Deep Reinforcement Learning Framework for Real-Time Personalized Travel Route Recommendation via LSTM-CNN and Multi-Head Attention Fusion

Dan Zhang
Hongqing vocational college of applied technology, College of Health Industry, Chongqing 401520, China
E-mail: sengsengyou5945@163.com

Keywords: Deep reinforcement learning, personalized recommendation model, real-time context awareness, user preferences, travel routes

Received: August 21, 2025

With the development of smart tourism, traditional static recommendations struggle to cope with the dynamic changes in RTCs (Real-Time Contexts) such as traffic and weather in urban environments. Furthermore, they cannot integrate UPs (User Preferences) with real-time contextual awareness, resulting in poor recommendation adaptability. This paper aims to design a highly adaptable, personalized, and dynamic TR (Travel Route) recommendation model. The model leverages LSTM-CNN for feature extraction and Multi-Head Attention Mechanism (MHAM) for feature fusion. The system is trained using an Actor-Critic (AC) framework. Evaluation metrics such as HR@5, HR@10, coverage, and median response latency (MRL) are used to assess performance. Based on DRL (Deep Reinforcement Learning), this model captures UP differences through the construction of an LSTM-CNN (Long Short-Term Memory-Convolutional Neural Network) network, achieving personalization. A MHAM (Multi-Head Attention Mechanism) is applied to deeply integrate UPs with real-time contextual states such as traffic and weather. A CRF (Composite Reward Function) is designed by jointly modeling preferences and context, and end-to-end training is achieved using an AC (Actor-Critic) framework. Experiments show that on the FS-NYC (Four Square–New York City dataset) and TCI (Tokyo Check-ins dataset), the paper's model achieves a Top-5 hit rate of 53% and a Top-10 hit rate of 84%, with a MRL (Median Response Latency) of 1.07 seconds. It also significantly improves adaptability to dynamic scenarios compared to baseline methods. This research provides a personalized recommendation paradigm that combines high accuracy with real-time responsiveness for dynamic travel scenarios, effectively improving user experience and service quality.

Povzetek: Članek predlaga prilagodljiv prilagojen model za priporočanje potovalnih poti, ki z LSTM–CNN in večglavno pozornostjo združi uporabniške preference z realnočasovnimi konteksti.

1 Introduction

With the rapid advancement of science and technology, all industries are integrating new technologies for development, and the tourism industry is becoming increasingly intelligent. Route recommendations are an indispensable part of travel planning and onboarding. With the development of smart tourism, traditional static recommendations cannot meet users' individual requirements in complex and dynamic environments [1], [2]. Traditional methods often ignore real-time contextual changes and lack joint modeling of UPs and environmental interactions. They recommend only a fixed number of attractions, resulting in poor adaptability and a subpar experience. Especially in cities with volatile traffic, weather, and crowd conditions, tourist routes can change anytime [3], [4]. Fixed route planning is prone to failure, necessitating an intelligent recommendation mechanism with real-time responsiveness.

This study focuses on the problem of dynamic route recommendation in urban tourism scenarios, taking the user's real-time location, preference characteristics, and multi-source contextual data as key objects, aiming to achieve highly adaptable and personalized serialized scenic spot recommendations, and enhance the quality of tourism services and user experience. Wilkins and Horne [5] pointed out that the weather has an important impact on tourists, and Marsanic et al. [6] believed that good traffic conditions can improve the quality of tourists' travel. These two studies prove the importance of RTC during travel. Kay Smith et al. [7] studied tourists' interests and the different activities they participated in, and pointed out that tourists' interests have a great influence on their behavior; Saxena et al. [8] proposed that tourists attach great importance to the accessibility and activities of scenic spots, verifying the necessity of a multidimensional context; Xin et al. and Prahadeeswaran believed that personalization can better enhance the experience of travel recommendation systems [9], [10]; Vada et al. and Anuar and Marzuki believed that suitable TRs require good infrastructure, and providing more personalized choices is an emerging trend [11], [12]. Research proposed adaptive fuzzy sliding-mode controllers with non-singular fixed-time sliding surfaces, which effectively addressed the issue of system uncertainties [13]. The study proposed an output-feedback controller based on adaptive fuzzy systems and a variable-structure framework to ensure system stability [14]. The study proposed a robust and indirect neural adaptive control scheme for uncertain nonlinear multivariable systems, which could effectively compensate for disturbances and ensure system stability [15]. The study proposed an adaptive backstepping control method based on Lyapunov stability theory, which ensured that the tracking error asymptotically converged to zero [16]. The study proposed a nonlinear optimal H-infinity control method for gas centrifugal compressors driven by asynchronous motors, aiming to achieve robust state estimation under uncertainty conditions [17]. The study proposed an adaptive backstepping control method that enabled the tracking error to asymptotically converge to zero [18]. These studies collectively indicate that dynamic travel recommendations must comprehensively consider user status and environmental evolution.

Table 1: Related work comparison

Method	User Preferences	Real-Time Context	DRL Technique	Evaluation Datasets	Reported Performance
Zhang et al.	Yes	Yes	No	AmazonDataset	HR@10:52.40%, 75.57%, 72.43%
Liu et al.	Yes	No	No	Ciao	RMSE: 1.9136,MAE: 1.4937
Zhang et al.	Yes	Yes	Yes	ASSISTments0910	Difficulty:0.7
Chen et al.	Yes	No	No	2400 international and domestic tourists in Pokhara	Accuracy:94%,99%
Yoon and Choi	Yes	Yes	No	Jeju Tourism Dataset	Accuracy:77.3%
Wang	Yes	No	No	obtained by web crawling information about attractions in a city	MAE:0.47235
Nan and Wang	Yes	No	No	A tourism dataset	Accuracy:91.04%

Table 1 presents information such as datasets and evaluation metrics related to the relevant works. Regarding recommendation model technology, in terms of path planning, Ma and Zhu proposed a recommendation model based on Deep Reinforcement Learning, which had the advantage of flexible scheduling [19], [20]. Zhang et al. and Liu et al. combined graph neural networks to extract UPs from graphs. However, this method relies on historical patterns and is challenging to respond to sudden situations [21], [22]. While Shyam and Zhang et al. proposed a method that incorporated DRL, it struggled to effectively integrate deep UPs with RTC [23], [24]. Existing models still suffer from insufficient coupling between state representation and reward design and weak generalization capabilities. Shrestha et al. and Nunez et al. examined the utilization of machine learning in tourism and travel recommendations [25], [26]. Chen et al. and Yoon and Choi proposed tourism analysis models and recommendation models that could perceive RTC, respectively, but neither modeled the dynamic evolution of UPs [27], [28]. Based on collaborative filtering, Wang and Nan and Wang integrated UPs, which improved the accuracy of recommendations, but still had shortcomings in context perception [29], [30]. Liu et al. [31] applied an attention mechanism to weight historical visits to determine UPs, but ignored the impact of real-time weather on the action space. Tsai et al. [32] studied the implicit and dynamic information of points of interest, taking into account UPs, but did not consider the impact of real-time scenarios such as traffic and weather. Mou et al. and Zhou et al. studied user trajectories, emphasizing the main behavioral intentions of tourists. They can effectively understand tourists' travel patterns, but cannot adapt to unexpected situations [33], [34]. The above methods still have difficulty balancing the depth of personalization and dynamic adaptability.

This paper designs a dynamic TR recommendation model based on DRL. The primary research questions addressed by this study are: (1) Can a DRL model integrating real-time context and user preferences outperform baseline recommendation systems in dynamic travel scenarios? (2) Does MHAM improve temporal personalization in dynamic travel recommendation? A joint high-dimensional vector that includes users' long-term preferences, real-time multidimensional context, and current state is constructed, and an LSTM-CNN network is adopted for feature extraction. The novelty of this integration lies in the specific combination of LSTM-CNN for capturing user preference patterns with MHAM for real-time contextual fusion, which is designed to enhance the adaptability of the recommendation in dynamic environments. While previous DRL-based systems address user preferences and contextual information, they do not employ such a tightly integrated feature fusion mechanism, particularly for real-time contextual shifts such as changes in weather or traffic patterns. This approach ensures that the model not only prioritizes user preferences but dynamically adapts to immediate situational changes, which has been underexplored in existing literature on mobility-based or temporal recommendation systems. The LSTM network is used to model the temporal dependencies of users' historical visit sequences, and the CNN network is used to process structured real-time contextual data. An MHAM is applied to achieve deep feature fusion. The decision mechanism is based on the asynchronous advantage AC framework for end-to-end training, and the Dueling DQN structure is used to decouple state value and action advantage to improve the stability of Q-value estimation. Finally, a compound reward function that includes preference matching, time rationality, situational adaptability, and repeated punishment is designed. Additionally, how attention mechanisms can provide insights into the decision-making process, helping end-users and tourism operators understand why certain recommendations are made, is explored, aiming to enhance user trust in the system by leveraging attention mechanisms not only for performance but also for interpretability. Comparisons with explainable recommendation systems can be considered in future work. Combined with a prioritized experience replay mechanism, learning efficiency under sparse rewards is improved, and the learning process is optimized, providing a learnable and evolvable decision-making paradigm for the intelligent tour guide system.

2 Algorithm design

2.1 State space construction

Accurate state space modeling is fundamental to enabling effective decision-making in dynamic travel recommendations using DRL. This study constructs a high-dimensional, semantically rich joint state vector S_t to simultaneously represent user personalization and real-time environmental changes. This implementation involves four steps.

First, UP encoding uses a two-layer unidirectional LSTM network to process the user's historical visit sequence [35], [36]. The input is a time-ordered sequence of scenic spot ID $\{v_1,v_2,...,v_n\}$, which is mapped into a 64-dimensional dense vector (Embedding Size = 64) through the embedding layer and sent to the LSTM (hidden layer dimension 128, sequence length limit 50). The LSTM updates its hidden state moment by moment, ultimately outputting a hidden vector $h_n \in R^{128}$ as a compressed representation of the user's long-term interests. This vector is further nonlinearly transformed through a fully connected layer to generate a fixed-dimensional preference embedding $\mathrm{h}_u \in \mathbb{R}^{128}$, preserving the interest evolution pattern in temporal behavior.

Secondly, RTC collection and vectorization encompass multi-source heterogeneous data. The system obtains the current weather conditions (sunny, rainy, snowy, high temperature, etc.) through the OpenWeatherMap API, one-hot-encodes them, and normalizes them into a 16-dimensional vector. The system also obtains the traffic congestion index (0–10) for the user's area through the Baidu Maps API and linearly normalizes it to the interval [0,1]. The real-time visitor flow ratio (current number of people/maximum capacity) of the target attraction is obtained through the scenic spot ticketing system interface and similarly normalized. The current time (hour encoded as a sin/cosine cycle feature) and whether it is a holiday (a binary flag) are combined to form a 64-dimensional context vector c_t.

Third, location status represents the user's current geographic and behavioral state. The latitude and longitude coordinates of the user's last checked-in attraction are used as the reference. These coordinates are converted to a spatial index using GeoHash encoding (6-digit precision) and co-encoded with the duration of stay (in minutes, truncated to 300 minutes). The duration of stay is logarithmically transformed and concatenated with the GeoHash vector, and is then mapped to a 32-dimensional position vector p_t through a fully connected network (32→32 ReLU), effectively capturing the user's current activity intensity and spatial anchor point.

Finally, the state fusion mechanism concatenates the three vectors to form a joint state representation:

$$s_t=[h_u;c_t;p_t]\in R^{224} \tag{1}$$

This joint vector serves as the state input for DRL, fully encompassing the user's intrinsic preferences, external environment dynamics, and current location information. The final concatenated state vector has a dimensionality of 160, comprising the 64-dimensional user preference vector, 64-dimensional contextual vector, and 32-dimensional location status vector. To ensure input consistency, all components are Z-score normalized (mean 0, variance 1) before entering the network, and are calculated offline based on statistical parameters from the training set. The normalization parameters (mean and variance) are computed from the training set and maintained across both the training and test phases to prevent data leakage.

2.2 Action space definition

The design of the action space directly determines the feasibility and real-time adaptability of the recommendation system. This study models each recommendation step as a discrete decision problem involving selecting the next destination from a set of candidate attractions. Action $a_t \in A_t$ represents the unique identifier of the recommended attraction at the time *t*. To ensure the enforceability of the recommendation results across geography, time, and user behavior, the action space A_t is not a fixed set, but a dynamically generated subset based on multidimensional constraints.

First, accessibility screening is centered around the user's current location, establishing spatial constraints. Using GPS to obtain the user's real-time (x_t,y_t) coordinates, an R-tree index is used to retrieve all candidate attractions within a 5-kilometer radius from the attraction database, forming an initial set C_{geo}. The 5-kilometer radius is chosen based on the average walking distance in urban environments, considering the practical travel limits for tourists and the density of attractions within this area. This range takes into account the city's average traffic density and the feasibility of walking/short-distance connections, avoiding jumpy recommendations across regions. Attraction closure events are simulated in the training phase as part of the environment, but are not present in the test phase to simulate real-world unpredictability. Geographic queries are supported by the PostGIS spatial database. The R-tree index is used to efficiently filter candidate attractions within the defined spatial range (5 km). The GeoHash encoding aids in determining the geographical proximity of attractions, and the R-tree provides an optimized search for nearest attractions.

Next, temporal feasibility pruning is performed based on the current system time τ_t and the opening schedule $[o_j,c_j]$ of each candidate attraction. Only attractions that meet the criteria $\tau_t+d(x_t,x_j)/v<c_j$ are retained, where $d(x_t,x_j)$ is the shortest road distance from the current location to the candidate attraction j (calculated using the OSRM routing engine), and v is the preset average moving speed (set to 8 km/h in urban areas). At the same time, attractions whose last entry time for the day has passed are eliminated to ensure that the recommended action can be completed in time. The preset average moving speed of 8 km/h is chosen based on typical walking speeds in urban tourism areas, which balances efficiency and user comfort.

Third, itinerary consistency constraints exclude attractions that the user has already visited. A dynamic set $V_t=\{v_1,v_2,...,v_t\}$ is maintained, recording the user's historical check-in sequence. All attractions in $j\in V_t$ are removed from the candidate set to prevent duplicate recommendations. Furthermore, if an attraction has been recommended but the user has not chosen it and is relatively close, the probability of it being recommended again is reduced over the next 30 minutes, achieving recommendation memory deduplication through status tagging.

Fourth, adaptive optimization of remaining time applies a path duration estimation mechanism based on the Dijkstra algorithm [37], [38]. A weighted graph is constructed based on urban road network data, with edge weights representing travel time (integrated with real-time traffic indices). The shortest arrival time $t_{arrive}(j)$ from the current node to each candidate attraction is calculated. This is combined with the recommended duration $t_{stay}(j)$ of the attraction. If $t_{arrive}(j)+t_{stay}(j)>T_{remain}$, where T_{remain} is the remaining time preset by the user or predicted by the model, the candidate is eliminated. This pruning strategy effectively avoids recommending infeasible actions that exceed the time budget.

Finally, the action space A_t is defined as the intersection of the above four filtered sets:

$$A_t=C_{geo}\cap C_{time}\cap \forall_t\cap\{j|t_{arrive}(j)+t_{stay}(j)\leq T_{remain}\} \qquad (2)$$

In cases where the filtered set is too small, the system expands the spatial range or allows for recommendations from attractions visited earlier within the trip, ensuring a minimum number of recommendations (K = 5) is maintained. This dynamic action space is updated every step, synchronized with state awareness (triggered every 30 seconds or when the user's location changes by >200 meters). When $A_t=\emptyset$, the termination action a_t=END is triggered, signaling the end of the trip. All candidate actions are sorted by Q value, and a top-K recommendation list (K = 5) is generated. This list is pushed to the client in real-time via the gRPC interface.

2.3 Reward function design

The design of the reward function directly affects the optimization direction of the DRL strategy and the rationality of the recommended behavior. The design structure is illustrated in Fig. 1.

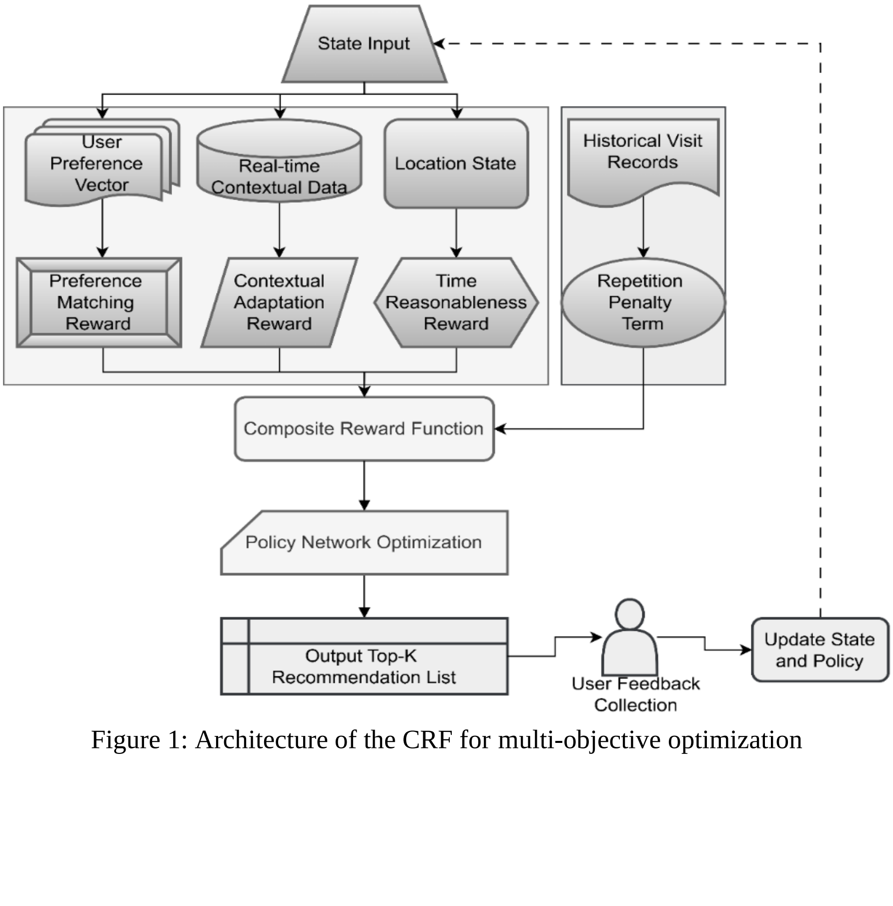
State Input
User Preference Vector
Real-time Contextual Data
Location State
Historical Visit Records
Preference Matching Reward
Contextual Adaptation Reward
Time Reasonableness Reward
Repetition Penalty Term
Composite Reward Function
Policy Network Optimization
Output Top-K Recommendation List
User Feedback Collection
Update State and Policy

Figure 1: Architecture of the CRF for multi-objective optimization

Fig. 1 shows the structure of the reward function, and illustrates the Composite Reward Function (CRF) using a weighted sum of preference matching, time rationality, context adaptability, and duplicate penalty to balance personalization, dynamic responsiveness, and recommendation consistency. This study constructs a CRF R_t , which achieves multi-objective collaborative optimization through a weighted linear combination to ensure that the model strikes a balance between preference matching, time rationality, situational adaptation, and recommendation specifications. The specific form is as follows:

$$R_t=\alpha R_{pref}+\beta R_{time}+\gamma R_{context}-\delta R_{penalty} \quad (3)$$

The weights of each item are determined through grid search as: $\alpha=0.22$, $\beta=0.18$, $\gamma=0.27$, and $\delta=0.33$, to enhance the response priority to situational changes and inhibit repetitive behaviors. A grid search is performed over the following parameter ranges: preference matching weight (0.1–0.5), time rationality weight (0.05–0.2), context-adaptive weight (0.1–0.4), and duplicate penalty weight (0.2–1.0). These ranges are chosen to balance personalization with situational responsiveness and penalize redundant recommendations.

The preference-matching reward R_{pref} quantifies the consistency of the recommendation results with the user's long-term interests. The input is the UP vector $h_u \in R^{128}$ (generated by LSTM encoding) and the category embedding $e_j \in R^{64}$ of the target attraction j. The cosine similarity between the two is calculated:

$$R_{pref}=\cos(h_u,e_j) \quad (4)$$

This value ranges from [-1, 1] and is linearly mapped to the interval [0, 1] to serve as the base preference score. This design encourages the model to recommend attractions that are semantically similar to the user's historical behavior, improving personalization accuracy.

The time rationality reward, R_{time} , assesses the suitability of the recommended timing. The optimal visiting hours are predefined based on the attraction type: 9:00–11:00 for museums, 11:30–1:30 for restaurants, and 18:00–21:00 for night scenes. If the predicted arrival time, $\hat{\tau}_{arrive}$, falls within the corresponding interval, $R_{time}=1.0$; if it falls within opening hours but not during peak hours, R is assigned a value of 0.3; if it is near closing time (remaining available time < 30 minutes), it is set to 0. The arrival time is calculated by adding the estimated travel time from the current location using the OSRM (OpenStreetMap Routing Machine) path planning engine to ensure that the time judgment is based on real traffic conditions.

The context-adaptive reward $R_{context}$ achieves responsiveness to dynamic environments, and makes logical decisions according to the current weather conditions and the attributes of the attraction: if the weather is "raining" or "snowing", and the recommended attraction is indoors (e.g., a museum or shopping mall), then $R_{context}=+1$; if the recommended attraction is an outdoor attraction (e.g., a park or square), then $R_{context}=-1$. Recommending an outdoor attraction on a sunny day can earn +0.8, and 0 otherwise. This mechanism forces the model to prioritize safe and comfortable indoor locations during inclement weather, improving user experience and safety. A duplicate penalty term, $R_{penalty}$, prevents invalid recommendation loops. If the attraction j corresponding to action a_t already exists in the user's historical visit set V_t, a fixed penalty of -2 is applied. If the attraction is recommended for the most recent trip but is not chosen, an additional penalty of -1 is applied. This design uses negative incentives to prevent the model from repeatedly outputting the same candidate, enhancing recommendation diversity.

All rewards are calculated immediately after each decision, normalized using the Z-score to eliminate dimensionality, and then weighted and summed. The final scalar reward R_t serves as an immediate feedback signal for RL (Reinforcement Learning), driving the policy network to optimize long-term cumulative benefits. To further investigate the impact of trade-offs between accuracy, latency, and personalization, sensitivity analyses are conducted on the weights of the CRF. Specifically, how varying the balance between preference satisfaction and contextual adaptation affects recommendation performance and responsiveness is explored. Additionally, the feasibility of integrating a multi-objective reinforcement learning framework is considered to provide a more structured approach to handling these trade-offs systematically. This reward mechanism addresses the decision bias caused by the single-goal orientation of traditional recommendation systems. A multidimensional reward structure enables the model to simultaneously address users' intrinsic preferences, external environmental constraints, and behavioral rationality; differentiated weighting enhances sensitivity to critical contexts; an explicit penalty mechanism improves the logical consistency of the recommendation sequence. Experiments demonstrate that this design significantly improves the strategy's robustness and practicality in complex urban tourism scenarios. The numerical values or ranges of each component are shown in Table 2.

Table 2: Model components, their expected impacts, and parameter settings

Component	Parameter Name	Value / Range
LSTM	Embedding Size	64
	Hidden Layer Dimension	128
	Sequence Length Limit	50
CNN-MLP	Conv1D Kernel Size	3
	Conv1D Filters	32
	MLP Hidden Layers	64→32, 32→32
MHAM	Attention Dimension	64

Action Space Pruning	Spatial Radius	5 km
	Average Moving Speed	8 km/h
	Minimum Recommendations	5
CRF	Preference	0.22
	Time	0.18
	Context	0.27
	Penalty	0.33
Asynchronous AC Training	Number of Parallel Threads	8
	n-step return	5
Prioritized Experience Replay (PER)	PER Priority Exponent	0.6
	PER Importance Sampling	0.4
Exploration Strategy	ε-Greedy Rate	0.1
	Softmax Temperature	0.8
Network Optimization	Discount Factor	0.9
	Dropout Rate	0.2
	L2 Weight Decay	0.01

2.4 Deep policy network architecture

The network architecture consists of a preference-context joint encoding layer and a fusion decision layer, with parameters jointly optimized via end-to-end backpropagation.

The preference layer is specifically designed to model the temporal dependencies of a user's historical behavior. The input layer for the user preference subnetwork has a size of 64, corresponding to the embedding dimension of the user's attraction IDs. The context subnetwork receives a 96-dimensional input, combining both contextual features and location status. The total number of parameters in the entire network is approximately 1.5 million, and regularization techniques such as dropout (with a rate of 0.2) and L2 weight decay (λ=0.01) are applied to the fully connected layers. The average inference time per recommendation step is 0.03 seconds. The input is a sequence $\{v_{t-9},...,v_t\}$ of attraction ID from the user's last ten check-ins. This is mapped into a 64-dimensional dense vector (Embedding Size = 64) by the embedding layer and fed into a two-layer unidirectional LSTM (hidden dimension 128, tanh activation). The second-layer LSTM outputs a hidden state $h_i \in R^{128}$ at each time step, forming a set of sequence representation $\{h_1,...,h_{10}\}$. In the following step, a MHAM is applied to dynamically weight UPs [39], [40]. This mechanism concatenates the individual heads after calculating attention for different subspaces and combines the results to generate a final preference representation. This mechanism can understand UPs from multiple subspaces and achieve a dynamic and focused interpretation of a user's historical access behavior. The current state is considered the focus of attention, with the current LSTM hidden state's as the query vector (query), and the LSTM hidden states of all historical accesses as the key/value pairs (key/value), to calculate the alignment weight:

$$e_i=v^T\tanh(W[h_i;s_t]),\alpha_i=\frac{\exp(e_i)}{\sum_j \exp(e_j)} \tag{5}$$

$W\in R^{k\times 256}$, $v\in R^k$ are learnable parameters (k=64). After calculating the attention weights, the model uses these weights to perform a weighted summation of all historical hidden states to obtain a dynamic aggregated final preference representation:

$$h_u=\sum_{i=1}^{10} \alpha_i h_i \tag{6}$$

This mechanism enables the model to dynamically concentrate on the historical accesses that are most relevant to the current decision, enhancing the semantic sensitivity of personalized representation.

The context component processes structured real-time input $c_t\in R^{64}$ and a position vector $p_t\in R^{32}$. These two are concatenated into a 96-dimensional input. Local features are extracted through a one-dimensional convolutional layer (Conv1D, kernel size 3, number of filters 32, ReLU activation), outputting 32 feature maps of length 94. These are then flattened and further nonlinearly transformed through two fully connected MLP layers (64→32 ReLU, 32→32 ReLU), outputting a 32-dimensional context feature vector f_c. This CNN-MLP architecture effectively captures the interactions between multiple context variables. For example, "high congestion combined with low passenger flow" may indicate an abnormal event.

The feature fusion and decision layer concatenate the preference representation $h_u\in R^{128}$ and the contextual features $f_c\in R^{32}$ into a 160-dimensional joint vector $z=[h_u;f_c]$, which is then fed into a three-layer MLP (256→128ReLU, 128→64ReLU, and 64→64ReLU) for high-level abstraction. The output layer employs a Dueling DQN architecture, connecting two branches: the value stream and the advantage stream. The value stream is a single-neuron, fully connected layer that outputs a state value estimate $V(s_t)$; the advantage stream outputs an action advantage vector $A(s_t,a)\in R^{|A_t|}$, which is then combined into a Q value after mean reduction:

$$Q(s_t,a)=V(s_t)+(A(s_t,a)-\frac{1}{|A_t|}\sum_{a'} A(s_t,a')) \tag{7}$$

This structure decouples state value and action difference, improves the stability of Q-value estimation, and is especially suitable for scenarios where the action space changes dynamically.

The network output action probability distribution $\pi(a|s_t)$ is generated by the Actor branch through SoftMax normalization:

$$\pi(a|s_t)=\frac{\exp\left(\frac{Q(s_t,a)}{\tau}\right)}{\sum_{a'}\exp\left(\frac{Q(s_t,a')}{\tau}\right)} \tag{8}$$

The temperature parameter τ=0.8 controls the exploration intensity.

This LSTM-CNN architecture addresses the inadequate modeling capabilities of traditional single-stream networks for heterogeneous inputs [41], [42]. The LSTM-Attention structure accurately captures evolving user interests; the CNN-MLP efficiently handles multidimensional contexts; the Dueling architecture enhances the robustness of value estimation. The overall network achieves a deep joint representation of personalized and dynamic environments while maintaining parameter efficiency.

2.5 Asynchronous advantage actor-critic training mechanism based on experience replay

This study uses the asynchronous advantage AC framework to implement distributed policy training to improve sample efficiency and convergence stability [43]. The entire training process is carried out in 8 parallel execution environment threads, each of which independently simulates a user's decision trajectory in the urban tourism scenario. The simulation of user trajectories is based on a synthetic environment that models urban tourism scenarios, taking into account dynamic changes such as weather and traffic conditions.

To explain why DRL with Actor-Critic is chosen over other adaptive control methods (e.g., backstepping optimization or robust adaptive models), it is noted that this paper’s approach focuses on dynamic, real-time decision-making, balancing long-term user preferences with immediate contextual factors. Backstepping and robust adaptive models, though effective in predictable systems, struggle with unpredictable contextual changes like traffic or weather. Additionally, DRL with Actor-Critic has proven more flexible in optimizing complex, multi-dimensional rewards in real-time dynamic settings, as shown in the experiments.

Each thread initializes a local copy of the policy network, whose parameters are synchronized with the global network. In each trajectory, the system selects action a_t based on the current state’s using an ε-greedy policy (ε = 0.1). After execution, it obtains the reward r_t and the next state’s from the simulation environment and stores the experience tuple (s_t,a_t,r_t,s_{t+1}) in a local replay buffer. An asynchronous gradient update is initiated when the buffer accumulates 32 steps or when the trajectory terminates.

To improve learning efficiency under sparse rewards, this study applies Prioritized Experience Replay (PER). The priority p_i of each experience is determined by its TD (Temporal-Difference) error $\delta_i=|r_t+\gamma V(s_{t+1})-V(s_t)|$, and the sampling probability is calculated based on $P(i)\propto p_i^{\alpha}$ (α =0.6). During training, 16 samples are sampled from the local buffer according to the priority, and the gradient is corrected using the importance sampling weight $w_i=(\frac{1}{N\cdot P(i)})^{\beta}$ （β=0.4） to correct for sampling bias.

Gradient calculation is based on an n-step Q-learning objective. For the sample sequence, the n-step return is calculated:

$$R_t^{(n)}=\sum_{k=0}^{n-1}\gamma^k r_{t+k}+\gamma^n V(s_{t+n}) \tag{9}$$

The critic loss function is the mean square error:

$$L_v=(R_t^{(n)}-V(s_t))^2 \tag{10}$$

Actor loss combines policy gradient and entropy regularization:

$$L_\pi=-\log\pi(a_t|s_t)\cdot A(s_t,a_t)-\lambda H(\pi(\cdot|s_t)) \tag{11}$$

The advantage function $A(s_t,a_t)=R_t^{(n)}-V(s_t)$ and the entropy term H enhance exploration capabilities, with a weight of λ=0.01.

After each update, the local network's gradients are uploaded to the global shared network, and the parameters (learning rate lr=3×10^{-4}, decay rates ρ=0.99, ϵ=10^{-5}) are updated using the RMSprop optimizer. The global network is synchronized to all threads every 10 asynchronous update cycles to ensure consistent policy evolution.

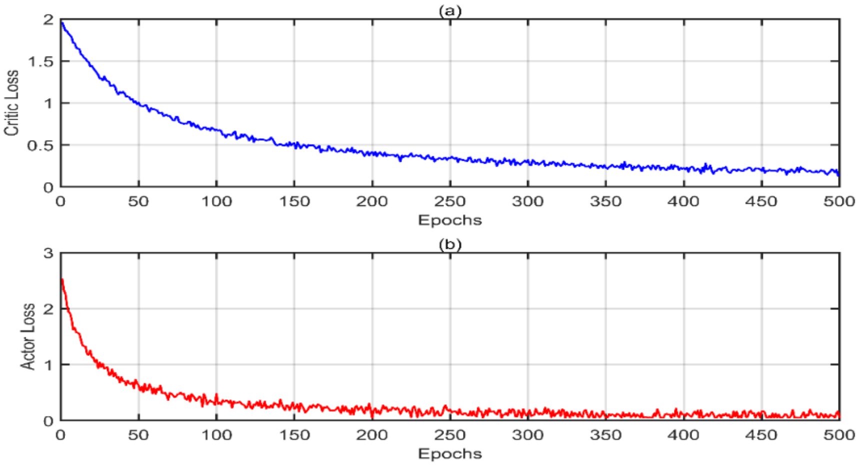

Figure 2: Loss function variation curves; (a). Critic loss variation curve, (b). Actor loss variation curve

Fig. 2 shows how the loss of critics and actors changes with the number of training rounds. Fig. 2(a) depicts the critic loss curve. As training progresses, the critic loss gradually decreases and stabilizes, indicating that the model's estimation of state values is becoming increasingly accurate. Fig. 2(b) shows the actor loss curve. It gradually decreases as training progresses, reflecting the dynamic balance between Exploration and Exploitation (E&E) in the policy network. The loss functions of both the critic and actor networks show a favorable downward trend, validating the model's efficiency. In the AC framework, the critic network continuously optimizes its predictions of state values utilizing the mean squared error loss function. As training progresses, the predicted values become closer to the true values, resulting in a decrease in loss. In the early stages of training, the actor network tends to explore more unknown states, leading to greater loss fluctuations. However, as training progresses, the network gradually learns to make better decisions within known states, resulting in a decrease in loss.

2.6 Recommendation generation mechanism

The recommendation generation mechanism implements a closed-loop deployment from trained policy models to online services, ensuring the system can deliver personalized, dynamically adjusted route recommendations in real-time in real-world travel scenarios. This mechanism, with its core process of state perception, decision-making inference, and feedback updates, operates on a low-latency service architecture.

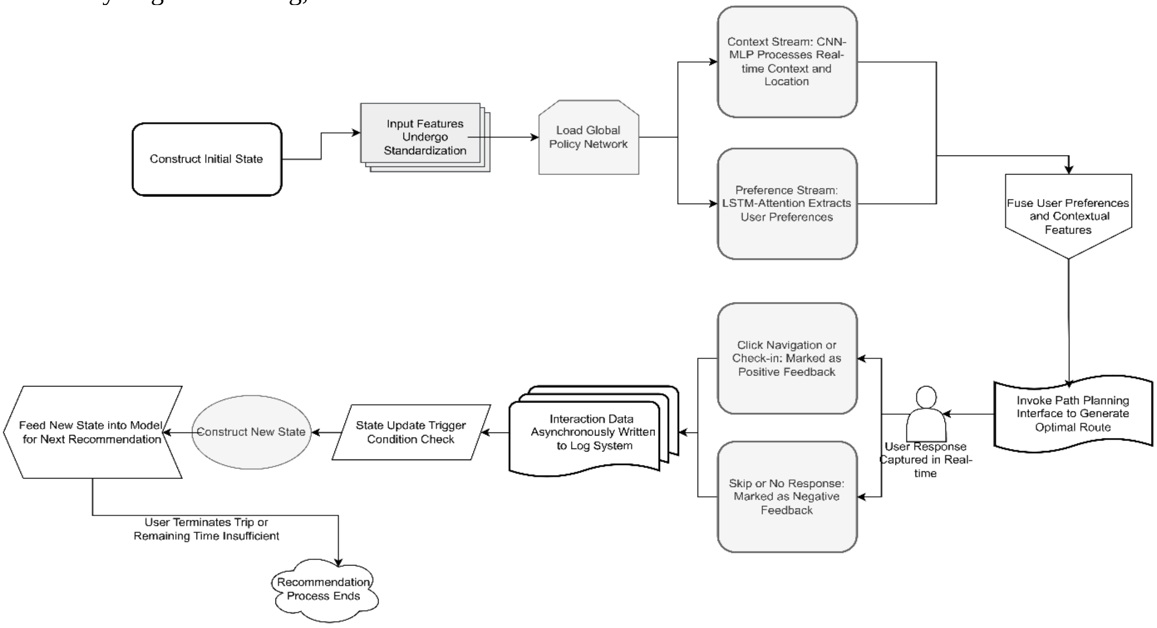
Construct Initial State
Input Features Undergo Standardization
Load Global Policy Network
Context Stream: CNN-MLP Processes Real-time Context and Location
Preference Stream: LSTM-Attention Extracts User Preferences
Fuse User Preferences and Contextual Features
Invoke Path Planning Interface to Generate Optimal Route
User Response Captured in Real-time
Click Navigation or Check-in: Marked as Positive Feedback
Skip or No Response: Marked as Negative Feedback
Interaction Data Asynchronously Written to Log System
State Update Trigger Condition Check
Construct New State
Feed New State into Model for Next Recommendation
User Terminates Trip or Remaining Time Insufficient
Recommendation Process Ends

Figure 3: Closed-loop architecture for real-time recommendation generation

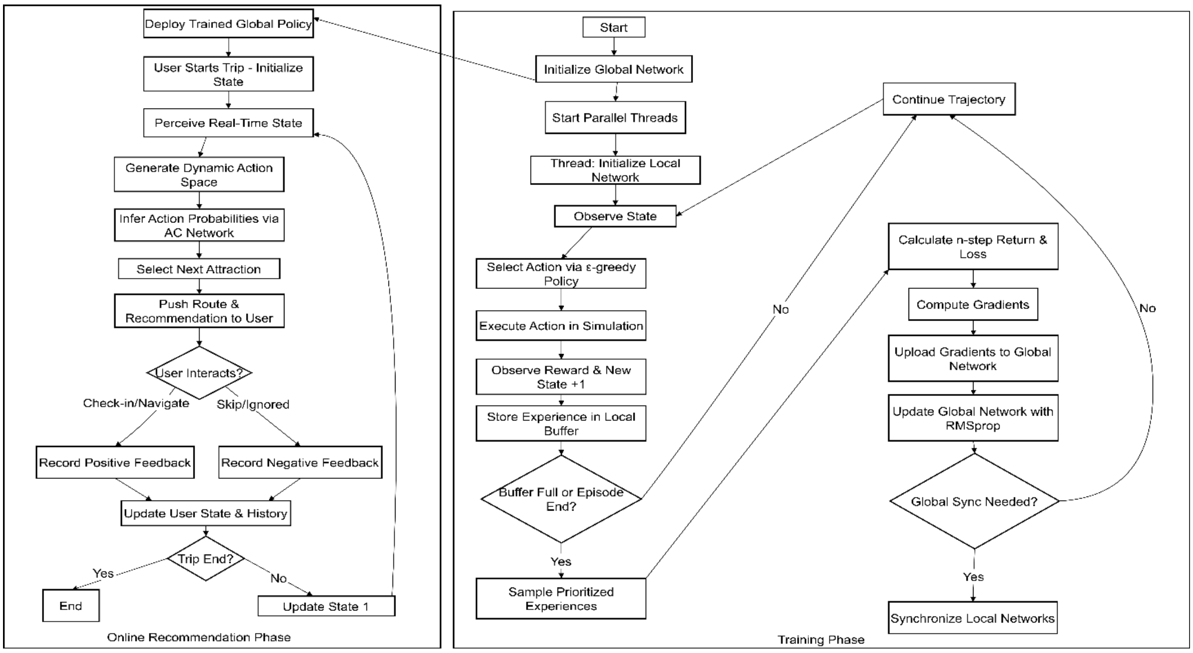
Deploy Trained Global Policy
User Starts Trip - Initialize State
Perceive Real-Time State
Generate Dynamic Action Space
Infer Action Probabilities via AC Network
Select Next Attraction
Push Route & Recommendation to User
User Interacts?
Check-in/Navigate
Skip/Ignored
Record Positive Feedback
Record Negative Feedback
Update User State & History
Trip End?
Yes
No
End
Update State 1
Online Recommendation Phase
Start
Initialize Global Network
Start Parallel Threads
Thread: Initialize Local Network
Observe State
Select Action via ε-greedy Policy
Execute Action in Simulation
Observe Reward & New State +1
Store Experience in Local Buffer
Buffer Full or Episode End?
Yes
No
Sample Prioritized Experiences
Continue Trajectory
Calculate n-step Return & Loss
Compute Gradients
Upload Gradients to Global Network
Update Global Network with RMSprop
Global Sync Needed?
Yes
No
Synchronize Local Networks
Training Phase

Figure 4: Overall algorithm workflow

Fig. 3 illustrates the output structure of the recommended path, and depicts the end-to-end deployment pipeline, where real-time context updates and user feedback form a dynamic recommendation chain for continuous personalization and adaptation throughout the user's journey. After constructing the initial state, all input features are normalized and fed into the loaded global policy network. A CNN-MLP then processes the RTC and location, while an LSTM-Attention framework extracts UPs. The model uses a shared AC network architecture for inference: the current state’s is input, and the encoding branch extracts UPs and contextual features in parallel. After fusion, the MLP (Multilayer Perceptron) and Dueling architecture output the Q-value for each candidate action. For each legal location j in the action space A_t, its Q-value $Q(s_t,j)$ is extracted and converted to an action probability distribution using a SoftMax function:

$$\pi(a=j|s_t)=\frac{\exp(Q(s_t,j)/\tau)}{\sum_{k\in A_t}\exp(Q(s_t,k)/\tau)} \tag{12}$$

The temperature parameter $\tau = 0.8$ controls the smoothness of the output distribution to avoid excessive concentration on a single option.

During the action selection phase, an ε-greedy strategy is utilized to balance E&E: action $a_t=\arg\max_{j\in A_t}Q(s_t,j)$ with the highest Q value is chosen with a 90% probability, and a uniform random sample is taken from A_t with a 10% probability. After selecting an attraction ID, the system invokes a path planning API to generate the optimal route from the current location to the target attraction (including transportation options and estimated travel time). This route is then pushed to the client along with the reasoning for the recommendation (e.g., "This matches your preference for cultural attractions" or "The current weather is suitable for indoor activities").

User responses are captured in real-time: if a user clicks on navigation or checks in to a destination, this is marked as positive feedback and recorded as a valid recommendation. If a user skips a recommendation or remains unresponsive for an extended period, this is considered negative feedback, triggering a signal for fine-tuning the local strategy. All interaction data is asynchronously written to the log system via a message queue for subsequent offline training data updates.

When a user completes their current stop at a scenic spot and moves to a new location, the system triggers a state update. Using a timer (every 30 seconds) or location change detection (displacement > 200 meters), the system recollects real-time contextual data (weather, traffic, and crowd flow), updates the user's location and time state, constructs a new state’s, and re-enters the model to generate the next recommendation, forming a dynamic recommendation chain. This process continues until the user actively terminates their trip or the system determines that there is insufficient time left to visit any new attractions. The overall workflow of the training and online recommendation processes is illustrated in Figure 4.

3 Experiment and verification

3.1 Experimental design

The experimental design aims to validate the comprehensive performance of a dynamic TR recommendation model based on DRL in a real-world urban tourism scenario. A reproducible, high-fidelity simulation evaluation environment is constructed. All experiments are run on a server cluster equipped with NVIDIA Tesla V100 GPUs, using Python 3.9 and PyTorch 1.12.

The data set utilizes FS-NYC and TCI, which hold check-in data gathered in NYC and Tokyo, spanning about 10 months (from April 12, 2012, until February 16, 2013), including 227,428 check-ins for NYC and 573,703 check-ins for Tokyo. Every check-in has a timestamp, GPS location, and a semantic label (indicated by a specific venue type). POI (Point of Interest) category information is supplemented via the Foursquare API, covering 16 categories (such as museums, parks, restaurants, and shopping malls). Ancillary data is acquired in real-time through APIs: weather data comes from the OpenWeatherMap API (updated hourly); traffic congestion index is provided by the Baidu Maps API (based on floating vehicle data); attraction opening hours are retrieved from official websites and stored in a structured format.

The data preprocessing process is as follows: first, abnormal stops with check-in intervals less than 5 minutes are filtered to prevent missed check-ins or short stops from interfering with trajectory continuity; second, the visit sequences of each user are sorted by time, and only valid users with at least 5 check-ins are retained, ultimately retaining 500 users; then, the Word2Vec model is used to train attraction category embedding vectors on all check-in sequences, with a dimension set to 64, for preference matching calculations in the reward function; finally, the original timestamps are parsed into hour and weekday/holiday symbols, and aligned with external data such as weather and traffic by time to construct a context vector corresponding to each check-in.

Data is partitioned using a chronological splitting method: the first 80% of the check-in data on the timeline is used as the training set; the middle 10% is used as the validation set (for hyperparameter tuning and early stopping); the last 10% is used as the test set. This ensures that test user behavior patterns are not leaked during training, preventing future information leakage issues with time series data.

This document presents a dynamic TR recommendation model based on DRL. By building an LSTM-CNN network and applying an MHAM, it deeply integrates UPs and real-time contextual status, designs a multi-objective reward function, and implements end-to-end training based on the AC framework.

The baseline model includes four representative methods:

DQN: Deep Q Network (DQN) uses the same state input, action space, and reward function in this paper;

PageRank-based: this method constructs a transition probability matrix based on the user-attraction interaction graph, calculates attraction importance using the PageRank algorithm, and generates static Top-K recommendations;

PredRNN: a spatiotemporal prediction sequence RNN (Recurrent Neural Network) travel recommendation model that takes user history sequences as input, models spatiotemporal patterns through LSTM, and outputs next visit predictions.

After initializing the user state for each test trajectory, each model runs sequentially until the end of the trip (three consecutive recommendation failures or timeout). The system automatically records the match between each recommendation result and the actual check-in. Hyperparameter settings are determined through grid search, and an early stopping strategy is employed, where training is halted if the validation loss does not improve for 10 consecutive iterations. The parameter values are illustrated in Table 3.

Table 3: Hyperparameter setting values

Parameter	Value
AC Learning Rate	3×10^{-4}
n-step	5
PER Parameter α	0.6
PER Parameter β	0.4
ε-Greedy Exploration Rate	0.1
Discount Factor	0.9

Table 3 shows the parameter settings. This experimental design ensures fair evaluation and real-world relevance. Time division prevents data leakage, multi-source data fusion restores real-world scenarios, and a unified simulation environment eliminates platform differences. The constructed test framework supports automated batch execution and metric collection, providing a reliable data foundation for subsequent performance comparisons.

3.2 Comparison of recommendation accuracy

To quantify the accuracy of the model in personalized recommendations, this study uses the Top-K Hit Ratio (HR@K) as a core evaluation metric to measure the ability of the recommendation list to cover users' actual behavior. The experiment is conducted on the test set constructed in Section 3.1. All models start with the same initial state, generating recommendations round by round and comparing them with the user's actual check-in sequence. The particular execution procedure follows: for each user in the test set, the system extracts the current state S_t from their historical trajectory and inputs it into various models to generate a top-K recommendation list (K=5 and K=10). The set of attraction IDs corresponding to the recommended action a_t is denoted as $R_t^K \subset A_t$. If the attraction g_t that the user actually visits in the next step is in R_t^K, the recommendation is considered a hit. This process is executed slidingly across the entire test trajectory, covering all evaluable time steps.

The hit rate is calculated using the global average form:

$$\mathrm{HR@K} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{I}\left(g_i \in \mathrm{R}_i^K\right) \tag{13}$$

Here, N is the total number of valid evaluation samples (i.e., the number of decision steps where the action space is non-empty and a true next point exists), and I () is the indicator function. This metric reflects the model's capability to forecast the user's next behavior in a dynamic environment.

To ensure evaluation consistency, all models use the same candidate set generation logic and time window alignment mechanism. The proposed model and the DQN dynamic model update their state step by step and make new recommendations. PageRank-based and PredRNN, as sequence prediction models, output fixed-length rankings based on the global graph structure and LSTM hidden states, respectively, and select the top K items as recommendations.

HR@K indicates the percentage of top K recommended attractions that the user actually visits. For each user in the test set, the system extracts their current state from their historical trajectory, generates a top-K recommendation list, and compares this list with the user's actual check-in sequence. If the attraction the user actually visits next is on the recommended list, the recommendation is considered a hit. The hit rate is calculated as a global average, representing the proportion of hits across all valid evaluation samples. Fig. 5 shows the HR@K of each model.

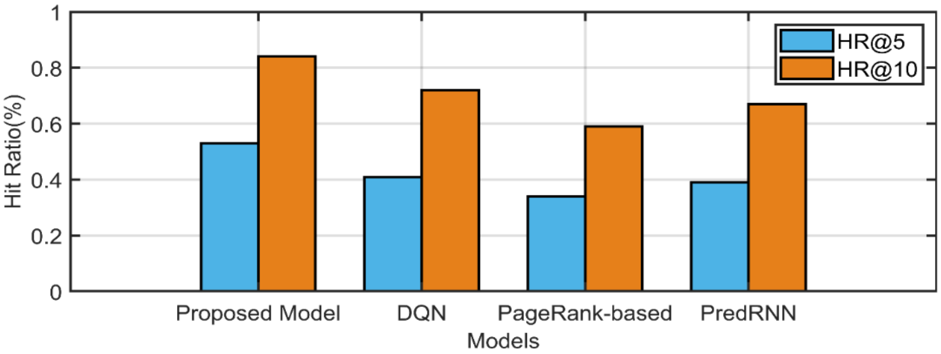

Figure 5: HR@K hit rate

Fig. 5 shows the HR@K hit rate. Under the HR@5 metric, this paper's model achieves a hit rate of 53%, exceeding baseline models such as DQN (41%), PageRank-based models (34%), and PredRNN (39%). When the K value is expanded to 10, the HR@10 hit rate of the paper's model reaches 84%, surpassing the three baseline models of DQN (72%), PageRank-based models (59%), and PredRNN (67%). The paper's model maintains a clear advantage. The HR@10 of the experimental group is higher than that of the GNN recommendation algorithm in Zhang et al.'s study (achieving 52.40%, 75.57%, and 72.43% on the Amazon-Beauty, Amazon-Games, and Amazon-CDs datasets, respectively). This demonstrates that the paper's model not only achieves high-precision recommendations for the first few attractions in the recommendation list, but also maintains high accuracy across a wider range of recommendations, providing users with more diverse choices. The standard deviations and confidence intervals are shown in Table 4.

Table 4: Top-K hit rate statistical significance (Mean ± Standard deviation, 95% Confidence interval)

Model	HR@5	HR@10
Proposed Model	53.0% ± 2.1% [52.1%, 53.9%]	84.0% ± 1.8% [83.3%, 84.7%]
DQN	41.0% ± 2.8% [40.0%, 42.0%]	72.0% ± 2.3% [71.2%, 72.8%]
PredRNN	39.0% ± 3.1% [38.0%, 40.0%]	67.0% ± 2.6% [66.1%, 67.9%]
PageRank-based	34.0% ± 3.5% [33.0%, 35.0%]	59.0% ± 3.0% [58.0%, 60.0%]

Table 4 presents the mean, standard deviation, and 95% confidence intervals for HR@5 and HR@10, calculated over 50 independent runs. The non-overlapping confidence intervals between the proposed model and all baselines confirm its statistically significant performance advantage.

To further compare recommendation accuracy, a coverage metric is added, which is defined as the ratio of the number of unique recommended attractions to the total number of attractions. This metric reflects the breadth of the recommendation system and its ability to discover low-hanging fruit. A high-coverage model can recommend not only popular attractions but also less popular ones that meet UPs, providing users with a richer and more diverse selection. The coverage data is shown in Table 5.

Table 5: Coverage statistics

Model	Coverage (%)
Proposed Model	78.8
DQN	54.7
PredRNN	62.1
PageRank-based	31.5

Table 5 shows that the paper's model has the highest coverage, reaching 78.8%, followed by PredRNN at 62.1%, DQN at 54.7%, and PageRank-based at 31.5%. Due to the paper's model's sensitivity to context and its penalty for repeated behavior, it can break out of its comfort zone of focusing on popular attractions and generate differentiated recommendations for different users and contexts, thus covering a wider range of attractions in the inventory. The PredRNN model can make personalized recommendations based on user history, but lacks exploration capabilities. The DQN model has exploration potential, but its ability to integrate UPs and context is weak. The PageRank-based model, driven by global popularity, repeatedly recommends a small number of popular attractions, often overlooking less popular ones that meet user needs.

3.3 Route rationality assessment

To assess the geographic coherence of routes, the experiment uses three quantitative metrics: average travel time, cross-region rate, and actual travel time. Using actual road network data, the shortest travel time between adjacent recommended attractions is calculated and compared with the model's recommended routes to verify whether they followed optimal or feasible transportation paths. Unreasonable spatial jumps within the route, such as long distances across different zones, are checked, indicating an illogical recommendation logic. Then, considering the overall duration of the recommended route relative to the user's actual available travel time, recommending unfeasible itineraries that exceed the user's time budget are avoided. Calculating the proportion of actual travel time to the total time needs to finish the route to reflect the time efficiency of the recommended route. Fig. 6 shows the average travel time, cross-zone rate, and actual travel time percentage.

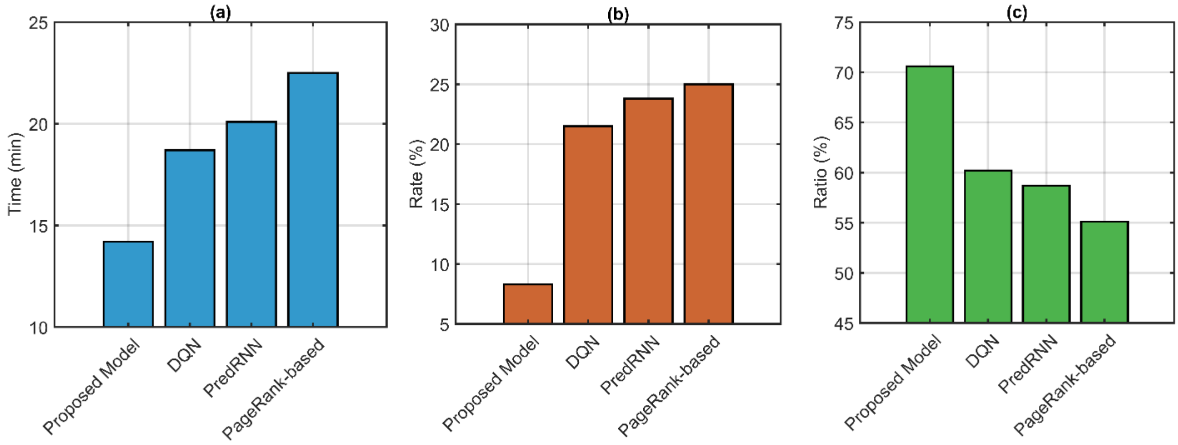

Figure 6: Average travel time, cross-region rate, and percentage of actual travel time; (a) Average travel time, (b) Cross-region rate, (c) Percentage of actual travel time

Fig. 6 shows that the paper's model significantly outperforms the baseline model in average travel time (14.2 minutes). The proposed model shortens average travel time by 8.3 minutes compared to the PageRank-based model, and the proposed model's cross-region rate (8.3%) is 16.7% lower than the PageRank-based model (25%). Compared to DQN, the paper's model shortens average travel time by 4.5 minutes and has a lower cross-region rate than DQN, demonstrating that the paper's model effectively integrates geographic information and generates coherent TRs. The actual travel time in the paper's model accounts for 70.6%, significantly higher than DQN (60.2%), PredRNN (58.7%), and PageRank-based models (55.1%). The paper's model recommends routes with shorter travel times and shorter waiting times, demonstrating superior rationality to the three baseline models.

3.4 Personalized matching satisfaction evaluation

To quantitatively evaluate how well recommendations match users' inherent preferences, this study uses user satisfaction scores as a key metric to assess the model's personalized performance. In experiments, the system creates personalized recommendation routes based on test users' historical check-in data. A panel of 30 evaluators (15 domain experts with advanced degrees and research experience, and 15 experienced travelers) conducts blind reviews. Inter-rater reliability, assessed via Cohen’s Kappa on a random subset, is 0.78 (95% CI [0.72, 0.84]), showing substantial agreement and confirming the evaluation's robustness. The scoring system uses a 5-point Likert scale, with 1 indicating "completely inconsistent with the user's interests" (e.g., recommending a high-intensity outdoor sports venue to a user who prefers cultural and artistic attractions) and 5 indicating "highly consistent with the user's preferences." The evaluation criteria cover five aspects: interest type matching: the consistency of the recommended attractions with the user's historical preferences (e.g., natural landscapes, historical sites, food streets, etc.). Tour pace adaptability: this refers to the degree to which the recommended itinerary's schedule (e.g., a packed morning of sightseeing, a leisurely afternoon) matches the user's historical behavior patterns. Preference intensity responsiveness: this refers to the ability to prioritize frequently visited attractions (e.g., recommending highly relevant museums to a "museum enthusiast"). Dynamic interest tracking: this refers to the ability to capture temporary shifts in user interest during an itinerary (e.g., a shift to indoor attractions during a sudden downpour) while maintaining consistent preferences. Recommendation logic explainability: this refers to the clarity and user understanding of the recommendation rationale (e.g., "Based on your visits to three art galleries last week, I recommend new museums of the same type"). User satisfaction evaluations are shown in Fig. 7.

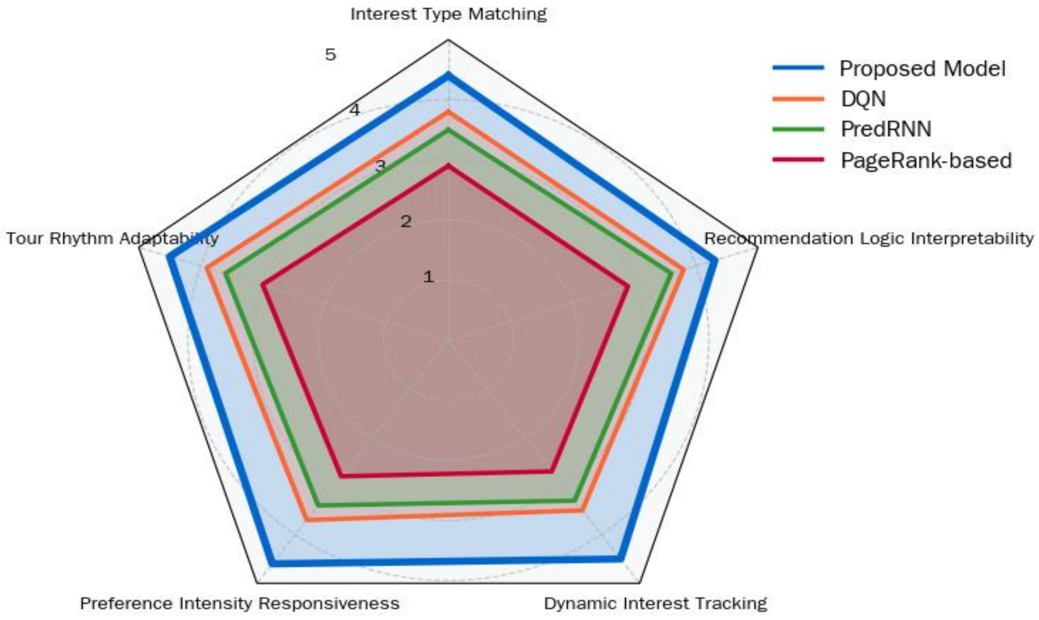

Figure 7: Satisfaction radar chart

As shown in Fig. 7, the satisfaction score for the model in this paper is notably greater than the satisfaction score for the baseline model. The recommended paths generated by the paper's model score 4.4, 4.5, 4.6, 4.5, and 4.3 in the five dimensions of interest type matching, tour rhythm adaptability, preference intensity responsiveness, dynamic interest tracking ability, and recommendation logic interpretability, respectively, with an average score of 4.46. DQN scores 3.8, 3.9, 3.7, 3.5, and 3.8, respectively, with an average score of 3.74. PredRNN scores 3.5, 3.6, 3.4, 3.3, and 3.6, respectively, with an average score of 3.48. The PageRank-based scores are 2.9, 3.0, 2.8, 2.7, and 2.9, respectively, with an average score of 2.86. This shows the effectiveness of the paper's model in continuously tracking UPs during dynamic interactions. In contrast, baseline models, either due to a lack of an explicit preference-context fusion mechanism or the limitations of static ranking logic, struggle to maintain personalization under environmental perturbations. This demonstrates that the paper's model can achieve a higher level of personalized matching.

3.5 Dynamic event and response delay testing

To evaluate the model's robustness and strategy adaptability during unexpected events, this study simulates a "temporary closure of a tourist attraction" to measure the system's reliability and responsiveness in providing alternative recommendations under extreme conditions. The assessment focuses on the model's closed-loop performance from plan failure to new route generation, demonstrating its fault tolerance and real-world adaptability in tourism.

The specific implementation process is as follows: during the testing phase, when a user completes their stop at a current attraction and is about to proceed to the next recommended destination, the system determines whether the destination is a "park-type" POI. If so, a "temporary closure" event simulation is triggered. The closed attraction is forcibly removed from the candidate set, and all subsequent recommendations are generated with respect to the updated action space. Upon activation, the system marks the attraction as "closed" and forcibly removes it from the candidate action space A_t. The weather variable is injected deterministically based on real-time weather data, ensuring that the simulated conditions are as realistic as possible. Simultaneously, the context vector c_t is updated, injecting "weather deterioration" or "crowd limit exceeded" flags to simulate real-world closure reasons.

After a trigger event, the system immediately re-executes the recommendation process: based on the updated state' s , the set of reachable candidates is recalculated, excluding closed attractions and similar high-risk outdoor POIs. Accessible, open, and complementary alternative attractions (such as museums, shopping malls, and indoor exhibition halls) are prioritized. Any recommendations that are repeats from previous attractions are penalized with a -1 score to discourage repetition. The model outputs a new action probability distribution $\pi(a|s_t)$. If a legitimate and non-duplicate alternative attraction is recommended within one minute, it is considered a "successful transfer". The experiment is tested with 50 independent events to ensure statistical significance.

Two core metrics are measured: transfer success rate and response latency. The transfer success rate reflects the model's ability to adjust its strategy within a constrained action space, measuring the rate at which the model successfully re-executes recommendations during a "temporary shutdown" event. A higher rate indicates a stronger ability to propose a new plan when the original plan fails. Fig. 8 illustrates the transfer success rate curve.

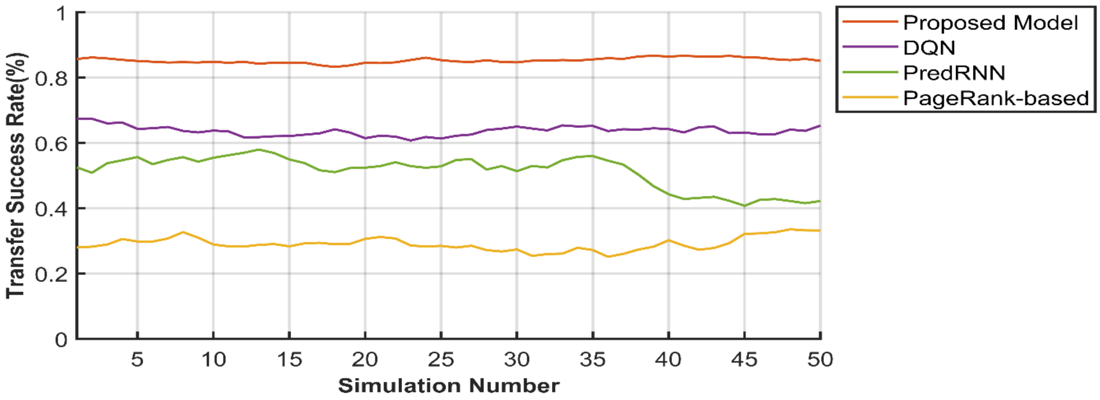

Figure 8: Transfer success rate curve

Fig. 8 shows that the paper's model's success rate significantly outperforms the three baseline models, reaching an average success rate of 85.3% with minimal fluctuation, demonstrating its ability to successfully handle unexpected situations in the vast majority of cases. In contrast, the DQN model has a lower average success rate of approximately 63.7%. The PredRNN and PageRank-based models perform even worse, with average success rates of 51.3% and 29.1%, respectively, and exhibiting significant fluctuations, indicating their limited adaptability to dynamic events. In the paper's model, when the "Attraction Closed" flag in the state vector is updated, the policy network immediately detects this change, enabling efficient and stable migration. While DQN also uses RL, it lacks deep modeling of UPs and an explicit penalty mechanism, resulting in sluggish and unstable responses to sudden state changes. PredRNN, as a sequence prediction model, relies too heavily on historical access patterns, making it difficult to dynamically adjust beyond the preset path. PageRank-based models are rarely able to generate effective alternatives and have the lowest success rate.

Response latency is the time interval (in seconds) from event triggering to the output of a new recommendation, accurately recorded by system logs. A latency of less than 3 seconds is considered efficient, while a latency exceeding 5 seconds may affect user experience. Response delay box plot is shown in Fig. 9.

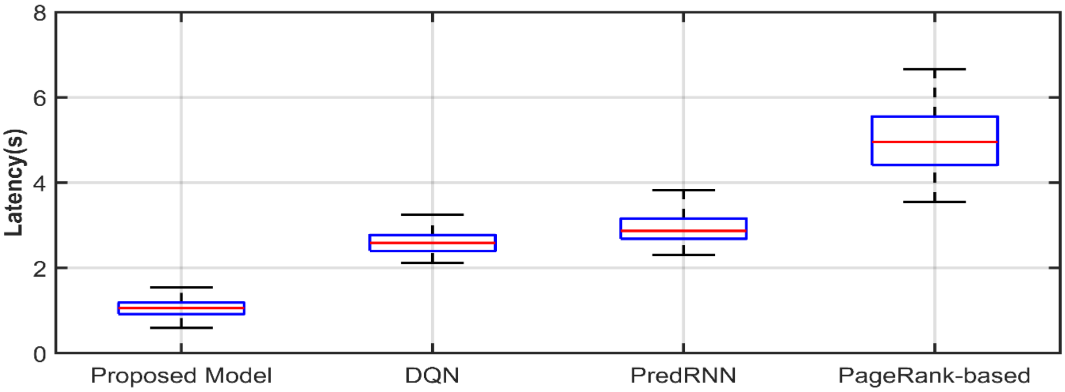

Figure 9: Response delay box plot

Fig. 9 compares response latencies. The horizontal axis signifies the four models, and the vertical axis denotes response latency. The data is based on the results of 50 independent tests. The data shows that the proposed model has extremely low response latency, with a maximum latency of 1.56 seconds and a minimum latency of 0.61 seconds, with a median of approximately 1.07 seconds. The overall distribution is compact, and the response is efficient and stable. In contrast, the DQN and PredRNN models experience significantly increased latency, with medians of approximately 2.61 seconds and 2.89 seconds, respectively. These wider bins indicate that their inference processes take longer and are more volatile, with maximum latencies of 3.26 seconds and 3.85 seconds, respectively, indicating slightly slower response times. The PageRank-based model has the highest latency, with a median of 4.97 seconds and a maximum of 6.68 seconds. This sluggish response to dynamic events may impact user experience. The paper's model can quickly calculate the new action probability distribution through efficient inference after the state vector is updated. However, standard deep learning models such as DQN and PredRNN require high computational overhead when processing high-dimensional states, while the latter requires reprocessing the entire historical sequence, resulting in high inference latency. The mean, median, standard deviation (SD), minimum, and maximum values are presented in Table 6.

Table 6: Statistical summary of response latency (in seconds)

Model	Mean	Median	SD	Minimum	Maximum
Proposed Model	1.05	1.07	0.18	0.61	1.56
DQN	2.58	2.61	0.22	2.13	3.26
PredRNN	2.84	2.89	0.29	2.33	3.85
PageRank-based	4.92	4.97	0.65	3.57	6.68

4 Discussion

This study proposes a deep reinforcement learning framework integrating LSTM-CNN and Multi-Head Attention Mechanism (MHAM) for real-time personalized travel route recommendation. On the FS-NYC and TCI datasets, the model achieves HR@5 of 53% and HR@10 of 84% (Figure 5), outperforming baselines due to its effective integration of User Preferences (UPs) and Real-Time Contexts (RTCs), where LSTM captures long-term behavior patterns, CNN-MLP processes contextual data (e.g., weather, traffic), and MHAM enables dynamic, fine-grained interest modeling by attending to relevant historical visits. The model also achieves high coverage (78.8%, Table 5), indicating strong diversity, driven by the Composite Reward Function (CRF) which uses a duplicate penalty and context-adaptive reward to promote exploration and mitigate the "filter bubble." Under dynamic events like attraction closures, it achieves an 85.3% migration success rate and a 1.07-second median response latency (Figures 8–9), demonstrating robust adaptability through its end-to-end AC architecture, which responds immediately to state changes, unlike the slower DQN, PredRNN, and static PageRank-based models.

In summary, the model's performance arises from the synergistic integration of LSTM-CNN, MHAM, CRF, and AC, enabling accurate, diverse, and highly adaptive real-time recommendations.

5 Conclusion

This study addresses the poor adaptability of recommendation systems in dynamic tourism scenarios by proposing a DRL model that integrates UPs with real-time contextual awareness. The proposed framework uses an LSTM-CNN-MHAM architecture within an Actor-Critic framework, guided by a Composite Reward Function, to achieve adaptive personalization by dynamically focusing on relevant historical behaviors based on real-time context. Based on the AC framework, a CRF is designed to drive the model to learn personalized and context-adaptive decision-making strategies. Experimental results show that the model achieves a Top-5 hit rate of 53% and a Top-10 hit rate of 84% on the FS-NYC and TCI, with a MRL of 1.07 seconds. It can be recognized that the FS-NYC and TCI datasets, though valuable, may not fully capture global travel diversity. Future work can test datasets from developing cities with less structured data and explore solutions to cold-start problems for new users, possibly using collaborative filtering or hybrid methods to enhance initial recommendations. This research effectively achieves collaborative modeling of user needs and dynamic environments, providing a personalized recommendation solution that combines high precision and real-time performance for smart tourism. The proposed model performs effectively in dynamic urban tourism but faces limitations like the 'cold-start' problem for new users without historical data, solvable through collaborative filtering or hybrid models. Scaling may raise computational costs from real-time processing, alleviated by model pruning or distributed computing. Future research can boost scalability for more users and contexts and incorporate adaptive event-triggered strategies to enhance responsiveness in complex urban environments.

Authorship contribution statement

Dan ZHANG: Supervision, Conceptualization, Project administration, Writing-Original draft preparation.

Conflicts of interest

The authors state that they have no conflict of interest concerning the publication of this paper.

Author statement

All authors have read and approved the manuscript, fulfilling the authorship criteria outlined earlier, and each author affirms that it represents honest work.

Funding

This work was supported by project of Chongqing Municipal Education Commission Humanities and Social Sciences Research Project: Research on the Mechanism of AI-Enabled Integrated and Innovative Development of Chongqing's Intangible Cultural Heritage and Tourism (No.25SKGH367).

Ethical approval

All authors have personally contributed significantly to the work behind this paper and will publicly stand by its content.

Reference

[1] Y. Zhang, M. Sotiriadis, and S. Shen, “Investigating the impact of smart tourism technologies on tourists’ experiences,” *Sustainability*, 14(5): 3048, 2022. https://doi.org/10.3390/su14053048

[2] C. Huda, A. Ramadhan, A. Trisetyarso, E. Abdurachman, and Y. Heryadi, “Smart tourism

recommendation model: a systematic literature review," *International Journal of Advanced Computer Science and Applications*, 12(12): 2021. DOI:10.14569/IJACSA.2021.0121222

[3] Y. Wang, M. Wang, K. Li, and J. Zhao, "Analysis of the relationships between tourism efficiency and transport accessibility—A case study in Hubei province, China," *Sustainability*, 13(15): 8649, 2021. https://doi.org/10.3390/su13158649

[4] C. M. Hall and Y. Ram, "Weather and climate in the assessment of tourism-related walkability," *Int J Biometeorol*, 65(5): 729–739, 2021. https://doi.org/10.1007/s00484-019-01801-2

[5] E. J. Wilkins and L. Horne, "Effects and perceptions of weather, climate, and climate change on outdoor recreation and nature-based tourism in the United States: A systematic review," *PLOS Climate*, 3(4): e0000266, 2024. https://doi.org/10.1371/journal.pclm.0000266

[6] R. Maršanic, E. Mrnjavac, D. Pupavac, and L. Krpan, "Stationary traffic as a factor of tourist destination quality and sustainability," *Sustainability*, 13(7): 3965, 2021. https://doi.org/10.3390/su13073965

[7] M. Kay Smith, I. Pinke-Sziva, Z. Berezvai, and K. Buczkowska-Gołąbek, "The changing nature of the cultural tourist: motivations, profiles and experiences of cultural tourists in Budapest," *Journal of Tourism and Cultural Change*, 20(1–2): 1–19, 2022. https://doi.org/10.1080/14766825.2021.1898626

[8] A. Saxena, N. K. Sharma, D. Pandey, and B. K. Pandey, "Influence of tourists satisfaction on future behavioral intentions with special reference to desert triangle of Rajasthan," *Augmented Human Research*, 6(1): 13, 2021. https://doi.org/10.1007/s41133-021-00052-4

[9] A. S. K. Xin, H. Y. Ting, and A. F. Atanda, "Trends in tourism recommendation systems: A review," *Journal of Computing Research and Innovation*, 9(2): 85–107, 2024. DOI:10.32628/CSEIT23902105

[10] R. Prahadeeswaran, "A comprehensive review: The convergence of artificial intelligence and tourism," *International Journal for Multidimensional Research Perspectives*, 1(2): 12–24, 2023.

[11] S. Vada, K. Dupre, and Y. Zhang, "Route tourism: a narrative literature review," *Current Issues in Tourism*, 26(6): 879–889, 2023. https://doi.org/10.1080/13683500.2022.2151420

[12] Mu. A. K. Anuar and A. Marzuki, "Critical elements in determining tourism routes: A systematic literature review," *Geografie*, 127(4): 319–340, 2022. 10.37040/geografie.2022.010

[13] Boulkroune, F. Zouari, and.A. Boubellouta, "Adaptive fuzzy control for practical fixed-time synchronization of fractional-order chaotic systems," *Journal of Vibration and Control*, 10775463251320258, 2025. https://doi.org/10.1177/10775463251320258

[14] Boulkroune, Abdesselem, et al. "Output-Feedback Controller Based Projective Lag-Synchronization of Uncertain Chaotic Systems in the Presence of Input Nonlinearities," *Mathematical Problems in Engineering* , 2017 (1): 8045803, 2017. https://doi.org/10.1155/2017/8045803

[15] Zouari, Farouk, K. Ben Saad, and M. Benrejeb, "Robust neural adaptive control for a class of uncertain nonlinear complex dynamical multivariable systems," *International Review on Modelling and Simulations* , 5(5): 2075-2103, 2012.https://www.scopus.com/pages/publications/84873265173

[16] Zouari, Farouk, Kamel Ben Saad, and Mohamed Benrejeb. "Adaptive backstepping control for a class of uncertain single input single output nonlinear systems." *10th International Multi-Conferences on Systems, Signals & Devices 2013 (SSD13)*. IEEE, 2013. DOI: 10.1109/SSD.2013.6564134

[17] Rigatos, G., et al. "Nonlinear optimal control for a gas compressor driven by an induction motor." *Results in Control and Optimization* 11: 100226, 2023. https://doi.org/10.1016/j.rico.2023.100226

[18] Zouari, Farouk, Kamel Ben Saad, and Mohamed Benrejeb. "Adaptive backstepping control for a single-link flexible robot manipulator driven DC motor." *2013 International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, 2013: 864-871, 2013. DOI: 10.1109/CoDIT.2013.6689656

[19] Ma, Xiaohang, Zhanyong Wu, and Ling Hu, "Deep Reinforcement Learning for Personalized Route Planning in Agricultural Tourism: A DDPG and Genetic Algorithm Approach," *Informatica*, 49(28), 2025. https://doi.org/10.31449/inf.v49i28.6865

[20] Zhu X. "Multi-Task Deep Reinforcement Learning for Intelligent Logistics Path Planning and Scheduling Optimization," *Informatica*, 49(20), 2025. https://doi.org/10.31449/inf.v49i20.7996

[21] M. Zhang, S. Wu, X. Yu, Q. Liu, and L. Wang, "Dynamic graph neural networks for sequential recommendation," *IEEE Trans Knowl Data Eng*, 35(5): 4741–4753, 2022. DOI: 10.1109/TKDE.2022.3151618

[22] Z. Liu, L. Yang, Z. Fan, H. Peng, and P. S. Yu, "Federated social recommendation with graph neural network," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(4): 1–24, 2022. https://doi.org/10.1145/3501815

[23] G. K. Shyam, "DRL-HIFA: a dynamic recommendation system with deep reinforcement learning based Hidden Markov Weight Updation and factor analysis," *Multimed Tools Appl*, 83(29):72819–72843, 2024. https://doi.org/10.1007/s11042-024-18296-8

[24] X. Zhang, Y. Shang, Y. Ren, and K. Liang, "Dynamic multi-objective sequence-wise recommendation framework via deep reinforcement learning," *Complex & Intelligent Systems*, 9(2): 1891–1911, 2023. https://doi.org/10.1007/s40747-022-00871-x

[25] D. Shrestha, T. Wenan, D. Shrestha, N. Rajkarnikar, and S.-R. Jeong, "Personalized Tourist recommender system: a data-driven and machine-learning approach," *Computation*, 12(3): 59, 2024. https://doi.org/10.3390/computation12030059

[26] J. C. S. Núñez, J. A. Gómez-Pulido, and R. R. Ramírez, "Machine learning applied to tourism: A systematic review," *Wiley Interdiscip Rev Data Min Knowl Discov*, 14(5): e1549, 2024. https://doi.org/10.1002/widm.1549

[27] X. Chen, H. Zhang, C. U. I. Wong, and Z. Song, "Context-Aware Markov Sensors and Finite Mixture Models for Adaptive Stochastic Dynamics Analysis of Tourist Behavior," *Mathematics*, 13(12): 2028, 2025. https://doi.org/10.3390/math13122028

[28] J. Yoon and C. Choi, "Real-time context-aware recommendation system for tourism," *Sensors*, 23(7): 3679, 2023. https://doi.org/10.3390/s23073679

[29] Z. Wang, "Intelligent recommendation model of tourist places based on collaborative filtering and user preferences," *Applied Artificial Intelligence*, (37): 1, p. 2203574, 2023. https://doi.org/10.1080/08839514.2023.2203574

[30] X. Nan and X. Wang, "Design and implementation of a personalized tourism recommendation system based on the data mining and collaborative filtering algorithm," *Comput Intell Neurosci*, 2022(1): 1424097, 2022. https://doi.org/10.1155/2022/1424097

[31] G. Liu *et al.*, "Individualized tourism recommendation based on self-attention," *PLoS One*, 17(8): e0272319, 2022. https://doi.org/10.1371/journal.pone.0272319

[32] C.-Y. Tsai, K.-W. Chuang, H.-Y. Jen, and H. Huang, "A tour recommendation system considering implicit and dynamic information," *Applied Sciences*, 14(20): 9271, 2024. https://doi.org/10.3390/app14209271

[33] N. Mou *et al.*, "Personalized tourist route recommendation model with a trajectory understanding via neural networks," *Int J Digit Earth*, 15(1): 1738–1759, 2022. https://doi.org/10.1080/17538947.2022.2130456

[34] F. Zhou, P. Wang, X. Xu, W. Tai, and G. Trajcevski, "Contrastive trajectory learning for tour recommendation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(1): 1–25, 2021. https://doi.org/10.1145/3462331

[35] H. Lee and Y. Kang, "Mining tourists' destinations and preferences through LSTM-based text classification and spatial clustering using Flickr data," *Spatial Information Research*, 29(6): 825–839, 2021. https://doi.org/10.1007/s41324-021-00397-3

[36] X. Xiao, C. Li, X. Wang, and A. Zeng, "Personalized tourism recommendation model based on temporal multilayer sequential neural network," *Sci Rep*, 15(1): 382, 2025. https://doi.org/10.1038/s41598-024-84581-z

[37] S. Alshammrei, S. Boubaker, and L. Kolsi, "Improved Dijkstra algorithm for mobile robot path planning and obstacle avoidance," *Comput. Mater. Contin*, 72(3): 5939–5954, 2022. DOI: 10.32604/cmc.2022.028165

[38] L. Liu *et al.*, "Path planning for smart car based on Dijkstra algorithm and dynamic window approach," *Wirel Commun Mob Comput*, 2021(1):8881684, 2021. https://doi.org/10.1155/2021/8881684

[39] X. Feng, Z. Ma, C. Yu, and R. Xin, "MRNDR: multihead attention-based recommendation network for drug repurposing," *J Chem Inf Model*, 64(7): 2654–2669, 2024. https://doi.org/10.1021/acs.jcim.3c01726

[40] G. Liao, X. Deng, C. Wan, and X. Liu, "Group event recommendation based on graph multi-head attention network combining explicit and implicit information," *Inf Process Manag*, 59(2): 102797, 2022. https://doi.org/10.1016/j.ipm.2021.102797

[41] H. An and N. Moon, "Design of recommendation system for tourist spot using sentiment analysis based on CNN-LSTM," *J Ambient Intell Humaniz Comput*, 13(3): 1653–1663, 2022. https://doi.org/10.1007/s12652-019-01521-w

[42] T. Nguyen-Da, Y.-M. Li, C.-L. Peng, M.-Y. Cho, and P. Nguyen-Thanh, "Tourism demand prediction after COVID-19 with deep learning hybrid CNN–LSTM—Case study of Vietnam and provinces," *Sustainability*, 15(9): 7179, 2023. https://doi.org/10.3390/su15097179

[43] M. Bukhari, M. Maqsood, and F. Adil, "An actor-critic based recommender system with context-aware user modeling," *Artif Intell Rev*, 58(5): 138, 2025. https://doi.org/10.1007/s10462-025-11134-9