Travel Package Recommendation Based on Reinforcement Learning and Trip Guaranteed Prediction

Jui-Hung Chang[1], Hung-Hsi Chiang[2], Hua-Xu Zhong[3], Yu-Kai Chou[2]

[1] Computer and Network Center, and Department of Computer Science and Information Engineering, National Cheng Kung University, Taiwan
[2] Department of Computer Science and Information Engineering, National Cheng Kung University, Taiwan
[3] Department of Engineering Science, National Cheng Kung University, Taiwan
changrh@mail.ncku.edu.tw, dennisps999@gmail.com, k43122003@gmail.com, b94dg010@gmail.com

Abstract

Trip planning research and travel package recommendation benefit from current trends in Location Based Social Networks and trajectory related sites nowadays. Travel package recommendation requires the extraction of characteristics of points of interest and setting up a ranking method. Traditional research used to rely on questionnaires without statistical validation methodologies. We proposed a recommendation framework based on reinforcement learning. To reach the objective of generating successful travel packages, we introduced a reward function for ranking points of interest. Based on labeled travel package data provided by travel agencies, two trip guaranteed prediction methods (deep learning and trajectory similarity) were used for travel guarantee prediction. The results of the accuracy and performances of these methodologies showed the prediction models are reliable. We found no statistically significant difference between the recommended and the uncancelled package groups.

Keywords: Reinforcement learning, Recommendation system, Deep learning, Neural network, Trajectory similarity

1 Introduction

1.1 Background and Motivation

POIs (Points of Interest) is a fundamental current concept in social media: it refers to places that visitors (users of social media software and hardware) may be interested in. They are usually stored in terms of spatial geo-coordinates. In the past few years, various LBSNs (Location-Based Social Networks) have been established. People can not only explore the features of different places, but also share their experiences in the form of reviews through platforms such as Foursquare, Flickr, and Google.

Along with the rapid development of LBSNs, trip planning services sprang into existence. They help users find places of interest at unfamiliar places. Travel package recommendation is an extension of POI recommendation in a systematic form. Several POIs form a traveling route, which satisfy the users' preset conditions.

Travel package recommendations fall into two categories. (1) Trajectory recommendations recommend routes. However, raw trajectory data is less often sourced by websites, and currently they refer to mostly bike routes [1]. (2) Travel packages are based on POI recommendation. Their methodology is based on collecting user preferences and ratings of POIs. They also use a ranking technique to suit POIs for a trip, and arrange zthem into a sequence [2]. The main difficulty is to choose the POI characteristics for evaluation, and to weight them among one another. On the contrary, the validation of these travel packages are mostly done by subjective questionnaires, lacking precision and objectivity. Last but not least, the current methodologies only aim at providing day trip recommendations. They do not work for long trips like visiting a country.

According to AlphaGo, RL (Reinforcement Learning) may be a powerful solution for gaming and smart scheduling developments, while it may also be applicable for the generation of travel packages. RL grabs the relationship between the agent and environment. The agent evaluates the environment and the relationship between different states by defining a reward function. The reward function of the RL framework can help ranking the POIs. In each iteration of agent learning, a sequence of states (POI) can be retrieved, which can be used as a recommended travel package.

This article aims to guarantee the accuracy of trip recommendation by the use of RL framework. The accuracy of the recommended packages can be verified by calculating the average cancellation rate of similar

*Corresponding Author: Jui-Hung Chang; E-mail: changrh@mail.ncku.edu.tw
DOI: 10.53106/160792642021112206013

packages. For travel agencies, prediction of a travel package will be canceled or not will be a significant fact. Questionnaire methodologies may be useful and common in many kinds of recommendation system, however it may be difficult to collect enough samples of the travelers who can really test the package in real and the method is also not convincible enough to prove if the package can be accepted by the crowd or not. In that case, we propose two models to deal with the trip guaranteed prediction problem in the future. (1) Deep learning based model: Trip Prediction Network (TPN). We proposed it for predicting the probability of canceling packages (2) the trajectory similarity model. Which aims to search the travel package data set and retrieve similar label packages. The accuracy of the recommended packages can be verified by calculating the average cancellation rate of similar packages.

1.2 Objective and Contribution

The contributions of this paper are listed as follows:

(1) Based on reinforcement learning, a travel recommending framework is proposed. We define a reward function to evaluate and rank POIs in order to recommend travel packages.

(2) We use the TPN and the Trajectory Similarity models to predict if the transaction will be made positive or negative among the recommended travel packages.

(3) The experiments were taken on real dataset and further discussions were given.

1.3 Organization

Section 2 reviews the related works. In Section 3, an overview of Reinforcement learning framework and model design are given. The mathematical and informatic details of the recommendation system and the success prediction models are provided in Section 4. Section 5 describes the experiments of case studying and several feature tunings of the validation step. We include a case study of the proposed model along with performance and precision evaluations. Finally, in Section 6, suggestions for future work are mentioned.

2 Related Work

Reinforcement learning (RL) become a popular issue nowadays, the algorithm can learn pattern by each iterations and make decisions from various states. By beating the world champion of the game of Go, the Alpha-Go program has gain attention around the world based on the algorithm of RL [3]. RL defines a class of algorithms solving problems modeled as a Markov Decision Process (MDP), which can be denoted by the notation list [S, A, $\boldsymbol{P}$, r, γ, $\boldsymbol{T}$], notations are shown as follow Table 1 :

Table 1. Markov Decision Variable symbols and their definitions

Symbol	Meaning
S	The set of states
A	The set of actions
$\boldsymbol{P}(\boldsymbol{s})$	The policy function to decide the next state s' and action a at state s, where s, $s' \in$ S and $a \in$ A
r	The reward function, was set to evaluate the reward of a pair of action and state
γ	The discount factor that specifies how much long term reward is kept
$\boldsymbol{T}$	The termination condition which decides if an episode is terminated

After giving the definition of the basic notations, Q-learning is a baseline reinforcement learning algorithm, which lets an agent learn the patterns of an environment and record the experience and the reward in the Q-table.

In each episode, an initialization state $\boldsymbol{s}$ was given and the agent continued choose action a by the policy P, then the reward r and next state 's' was evaluated. Meanwhile, $\boldsymbol{Q}$ is the Q-table while $\boldsymbol{Q}(\boldsymbol{s}, \boldsymbol{a})$ is the possible reward by acting action $\boldsymbol{a}$ at state $\boldsymbol{s}$. $\propto$ is the learning-rate of the updating, and γ is the maximum expected reward for the next state, while α is the evaluated reward of Reinforcement learning is fundamental for AI (artificial intelligence) constructions [4]. Recently, biomedical, system security, and Internet of Things (IOT) research domains applied RL. The applications included biomedical data [5], Smart Grid construction, electronic product error detection, and power system related improvements [6]. System security problems can be detected [7]. The RL technique may also be used in Smart City applications, such as indoor positioning [8].

Many innovations and researches on RL aimed on the design of different reward functions in order to work on different tasks. For instance, if the work performed by the agent is closer to human life, there is a need to understand the legal norms. If the agent went to the pharmacy to buy medicine, it is stolen without purchasing. The paper proposed the concept of ethical reward based on moral data. A goal can also be achieved without ethical norms, such as a driverless car can dodge the animals on the road while driving [9]. RL structures were also designed for multi-agent tasks such as the road mass transit system [10]. The environment state space will be compressed if there are multiple agents, and the agent's own reward will appear. The paper also proposed the notion of difference reward based on the environment state. Compared with a general reward target (shortest travel time), a multi-agent task is to maximize the average reward and to minimize system load [11]. The role of a human reward function is to bring the agent closer to human actions. This can be applied to robotic or

android design such as arm movement [12]. Some studies raised the idea of replacing the action field by a Continuous Action Field. This can enhance the reality of simulated walking [13]. Last but not least, multi-agent RL framework was also applied to tackle the feature selection problem [14].

Recently, the trend of research and implementation on has changed, researchers tend to focus on collaborative techniques. Rather than letting the trained machine agent work in single, Human-centered collaborative robots may also be trained by Reinforcement learning techniques, which can provide more fluent coordination between human and robot partners than baseline algorithms [15]. As to multi-agent tasks, sample efficiency and learning robust policies may be significant since difficulties may existed while learning in a dynamic environment, by recurrent neural network-based actor-critic networks and deterministic policy gradients techniques, the agents weights may be trained centralized and be more reasonable [16], which may also be useful and implementable on medical issues Functional Electrical Stimulation [12] or Large-scale Traffic Signal Control [17]. Finally, in order to tackle the time consuming problem of deep reinforcement learning, the methodology of broad reinforcement learning was proposed and tested well on fast autonomous IoT [18].

Thus, talk about Package recommend issue. Trajectory recommendation systems depend on GPS (Global Positioning System) study proposed a PATS recommendation system for climbers and cyclists. This system considers the traversal behavior of the particular region of interests (ROI) and adds the user preferences for the recommendation [1]. However, the difficulty in trajectory recommendation lies in the data collection. LBSN also provides open trajectory data bases such as Bike-map and Every-trail. However, it is still relatively difficult to obtain a large amount of data.

POI popularity ranking can be accomplished by public data such as POI check-in records from LBSN [2]. [19] Proposed crawling and parsing the Travelogue data, Topic package to classify the POIs into different topics and sub-topics. This topic classification model added the users' choices and POI check-in data to customize travel packages. A similar model called TRAST (tourist-relation-area-season topic) mined characteristic features from historic data (area and season) for effectively forming travel groups [20].

In recent years, the neural network technique has been enriched by the Backpropagation algorithm with automatic training weights. This enables to find the most suitable proportion of input data for achieving high accuracy predictions. Neural networks became popular for tourism demand and travel time prediction.

Tourism demand research aims at predicting passengers and tourism. The training data comes from information of the Tourism Bureau, news data, and Google Trend [21]. Some feedforward artificial intelligence neural networks (ANNs) use different input functions and hidden nodes to test and predict travel time series dedicated ANNs have better output performance than model [22]. The incorporation of seasonality and volatility are important influencing the choice between linear and non-linear models [23-24]. [25] Used ANNs to model and predict tourism demand in Mozambique from January 2004 to December 2013. They included the number of overnight stays at hotels, which is representative of tourism demand. High-precision predictions are also important for planning economic activities and development. [26] has been observed that the popularity of tourism keywords on Google was highly correlated to the number of Taiwanese tourists traveling to Japan through ANNs model to analyze.

[27] Proposed a neural network model for improving the prediction accuracy of recommendation systems. Their neural network was trained using a simulated annealing algorithm integrated with two single-level recommendation system samples. The study presented experimental results using two single scoring techniques and their corresponding neural network models. They compared the performance of each single technique with that of the proposed multi-standard model. The combined model was superior to.

As to including public transportation in travel plans, recurrent neural network (RNN) offers a time consuming solution with high accuracy. DeepSTCL (Spatio-temporal ConvLSTM) and Gated recurrent unit neural networks (GRUNN) experimented by bus data. The baseline model for comparison is ARIMA (Autoregressive Integrated Moving Average model) [28-29]. Besides, passenger profiling may also be part of tourism demand, while fuzzy deep Boltzmann machine outperformed the existing baseline profilers [30].

Travel time prediction is actually based on shallow learning architectures without feature learning capability. Deep belief network (DBN) can achieve higher precision when high dimensional features [31]. Based on LSTM-DNN (Long Short-Term Memory Deep Neural Network), short-term travel time was predicted from maintained by Caltrans Performance Measurement System (PeMS) and the results were fine-tuned with 16 different parameter settings towards the highest accuracy [32].

Furthermore, time-series data related issue can be well evaluated by RNN based models, such as fusing CNNs and LSTMs on express data [33], applying minimal gated unit on RNN [34], and prediction continuous travel time for traffic signal priority (TSP) systems [35].

3 Travel Package Recommendation

3.1 Structure Overview

Our travel package recommendation framework is based on reinforcement learning. An agent visits and travels through states and does traveling actions by moving from a POI to another to learn the environment. The limit iteration step is when iteration stops.

The schema of the proposed framework is shown in Figure 1 Data collection was based on travel package datasets provided by travel agencies. Word segmentation and extraction were applied to collect check-in data. The POI related data were retrieved by web crawlers such as the LBSN based API. The agent is making actions in the environment based on Epsilon greedy technique. The Prediction models box contains two models: Trip Prediction Network and Trajectory Similarity.

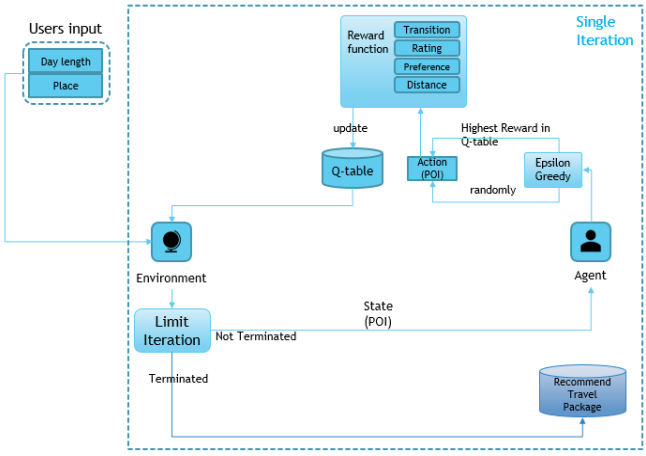


Figure 1. Framework structure overview

3.2 Reinforcement Learning Model

The schema of the reinforcement learning model is displayed in Figure 2 Users provide a day length and the place they want to travel to as input. In the initial state of each training iteration, the Environment may be at a random State (POI). The agent can apply an action to move on to the next State decision making is based on the Epsilon greedy policy. After an action is decided, the Reward of the Action is evaluated and updated in the Q-table. Limit iteration decides when iteration is terminated. Since the Q-table records the experience after the Agent's actions, many iterations are needed to train the Q-table. Limit iteration is explained in Sec. 3.3, and the Reward function design is depicted in Sec. 3.4. The collection of variables used in this paper is displayed in Table 2. The algorithm of our reinforcement learning framework is as follows:

Algorithm 1. Limit iteration

Input: Day_{Total}

1. $Day_{Travel} = Day_{Total} - Day_{flight}$
2. For each Day in Day_{Travel}:
3. $\quad Time_{daily} = 9$
4. $\quad$ While $Time_{daily} > 0$:
5. $\quad\quad Time_{daily} = Time_{daily} - Time_{POI} - Time_{Transport}$
6. $\quad\quad$ If $Time_{daily} \le 0$:
7. $\quad\quad\quad$ Break While
8. End For
9. $Limit = True$

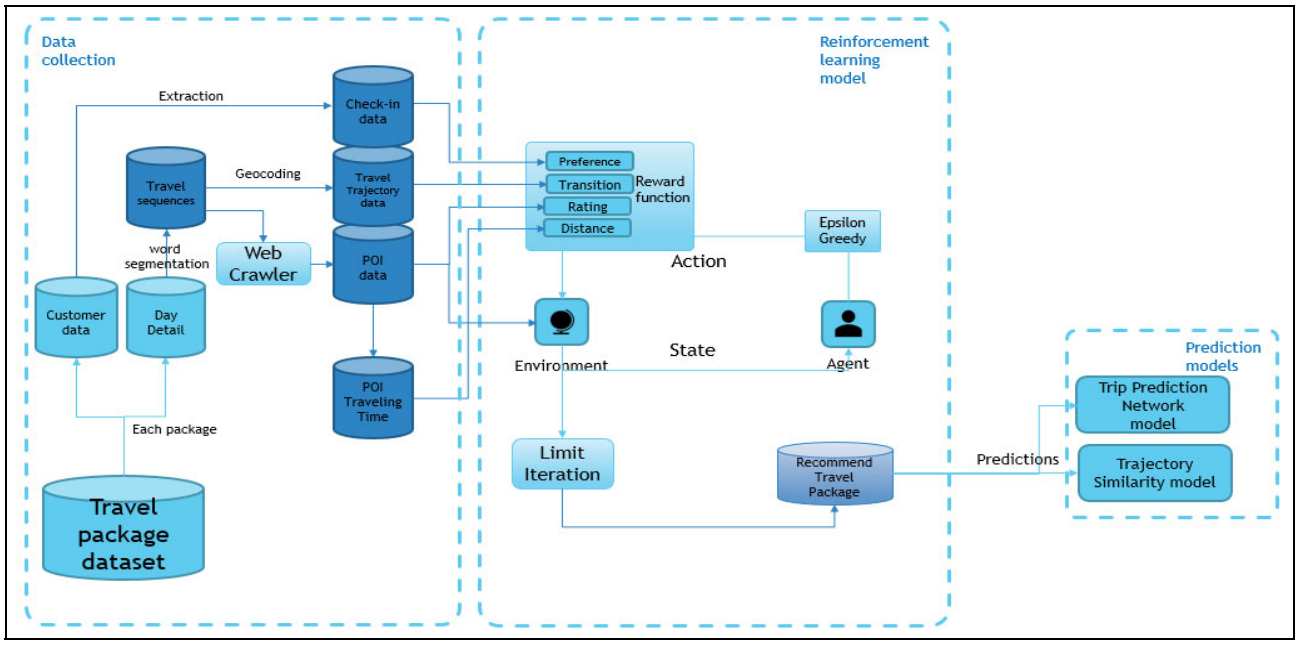


Figure 2. Reinforcement learning model

Table 2. Variable symbols and their definitions

Symbol	Meaning
S	The set of states
T	The termination condition for each episode
$\in$	The threshold of epsilon greedy
Day_{Travel}	The actual number of days spent by traveling
Day_{Total}	Total day of the travel package inputed by user
Day_{flight}	The number of days spent by flying
$Time_{POI}$	The time (hours) spent by visiting *POIs* per day
$Time_{daily}$	The time (hours) spent by travelling between and visiting *POIs* per day
$Time_{Transport}$	The total time (hours) spend by transportation (hours)
l	Grid length
G	The set of all grids g_i
g_i	The ith grid after grid slicing
θ	The threshold for ROI (Region of interest) determination
RD	The set of *ROIs*
R_i	The ith *ROI* in *RD*
TRD	The trajectory dataset generated from travel packages
TR_i	The ith trajectory in *TRD*
p_{ij}	The *jth POI* in TR_i
$\propto$	The Learning rate of Markov chain
$S^i(R_i)$	The transition score of R_i after ith iteration
$S(R_i)$	The final transition score of R_i
$V_i(S, A)$	Original values for evaluating the reward $R_i(S, A)$
$R_i(S, A)$	The normalization of $V_i(S, A)$
$R(S, A)$	The reward function constructed from $R_i(S, A)$
(t, p, d, r)	(transition, preference, distance, rating)
P	Preference matrix
P_{ji}	The value in P at the jth row and ith column
D	The User-Item matrix of check-in data
N	The pairwise distance matrix of P
∂	Spatial threshold for distance similarity evaluation

The domains S and A are the set of POIs. The Q-table is a $n \times n$ zero matrix initially. For each episode, the initial state is a random POI from S. The While loop may choose an action by epsilon greedy as follows:

$$\begin{aligned} &\varepsilon - greedy = \\ &Random(1, 0)\{< \varepsilon \\ &\rightarrow random\ action\ else \\ &\rightarrow choose\ action\ a\ in\ max(Q(s, a)) \end{aligned} \quad \textbf{(1)}$$

$$While\ \varepsilon = \varepsilon' \cdot \Delta\varepsilon \frac{2 \cdot recent\ iteration}{Max\ iteration}$$

Here $\varepsilon' = 1$ is the initial value of ε

After choosing action a, the s' is determined, and the reward r is evaluated by the reward function (R). The Q-table is updated in line 10. The While loop is terminated when the limit iteration (Limit) becomes True.

3.3 Limit Iteration

$Time_{daily}$ is defined as the duration between 9 am and 18 pm. The starting point of 9 am signifies the end of the breakfast, while the ending point of 18 pm signifies the beginning of the dinner. Meanwhile, Day_{Travel}, which is the traveling days of the trip is defined as:

$$Day_{Travel} = Day_{Total} - Day_{flight} \quad \textbf{(2)}$$

To terminate the travel part of a day, we used the following inequality:

$$Time_{daily} \leq \Sigma Time_{POI} + \Sigma Time_{Transport} \quad \textbf{(3)}$$

When the RHS of the inequality first exceeds the LHS, we cannot pack more program or travel (including the last proposed item) into the daily plan. Obviously, the opposite direction of the inequality is maintained during the whole day.

The termination of the iteration takes place after the termination of a day happened Day_{Travel} times. At that point, the POIs (states) visited by the agent can be retrieved.

The algorithm of Limit iteration is as follows Algorithn 1. Figure 3 shows an example of Limit iteration evaluation. In the figure, the episode terminates, and the visited POIs are retrieved as a package.

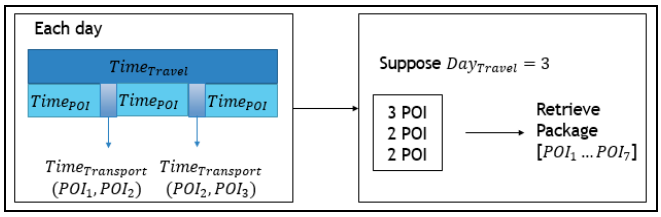


Figure 3. Example of limit iteration evaluation

3.4 Reward Function Design

The reward function is enabled with the capability to choose popular POIs. It is formed by 4 components: Transition score, Item preference, Distance, and Rating.

(1) Transition score

The transition score comes from the historic trajectory database. It symbolizes the travel behavior of earlier visitors. POIs with higher likelihood of being visited have higher transition score. The transition score evaluation is shown in Figure 4.

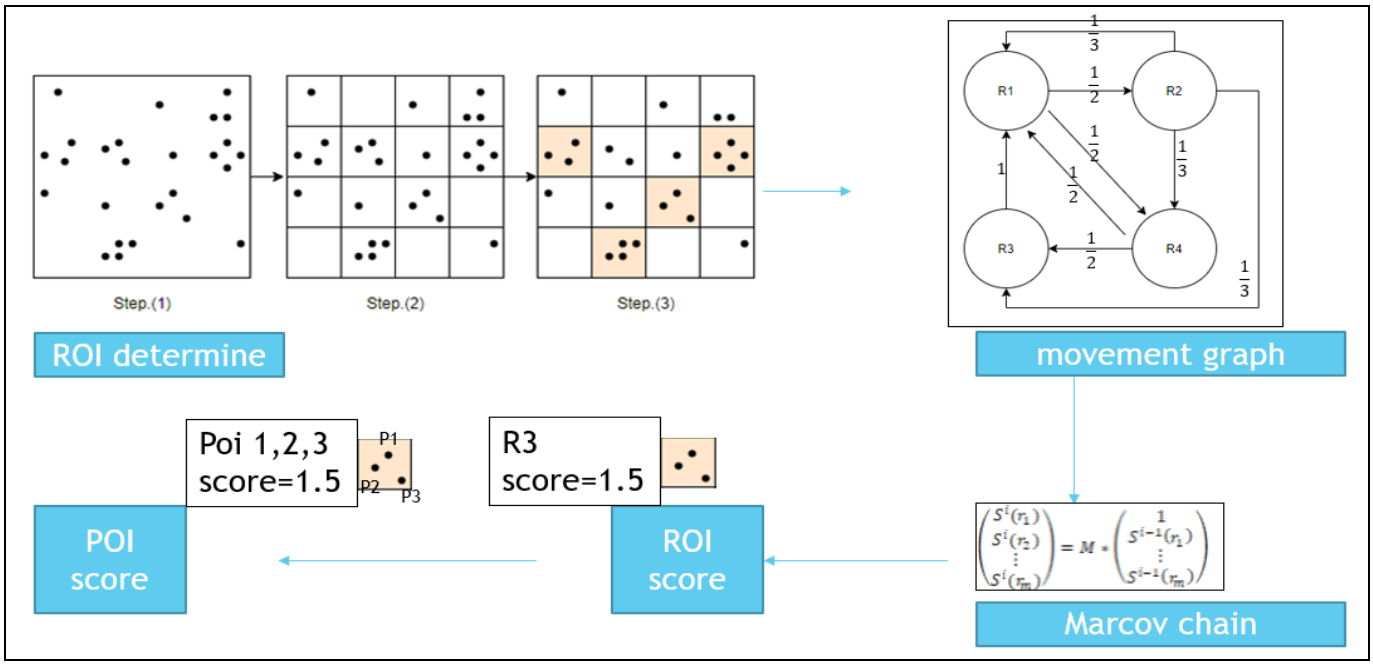


Figure 4. Transition score process

(a) ROI determination

The schema of ROI determination is shown in Figure 5 Given a spatial map region, we can mark the POIs in the trajectories dataset on the map. After slicing the map with a grid length of *l*, a sequence of grids G = $[g_1, g_2, g_3, \ldots, g_{16}]$ is generated. Let θ denote an ROI threshold. If for every $g_i \in$ G, the number of POIs in g_i is higher than θ, g_i is determined as a ROI $R_i \in RD$, and *RD* is the ROI set. After the determination of ROIs, each trajectory Tr_m is converted into an ROI sequence TR_m. Figure 4 provides an example. When considering the trajectory $Tr = [p_1, p_2, p_3, p_4, p_5]$, p_1 and p_5 don't belong to any RO I, while p_2, p_3, p_4 belong to R_1, R_2, R_3 respectively, so the ROI sequence *TR* converted from *Tr* has the form of $TR = [R_1, R_2, R_3]$.

(b) Movement graph

The movement graph is constructed from the traversal relationships between the ROIs. Table 3 illustrates an ROI sequence dataset. Figure 6 displays the movement graph constructed from Table 3.

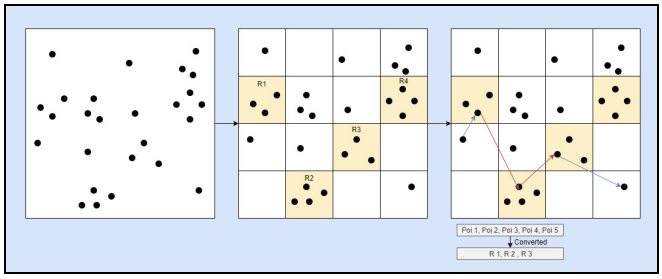


Figure 5. ROI determined

Table 3. ROI sequence example

Sequence name	Sequence content
Tr_1	$R_1 \rightarrow R_2 \rightarrow R_3$
Tr_2	$R_2 \rightarrow R_4 \rightarrow R_1$
Tr_3	$R_3 \rightarrow R_1 \rightarrow R_2$
Tr_4	$R_2 \rightarrow R_1 \rightarrow R_4 \rightarrow R_3$

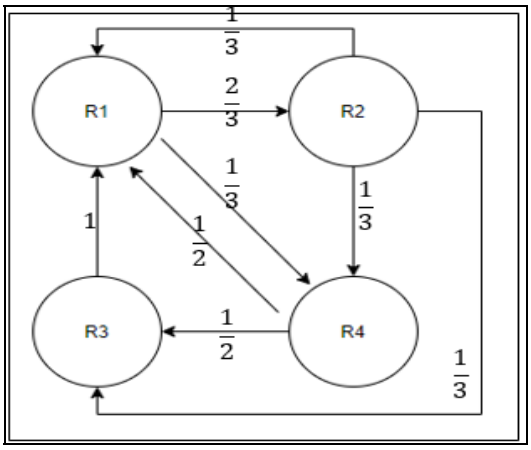


Figure 6. User movement graph

The weight evaluation formula of the graph is as follows:

$$w_{<R_i, R_j>} = \frac{\Sigma_{Tr_h \in TRD, <R_i, R_j> \subset Tr_h} \frac{1}{\deg^+(R_i \mid Tr_h)}}{|\bigcup_{R_l \in R, R_i \neq R_l} \{Tr_k \mid Tr_k \in TRD, <R_i, R_l> \subset Tr_k\}|} \quad (4)$$

Note that $w_{<R_i, R_j>}$ is the weight of the edge $<r_i, r_j>$. The individual outer degree of an ROI in a trajectory is defined as follows:

$$deg^+(Tr) = |\{R_j \mid <R_i, R_j> \subset Tr, R_i \neq R_j\}|$$

We can demonstrate [what] by taking $w_{<R_1, R_2>}$ as an example. The denominator [of what] will be the number of trajectories including the edge $<R_1, R_l>$. Note that $R_l \in R$ is an arbitrary element in the ROI set R. The denominator becomes $|Tr_1, Tr_3, Tr_4| = 3$, the numerator of (4) become

$$\sum_{Tr_h \in TRD, <R_1, R_2> \subset Tr_h} \frac{1}{\deg^+(R_1 \mid Tr_h)} = \frac{1}{\deg^+(R_1 \mid Tr_1)} + \frac{1}{\deg^+(R_1 \mid Tr_3)}$$

Since only Tr_1 and Tr_3 include the edge $<R_1, R_2>$ $\frac{1}{\deg^+(Tr_1)} + \frac{1}{\deg^+(Tr_3)} = \frac{1}{1} + \frac{1}{1} = 1 + 1 = 2 \frac{1}{\deg^+(Tr_1)} + \frac{1}{\deg^+(Tr_3)} = \frac{1}{1} + \frac{1}{1} = 1 + 1 = 2$. Finally, $w_{<R_1, R_2>}$ can be obtained by dividing the numerator by the denominator, i.e., w_<R1, R2> $= \frac{2}{3}$.

(c) Marcov Chains

We used Marcov chains to evaluate weighted graph that presents the traversal relationships of ROIs. This principle is similar to the Page-rank. The common feature is that the chain and state may finally come to a stable situation. We can present the transition matrix M as:

$$M = \begin{pmatrix} (1-\alpha)\alpha(w_{<R_1, R_1>}) & 1-\alpha)\alpha(w_{<R_1, R_2>}) & \cdots \\ \alpha(w_{<R_m, R_1>}) & \alpha(w_{<R_m, R_2>}) & \vdots \\ 1-\alpha)\alpha(w_{<R_1, R_m>}) & \cdots & \alpha(w_{<R_m, R_m>}) \end{pmatrix} \quad (5)$$

Here, the parameter $\alpha \in [0, 1)$ is the learning rate, $w_{<R_i, R_i>} = 0$ for all $R_i \in RD$, and for each iteration of the Marcov chain process, we have:

$$(S^i(R_1) \vdots S^i(R_m)) = M \cdot (1 \vdots S^{i-1}(R_m)) \quad (6)$$

Here, $S^i(R_i)$ is the transition score of R_i after the *ith* iteration, with $S^0(R_i) = 1, \forall R_i \in RD$. The iteration is terminated when:

$$(S^i(R_1) \vdots S^i(R_m)) = (S^{i-1}(R_1) \vdots S^{i-1}(R_m)) \quad (7)$$

In this case, the score of each ROI is stabilized, and $S(R_i) = S^i(R_i)$ is the transition score of R_i. Recall that the transition score of POI p_i is defined as:

$$V_t(s, p_i) = S(p_i), A \in R_i \quad (8)$$

(2) Item preference

The concept of item preference is based on collaborative filtering. We attempt to evaluate the personal preference for an item from the user check-in data. The check-in data is extracted from the customers' data and the POI they visited. There are 16175 users and 400027 check-in items in total. The preference matrix is defined as:

$$P = \frac{D \cdot N}{\sum_{j=1}^{m} N_{ij}} \quad (9)$$

Here, D is the User-Item matrix of check-in data, N is the matrix obtained from D by cosine similarity, and the denominator is the ith column sum of N. The size of P is $m \times n$, m is the number of POIs, and n is the number of users. In the preference matrix P, P_{ij} is the preference score of user i towards POI j. For POI j, the preference of all users is:

$$V_p(p_j) = \sum_{j=1}^{N} P_{ji} \quad (10)$$

(3) Distance and Rating

The distance between POIs and their rating data were collected from the Google API. The physical distances between pairs of POIs were calculated by Google's direction API, while the rating data were evaluated by Google's place API.

Our distance score from S to A is defined as:

$$V_d(S, A) = -dist(S, A) \quad (11)$$

Where $dist(S, A)$ is the road travel time between S and A. The rating score from S to A is defined as:

$$V_r(S, A) = rating(A) \quad \textbf{(12)}$$

(4) Normalization

Now, we can define the reward function value of State S doing Action A as:

$$R(S, A) = \frac{R_t(S, A) + R_p(S, A) + R_d(S, A) + R_r(S, A)}{4*3} \quad \textbf{(13)}$$

Here, $R_i(S, A)$ for $i \in (t, p, d, r)$ is the normalization of $V_i(S, A)$:

$$R_i(S, A) = \frac{V_i(S, A) - \mu_i}{\sigma_i} \quad \textbf{(14)}$$

The mean value and the standard deviation of all. According to the rule about standard normal distribution---99.73% of the values lie within three standard deviations of the mean--- we set the values further than that as 3 after normalization that is:

$$if\ R_i(S, A) > 3 \rightarrow R_i(S, A) = 3 \quad while\ i \in (T, P, D, R) \quad \textbf{(15)}$$

The denominator of $R(S, A)$ is set as 4*3, where the factor 4 comes from the 4 components in the reward function, and the factor 3 comes from the [-3, 3] domain of $R_i(S, A)$. This way, $R(S, A)$ falls into the interval [-1, 1].

4 Prediction Model

We obtained labeled travel package datasets from travel agencies due to the industry school corporation. We can use the historic cancellation data from these databases to make predictions about the cancellation probabilities of our generated travel packages. We used two models, TPN and Trajectory Similarity for a classifier, and predict the recommend travel package by our fine-tuned models.

4.1 Trip Prediction Network

For travel agencies, the most significant characteristic of a travel package is whether it is accepted by the customer or canceled in the end. The extracted features from the labeled travel packages of the travel agencies' data formed a deep artificial neural network, see Figure 7 According to Table 4, there were 7 kinds of extracted features for model training in a total of 66 dimensions. Since one hot encoding was applied on the Month and Season features, we had to remove the Money and Date features in order to gain better precision. Meanwhile, the unit of money is U.S dollars and the Type features were collected from the type of Google POI, there are 69 types in total but only the most relative ones to our topic were left, so 46 Type features in the end.

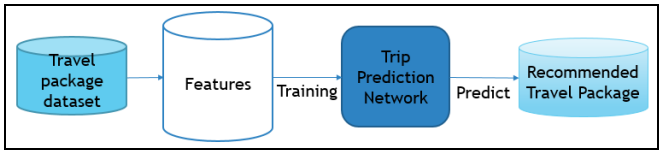


Figure 7. Trip prediction network

Table 4. Table of used features

Feature name	Dimension	Remarks
Money	1 dimension	Removed
Date	1 dimension	Removed
Days	1 dimension	
Customers	1 dimension	
Type	46 dimensions	
Month	12 dimensions	One hot encoding
Season	4 dimensions	One hot encoding

The network structure schema is shown in Figure 8 there are 5 hidden layers, and the number of neurons in each layer is a power of 2. The network holds more neurons in the middle than at the edges (Input and Output layers). We applied batch normalization on each layer, and we set ReLU as the activation function for each layer except the Output, for which Softmax was set. We used the Adam optimizer for training the model and chose the Loss function as the reduced mean of sparse Soft-max Cross Entropy.

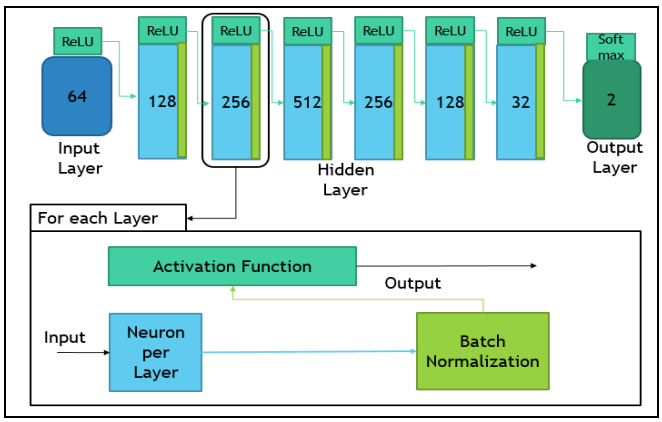


Figure 8. The structure of TPN

4.2 Trajectory Similarity Model

This model is used for making a trajectory search. For each recommended package, it queries and retrieves the similar packages-as Similar Travel packages dataset---from the original database, see Figure 9 the chosen query process is Top similarity. We predicted the recommended trip cancellation rate based on the average cancellation rate of the retrieved trajectories in the Similar Travel packages dataset. Let

Tr_m be a trajectory in dataset, and Tr_{target} be a recommended travel package. The similarity between the POIs p_{target_i} & p_{mj} is evaluated from the type feature they hold and their spatial distance as follows:

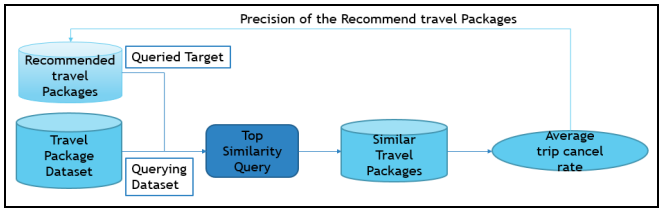


Figure 9. The trajectory similarity model

$$Sim(p_{target_i}, p_{mj}) = f_{dis} + f_{type}(p_{target_i}, p_{mj}) \quad while\ p_{target_i} \in Tr_{target}, p_{mj} \in Tr_m \tag{16}$$

For a spatial threshold $\in$, $f_{dis}(p_{target_i} - p_{mj})$ denotes the distance similarity:

$$f_{dis}(p_{target_i}, p_{mj}) = 1 - \frac{dist(p_{target_i}, p_{mj})}{\in} \tag{17}$$

Recall that $dist(p_{target_i} - p_{mj})$ is the distance between p_{target_i} & p_{mj}, we also need the Jaccard similarity $f_{type}(p_{target_i} - p_{mj})$ defined as:

$$f_{type}(p_{target_i}, p_{mj}) = \frac{|type(p_{target_i}) \bigcap type(p_{mj})|}{|type(p_{target_i}) \bigcup type(p_{mj})|} \quad type(p_{target_i})\ is\ the\ type\ set\ of\ p_{target_i} \tag{18}$$

For a POI p_{target_i} in Tr_{target}, the optimal point of Tr_m and p_{target_i} is defined as:

$$Opt(p_{target_i}, Tr_m) p_{mj}, p_{mj} \in Tr_m \quad if \quad Sim(p_{target_i}, p_{mj}) > Sim(p_{target_i}, p_{mk}) \forall p_{mk} \in Tr_m \tag{19}$$

Take Figure 10 as an example, it shows that each point in Tr_{target} will find an optimal point in Tr_m.

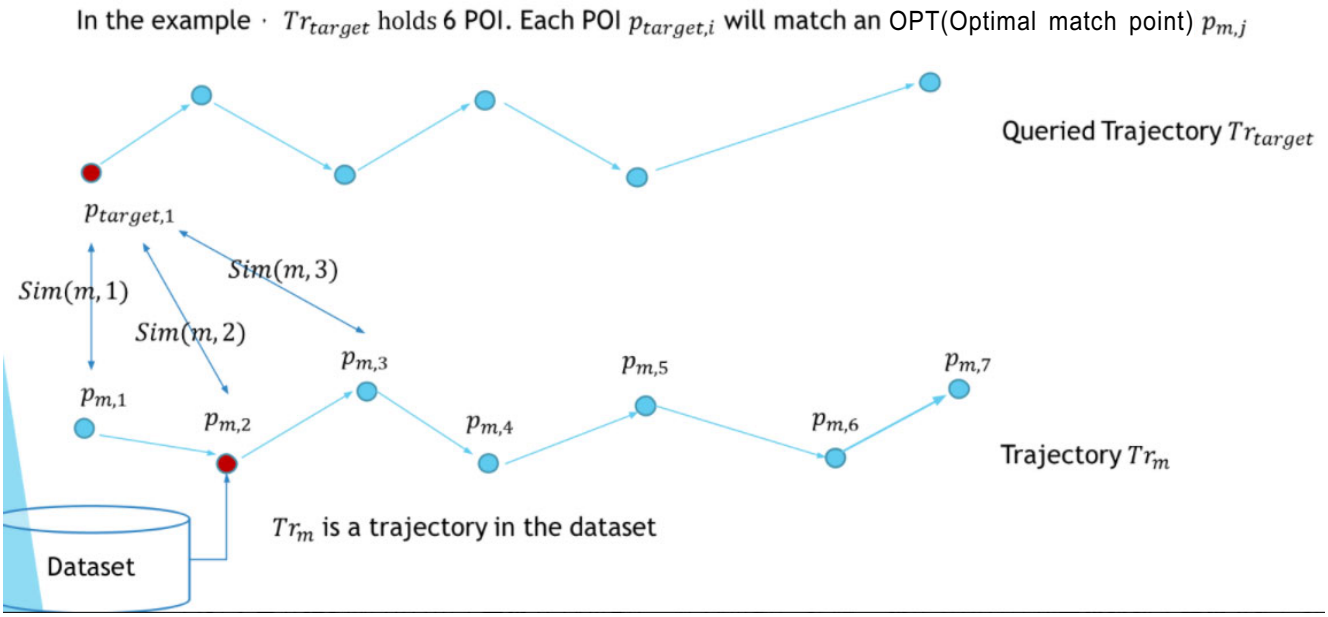


Figure 10. Optimal point decision

If p_{target_i}'s optimal point in Tr_m is p_j, then $Sim(p_{target_i}, p_{mj})$ has higher score than $Sim(p_{target_i}, p_{mk})$ when p_{mk} is a POI in Tr_m different from p_{mj}. We evaluate the trajectory similarity between Tr_m, Tr_{Target} in the following way:

$$Sim_{Tr}(Tr_m, Tr_{Target}) = normalized(\Sigma Sim(p_{target_i}, Opt(p_{target_i}, Tr_m))) \quad while\ p_{target_i} \in Tr_{Target} \tag{20}$$

For each POI p_{target_i} in Tr_{target}, an optimal point can be found in Tr_m, which we denote by $Opt(p_{target_i}, Tr_m)$. Note that the similarity of trajectories Tr_m, Tr_{target} equals to the normalized summation of the optimal points $p_{target_i} \in Tr_{target}$ over index i.

The algorithm of finding the most similar trajectory to Tr_{target} is as follows:

Algorithm 2. Top Similarity Query

Input: Tr_{target}, TRD

$Top\_K = \phi$
For num from 1 to K:
  $Max_{sim} = 0$
  $Max_{Tr} = 0$
  For All $Tr_m \in Dataset$:
    If $Max_{sim} < Sim_{Tr}(Tr_m, Tr_{Target})$:
      $Max_{sim} = Sim_{Tr}(Tr_m, Tr_{Target})$
      $Max_{Tr} = Tr_m$
  End For
  $Sim_{Tr}(Tr_m, Tr_{Target}) = 0$
  $Top\_K = Top\_K \bigcup Max_{Tr}$
End For

For a trajectory target Tr_{Target}, evaluate $Sim_{Tr}(Tr_m, Tr_{Target})$ for all $Tr_m \in Dataset$. The Tr_m that maximizes $Sim_{Tr}(Tr_m, Tr_{Target})$ should be added to the Top_K array. Retrieved after looping K times.

5 Experiments and Results

5.1 Data Collection

The travel agency provided us with a travel package database for the duration between 2015 and 2017. This database contained trips from Taiwan to Japan. Figure 11 shows the structure of the database along with our preprocessing.

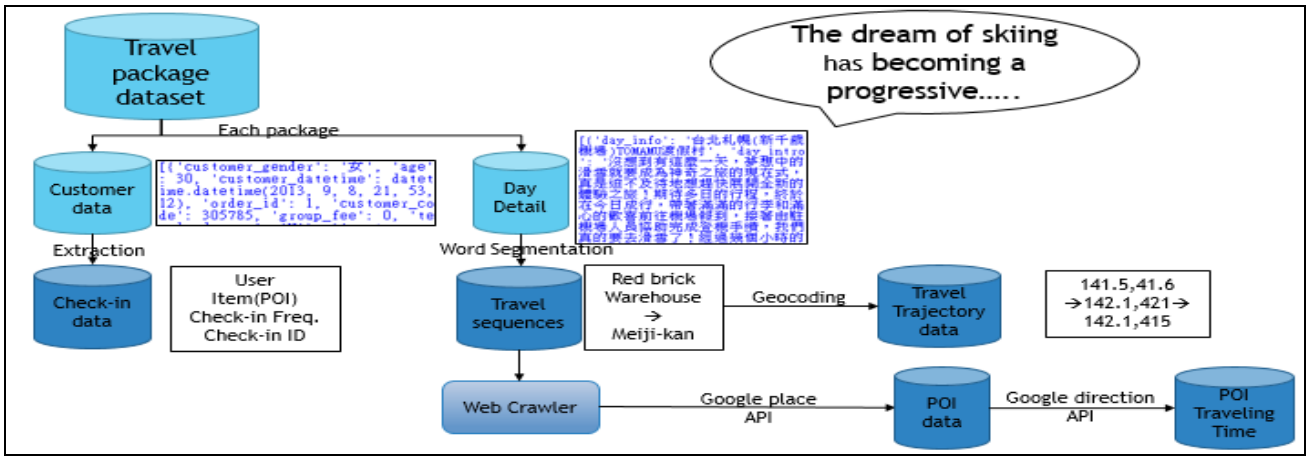


Figure 11. Data collection process

5.1.1 Preprocessing

We used word segmentation to extract the attributes we needed, the detail was shown in Table 5 and Table 6.

5.1.2 Web Crawling

In order to find the detail (address, location, and rating) of the POIs in the travel sequences, we used the Google place API. In order to find the physical distance between POIs, we used the Google direction API. The administrative data was crawled over the Japanese government site to do. Finally, the travel sequences were geocoded to travel trajectories to be used for the trajectory similarity model.

5.2 Recommendation Visualization

Figure 12 displays 3 recommended packages in the Hokkaido area for 5 days of travel. Plan (a) starts from South Hakotade Market and ends at Tokachi Millennium Forest. The total transportation time is estimated 14 hours. The number of POIs to visit is 11, which requires a total visiting time of 22 hours. The total traveling time is estimated 36 hours, feasible for a 5-day trip. The average rate of POIs in this package is 3.8. The recommended packages differ in their starting states (POI). Plan (b) starts from the west side of Hokkaido-Cape Kamui, and it travels towards east. Plan (c) starts from the northern POI-Rishirifuji, and it turns south and west afterwards.

5.3 Trip Prediction Network Model Experiments

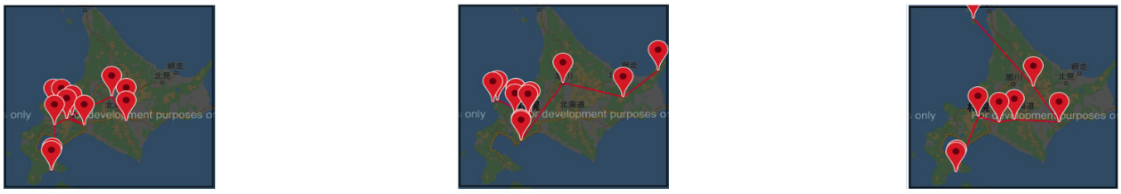
The data was split randomly, 80% for training, 20% for testing. The same process was repeated 5 times with different splits. We took the mean value as the final results. Figure 13 displays results in 4 evaluation metrics: Training Accuracy, Training AUC, Testing Accuracy, and Testing AUC. The training took 10000 epochs. The domain of training precision was about [80, 90], while for testing precision, it was [70, 80].

Table 5. Dataset statistics

Dataset	Attribute type	Example
POI Dataset	POI name	Tokyo Skytree
	Address	1Chome-1-2 Oshiage, Sumida, Tokyo 131-0045, Kanto
	POI types	Establishment, point_of_interest
	Longitude	139.8107004
	Latitude	35.7100627
	Region	Kanto
	Prefecture	Tokyo
	City	Sumida
	Rating	4.1
Travel Dataset	Package id	28
	Sale price	28,800
	Area	Japan
	Detail area	Hokkaido
	Group startday	
	Days	3
	Sequence	Kanemori Red Brick Warehouse, Meijikan……
	Trajectory	(141.14,42.49) (140.72,41.76)
	Tour YN	Y(Uncanceled)
Check-in Dataset	Customer ID	305785
	POI name	Kanemori Red Brick Warehouse
	Check-in Frequency	2
	Check-in ID	102

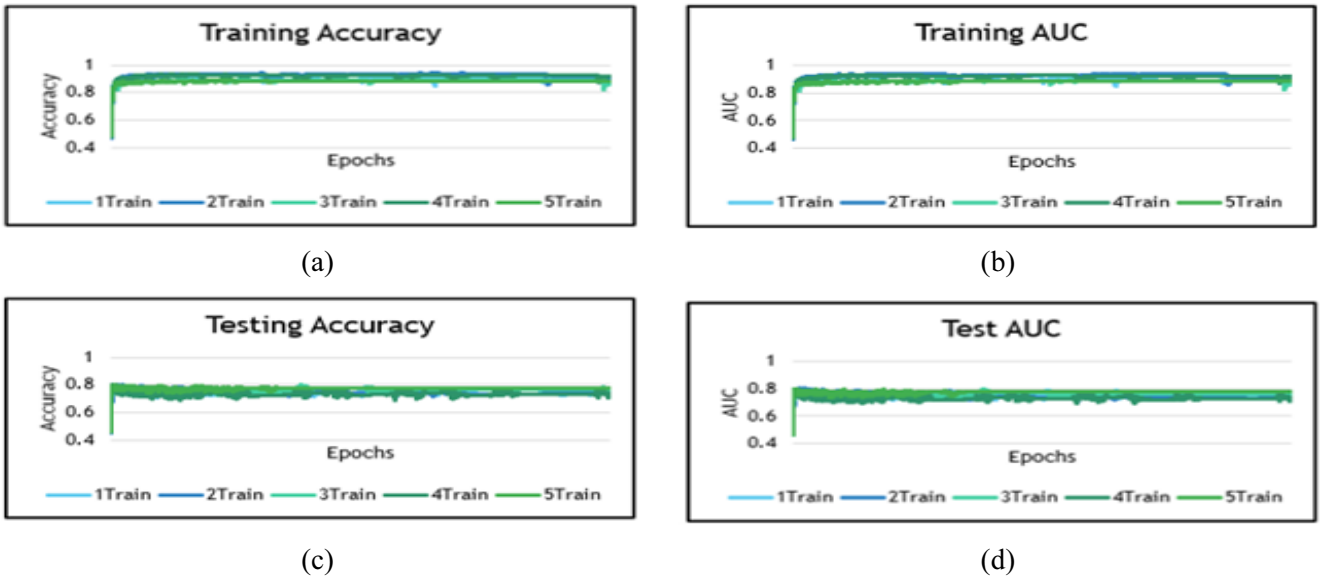
Table 6. Dataset detail overview

Dataset	#Users	#POIs	#Check-ins
Check-in Data	16175	3851	400027



(a) (b) (c)

Figure 12. Example of recommendation visualization



(a) (b)

(c) (d)

Figure 13. Training and Testing Curves

The running time increased with the number of layers. When we switched from 6 layers to 7 or 8 layers, the longer running times failed to achieve higher precision (Testing AUC). Interestingly, even the 1-layer network performed well: it was time-saving, and it had a precision only 1% lower than that of the 6-layer model. For this, it took only one-sixth of its training time.

Figure 14 shows a comparison between Money, Days, Customer, Season, Month, and POI as listed in Table 7. Obviously, the Customer feature is the most significant factor affecting prediction precision. The Date feature is also important to improve precision. However, the set with the highest test AUC score (0.798) is the basic set without the Money feature (set # 7).

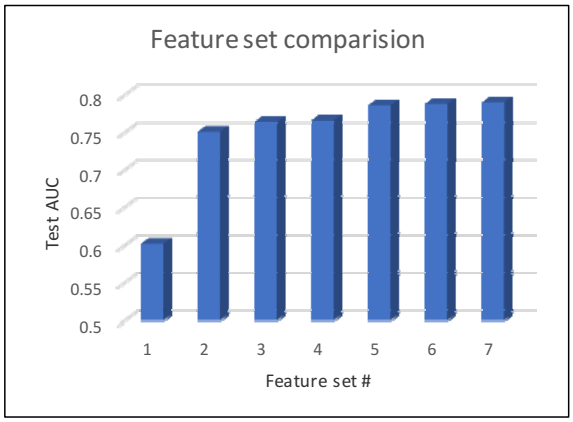


Figure 14. precision varying by the index of feature sets

Table 7. Table of different feature sets

Feature set	Features included	Features in basic set
1	Basic set without Customer	Money Days Customer Season Month POI
2	Basic set without Season and Month	
3	Basic set without Days	
4	Basic set without POI types	
5	Basic set plus Date	
6	Basic set	
7	Basic set without Money	

Trip guaranteed prediction. Using AI to predict a customer's acceptance of a recommended travel package has become a sport in Taiwan. T-brain---an organization holding AI and machine learning competitions-held a competition in December 2018 on this subject [31]. The winner scored 0.765 for Test AUC precision. In comparison, the highest score of our proposed models was 0.798.

5.4 Trajectory Similarity Model Experiments

We wanted to show that trajectory similarity can classify the cancelled and uncancelled travel packages. The spatial threshold $\in$ was set as 10 (km). To evaluate the performance of model, dataset was split into 2 classes: uncancelled and cancelled. Since the two classes were imbalanced, the uncancelled data was partly eliminated. Both datasets were randomly split: 80% for training, 20% for testing. T-test is a type of inferential statistics to determine significant difference between the means of two groups. We set the confidence interval at 95%. Table 8 shows the group statistics of three sets: uncancelled, cancelled, and recommended packages by our proposed framework. The values among those sets were the trajectory similarity precision ordered from the highest. We considered the top 20 of this list.

Table 8. Data statistics of Top K precision

guaranteed	N	Mean	Std deviation	Std error mean
Uncanceled	20	0.5504	0.02297	0.00514
Canceled	20	0.7659	0.01547	0.00346
Recommend	20	0.7639	0.03059	0.00684

Figure 15 shows the Top K curves of the average accuracy of datasets. Since the top K recommendation curves have a difference of more than 10%, model can correctly classify packages likely to be cancelled.

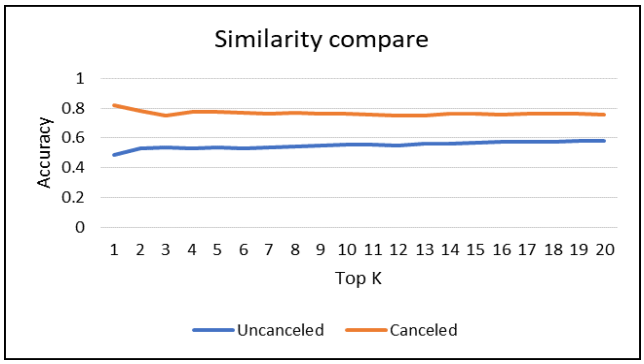


Figure 15. Precision curve for Top K precision

The set of recommended packages depends on the components of the reward function. We omitted some of the 4 compon ents of the reward function for testing there, see Figure 16. Each set had 100 packages. Finally, the proposed set with 4 elements outperformed.

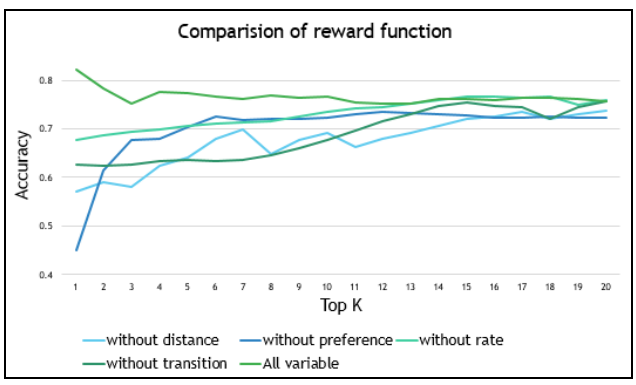


Figure 16. Precision curve for different reward function settings

As Figure 17 shows the T test score of the uncanceled, canceled and Recommend pair of data. We can see obviously the mean of both Canceled Package and the Recommend package were higher than the Uncanceled one. Further more due to the significance results, we can know that the U & C pair and C & R pair were really similar to each other, while only the Uncanceled one was different.

Trip guaranteed precision sequence		N	Mean	Std deviation	Std error mean
	Uncanceled	20	0.5504	0.02297	0.00514
	Canceled	20	0.7659	0.01547	0.00346
	Recommend	20	0.7639	0.03059	0.00684

T-test for Equality of Means

Variable set	T	df	Sig. (2-tailed)	Mean difference	Std error Difference
U & C	-34.797	38	0.000	-0.216	0.006
C & R	-24.959	38	0.000	-0.214	0.009
R & U	0.258	38	0.798	0.002	0.008

U=Uncanceled
Precision value
C=Canceled
Precision value
R=Recommend
Precision value

Figure 17. T test of the top 20 recommendation

6 Conclusion and Future Work Plan

6.1 Conclusion

The reinforcement learning methodology successfully capture the by related functions and recommend. The TPN model outperformed than the Taiwan T-brain competition, many experiments and variable tuning were done to make sure the model may have the best accuracy and AUC. The Trajectory Similarity model may able to classify among Failure packages and Success packages. The proposed package is much similar to Success packages according to the T test methodology.

6.2 Future Work Plan

As to the future work, the system may be able to be worldwide useable by collecting more data and wider range. In addition, the running time may be a significant factor for recommendation systems, not even to mention real-time characteristic. Enhanced technique on reinforcement learning may be possible to apply on our framework to reach better performance and better recommendation result, such as Sarsa technique and Deep Q-network.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. This research work was supported in part by the Ministry of Science and Technology of Taiwan, ROC. (MOST 108-2221-E-006-173-MY2)

References

[1] L. Y. Wei, W. C. Peng, W. C. Lee, Exploring pattern-aware travel routes for trajectory search, *ACM Transactions on Intelligent Systems and Technology (TIST)*, Vol. 4, No. 3, pp. 1-25, June, 2013..

[2] Z. W. Yu, Y. Feng, H. Xu, X. S. Zhou, Recommending travel packages based on mobile crowdsourced data, *IEEE Communications Magazine*, Vol. 52, No, 8, pp. 56-62, August, 2014.

[3] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. v. d. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, Mastering the game of Go with deep neural networks and tree search, *Nature*, Vol. 529, No. 7587, pp. 484-489, January, 2016.

[4] M. S. Emigh, E. G. Kriminger, A. J. Brookmeier, J. C. Príncipe, P. M. Pardalos, Reinforcement learning in video games using nearest neighbor interpolation and metric learning, *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 8, No. 1, pp. 56-66, March, 2016.

[5] M. Mahmud, M. S. Kaiser, A. Hussain, S. Vassanelli, Applications of deep learning and reinforcement learning to biological data, *IEEE transactions on neural networks and learning systems*, Vol. 29, No. 6, pp. 2063-2079, June, 2018.

[6] D. Zhang, X. Han, C. Deng, Review on the Research and practice of deep learning and reinforcement learning in smart grids, *CSEE Journal of Power and Energy Systems*, Vol. 4, No. 3, pp. 362-370, September, 2018.

[7] T. Mannucci, E. J. Kampen, C. Visser, Q. Chu, Safe exploration algorithms for reinforcement learning controllers, *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 29, No. 4, pp. 1069-1081, April, 2018.

[8] M. Mohammadi, A. Al-Fuqaha, M. Guizani, J. S. Oh, Semisupervised deep reinforcement learning in support of IoT and smart city services, *IEEE Internet of Things Journal*, Vol. 5, No. 2, pp. 624-635, April, 2018.

[9] Y. H. Wu, S. D. Lin, A low-cost ethics shaping approach for designing reinforcement learning agents, *Proceedings of the AAAI Conference on Artificial Intelligence*, New Orleans, Louisiana, USA, 2018, pp. 1687-1694.

[10] T. Y. Ma, P. Gerber, A Hybrid Learning Algorithm for Generating Multi-Agent Daily Activity Plans, *Journal of Internet Technology*, Vol. 17, No. 5, pp. 959-969, September, 2016.

[11] R. Grunitzki, G. d. O. Ramos, A. L. C. Bazzan, Individual versus difference rewards on reinforcement learning for route

choice, *2014 Brazilian Conference on Intelligent Systems*, Sao Paulo, Brazil, 2014, pp. 253-258.

[12] K. M. Jagodnik, P. S. Thomas, A. J. van den Bogert, M. S. Branicky, R. F. Kirsch, Training an Actor-Critic Reinforcement Learning Controller for arm movement using human-generated rewards, *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, Vol. 25, No. 10, pp. 1892-1905, October, 2017.

[13] J. L. Lin, K. S. Hwang, W. C. Jiang, Y. J. Chen, Gait balance and acceleration of a biped robot based on Q-learning, *IEEE Access*, Vol. 4, pp. 2439-2449, May, 2016.

[14] K. Liu, P. Wang, J. Zhang, Y. Fu, S. K. Das, Modeling the interaction coupling of multi-view spatiotemporal contexts for destination prediction, *Proceedings of the 2018 SIAM International Conference on Data Mining*, San Diego, CA, USA, 2018, pp. 171-179.

[15] A. Ghadirzadeh, X. Chen, W. Yin, Z. Yi, M. Björkman, D. Kragic, Human-Centered Collaborative Robots With Deep Reinforcement Learning, *IEEE Robotics and Automation Letters*, Vol. 6, No. 2, pp. 566-571, April, 2021.

[16] Y. J. Park, Y. J. Lee, S. B. Kim, Cooperative Multi-Agent Reinforcement Learning With Approximate Model Learning, *IEEE Access*, Vol. 8, pp. 125389-125400, July, 2020.

[17] T. Chu, J. Wang, L. Codecà, Z. Li, Multi-Agent Deep Reinforcement Learning for Large-Scale Traffic Signal Control, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 21, No. 3, pp. 1086-1095, March, 2020.

[18] X. Wei, J. Zhao, L. Zhou, Y. Qian, Broad Reinforcement Learning for Supporting Fast Autonomous IoT, *IEEE Internet of Things Journal*, Vol. 7, No. 8, pp. 7010-7020, August, 2020.

[19] S. Jiang, X. Qian, T. Mei, Y. Fu, Personalized Travel Sequence Recommendation on Multi-Source Big Social Media, *IEEE Transactions on Big Data*, Vol. 2, No. 1, pp. 43-56, March, 2016.

[20] Q. Liu, E. Chen, H. Xiong, Y. Ge, Z. Li, X. Wu, A Cocktail Approach for Travel Package Recommendation, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 26, No. 2, pp. 278-293, February, 2014.

[21] J. Chang, C. Tseng, R. Hwang, M. Ma, Using ANN to Analyze the Correlation Between Tourism-Related Hot Words and Tourist Numbers: A Case Study in Japan, *2017 IEEE 7th International Symposium on Cloud and Service Computing (SC2)*, Kanazawa, Japan, 2017, pp. 132-137.

[22] J. P. Teixeira, P. O. Fernandes, Tourism time series forecast with artificial neural networks, *Tékhne*, Vol. 12, No. 1-2, pp. 26-36. January-December, 2014.

[23] A. Majid, L. Chen, G. Chen, H. T. Mirza, I. Hussain, J. Woodward, A context-aware personalized travel recommendation system based on geotagged social media data mining, *International Journal of Geographical Information Science*, Vol. 27, No. 4, pp. 662-684, April, 2013.

[24] O. Claveria, S. Torra, Forecasting tourism demand to Catalonia: Neural networks vs. time series models, *Economic Modelling*, Vol. 36, pp. 220-228, January, 2014.

[25] H. A. Constantino, P. O. Fernandes, J. P. Teixeira, Tourism demand modelling and forecasting with artificial neural network models: The Mozambique case study, *Tékhne*, Vol. 14, No. 2, pp. 113-124, July-December, 2016.

[26] J. H. Chang, C. Y. Tseng, Analyzing Google Trends with Travel Keyword Rankings to Predict Tourists into a Group, *Journal of Internet Technology*, Vol. 20, No. 1, pp. 247-256, January, 2019.

[27] M. Hassan, M. Hamada, A Neural Networks Approach for Improving the Accuracy of Multi-Criteria Recommender Systems, *Applied Sciences*, Vol. 7, No. 9, Article No. 868, September, 2017.

[28] D. Wang, Y. Yang, S. Ning, DeepSTCL: A Deep Spatio-temporal ConvLSTM for Travel Demand Prediction, *2018 International Joint Conference on Neural Networks (IJCNN)*, Rio de Janeiro, Brazil, 2018, pp. 1-8.

[29] J. Ji, J. Hou, Forecast on Bus Trip Demand Based on ARIMA Models and Gated Recurrent Unit Neural Networks, *2017 International Conference on Computer Systems, Electronics and Control (ICCSEC)*, Dalian, China, 2017, pp. 105-108.

[30] Y. Zheng, W. Sheng, X. Sun, S. Chen, Airline Passenger Profiling Based on Fuzzy Deep Machine Learning, *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 28, No. 12, pp. 2911-2923, December, 2017.

[31] C. Siripanpornchana, S. Panichpapiboon, P. Chaovalit, Travel-time prediction with deep learning, *2016 IEEE Region 10 Conference (TENCON)*, 2016, pp. 1859-1862.

[32] Y. Liu, Y. Wang, X. Yang, L. Zhang, Short-term travel time prediction by deep learning: A comparison of different LSTM-DNN models, *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Yokohama, Japan, 2017, pp. 1-8.

[33] Z. Zhang, P. Chen, Y. Wang, G. Yu, A hybrid deep learning approach for urban expressway travel time prediction considering spatial-temporal features, *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Yokohama, Japan, 2017, pp. 795-800.

[34] A. Dong, Z. Du, Z. Yan, Round Trip Time Prediction Using Recurrent Neural Networks With Minimal Gated Unit, *IEEE Communications Letters*, Vol. 23, No. 4, pp. 584-587, April, 2019.

[35] X. Gang, W. Kang, F. Wang, F. Zhu, Y. Lv, X. Dong, J. Riekki, S. Pirttikangas, Continuous Travel Time Prediction for Transit Signal Priority Based on a Deep Network, *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Gran Canaria, Spain, 2015, pp. 523-528.

Biographies

Jui-Hung Chang is currently a full-time Professor with the Computer and Network Center, Department of Computer Science Information Engineering, National Chen Kung University. Her current research interests include data base

management, digital learning system, system analysis and object oriented design, program design, data mining, cloud calculation application, and project management and planning. Furthermore, she is also interested in active participation in related travel information prediction and recommendation, geographical system, agricultural statistical big data, biological big data mining with industry academy cooperation plan and research projects.

Hung-Hsi Chiang, received his diploma in 2016 from the Mathematics department of National Chung Cheng University, and received his M.S. degree in Computer Science and Information Engineering at National Cheng Kung University in 2019. His research interests include deep learning and reinforcement learning, focusing his research on spatial issues and recommendation systems.

Hua-Xu Zhong, received the M.S. degree in Department of E-learning Design and Management from National Chiayi University, Taiwan. He is a Ph.D. student of Engineering Science at National Cheng Kung University, Taiwan. His research interests include e-learning, Flipped classroom, educational technology and STEM education.

Yu-Kai Chou, received his diploma in 2017 from the Applied Mathematics department of National Chengchi University and is currently pursuing his M.S. degree in Information Engineering at National Cheng Kung University. His research topics include machine learning and time series.