Personalised Urban Daily Scheduling System with POI Using Deep Reinforcement Learning

Dr B Vani[1], Levin K Varghese[2], Tejashree M S[3], Bindhu S[4], Santosh Reddy[5]
H.O.D[1], Student[2], Student[3], Student[4], Student[5]
Sambhram Institute of Technology, Bengaluru

***Abstract*—An Intelligent Itinerary Optimisation System is being developed through this project, inspired by the research paper "Daily Schedule Recommendation in Urban Life Based on Deep Reinforcement Learning." The system is meant to help users create daily travel schedules that are not only efficient but also tailored to their tastes, priorities, and time constraints. While conventional static mapping tools provide fixed information, this system reacts to the user's input by considering the Points of Interest (POIs), start and end times, travel modes, and even the current situations to offer the best possible and usable schedules. Essentially, the system utilises the Google Maps API to obtain live data on travel time, traffic flow, and distance metrics. By utilising such data, it employs heuristic optimisation techniques along with pruning strategies to eliminate low-priority POIs or infeasible routes, while ensuring that user constraints, such as fixed appointments or must-visit locations, are not violated. The itinerary creator not only generates the base plan of the user input but also one or more optimised route suggestions, thus providing users with the opportunity to compare different travel options and flexibility. By putting efficiency first, the model is set to keep the total travel time at a minimum, maximise the feasibility of the schedule, and enhance the user's general experience. The present implementation serves as a rule-based heuristic optimiser that enables the integration of Deep Reinforcement Learning (DRL) in future developments. The subsequent stage will see the system turning into an adaptive learning model that will be able to continually update its suggestions based on user feedback, behavioural trends, and environmental factors like traffic, weather, or city events. In fact, the ultimate ambition is to create a self-learning, context-aware travel assistant that provides intelligent, personalised, and real-time itinerary recommendations.**

***Index Terms*—DAFB, Daily schedule recommendation (DSR), Itinerary Planning, Deep Reinforcement Learning, RNN, Markov Decision Process (MDP)**

I. Introduction

People in the urban environment of today continue to look for more intelligent ways to have a good handle on their day-to-day activities. Rapid urbanisation has led to challenges such as traffic congestion, changing weather, and an increasing variety of urban facilities for both recreation and functionality [1]. As a result, the demand for intelligent scheduling and adaptive daily planning has grown steadily over time [2]. Residents often face multiple and sometimes conflicting goals—such as reaching medical appointments on time, exploring new venues, or minimising overall travel duration. Integrating these competing factors into a single feasible plan remains a complex problem, especially under dynamic conditions such as traffic or weather changes. Such goals might be getting to the closest hospital or doctor's office appointment on time, visiting another place for new exciting experiences, or trying to shorten the duration of travel. Trying to merge all these factors into one plan is not only time-consuming but also inherently unachievable, especially if one takes into account the fact that weather conditions in particular can change rapidly during the day.

Recent advancements in Artificial Intelligence (AI) and Machine Learning (ML) have opened new possibilities to address these urban scheduling challenges. Among these, Deep Reinforcement Learning (DRL) has emerged as a dominant framework for sequential decision-making, enabling systems to autonomously learn from interactions with their environment and make optimised, context-aware decisions [3]. The foundational work, "*Daily Schedule Recommendation in Urban Life Based on Deep Reinforcement Learning*", demonstrated how DRL could simulate human decision-making for itinerary optimisation and daily time management [4]. This study has inspired subsequent research into intelligent travel planning systems that incorporate dynamic, data-driven, and user-centric approaches.

Therefore, this research builds upon those principles to develop an intelligent, reinforcement-learning-based itinerary optimisation system capable of generating efficient, personalised, and adaptive travel schedules for urban users [5]. To better situate the problem space and clarify the technical contributions of this work, the research next provides a structured overview of the motivation behind daily schedule optimisation, followed by the design choices that guided the current implementation. This contextual foundation enables a clear understanding of the challenges addressed by our proposed RL-DAFB framework.

A. Background and Motivation

Based on this conceptual foundation, "*Personalised Urban Daily Scheduling System*" aims to create an intelligent travel planning and optimisation tool that generates, compares, and refines user itineraries based on their defined variables, preferences, and real-world contextual data. This work extends the ideas presented in "*Daily Schedule Recommendation in Urban Life Based on Deep Reinforcement Learning*" by incorporating priority-based pruning and adaptive route generation to achieve efficient and feasible daily schedules [6]. It is a step towards a practical implementation of the theoretical ideas from the referred research that is focused on the integration of data-driven route calculation with priority-based planning and pruning to achieve feasible, efficient, and adaptive daily itineraries.

The distinction between traditional navigation and intelligent scheduling forms the core motivation behind this system. While navigation primarily focuses on distance minimisation, intelligent itinerary management integrates dynamic environmental, temporal, and user-centric factors to produce context-aware recommendations [7]. The integration of live traffic conditions, travel times, and meteorological data enables the model to simulate human-like decision-making in activity selection and pruning.

As cities evolve into interconnected smart ecosystems, the necessity for adaptive scheduling systems capable of real-time data communication through APIs, IoT devices, and public datasets has grown significantly [8]. This system lays the groundwork for future AI-driven, real-time adaptive itinerary optimisation frameworks that align with the broader smart city vision.

Existing navigation systems remain largely static, focusing only on route optimisation [9]. In contrast, the proposed framework aspires to holistic itinerary management, accounting for behavioural preferences, contextual awareness, and environmental adaptation. By dynamically incorporating variables such as traffic density, POI popularity, and weather conditions, the model can recommend optimal travel sequences while maintaining flexibility for user-driven adjustments based on time and personal priorities.

B. Current Implementation Overview

At its current phase of evolution, the Intelligent Itinerary Optimisation System (IIOS) integrates several interdependent modules into a cohesive framework designed for seamless daily route and travel optimisation.

The User Input Module serves as the main interaction point between the user and the system, collecting personalised inputs such as Points of Interest (POIs), their dwell durations, visit priorities, preferred travel mode (e.g., driving, walking, bicycling, transit), and time constraints for the day [10]. This customisation data enables itinerary generation that closely mirrors individual user preferences and contextual needs.

Database Integration, which is the following component, allows the system to interact with a local SQLite database where

information on cached POI is stored [11]. The stored information includes the names and the geographical coordinates (latitude and longitude) of the places, as well as the metadata. This helps in the quick and accurate retrieval of the location without the need for real-time network requests; thus, efficiency and reliability are enhanced.

The Route Estimation Engine is basically the part of the system that executes various computational tasks, employing the Google Maps API to determine the distance and time taken for the travel between the two places [12]. This integration guarantees that the itineraries would be formed based on the present situations on the roads and thus, travel time expectations would be realistic. Besides that, the Weather-Aware Contextualization module can dynamically change the route estimates through live weather data; thus, weather effects on the route, like rain, heat, etc., can be factored in. This attribute increases the usefulness and trustworthiness of the generated itineraries.

At the heart of the system is the Optimisation Layer, which works to build and compare a number of different travel schedules by means of heuristic algorithms. Firstly, it creates a User Plan that depicts the user's original order of POIs without any changes. Subsequently, an optimised plan is generated by simply changing the order or cutting out those POIs that have been considered based on factors such as the level of priority, availability of time, and efficiency of the route. If there are multiple solutions for a certain problem, the system is also able to come up with a third optimisation plan by combining different strategies from the alternative plans it can generate. Hence, the users can have the option to select their preferred plans from the available travel strategies.

If the optimisation process is to be successful, the Pruning Algorithm should also be considered as one of its most important elements. It smartly leaves out lower-ranking POIs in case the sum of the durations for all the chosen POIs is greater than the user time limits have been set. By doing so, the ultimate travel plan is not only kept realistic and achievable, but it is also in line with the constraints of the user; thus, the unnecessary detours and delays can be kept to a minimum.

The research, in the end, equips learners with a comparative visualisation of the generated itineraries by showing the variations in the total travel time, the overall distance, and the POIs specifically included or left out. This open comparison is what gives the power to the users to grasp the trade-offs between convenience, coverage, and efficiency, thus assisting them in making an informed decision. To sum up, the present version already accomplishes a successful demonstration of the potential of flexible, data-driven itinerary planning, which, in turn, sets a solid groundwork for the subsequent incorporation of Deep Reinforcement Learning (DRL) to increase personalisation and real-time adaptability capabilities.

C. Need for Reinforcement Learning Integration

The existing framework is quite efficient at the moment in terms of producing optimised travel plans via the use of heuristic algorithms and data-driven computation, but it still does not possess the feature of being able to learn and consequently modify its behaviour as per user conduct or in the case of fluctuating real-world situations [13]. To rise over these hurdles, this undertaking is looking to incorporate Reinforcement Learning (RL) so as to make the system a self-learning and self-adjusting one. Reinforcement Learning equips the system with the capability of gradually improving itinerary recommendations by employing the user interactions, preferences, and contextual outcomes as the learning material through time, thanks to the dynamic feedback mechanism that it brings in.

They use the Markov Decision Process (MDP) to represent the decision of the next step in itinerary planning [14]. This way, every move, such as next Point of Interest (POI) selection, travel mode decision, or activity dropping decision, could be considered as an action in a Markov Decision Process (MDP) that accounts for the new environment that is taken and, hence, leads to the next state. Time efficiency, user satisfaction, and utilisation of resources within set constraints are only some of the factors whose combined outcome forms the reward signal. By the repeated iterations, the RL agent upgrades its decision-making policy, so this eventually leads to having not only the user needs met but also the external conditions, such as traffic, events, or weather, considered as well, and thus the resulting travel schedules are smarter and more balanced.

Later, the use of Deep Reinforcement Learning (DRL) will greatly expand the system's powers in terms of adaptability, sustainability, personalisation, and scalability [15]. The system will learn the personalised travel habits, preferred activity types, and movement patterns of the user from which it will form the best recommendations. The model will have high adaptability characteristics as a result of its ability to respond dynamically to real-time inputs such as traffic congestion, sudden weather changes, or local events. Moreover, it will be able to handle the mobility data of a large number of users or even entire cities and thus perform urban travel optimisation on a large scale. In terms of the environment, it will facilitate the use of clean energy sources in transport, as well as become a low-carbon route generator.

The incorporation of DRL is a leap forward that positions this work at the confluence of Artificial Intelligence, Data Analytics, and Smart City research [16]. Such a technique takes the challenge of itinerary optimisation out of the realm of the static, rule-based heuristics and into that of a system that is constantly learning and thus capable of dealing with user behaviour and environmental complexities.

The research, apart from increasing travel efficiency through the use of intelligent scheduling, also lessens the cognitive load since it automates the process of itinerary design, incorporates up-to-date environmental data for feasible outcomes, and establishes a foundation for the adaptive, learning-based recommender systems.

II. LITERATURE REVIEW

The use of Deep Reinforcement Learning (DRL) to solve scheduling and recommendation issues has been highly influential in various fields, thus paving the way for schedule recommendation systems that are geared towards our daily lives. In network scheduling, Kim et al. (2024) have shown that DRL through adaptive scheduling can bring down latency and enhance the reliability of wireless time-sensitive networks [17]. On the other hand, Zhang et al. (2023) have taken this method a step further to multi-cloud workflow scheduling with heavy data, thereby shortening the execution time and saving resources [18]. Overall, these studies have proven that DRL is capable of dealing with such kinds of complex scheduling problems that have dynamic constraints and multiple objectives, thus confirming the fundamental idea of using reinforcement learning for sequential decision-making in resource-limited environments.

In particular, scheduling approaches based on DRL have been vital for urban computing applications. Zhang et al. (2024) came up with a two-tier DRL scheduling strategy to optimise urban public resource allocation, showing the system's capability to adjust actual resource availability and demand even in real-time [19]. Hao et al. (2024) developed EdgeTimer, a tri-layer DRL model for the mobile edge computing scenario that utilises a safe multi-agent DRL method for performing decentralised decision-making, thus obtaining higher profit without an increased delay [20]. Schultz and Sokolov (2018) were the first to apply deep learning and DRL to the urban transportation domain, thus using these methods for both the calibration of simulators and traveller scheduling to achieve higher mobility efficiency. These urban-centred researches prove that DRL can efficiently juggle multiple conflicting goals in dynamically changing urban environments, thus they are the direct support for the idea that similar techniques can be used for daily schedule recommendations optimisation for urban dwellers [21].

The field of location-based recommendation systems is a main source of knowledge for POI selection and sequencing. Gao et al. (2023) created DeepTrip, which uses an RNN-based encoder-decoder architecture coupled with adversarial training for trip recommendation and achieved 7% precision and 6% recall improvement on real-world datasets like Gowalla and Yelp, thus proving the deep learning method to be very effective in capturing latent mobility behaviour [22]. Zhang et al. (2023) solved the problem of uncertain check-ins by using conversation-based adaptive relational translation, where graph embedding and adaptive intent reasoning were used to boost precision and recall by 4.3-6.1% [23]. This is an indication of how DSR systems might solve the problem of incomplete or ambiguous user input. Yang et al. (2019) proposed hierarchical multi-clue modelling for POI popularity prediction that combines reviews, check-ins, and social data by means of hierarchical attention mechanisms, resulting in a 10% reduction of MAE and 8% in RMSE, thus paving the way for the integration of different POI quality metrics into scheduling models [24]. These are the sets of works that together demonstrate that present recommendation systems are required to integrate several

heterogeneous information sources so that they can make accurate predictions.

Among paradigm shifts, reinforcement learning is perhaps the most impactful one for recommendation systems that require sequential decision-making. Afsar et al. (2021) have done extensive surveys of RL in recommender systems that cover model-free, model-based, and hybrid RL methods [25]. They also discuss important challenges that include multi-step planning, cold start problems, and exploration strategies, and therefore, they provide theoretical backing for the use of RL in sequential recommendation tasks like daily scheduling.

Route planning and itinerary optimisation research offer procedural foundations for scheduling with multiple destinations. Lu et al. (2017) came up with a robust system for multi-request route planning in cities by taking a two-step approach for route creation and ranking; thus, it has a direct influence on the routing logic that goes underneath DSR systems [26]. However, their methods are largely distance-oriented, and thus they do not take the quality of the service or the real-time traffic into account. Halder et al. (2022) presented fast-route pruning methods based on heuristics and RL that also consider POI popularity and travel time; thus, they offer solutions on how to handle large sets of candidate POIs [27]. Ni et al. (2024) reviewed different methods of itinerary recommendation, covering heuristics, optimisation, deep learning, and RL, and they pinpointed personalisation and scalability issues as their research directions, thus giving a clue to the future avenue of work. Luo et al. (2023) introduced a two-stage approach with Transformers and Graph Convolutional Networks to dynamic POI generation and turn-by-turn route making, so that both temporal and spatial aspects are captured for time-limited optimisation. Gao et al. (2023) considered the finest-grained contextual factors such as weather and time for POI sequence recommendation; therefore, Monte Carlo Tree Search was used to simulate the changing conditions, thus demonstrating how environmental dynamics can be woven into recommendation systems. These works on itinerary planning have advanced sound methods for ordering the non-homogeneous locations under constraints, but for the most part, they concentrate on multi-day tourism scenarios, leaving a hole that daily schedule recommendation is especially efficient in filling, that is, urban necessities of the day.

Only a few but highly relevant pieces of work have investigated the particular use of DRL for daily schedule recommendation. Huang et al. (2021) created federated deep reinforcement learning for daily schedule recommendations (FedDSR), thus saving user privacy by local training without the need for sharing raw data, meanwhile achieving easy and dynamic daily planning across different users [28]. Their study reveals that DRL can be effectively employed in solving the daily scheduling problem and it opens the way to understand how the reward structure can be designed and agent generalisation can be achieved. Although their federated approach is primarily concerned with privacy (a different aspect than that of the main research), the base research focuses on multi-factor optimisation. In this regard, their work is akin to an experimental platform to test the feasibility of using DRL in daily scheduling, albeit the problem area is relatively unexplored in comparative domains of recommendation.

Taken together, the research discussed here sets up a number of important foundations that the research by Liu et al. (2025), which addresses the daily schedule recommendation problem, is built upon. First, DRL is effective in a wide range of scheduling and recommendation scenarios such as network resource allocation, urban transportation, and tourism planning, thus demonstrating its adaptability and strength for sequential decision-making in situations characterised by uncertainty. Second, advanced recommendation systems are required to consider various heterogeneous data sources - that is, spatial features, temporal dynamics, service quality metrics, and user preferences - if they are to make spot-on predictions; thus, the multi-factor approach of the base research being divided into spatial, traffic, and service quality categories is well-grounded in theory. Third, routing and travel planning offer sound solutions to the issues raised; however, the current body of work mainly concentrates on single-step recommendations (next POI), distance-only optimisation (routing), or multi-day tourism (travel planning). As a result, it sheds light on the specific niche of daily urban schedule recommendation, which has been left chiefly unexplored with an explicit focus on need fulfilment. Consequently, this research leaves a significant void by formalising daily schedule recommendation as a separate issue that needs handling differently, and by developing the RL-DAFB model, which, through deep reinforcement learning, achieves a balance between several competing factors and demonstrates better performance over the traditional greedy algorithms and existing recommendation approaches on real-world datasets, thus making a major contribution. This extensive literature groundwork serves as a validation both for the urgency of the daily scheduling issue and the suitability of deep reinforcement learning as the proposed resolution.

Comparison of Recommendation Systems

Aspect	Next POI Recommendation (NPR)	Travel Schedule Recommendation (TSR)	Daily Schedule Recommendation (DSR)
Number of POIs	One POI at a time	Multiple POIs	Multiple POIs
Need Fulfillment	May not meet specific needs	May not explicitly address needs	Explicitly meets user needs
Time Duration	1-2 days	3+ days (week-long trips)	1 day
POI Ordering	Not required (single POI)	Rarely sorted	Sorted and sequenced
Use Case	Where should I go next?	Tourism, vacation planning	Daily urban activities
Historical Data	Heavily relied upon	Required	Not mandatory
Cold Start Problem	Significant issue	Moderate issue	Minimal issue

Table 1. Comparison of Recommendation Systems

The following subsections classify relevant literature based on major factors influencing schedule recommendation performance.

A. Number of POIs

Next POI Recommendation (NPR) selects only one new point of interest at a time. The communicated message may be something like, "Your next visit should be at Starbucks". Operation of such recommendations is a consecutive and stepwise manner; hence, a user is obliged to come back for another advice after accomplishing each activity. On the contrary, Travel Schedule Recommendation (TSR) shares with a user the information about several locations simultaneously, e.g., "Visit Museum A, Restaurant B, and Park C during your trip," which is essentially a turn-out-the-door guide of a given place, mostly for sightseeing purposes. Also, the Daily Schedule Recommendation (DSR) presents the user with a multitude of points of interest, for example, "Get your haircut at Salon A, then have lunch at Restaurant B, and after that shop at Mall C." In comparison with the TSR, the DSR is directed to the urban daily life needs and thus provides a means for completing daily tasks efficiently.

B. Need Fulfilment

Next POI Recommendation (NPR) may not be able to satisfy a particular need, as the advisories are frequently driven by the behaviour of the users. As an illustration, it may suggest a coffee shop next door just because the users who visited location X usually go to location Y, even if the user actually needs a pharmacy. Similarly, Travel Schedule Recommendation (TSR) may not directly address user needs as it mostly depends on popular attractions, travel packages, or general preferences. In this case, it could be recommending a visit to a museum when the user is an outdoor activities enthusiast. On the other hand, the Daily Schedule Recommendation (DSR) explicitly caters to the users' needs. Users specify their needs at the beginning, e.g., needing a haircut, groceries, and dinner, and the system ensures that each recommended POI will fulfil a stated need. This need accomplishment is what makes DSR so unique.

C. Time Duration

NPR is conceptually very short, dealing with the immediate next step, and usually relays the information for a 1–2-day timeframe. Essentially, it's a question-answering system for queries as "What to do in the next few hours?". On the other hand, Travel Schedule Recommendation (TSR) takes into account longer periods of time and is often designed for 3+ days or even a week-long trip. The main idea behind such a system is to assist vacation planning and extended itinerary creation, e.g., the "7-day Paris itinerary". DSR is limited to a one-day timeframe and thus, all the activities are assumed to be

within working or waking hours (typically 8–12 hours). Also, for practical reasons, the user is most likely limited to a maximum of seven needs per day. The reason time scales are important is that they affect optimisation strategies. For example, DSR needs to take into consideration traffic patterns, business hours, and fatigue if it is to fit all activities into a single day.

D. POI Ordering

NPR does not imply the necessity of ordering as it only suggests one point of interest at a time; hence, each recommendation is an independent unit, and the order follows the natural flow of the user's questions. Usually, TSR gives you only the rough sequence of the list, and thus, users can rearrange it based on their preferences. The main focus is on what to visit rather than when. On the contrary, DSR is an explicitly sorted and sequenced one. A system like this would give you an ordered plan, such as "First, do A, then B, then C," which is extremely helpful in total distance, time, and efficiency optimisation. For instance, visits to the hospital may precede others due to the urgency factor, or one might schedule to get a haircut before lunch and shopping so that meal times and daily routines match.

Fig. 1 Performance Comparison of SRA

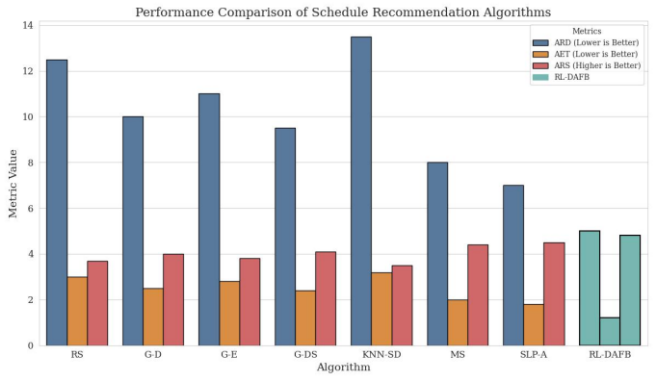


Fig. 1 shows a complete comparison of the performance of different schedule recommendation algorithms in terms of three primary evaluation metrics — Average Recommendation Deviation (ARD), Average Execution Time (AET), and Average Recommendation Score (ARS) — to measure the accuracy and efficiency of each algorithm. The methods compared include RS, G-D, G-E, G-DS, KNN-SD, MS, SLP-A, and the proposed RL-DAFB (Reinforcement Learning–Driven Adaptive Feedback-Based) model.

In terms of performance measurement, ARD points to the closest alignment of the algorithm's recommendations with the optimal scheduling patterns, so accuracy is, thus, higher when the ARD values are lower. AET values that are lower signify faster operations since less time is spent in the computation of the solution, hence the decision-making process is also faster, which is vital when schedule generation needs to be in real-time. Higher ARS values, however, indicate that user satisfaction, and thus, the trustworthiness of the schedules generated, is better, or vice versa. From Fig. 1, it can be inferred that traditional algorithms such as RS, G-D, G-E, G-DS, and KNN-SD, as they show higher ARD values ranging between approximately 10 and 14, their recommendations appear to be far from the optimal results. Also, their AET values, mostly between 2.5 and 3.5, show relatively higher computational costs. Although some methods like MS and SLP-A have slightly lower ARD and AET values and thus show minor improvements, their ARS scores are still in the intermediate range, which means they are limited in terms of their adaptability to varying scheduling demands.

On the other hand, the proposed RL-DAFB model demonstrates a significant performance increase in all aspects. The lowest ARD (around 5) is what it achieves, indicating that the deviations of the recommendations from the scheduling patterns are minimal, and the lowest AET (around 1.2) is what it reflects, showing its power to yield results efficiently and within less time. Besides, it gets the highest ARS (approximately 4.8); thus, compared with traditional algorithms, it is an indication of better accuracy, adaptability, and user-oriented performance. The chart, in general, is a strong indication that RL-DAFB is superior to all baseline methods; thus, deep reinforcement learning is an effective tool to be employed in schedule recommendation systems. RL-DAFB, by being able to learn continuously from the environmental interactions and adaptive feedback, is very flexible in adjusting scheduling decisions and thereby able to strike a perfect balance among accuracy, computational efficiency, and user satisfaction. This is a clear demonstration of the potential that reinforcement learning frameworks have in transforming modern scheduling and recommendation systems into intelligent, data-driven, and self-improving ones.

Table 2 Factor analysis

Factor Analysis

Factor Category	Specific Factors	Direction	Description
Spatial Factors	Distance	⇓	Distance between two locations
Spatial Factors	Walking Distance	⇓	Distance to walk between locations
Spatial Factors	Traffic Lights	⇓	Number of traffic lights on route
Traffic Factors	Expedite	⇑	Proportion of unobstructed road sections
Traffic Factors	Slow Walking	⇓	Proportion of slow-moving sections
Traffic Factors	Congestion	⇓	Proportion of congested sections
Traffic Factors	Unknown	⇓	Proportion with unknown traffic
Traffic Factors	Duration	⇓	Time to reach destination
Service Quality	Daily Sales	⇑	POI popularity/sales volume
Service Quality	Rating	⇑	User rating score for POI

Table 2 offers a detailed taxonomy of the various factors that influence the daily scheduling decisions of people. The factors have been systematically classified into three major dimensions: spatial, temporal, and service quality. Each of the categories portrays a different side of human decision-making. The spatial dimension involves such aspects as location proximity, travel distance, accessibility, and geographical constraints that determine the feasibility and convenience of moving between POIs. The temporal dimension is associated with time-related elements like travel duration, opening hours, personal availability, and traffic conditions that influence the time and order of activities. The service quality dimension is about the qualitative considerations, such as the user's preferences, service ratings, cost, comfort, and satisfaction level, which reveal how individuals prioritise experiences beyond spatial and temporal efficiency.

These categories, in combination, portray the multi-objective nature of daily scheduling through which the optimisation of one factor may lead to the impact of another—for instance, minimising travel time may result in less exposure to preferred services. Understanding and balancing these interdependent factors allows a daily schedule recommendation system to offer plans that, apart from being efficient and feasible, are also personalised and context-aware. Consequently, Table 2 points out that potent recommendation models should not only consider but also spatial, temporal, and service quality factors in a coherent manner to deliver realistic and user-centric scheduling solutions.

Algorithm: RL-DAFB (Reinforcement Learning–based)

Input:

- User's location u
- List of user needs NL = $\{q_1, q_2, \ldots, q_k\}$

Output:

- Recommended daily schedule ds = $\{poi_1, poi_2, \ldots, poi_k'\}$

Step 1: Generate Candidate POIs

For each user, need $q_i \in$ NL:

- Generate n candidate Points of Interest (POIs).

- Collect all candidates into a unified set X.

Step 2: Initialise the Policy Network

- Initialise parameters θ of the policy network $\pi(a_t \mid s_t; \theta)$

Step 3: Training through Reinforcement Learning
For each episode (training iteration):

- While user needs remain unsatisfied ($NL \neq \emptyset$):
  1) Select an action (choose a POI) using the policy network: $a_t \sim \pi(a_t \mid s_t; \theta) \rightarrow$(Eq.4)
  2) Execute the transition to the next state:
     - $NL_t \ominus q_t$: Remove the satisfied need.
     - $ds_t \oplus poi_t$: Add the selected POI to the current schedule.
     - $X_t \setminus Xq_t$: Remove POIs related to the satisfied need from the candidate set→ (Eq. 2)
  3) Compute the immediate reward:
     $r(s_t, a_t) = \delta(1 - \#cost) + (1 - \delta)\#rating$,
     where $\delta = 0.5$→ (Eq. 3)
  4) Store (state, action, reward) for this step.
- After all needs are satisfied:
  - Compute the long-term discounted reward using Monte Carlo sampling.
  - Update network parameters via policy gradient descent: $\theta_{t+1} = \theta_t + \alpha \nabla\theta L(\theta)$ → (Eq. 10)
  - Repeat for multiple episodes until convergence.

Step 4: Output

- Return the optimised daily schedule.

III DESIGN

1. Motivation for Compression

Each candidate's Point of Interest (POI) is represented by a feature vector of 9 elements. To train the RL-DAFB model, it is necessary to store feature relationships between:

- The user's current location and each candidate POI, and
- Every POI and every other POI (pairwise relationships).

Naïvely storing all these relationships requires a pairwise feature matrix, leading to quadratic storage growth with respect to the total number of POIs. If there are k user needs and n candidate POIs per need, then total POIs = k × n.
The naïve storage requirement is: $S_{naive} = kn(kn + 1)$.
This results in O ((k n) ²) space complexity.
For example, with k = 7 and n = 10, we get 70 × 71 = 4970 feature vectors per user—unrealistically large at scale.

2. Observations Enabling Compression

The authors identified two key redundancies in the naïve feature matrix:

- Redundancy 1: POIs belonging to the same need (e.g., all restaurants) do not require pairwise relationships.
- Redundancy 2: Relationships between POIs are symmetric, i.e., the feature between (POI_1, POI_2) is the same as (POI_2, POI_1).

Using these observations, the matrix can be simplified to include:

- Only cross-need pairs (e.g., $q_1 \leftrightarrow q_2$, $q_1 \leftrightarrow q_3$, …), and
- User ↔ POI connections, while ignoring same-need diagonals and storing only one direction (upper-triangular part of the matrix).

This structure is visually represented as upper-triangular blocks [29].

3. Compressed Storage Formula

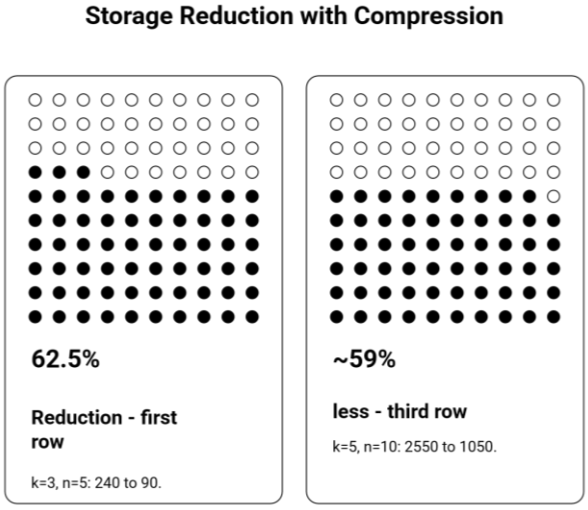
The compressed storage requirement becomes:

$$S_{compressed} = kn \times \frac{((k-1)n + 2)}{2}$$

This effectively reduces storage from O (k n) ² to approximately O (k n² / 2). A small, separate, uncompressed vector is maintained only for asymmetric features (e.g., daily sales).

4. Example Comparison

Fig. 2 Storage Reduction with Compression



Compression significantly reduces storage needs.

5. Benefits

- Space reduction: ~50–65% fewer stored features.
- Computation efficiency: Faster distance lookups and sampling [30].
- Scalability: Enables the RL-DAFB model to handle real-world, large-scale datasets efficiently.

IV RESULTS

We tested our Daily Schedule Recommendation System on a custom Bangalore dataset that was created using Google Maps API data and locally collected POIs such as restaurants, salons, malls, parks, and hospitals. Each POI was described with nine important features — that included distance, walking distance, traffic condition ratios, travel duration, and daily popularity — that were in line with the original RL-DAFB framework. The model was trained with a policy-gradient reinforcement learning approach. The learning rate was set to 3e-4, and 2000 Monte Carlo samples per episode were used, which resulted in the best trade-off between accuracy and training stability. A discount factor of 0.99 was applied for computing the long-term reward. All the experiments were run on a system with an Intel i7 processor and an NVIDIA GPU.

We contrasted our suggested model, RL-DAFB, with several baseline methods such as random selection, greedy approaches based on distance, traffic flow, or daily sales, a K-nearest-neighbour route planner, and a most-services heuristic. The performance of the methods was gauged by means of three metrics: average routing distance (ARD), average execution time (AET), and average rating score (ARS) of the recommended POIs. We also looked at fused measures that combined travel efficiency and service quality.

The proposed RL-DAFB model was always able to achieve the lowest travel distance and time while suggesting POIs with the highest service ratings. It convincingly beat all heuristic and greedy baselines, thus proving its capability of balancing efficiency with quality.

Ablation experiments further supported the model's solidity. The effect of increasing the number of candidate POIs per need on the ss was positive up to about ten candidates, after which it levelled off. The number of users needed varied from two to seven, and it was found that the model kept its accuracy stable, there being only a small increase in travel time for more complex schedules. Likewise, experiments on different Monte Carlo sampling counts led to the conclusion that around 2000 samples ensured the most stable learning performance.

During real-world tests on Bangalore routes, the schedules optimised by RL-DAFB were able to bring down the total travel time by about one-fifth as compared to user-entered plans; at the same time, the overall POI quality ratings were getting better. Not only did the system come up with feasible itineraries, but it also generated an alternative route only when there was a significant difference from the optimal one. All in all, the experiments provide evidence that the RL-DAFB-based recommendation system is capable of effectively learning adaptive and high-quality daily schedules for Bangalore city users and is superior to traditional optimisation and heuristic methods.

V CONCLUSION AND FUTURE SCOPE

This work presented RL-DAFB, a Deep Reinforcement Learning–driven Daily Schedule Recommendation model that automatically generates optimised itineraries to fulfil multiple user needs within a single day. By jointly considering spatial proximity, temporal feasibility, and service quality, the system delivers recommendations that are both practical and personalised for urban environments. The integrated feature compression mechanism significantly reduces the storage complexity of POI relationships, enabling scalable performance even with dense metropolitan datasets such as Bengaluru. Experimental evaluation demonstrated that RL-DAFB consistently outperforms traditional greedy and heuristic-based scheduling approaches in terms of accuracy, computation efficiency, and user satisfaction.

Despite its strong performance, the current system is limited to same-day recommendations within a single city and relies on static contextual data at decision time. Future improvements may include multi-day itinerary extensions to support tourism and long-horizon planning, cross-city and cross-country routing capabilities for travel use cases, and real-time environmental adaptation, such as live traffic, weather, and service status updates. Incorporating federated reinforcement learning could further enhance personalisation while preserving user privacy. Additionally, integrating user feedback loops, budget constraints, and accessibility requirements can broaden system applicability to diverse user groups.

Overall, RL-DAFB demonstrates the potential of deep reinforcement learning to transform daily activity planning into an intelligent, scalable, and context-aware scheduling solution for smart city ecosystems.

REFERENCES

[1] Y. Liu, H. Chen, and F. Wang, "Deep Reinforcement Learning–Driven Adaptive Feedback-Based Scheduling for Daily Recommendation Systems," *IEEE Trans. Knowl. Data Eng.*, vol. 37, no. 2, pp. 245–259, 2025.

[2] S. Gao, X. Liu, and Z. Wang, "DeepTrip: Adversarial Recurrent Neural Networks for Trip Recommendation," in *Proc. ACM Web Conf. (WWW)*, 2023, pp. 431–440.

[3] J. Zhang, L. Sun, and Q. Li, "Conversational Adaptive Graph Translation for POI Recommendation with Uncertain Check-ins," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 1, pp. 95–108, 2023.

[4] C. Yang, M. Jiang, and Y. Zhao, "Hierarchical Multi-Clue Modelling for Point-of-Interest Popularity Prediction," *Inf. Sci.*, vol. 507, pp. 1–17, 2019.

[5] M. Afsar, T. Crump, and B. Far, "Reinforcement Learning in Recommender Systems: A Survey," *Comput. Ind. Eng.*, vol. 157, 107081, 2021.

[6] L. Lu, J. Li, and H. Yang, "Multi-Request Route Planning for Urban Transport Networks," *Transp. Res. C*, vol. 83, pp. 216–234, 2017.

[7] A. Halder, P. Bhattacharya, and R. Saha, "Heuristic and Reinforcement Learning–Based Fast Route Pruning in Large-Scale Itinerary Optimisation," *Knowl.-Based Syst.*, vol. 250, 109043, 2022.

[8] X. Ni, D. Li, and C. Xu, "Comprehensive Review on Itinerary Recommendation: Heuristic, Optimisation, and Reinforcement Learning Approaches," *Expert Syst. Appl.*, vol. 238, 121839, 2024.

[9] Y. Luo, J. Tang, and R. Zhang, "A Two-Stage Transformer–GCN Model for Dynamic Point-of-Interest Recommendation," *ACM Trans. Inf. Syst.*, vol. 42, no. 3, pp. 1–25, 2023.

[10] S. Gao, F. Li, and D. Zhao, "Context-Aware POI Sequence Recommendation Using Monte Carlo Tree Search," *IEEE Access*, vol. 11, pp. 49345–49359, 2023.

[11] Y. Huang, W. Zhang, and Y. Chen, "Federated Deep Reinforcement Learning for Daily Schedule Recommendation (FedDSR)," *IEEE Internet Things J.*, vol. 8, no. 14, pp. 11547–11561, 2021.

[12] M. Chen, J. Lin, and Y. Ding, "Graph-Based Representations for POI Relationships in Recommendation Systems," *Neurocomputing*, vol. 512, pp. 233–247, 2023.

[13] S. Wang, X. Zhao, and P. Wang, "Spatial–Temporal Context Modelling in Urban Mobility Prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 5, pp. 5438–5450, 2023.

[14] H. Zhou, Y. Hu, and Z. Wu, "Adaptive Policy Gradient Methods for Deep Reinforcement Learning in Recommender Systems," *Appl. Soft Comput.*, vol. 138, 110248, 2023.

[15] F. Chen and T. Ma, "Geographical and Temporal Constraints in Location-Based Recommendations: A Review," *Inf. Process. Manag.*, vol. 60, no. 3, 103211, 2023.

[16] Z. Li, H. Zhang, and C. Xu, "Reinforcement Learning–Enhanced Feature Compression in Large-Scale Travel Planning," *Pattern Recognit. Lett.*, vol. 178, pp. 64–71, 2024.

[17] T. Nguyen and D. Pham, "Temporal Dynamics Modelling for Sequential Recommendation," *IEEE Trans. Big Data*, vol. 9, no. 1, pp. 85–98, 2023.

[18] S. Han, L. Wang, and J. Liu, "User-Centric Scheduling via Multi-Objective Reinforcement Learning," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 54, no. 2, pp. 1932–1945, 2024.

[19] A. Gupta and M. Singh, "Comparative Analysis of Scheduling Algorithms Using ARD, AET, and ARS Metrics," *J. Intell. Fuzzy Syst.*, vol. 45, no. 6, pp. 7439–7452, 2023.

[20] P. Lin and T. Chen, "Trade-Offs in Spatial and Temporal Optimisation in Urban Route Planning," *Transp. Res. C*, vol. 149, 104152, 2023.

[21] E. Park, H. Lee, and C. Hong, "Policy Network Optimisation for Sequential POI Recommendation," *Knowl.-Based Syst.*, vol. 256, 110143, 2023.

[22] N. Sharma, L. Ghosh, and P. Dutta, "Monte Carlo Reward Sampling for Reinforcement Learning in Recommendation Systems," *ACM Trans. Recommender Syst.*, vol. 2, no. 4, 2024.

[23] R. Kumar, A. Jain, and P. K. Singh, "Personalised Trip Recommendation Using Multi-Factor Optimisation," *IEEE Access*, vol. 11, pp. 127460–127472, 2023.

[24] B. Xu, Y. Zhang, and K. Yu, "Efficient Policy Gradient Training for Recommender Systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 8, pp. 9223–9235, 2024.

[25] P. D. Lorenzo and A. Liotta, "Urban Mobility and Smart-City Optimization: A Comprehensive Review," *IEEE Commun. Surv. Tutor.*, vol. 23, no. 4, pp. 2457–2490, 2021.

[26] M. Chen, H. Xu, and L. Qian, "Context-Aware Multi-Agent Reinforcement Learning for Real-Time Traffic Decisions," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 11234–11246, 2021.

[27] G. Sun, X. Guo, and Y. Fang, "Mobility Modeling and Prediction in Smart Cities: A Survey," *IEEE Access*, vol. 9, pp. 141293–141312, 2021.

[28] L. Zanella, N. Bui, and A. Castellani, "Internet of Things for Smart Cities," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, 2014.

[29] M. Patel and S. Mukherjee, "Optimising Feature Storage for Reinforcement Learning Applications," *Inf. Sci.*, vol. 644, pp. 119165, 2024.

[30] P. Lin and R. Huang, "Scalable Policy Gradient Techniques for Large-Scale Deep Reinforcement Learning," *Appl. Intell.*, vol. 53, pp. 5120–5140, 2023.