Personalized Flight Itinerary Ranking at Fliggy. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3340531.3412735

1 INTRODUCTION

Currently, online travel apps are becoming the most prevalent channels to book flight itineraries. In our case, Fliggy, one of the most popular online travel apps in China, concentrates on the flight itinerary ranking problem to quickly match flight itineraries to passengers' needs so as to improve user experience and increase the conversion rate (CVR).

Different from traditional e-commerce search systems or recommender systems deployed with various deep learning models (e.g., [16, 21]), it is still challenging to rank flight itineraries in tourism. The most commonly used strategy of current Online Travel Agencies (OTAs) is price-based or duration-based. For example, Expedia[1] ranks flight itineraries by price as default [2]. The main challenges of the flight itinerary ranking system are listed below:

- **Global context-aware information and mutual influences**: Unlike items with various attributes in traditional e-commerce search systems (e.g., Taobao [3]), flight itineraries have standard structures and the key factors (e.g., price or duration) are listed clearly for users to compare, which is shown in Figure 1 [4]. Models are expected to have the same ability as users to compare a list of itineraries with the global context-aware information and mutual influences.
- **User preferences**: The major challenge to model user preferences is the limited user information. We notice that more than 50% users have no booking records in one year at Fliggy. One possible solution is to adopt user real-time clicking behaviors as a complement. However, only about 40% users have more than 3 real-time clicking behaviors on flight itineraries at Fliggy, while the average number of short-term behaviors is about 6 at Taobao [14]. The sparsity of user behaviors may restrict performances of previous ranking or recommendation models in flight itinerary ranking.

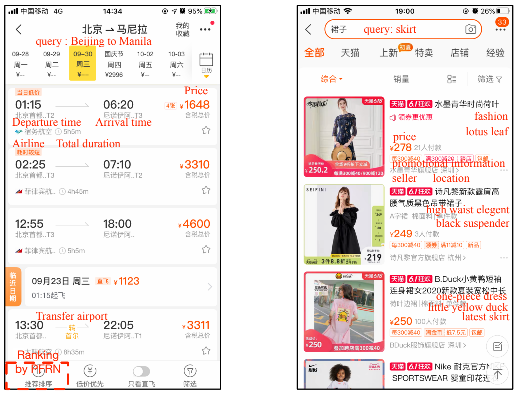

Figure 1: The listing page at Fliggy (left) and Taobao (right). In the listing page at Fliggy, it ranks flight itineraries by models (PFRN) as default rather than the rule-based methods. Furthermore, compared with items listed at Taobao, the key factors of flight itineraries are listed clearly at Fliggy and users can compare itineraries easily.

[1]https://www.expedia.com/Flights
[2]We observed the cases at Expedia on May 20, 2020.
[3]Taobao app is the largest e-commerce shopping app in China.
[4]Figure 1 is captured on May 20, 2020.

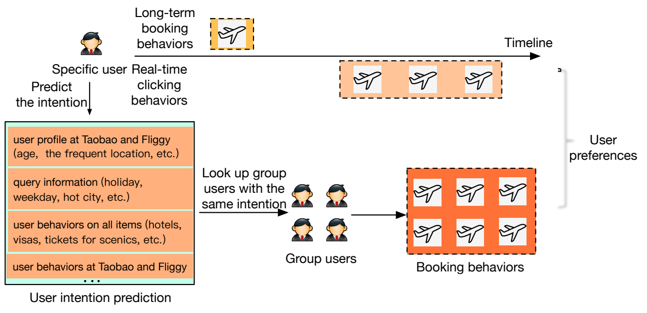
Long-term booking behaviors
Timeline
Specific user
Real-time clicking behaviors
Predict the intention
User preferences
user profile at Taobao and Fliggy (age, the frequent location, etc.)
query information (holiday, weekday, hot city, etc.)
user behaviors on all items (hotels, visas, tickets for scenics, etc.)
user behaviors at Taobao and Fliggy
...
Look up group users with the same intention
Group users
Booking behaviors
User intention prediction

Figure 2: The user preference learning from a specific user and group users with the same intention.

The existing researches discussing flight itinerary recommendation or ranking problems only focus on one of those mentioned challenges. For the global context-aware information problem, Mottini et al. [15] proposed a deep choice model (DCM) using recurrent neural networks (RNN) and the pointer mechanism. However, RNN only models the effect of previous inputs and ignores interactive influences of other potential itineraries. Furthermore, Monttini et al. only concentrated on flight features due to anonymous user sessions and lacking available user behavior history. For the user preference problem, Budell et al. [3] proposed to design cross-term features between item-specific and user-specific variables (e.g. the number of adults and the brands of user phones) manually and then adopt a ranking support vector machine (SVM) to select features. However, they focused on the user statistical features and the sequence behaviors are not considered to model the user preferences over time. Furthermore, all researches mentioned above were conducted offline on small datasets, 30,000 and 10,000 users respectively, and have not been deployed on online applications or been evaluated with online A/B tests.

In this paper, we propose a Personalized Flight itinerary Ranking Network (PFRN) to rank flight itineraries effectively and efficiently. Inspired by the recent successes of multi-head self-attention mechanism [10, 11, 18–20], we propose a Listwise Feature Encoding (LFE) structure in PFRN, by means of multi-head self-attention layer with relative position representations (RPR) [5, 17] to capture global context-aware information from all inputs and mutual influences between itineraries. Then dense neural networks [9] is incorporated after self-attention layer in the LFE structure to learn hierarchical features. Finally, a user attention mechanism is proposed to model the attention from users on flight itineraries, which also enables the model to learn user features and flight itinerary features by the LFE structure independently and simultaneously. Besides, in order to alleviate the data sparse phenomenon, we not only utilize real-time clicking behaviors of a specific user, but also leverage booking behaviors of group users with similar predicted intention (e.g., travel or business), which is shown in Figure 2.

The major contributions of this paper are listed bellow:

(1) To the best of our knowledge, this is the first work to study the real-time personalized flight itinerary ranking problem on large-scale dataset. The designed system is deployed on online travel apps for the first time and achieved surprising results.

(2) We propose a listwise feature encoding (LFE) structure to extract global context-aware information, model mutual influences and learn hierarchical features for flight itinerary representation and user preference representation.

(3) In order to model user preferences, we not only utilize user long-term booking behaviors and real-time clicking behaviors, but also leverage the booking behaviors from group users. Furthermore, we propose a user attention mechanism to model the attention between users and flight itineraries, which improves the performance and helps to speed up the training and testing procedures.

(4) We verify the PFRN on the Amadeus dataset and the Fliggy dataset [5] to demonstrate its effectiveness and efficiency. Furthermore, our model have been deployed on-line at Fliggy successfully. It is worth noting that the PFRN improves *Precision*@1 by 23% compared with the previous system.

Our paper is organized as follows. In Section 2, we summarize the related work. Section 3 formally defines our problem and presents the proposed method in detail. In Section 4, we present the experimental results and detailed analyses. We conclude this paper and point out the future work in Section 6.

2 RELATED WORK

Our work is relevant to two lines of work : 1) models for global context-aware information learning, and 2) works on user preference learning.

Global context-aware information learning. The state-of-the-art methods are RNN-based models, such as DCM for airline itinerary choice prediction [15], or Transformer-based models including personalized re-ranking model (PRM) at Taobao [16]. RNN-based models feed the sequential inputs and output the decoder which contains the global context-aware information. However, RNN-based approaches only model the influences of previous inputs, which have limited ability to model the interactions between items in the list. Transformer-based architecture [19] used the multi-head self-attention mechanism where any two items can interact with each other directly to model the mutual influences. In contrast to RNN, standard Transformer-based model introduced the absolute positions to capture the positional information. Recently, relative position representations have been proposed in natural language processing [5, 17] to allow attention to be informed by how far two positions are apart in a sequence.

User preference learning. Zhou et al. [21] proposed a deep interest network (DIN) to model user interests between a candidate item and the items from user behaviors for online advertising systems in Alibaba. For item recommendation problems, many works [10, 11, 18, 20] have been proposed to use the multi-head self-attention mechanism to learn the user preferences from user behaviors. However, these works concentrate on the behaviors from a specific user. When the user behaviors are sparse, the performance is limited. Airbnb [8] proposed to train embeddings at a level of user type, making users with the same type share the same embedding. Furthermore, it is still an open problem to utilize the feature vector in listwise models. Pei et al. [16] proposed to adopt a pre-trained model to generate personalized vectors and

[5]Fliggy dataset includes about 1 million bookings. Rich user behaviors are collected from logs. This dataset will be released soon.

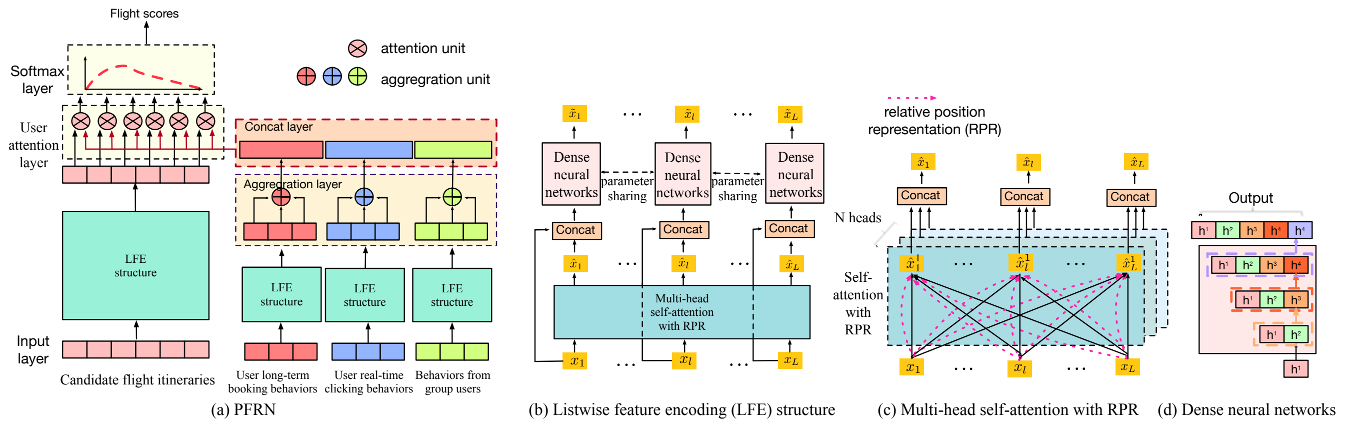
Flight scores
Softmax layer
attention unit
aggregration unit
User attention layer
Concat layer
Aggregration layer
LFE structure
Input layer
Candidate flight itineraries
User long-term booking behaviors
User real-time clicking behaviors
Behaviors from group users
(a) PFRN
Dense neural networks
parameter sharing
Concat
Multi-head self-attention with RPR
(b) Listwise feature encoding (LFE) structure
relative position representation (RPR)
N heads
Self-attention with RPR
(c) Multi-head self-attention with RPR
Output
(d) Dense neural networks

Figure 3: The structure of the proposed PFRN and its sub-modules. In this framework, the candidate flight itinerary representation and the user preference representation can be learned independently and simultaneously by the LFE structure. Then a user attention mechanism in the user attention layer will choose one flight itinerary based on the user preferences.

then concatenated personalized vectors and item features as the input of Transformer structure, where two models are required to be trained and be served online.

3 PROPOSED METHOD

In this section, we present the proposed method extensively. Firstly, we illustrate the flight itinerary ranking problem. Then we give an overview of the proposed PFRN. Finally, we introduce the sub-modules of PFRN in detail.

3.1 Problem Definition

Let $\mathcal{U} = \{u_1, u_2, \dots, u_M\}$ be a set of users. The long-term booking behaviors of $u \in \mathcal{U}$ with L_B records is a time-ordered sequence $\mathcal{B}^u = \{\boldsymbol{b}_1^u, \dots, \boldsymbol{b}_{L_B}^u\}$. The real-time clicking behavior sequence of u with L_R records is $\mathcal{R}^u = \{\boldsymbol{r}_1^u, \dots, \boldsymbol{r}_{L_R}^u\}$. Furthermore, the group users sharing the same intention with u are defined as $\mathcal{U}_G = \{u_{G_1}, \dots, u_{G_M}\}$, where $u_{G_m} \in \mathcal{U}$. The L_G real-time booking behaviors of $\mathcal{U}_G$ sharing the same intention with u are defined as $O^u = \{\boldsymbol{o}_1^u, \dots, \boldsymbol{o}_{L_G}^u\}$. Given a sequence of Z flight itineraries $F = \{f_1, \dots, f_Z\} \in \mathbb{R}^{Z \times D_F}$, our goal is to predict the probability $p(y_z | F, \mathcal{B}^u, \mathcal{R}^u, O^u)$ of the z-th flight itinerary chosen by user u.

3.2 Model Architecture

The architecture of PFRN is shown in Figure 3. The model mainly consists of two modules: the Listwise Feature Encoding (LFE) structure and the User Attention Layer. In this framework, the flight itinerary representation and the user preference representation can be learned independently and simultaneously by the LFE structure to speed up the training phases and testing phases. From the LFE structure, we can both learn global context-aware information from candidate itineraries and model global user preferences from different types of user behaviors. Then a user attention mechanism is proposed to measure the interaction between users and flight itineraries. Finally, we output the scores after the Softmax Layer. The detailed structures of PFRN are introduced separately in the following subsections.

3.3 Listwise Feature Encoding Structure

The LFE structure is the base structure in PFRN to capture global context-aware information, model mutual influences from inputs and learn hierarchical features. Given a sequential input $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_L\} \in \mathbb{R}^{L \times D}$, where $\boldsymbol{x}_l \in \mathbb{R}^{1 \times D}$, the LFE structure outputs $\tilde{X} \in \mathbb{R}^{L \times \tilde{D}}$ as

$$\tilde{X} = \mathbf{LFE}([\boldsymbol{x}_1, \dots, \boldsymbol{x}_L], \boldsymbol{W}_{LFE}, k_{max}) \tag{1}$$

where $\boldsymbol{W}_{LFE}$ is the learnable parameters in the LFE structure and k_{max} is the max position to measure the relative position information. The LFE structure is shown in Figure 3 (b). The LFE structure includes two sub-modules: multi-head self-attention with relative position representation (RPR) and dense neural networks. In the self-attention layer, the RPR is incorporated to capture the relative-distance relationship. Besides, dense connections in dense neural networks are adopted to learn hierarchical features.

Multi-head self-attention with RPR. In contrast to RNN, the multi-head self-attention mechanism with absolute position vectors in traditional Transformer directly models global context-aware information and mutual influences from any pair of items. However, compared with the absolute positions, the relative position representation may be more helpful to capture the mutual influences. Recently, the idea of relative positional encodings has been explored in machine translation and achieved significant improvements [5, 17]. For instance, when a query vector attends on the key vectors, it does not need to know the absolute position of each key vector to identify the temporal order. Instead, it suffices to know the relative distance between each key vector and itself. More importantly, it is more intuitive and generalizable to define the temporal bias in a relative manner. In multi-head self-attention layer with RPR, we employ N attention heads and concatenate each head to form the output $\hat{\boldsymbol{x}}_l \in \mathbb{R}^{1 \times \hat{D}}$ as

$$\hat{\boldsymbol{x}}_l = \mathbf{Concat}(\hat{\boldsymbol{x}}_l^1, \dots, \hat{\boldsymbol{x}}_l^N) \tag{2}$$

The n-th head $\hat{\boldsymbol{x}}_l^n \in \mathbb{R}^{1\times\frac{\hat{D}}{N}}$ in (2) is defined as

$$\hat{\boldsymbol{x}}_l^n = \sum_{j=1}^{L} \alpha_{lj}^n (\boldsymbol{x}_j \boldsymbol{V}^n + \boldsymbol{\alpha_{lj}}^V) \tag{3}$$

$$\alpha_{lj}^n = \frac{\mathbf{exp}(e_{lj}^n)}{\sum_{i=0}^{L} \mathbf{exp}(e_{li}^n)} \tag{4}$$

$$e_{lj}^n = \frac{\boldsymbol{x}_l \boldsymbol{Q}^n (\boldsymbol{x}_j \boldsymbol{K}^n + \boldsymbol{\alpha}_{lj}^K)^T}{\sqrt{\hat{D}/N}} \tag{5}$$

$$\boldsymbol{\alpha}_{lj}^K = \boldsymbol{W}_{\mathbf{clip}(j-l, k_{max})}^K \tag{6}$$

$$\boldsymbol{\alpha}_{lj}^V = \boldsymbol{W}_{\mathbf{clip}(j-l, k_{max})}^V \tag{7}$$

$$\mathbf{clip}(j-l, k_{max}) = \mathbf{max}(-k_{max}, \mathbf{min}(k_{max}, j-l)) \tag{8}$$

where $\boldsymbol{\alpha}_{lj}^V \in \mathbb{R}^{1\times\frac{\hat{D}}{N}}$ represents the relative position information between the l-th query vector and the j-th value vector, $\boldsymbol{\alpha}_{lj}^K \in \mathbb{R}^{1\times\frac{\hat{D}}{N}}$ represents the relative position information between the l-th query vector and the j-th key vector, and $\boldsymbol{W}^K = (\boldsymbol{w}_{-k_{max}}^K, \ldots, \boldsymbol{w}_{k_{max}}^K)$ and $\boldsymbol{W}^V = (\boldsymbol{w}_{-k_{max}}^V, \ldots, \boldsymbol{w}_{k_{max}}^V)$ is the RPR. Note that the representations can be shared across attention heads. Besides, $\boldsymbol{Q}^n$, $\boldsymbol{K}^n$, $\boldsymbol{V}^n \in \mathbb{R}^{D\times\frac{\hat{D}}{N}}$ are the learnable project matrices of the n-th head.

Dense Neural Networks. The core idea behind dense neural networks [9] is to propagate low-layer features to higher layers by dense connections to capture hierarchical features and encourage feature reuse

$$\boldsymbol{h}^m = \sigma(\mathbf{Concat}(\boldsymbol{h}^1, \ldots, \boldsymbol{h}^{m-1})\boldsymbol{W}^m + b) \tag{9}$$

where $\sigma()$ is the activation function, $\boldsymbol{h}_m \in \mathbb{R}^{1\times D_m}$ is the m-th layer in the dense neural networks, $\boldsymbol{h}^1 = \mathbf{Concat}(\boldsymbol{x}_l, \hat{\boldsymbol{x}}_l)$ is the concatenation of the input and the output of the self-attention layer and $\boldsymbol{W}^m \in \mathbb{R}^{(D_1+\cdots+D_{m-1})\times D_m}$ is the parameter.

Then the the output of $\boldsymbol{x}_l$ in the LFE structure is defined as $\tilde{\boldsymbol{x}}_l = [\boldsymbol{h}_l^1, \ldots, \boldsymbol{h}_l^m] \in \mathbb{R}^{1\times\tilde{D}}$. Notice that the output of the LFE structure $\tilde{\boldsymbol{X}} = \{\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_L\} \in \mathbb{R}^{L\times\tilde{D}}$ is listwise.

3.4 Flight Itinerary Representation

Different from items in traditional e-commerce search systems, flight itineraries have a standard structure and the key information is shown in the listing page, where users can compare itineraries easily with the global context-aware information. To this end, we propose to capture the global context-aware information and model mutual influences between flight itineraries from two aspects: designing the global features manually and learning the feature representation by the LFE structure.

On the one hand, instead of only using the raw information directly, we design some extended features to model the global context-aware information. For example, "*ratio_to_cheapest*" defines the ratio of the price to the cheapest price in the inputs.

On the other hand, with the LFE structure, the flight itinerary representation can be defined as

$$\tilde{\boldsymbol{F}} = \mathbf{LFE}([\boldsymbol{f}_1, \ldots, \boldsymbol{f}_Z], \boldsymbol{W}_{LFE}^F, k_{max}^F) \tag{10}$$

where $\tilde{\boldsymbol{F}} \in \mathbb{R}^{Z\times\tilde{D}_F}$ represents the features of the flight itineraries.

3.5 User Preference Representation

As mentioned above, we propose to capture the user preferences from three aspects: long-term booking behaviors from a specific user, real-time clicking behaviors from this user and booking behaviors from group users sharing the same intention, which is shown in Figure 2. For individual user behaviors, user long-term booking behaviors reflect user's customs, while user real-time clicking behaviors reflect user's demands. Unfortunately, in tourism, individual user behaviors are sparse. Instead of only learning the preferences from specific user behaviors, we firstly predict the user intention, which is an easier problem and have been applied in many applications at Fliggy. By leveraging group user behaviors, the sparsity of user behaviors on flight itineraries can be alleviated.

Due to the limited space, here we only give an overview of the user intention prediction task at Fliggy. The intention prediction considers the user profile at Taobao and Fliggy (e.g., age, job, the location and the birth place), user behaviors on all kinds of items (e.g., hotels, the tickets of tourism attractions or visas) at Taobao and Fliggy, the query information, etc. The user intention is divided into several classes: travel, business, going home and other purposes.

Similar to works [10, 11, 18, 20] using the multi-head self-attention on user preference learning, we use the LFE structure to learn user preferences based on the global context-aware information from user sequential behaviors on itineraries. Given the user long-term booking behaviors $\mathcal{B}^u$, the user real-time clicking behaviors $\mathcal{R}^u$ and the group user booking behaviors O^u, we define $\tilde{\boldsymbol{Q}}_*^u \in \mathbb{R}^{L_*\times\tilde{D}_*}$ as the output of these behaviors for a specific user u in LFE structure, where $*$ represents "longB", "realC" and "groupB" for these three type behaviors respectively. It is worth noting that we should set $k_{max}^{groupB} = 0$ for group user behaviors since there are no interactive influences between any pair of behaviors.

Since the output of the LFE structure is listwise, we adopt an aggregation layer to obtain a single feature vector

$$\tilde{\boldsymbol{q}}_*^u = \boldsymbol{w}_*^T \tilde{\boldsymbol{Q}}_*^u \tag{11}$$

where $\boldsymbol{w}_* \in \mathbb{R}^{1\times L_*}$ is the aggregation parameter and $\tilde{\boldsymbol{q}}_*^u \in \mathbb{R}^{1\times\tilde{D}_*}$. With $\tilde{\boldsymbol{q}}_*^u$, we simply concatenate them as user preferences

$$\tilde{\boldsymbol{q}}^u = \mathbf{Concat}(\tilde{\boldsymbol{q}}_{realC}^u, \tilde{\boldsymbol{q}}_{longB}^u, \tilde{\boldsymbol{q}}_{groupB}^u) \tag{12}$$

where $\tilde{\boldsymbol{q}}^u \in \mathbb{R}^{1\times\tilde{D}_u}$ and $\tilde{D}_u = \tilde{D}_{realC} + \tilde{D}_{longB} + \tilde{D}_{groupB}$.

3.6 User Attention Layer

It is straightforward to concatenate the raw input of each flight itinerary $\boldsymbol{f}_z$ and the user preferences $\tilde{\boldsymbol{q}}^u$ as the inputs in Transformer-based models [7]. However, this operation cannot represent the interaction between users and itineraries directly, which is also time consuming since we should firstly obtain $\tilde{\boldsymbol{q}}^u$. In order to learn the interaction, Pei et al. [16] proposed to adopt a pre-trained model to calculate the personalized matrix. This method requires to train two models and it is time consuming. In PFRN, we propose a user attention mechanism, where one flight itinerary is chosen based on the user preferences. In the user attention mechanism, $\tilde{\boldsymbol{q}}^u$ is the query vector and $\tilde{\boldsymbol{f}}_z$ is the key vector. Note that all the attention methods [2, 13] can be used in this layer. For simplicity, we use the dot-product attention, which is similar to (5). The user attention mechanism is defined as

$$\beta_z = \tilde{\boldsymbol{q}}^u \boldsymbol{W} \tilde{\boldsymbol{f}}_z^T + w_z \tag{13}$$

where β_z represents the attention on $\tilde{\boldsymbol{f}}_z$, $\boldsymbol{W} \in \mathbb{R}^{\tilde{D}_u \times \tilde{D}_F}$ is the attention matrix and w_z is the position information parameter. More importantly, in this method, we can learn $\tilde{\boldsymbol{f}}_z$ and $\tilde{\boldsymbol{q}}^u$ independently and simultaneously to speed up the training and testing phases. Besides, if there are no user features, (13) is modified as

$$\beta_z = \boldsymbol{W} \tilde{\boldsymbol{f}}_z^T + w_z \tag{14}$$

Then the user attention score after the Softmax function can be viewed as the probability chosen by user u

$$p(y_z | \boldsymbol{F}, \mathcal{B}^u, \mathcal{R}^u, O^u) = \frac{\mathbf{exp}(\beta_z)}{\sum_{k=0}^{Z} \mathbf{exp}(\beta_k)} \tag{15}$$

In the training phase, we use the booking flight itineraries as labels and minimize the loss function

$$L = -\sum_{u \in \mathcal{U}} \sum_{z \in Z} y_z log(p(y_z | \boldsymbol{F}, \mathcal{B}^u, \mathcal{R}^u, O^u)) \tag{16}$$

where $y_z \in \{0, 1\}$ is the label of the z-th flight itinerary.

4 EXPERIMENTS

In this section, we evaluate the performance of the proposed PFRN extensively. Firstly, we compare the PFRN with several models on the offline datasets, then the online performance at Fliggy is reported. Finally, we present some real-world cases.

4.1 Datasets

4.1.1 Amadeus Airline Itinerary Dataset. In [15], Amadeus constructed an airline itinerary dataset which considered travel requests concerning round trips and medium-haul markets. In total, there are 33951 unique users in the dataset. However, there is no user information in the dataset.

4.1.2 Fliggy International Flight Itinerary Dataset . To evaluate the performance of the proposed method, we collect the user logs at Fliggy and construct the training dataset and the testing dataset. Similar to Amadeus Airline Itinerary Dataset, we only consider the users who booked the international flight itineraries. In order to collect the user logs, we design a link id from search to booking in order to track the behaviors at Fliggy.

The user long-term booking behaviors with the same destination are collected from the last 180 days but the max number of long-term booking behaviors are limited to 15. The real-time clicking behaviors are collected from the last 15 behaviors before a new search. In the online platform, the real-time clicking behaviors delay less than 1s. The intentions of users are recorded in the logs when a user searches for itineraries. The length of group user behaviors is limited to 15. The max number of candidate flight itineraries is 30. More details of the Fliggy dataset are described in Table 1.

Table 1: Overview of Fliggy dataset.

Attribute	Training	Testing
# bookings	916, 697	94, 213
# bookings with user long-term booking behaviors	160,189	17,019
# bookings with user real-time clicking behaviors	606, 435	62,924
# bookings with group user behaviors	750,992	77,174
# flight itineraries	6,627,594	717,495
# features of flight itineraries	20	20
# features on each type user behaviors	7	7

4.2 Baseline Methods & Evaluation metrics

4.2.1 Baseline methods . We compare the model with three classes of previous models in flight itinerary choice problem or traditional e-commerce ranking systems: a) the rule methods that usually adopted in OTAs, b) pointwise models and c) listwise methods.

- **Cheapest**: Ranking the flight itineraries by price.
- **Shortest**: Ranking the flight itineraries by duration.
- **Gradient boosting decision tree (GBDT)** [6] is widely adopted in industrial applications.
- **Multi-layer perceptron (MLP)** [4] is viewed as a baseline method of pointwise model .
- **DCM**[15] [6] adopts the long short-term memory (LSTM) module to capture the global information from the listwise input for airline itinerary choice prediction. The user preference are not considered in this method.
- **DLCM** [1] [7] is a deep listwise context model with gate recurrent unit (GRU) structure.
- **PRM** [16] adopts the standard Transformer for listwise reranking at Taobao. Here we need to train an additional model to learn the interaction between users and items. Then the user vector is concatenated with the raw item features directly as the input of the standard Transformer.
- **DIN** [21][8] is an improved version of MLP with user short-term history behaviors.

4.2.2 Implementation details. We train GBDT using 500 trees with a learning rate of 0.01. For neural networks-based models, the numerical features are normalized between $[-1, 1]$ and the categorical features are embedded into low-dimensional spaces. All the neural networks have five layers with the number of hidden nodes $\{128, 64, 32, 16, 1\}$. Similar to [15], the flight itineraries are ranked by price as the initial order for listwise models. The number of multi-head in self-attention is chosen from $\{2, 4, 6, 8\}$. For PFRN, k_{max} is chosen from $\{0, 2, 4, 8, 16, 32\}$ and the number of candidate flight itineraries. The learning rate is selected from $\{10^{-6}, ..., 10^{-2}\}$. Finally, we use Adam [12] to optimize all deep neural network-based models. The max training step is set as 10^6.

4.2.3 Evaluation metrics. Similar to [15, 16], we use the precision metrics P@1 ($Precision$@1) and P@5 to measure the effectiveness of the proposed PFRN and the other baseline methods. Note that if the P@1 is improved, it usually means that it is convenient for

[6]The code is avaliable download from https://amadeus.app.box.com/s/uv5ctxle5u5p1pysh5kiofgf4s88mxks
[7]The code is avaliable download from https://github.com/QingyaoAi/Deep-Listwise-context-aware-Model-for-Ranking-Refinement
[8]The code is avaliable download from https://github.com/zhougr1993/DeepInterestNetwork

Table 2: Results on offline datasets.

Dataset	Accuracy	Rule methods		Models without user features						Models with user features			Impr.
		Cheapest	Shortest	GBDT	MLP	DCM	DLCM	PRM	PFRN	DIN	PRM	PFRN	
Amadeus	P@1	16.43%	15.41%	23.15%	24.72%	25.38%	25.62%	25.68%*	**26.44%**	-	-	-	+2.1%
	P@5	45.84%	43.16%	61.72%	64.68%	66.35%	66.54%	66.72%*	**67.15%**	-	-	-	+0.6%
Fliggy	P@1	34.78%	43.27%	57.91%	59.28%	60.25%*	60.08 %	59.87%	60.79%	70.54%	72.94%	**74.12%**	+23.0%
	P@5	88.37%	91.38%	94.24%	94.59%	94.96%*	94.79%	94.93%	94.95%	96.44%	96.89%	**97.09%**	+2.2%

Table 3: Results of the PFRN with different user features.

Accuracy	PFRN	PFRN -NO	PFRN -UL	PFRN -UR	PFRN -UL-UR	PFRN -GU
P@1	74.12%	60.79%	62.16%	71.21%	72.74%	62.93%
P@5	97.09%	94.95%	95.31%	96.17%	96.46%	95.54%

Table 4: Performance of the user attention mechanism on the Fliggy dataset.

Method	*P*@1	*P*@5	The number of training steps per second	Total training time (s)	Total testing time (s)
PRM	72.94%	96.89%	–	4124	975
PFRN-CONCAT	72.88%	96.78%	96	2891	593
PFRN	74.12 %	97.09%	161	2147	420

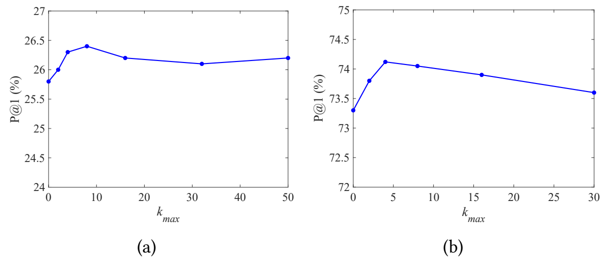

Figure 4: The influences of k_{max} on (a) Amadeus dataset and (b) Fliggy dataset respectively .

users to find the target flight itinerary and it could improve the user experience largely.

4.3 Offline Evaluation

4.3.1 Effectiveness of the LFE structure. The LFE structure adopts the multi-head self-attention with RPR and dense neural networks to capture global context-aware information, model mutual influences and learn hierarchical features. We firstly investigate the effectiveness of the LFE structure. In this experiment, both the Amadeus and Fliggy datasets for training and testing are without user behaviors. Hence, we use (14) to calculate the attention scores. The results are shown in Table 2. In Table 2, "Impr." means the improvement of the proposed PFRN compared with the best baseline method, which is marked with "*". From Table 2, we clearly find that the proposed method achieves the state-of-the-art results. The listwise models have better results than point-wise models, e.g., GBDT and MLP. Among all the listwise models, the proposed PFRN outperforms the other methods due to the LFE structure.

4.3.2 Effectiveness of user features. In order to evaluate the effectiveness of user features, we adopt the user features in three methods: PRM, DIN and PFRN. The results are presented in Table 2. "-" in Table 2 means that there is no user behaviors on Amadeus dataset. From Table 2, we find that methods with user features improve the *P*@1 largely compared with those without user features. For example, compared with MLP, DIN improves the *P*@1 from 59.28% to 70.54%. Notice that DIN is not suitable for fusing different types of user behaviors. Hence, we only uses the real-time clicking behaviors in DIN. The PRM improves the *P*@1 from 59.87% to 72.94%. Compared with PRM, the PFRN achieves better results due to the LFE structure. Compared with the best baseline method DCM without user features, the PFRN with user features improves the *P*@1 by 23% on the Fliggy dataset.

4.3.3 Model ablation of user features. We also conduct the experiments on the influences of different type user behaviors. Here, we define several model ablations: 1) PFRN-NO-U: the PFRN without any user features, 2) PFRN-UL: the PFRN with user long-term booking behaviors, 3) PFRN-UR: the PFRN with user real-time clicking behaviors, 4) PFRN-UL-UR: the PFRN with user long-term behaviors and user real-time clicking behaviors and 5) PFRN-GU: the PFRN with group user behaviors. The results are shown in Table 3. From Table 3, we find that the user real-time clicking behaviors improve the accuracy largely and the user long-term behaviors improve the accuracy slightly. The reasons may be: a) only 20% users have past booking behaviors on the same query and b) the user preferences may be varied. By introducing the group user behaviors, the *P*@1 is improved to 74.12%.

4.3.4 Effectiveness and efficiency of user attention mechanism. To justify the contributions of user attention mechanism, we test an ablation model called PFRN-CONCAT that concatenates the raw input features of flight itineraries and the user preferences in the input layer. Notice that we adopt a distributed implementation in training, which is described in more detail later, and run the testing phases on a single machine to simulate the online environment. The results are shown in Table 4. Since PRM requires to train two models, the number of training steps per second is not reported in Table 4. Comparing to PFRN, PRM is more time consuming due to two model training and serving. Without the user attention mechanism, PFRN-CONCAT has poor results due to lacking interactive influences from users and items. Furthermore, both the total training time and

Table 5: Performance improvements in online A/B tests.

Accuracy	Rule Method (Baseline)	DCM	PFRN
P@1	*	+6.6%	+23.8%
P@5	*	+0.5%	+2.3%
CVR	*	+1.4%	+3.9%

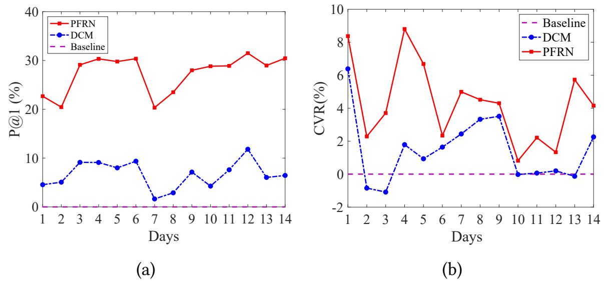

Figure 5: Online performances of PFRN compared with baseline in 14 days during February 2020.

the testing time of PFRN are reduced largely. Hence, the proposed PFRN not only achieves higher accuracy but also speeds up the training and testing phases.

4.3.5 Influence of k_{max}. Here, we give some analyses on the hyper parameter k_{max} in the LFE structure. The results of k_{max} are shown in Figure 4. It seems that the performance is firstly improved and then is decreased as k_{max} increases. The main reason may be that one page at Fliggy usually includes about 4 flight itineraries and users consider itineraries listed in one or two pages.

4.4 Online A/B Test

With the success in offline experiments, we deploy the PFRN in daily environment at Fliggy. In the online experiments, the baseline method is a rule method adopted in the previous system, and the rules are based on the expert experiences at Fliggy. Due to the limited users online per day, we only consider two methods: DCM and PFRN. DCM is proposed for flight itinerary choice problem in [15]. Furthermore, we consider an important metric CVR in online experiments to evaluate the methods

$$CVR = \frac{\#booking\ users}{\#all\ users} \tag{17}$$

The experiments are conducted during Feburary 2020 and the average improvements are shown in Table 5. Figure 5 shows the online results in 14 successive days. Considering the data confidentiality policy in Alibaba Group, we use "*" as the baseline to replace the absolute values at Fliggy online and report the improvements in Table 5 and Figure 5. From Table 5, we find that PFRN has a large improvement at $P@1$. The $P@1$ is improved by about 23% compared with the previous system. Furthermore, due to the high quality of user experiences, the CVR at Fliggy has been improved. Based on the A/B test results with p–value as 0.015, we deployed PFRN for all users at Fliggy on March, 2020.

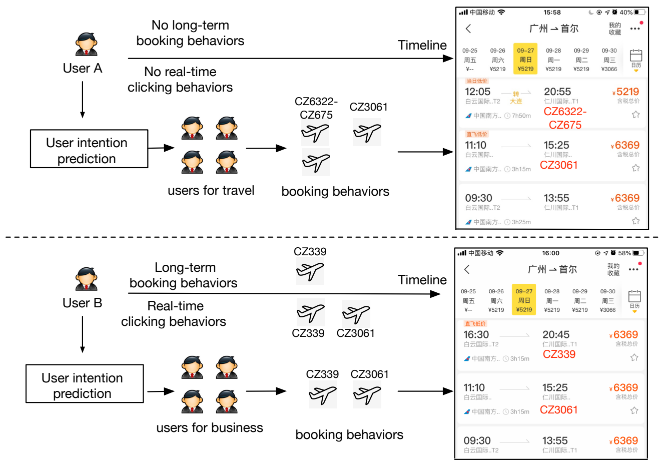

Figure 6: Two cases of the PFRN at Fliggy: user A (top) without any behaviors on flight itinerary and user B (bottom) with booking behaviors and real-time clicking behaviors.

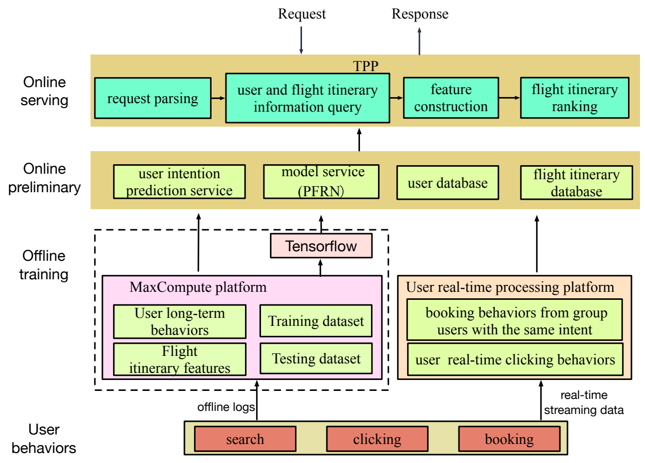

Figure 7: The workflow of the ranking systems at Fliggy.

4.5 Case study

We present two real-world cases to illustrate the effectiveness of the personalized flight itinerary system online, which are shown in Figure 6. User A is a user without any behaviors on flight itineraries. But we find that he searches for travel visas and tickets for scenic at Fliggy and Taobao, hence we predict his intention is for travel. Then we leverage group user behaviors for travel to rank itineraries. The flight itinerary "CZ6322-CZ675" including connecting flights has the lowest price and is chosen by many users for travel. Hence, we rank the flight itineraries "CZ6322-CZ675" at the first position, which is shown in Figure 6. Different from user A, user B had booking behaviors and clicked some flight itineraries in short-term. Furthermore, we find that he requested the invoice in his past bookings. Based on his behaviors, we predict his intention is for business. People for business prefer direct flight itineraries and have their preferences on the departure time and arrival time. Hence, direct flight itineraries and the clicked or booked itineraries are listed in front positions of the listing page at Fliggy.

5 SYSTEM IMPLEMENTATION AND DEPLOYMENT

In this section, we introduce the implementation and the deployment of PFRN at Fliggy. The workflow of the ranking system is shown in Figure 7.

5.1 Offline Training

In the offline training phases, we collected user logs and stored the logs on the MaxCompute platform, which is developed by Alibaba Group. Based on the logs, we constructed the training datasets and the testing datasets. Besides, we also created the table about user long-term booking behaviors and the table about flight itinerary features, which are prepared for online serving phases.

Considering large scale data on training, we conducted a distributed implementation in Tensorflow [9] to train PFRN. The parameter-server architecture of Tensorflow is adopted to provide a distributed solution for parameter storage, data-fetching and training. Specifically, we used 5 parameter servers along with 50 workers in this architecture. Each parameter server has 6 CPU cores with 32 GB memory, which stores part of parameters. Each worker is equipped with 6 CPU cores and 32 GB memory, which fetches part of training samples, calculates the gradients of parameters and then send the gradients to parameter servers. With the parameter-server architecture, the training time of the PFRN is less than 1 hour.

5.2 Online Serving

Before we deployed the PFRN online, we should prepare the user database, the flight itinerary database, the model service and the user intention prediction service.

When a user searches for flight itineraries, the server sends a request to TPP, which is an online service platform in Alibaba Group. The serving workflow in TPP is described as follows:

- Request parsing: TPP firstly explains the request to get the user id and the basic information of flight itineraries (e.g., the flight itinerary id and the real-time price).
- User and flight itinerary information query: With the user id, we run the intention prediction task to obtain the intention. Besides, we get the user behaviors and the group user behaviors from user database. We also get the flight itinerary information from flight itinerary database. Notice that all queries are simultaneous to speed up the serving phases.
- Feature construction: We preprocess the user information and flight itinerary information to construct the input.
- Flight itinerary ranking: With the prepared input, we request the model service to get the scores.

The average response time of the flight itinerary ranking system at Fliggy is about 20ms, which satisfies the requirements of online environment at Fliggy. Furthermore, the average time of running model is less then 5ms while the message transmission between different platforms by networks costs a lot of time.

[9]https://www.tensorflow.org/

6 CONCLUSION AND FUTURE WORK

In this paper, we propose a PFRN to rank flight itineraries. In PFRN, a LFE structure is proposed to extract global context-aware information, model mutual influences and learn hierarchical features. Besides, group user behaviors and individual user behaviors are utilized to alleviate the sparse phenomenon. Then an user attention mechanism is proposed to rank flight itineraries based on the user preferences. Offline experiments and online experiments have shown the effectiveness and the efficiency of PFRN. In the future, we propose to introduce the diversity in the flight itinerary ranking system because similar flight itineraries are potentially listed on one page. In this case, the exposure of similar flight itineraries is redundant and it cannot satisfy diverse demands of users.

REFERENCES

[1] Qingyao Ai, Keping Bi, Jiafeng Guo, and W Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. In *SIGIR*. 135–144.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[3] Gaby Budel, Lennart Hoogenboom, Wouter Kastrop, Nino Reniers, and Flavius Frasincar. 2018. Predicting User Flight Preferences in an Airline E-Shop. In *ICWE*. Springer, 245–260.

[4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[5] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860* (2019).

[6] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[7] Yu Gong, Yu Zhu, Lu Duan, Qingwen Liu, Ziyu Guan, Fei Sun, Wenwu Ou, and Kenny Q Zhu. 2019. Exact-k recommendation via maximal clique optimization. In *KDD*. 617–626.

[8] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time personalization using embeddings for search ranking at airbnb. In *KDD*. 311–320.

[9] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *CVPR*. 4700–4708.

[10] Xiaowen Huang, Shengsheng Qian, Quan Fang, Jitao Sang, and Changsheng Xu. 2018. Csan: Contextual self-attention network for user sequential recommendation. In *MM*. 447–455.

[11] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *ICDM*. IEEE, 197–206.

[12] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[13] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[14] Fuyu Lv, Taiwei Jin, Changlong Yu, Fei Sun, Quan Lin, Keping Yang, and Wilfred Ng. 2019. SDM: Sequential deep matching model for online large-scale recommender system. In *CIKM*. 2635–2643.

[15] Alejandro Mottini and Rodrigo Acuna-Agost. 2017. Deep choice model using pointer networks for airline itinerary prediction. In *KDD*. 1575–1583.

[16] Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Junfeng Ge, Wenwu Ou, et al. 2019. Personalized re-ranking for recommendation. In *RecSys*. 3–11.

[17] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. *NAACL* (2018).

[18] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from Transformer. In *CIKM*. 1441–1450.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*. 5998–6008.

[20] Shuai Zhang, Yi Tay, Lina Yao, Aixin Sun, and Jake An. 2019. Next item recommendation with self-attentive metric learning. In *AAAI*, Vol. 9.

[21] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *KDD*. 1059–1068.