ORIGINAL RESEARCH

Promoting sustainable tourism by recommending sequences of attractions with deep reinforcement learning

Anna Dalla Vecchia[1] · Sara Migliorini[1] · Elisa Quintarelli[1] ·
Mauro Gambini[1] · Alberto Belussi[1]

Abstract

Developing Recommender Systems (RSs) is particularly interesting in the tourist domain, where one or more attractions have to be suggested to users based on preferences, contextual dimensions, and several other constraints. RSs usually rely on the availability of a vast amount of historical information about users' past activities. However, this is not usually the case in the tourist domain, where acquiring complete and accurate information about the user's behavior is complex, and providing personalized suggestions is frequently practically impossible. Moreover, even though most available Touristic RSs (T-RSs) are user-focused, the touristic domain also requires the development of systems that can promote a more sustainable form of tourism. The concept of sustainable tourism covers many aspects, from economic, social, and environmental issues to the attention to improving tourists' experience and the needs of host communities. In this regard, one of the most important aspects is the prevention of overcrowded situations in attractions or locations (over-tourism). For this reason, this paper proposes a different kind of T-RS, which focuses more on the tourists' impact on the destinations, trying to improve their experiences by offering better visit conditions. Moreover, instead of suggesting the next Point of Interest (PoI) to visit in a given situation, it provides a suggestion about a complete sequence of PoIs (tourist itinerary) that covers an entire day or vacation period. The proposed technique is based on the application of Deep Reinforcement Learning, where the tourist's reward depends on the specific spatial and temporal context in which the itinerary has to be performed. The solution has been evaluated with a real-world dataset regarding the visits conducted by tourists in Verona (Italy) from 2014 to 2023 and compared with three baselines.

Keywords Recommendation system · Touristic itinerary · Deep reinforcement learning · Sustainable tourism · Overtourism

---

Extended author information available on the last page of the article

␧ Springer

1 Introduction

The development of touristic RSs (T-RSs) (Xiang et al. 2022) has encountered many problems due to the difficulty of acquiring information about the actual user behavior, as well as the sequence of experiences that travelers perform (Massimo and Ricci 2022). Indeed, tourists' behavior is usually inferred by their activity on social networks and their reviews posted on travel platforms, like TripAdvisors or Google Places, but this represents only a portion of the overall behavior or experience, in addition to the fact that such information is available for a subset of all tourists (Marchiori et al. 2013; Zhang and Fesenmaier 2018). The lack of a long list of preferences has been investigated in the research literature, and algorithms that provide recommendations without the existence of long-term preference profiles have been labeled as session-based (Ludewig et al. 2021). In many real-world situations, touristic applications deal with anonymous or occasional users interacting with a specific application for the first time during each journey or visit. Therefore, in the tourist domain, the development of personalized suggestions is an even more complex task requiring a relaxed form of personalization, like the adaptation to clusters or classes of users as in Massimo and Ricci (2022). T-RSs should be able to personalize the suggestions as much as possible with limited information at their disposal.

In analyzing the data for a T-RS, the context in which previous visits have been performed or the suggestion will be provided also represents an essential aspect to be considered. Indeed, user preferences can change dynamically based on different factors, like the composition of the group that will enjoy some activities (with friends, with the family, or with some colleagues), the season or period of the year when travel is planned, or weather conditions in which a tourist visit will take place. In general, introducing the notion of context in a recommender system allows the production of more practical and tailored suggestions with a more significant benefit for users (Adomavicius et al. 2022; Chen and Chen 2015; Villegas et al. 2018). Typical contextual dimensions are represented by the temporal one, specified at different granularity levels, the space or position in which the user is located, the social conditions, and, in general, all the variables that can somehow modify the preferences of target users. As highlighted in Baltrunas et al. (2011), one of these is undoubtedly the level of crowding in a given PoI when the visit is performed. To reduce the amount of time wasted in a queue before entering an attraction, a good RS should be able to balance the number of tourists visiting each PoI in a given moment (Migliorini et al. 2021, 2018).

Currently, available T-RSs are typically user-focused; they act on behalf of users, trying to produce the best suggestion for the single tourist. However, sooner or later, this can negatively impact both the environment and the local communities (Merinov 2023; Merinov et al. 2022). More specifically, existing user-focused T-RSs can lead to overcrowded situations: an attraction is said to be overcrowded when it attracts more tourists than it can sustain. The problem has become so interesting that the term *over-tourism* has been coined to represent a situation in which both tourists and locals feel that the destination is too busy and over-visited. On the one hand, over-tourism negatively impacts the environmental

landscape and the life of the host communities. On the other hand, it could also negatively impact tourist satisfaction and safety (Merinov 2023; Patro et al. 2020; Yu and Egger 2021), compromising the overall touristic experience. Sustainable tourism tries to balance tourism's positive and negative impact in a given region from multiple points of view (Mason 2020). It is a complex concept that covers the complete tourism experience, including concerns for economic, social, and environmental issues, as well as attention to improving the tourist experience and addressing the needs of host communities (Sustainable development 2023). In this regard, mitigating overcrowding and decreasing popularity bias are two essential aspects that a T-RS shall consider (Merinov 2023). For doing that, a reliable crowding forecaster system is essential. Forecasting the number of tourists in each attraction can be performed by applying several techniques and considering various contextual information (Belussi et al. 2022; Merinov et al. 2022).

Tourism authorities try to support and manage tourism in various ways, one of these is the offer of tourist cards or city passes, which cover the entrance to a predefined set of attractions or Points of Interest (PoIs), have a validity period, and a cost. To increase tourists' satisfaction and improve the experience in using such cards, tourism authorities can provide some suggestions about itineraries considering the covered PoIs. Such recommendations could also be used to drive tourist choices towards more sustainable forms of tourism, mainly to prevent overcrowding. This consideration leads to another specialization of T-RSs, from next-item recommendations to sequence recommendations. Indeed, the ability to organize in advance the best travel plan covering an entire holiday of one or more days can both increase user satisfaction and optimize a set of visits concerning budget and sustainability concerns (Wörndl et al. 2017). Moreover, presenting a sequence of suggestions to users where the visit to a popular PoI is only postponed, not eliminated, can better convince them to follow the suggestion. Clearly, the shift from a next-item suggestion to a sequence-based direction greatly complicates the problem (Migliorini et al. 2019, 2022).

In this paper, we propose a T-RS that suggests to tourists the best itinerary to follow, given a set of predefined PoIs and the context in which the visits will be performed. Relative to the context, we will extract it from the tourist's location and the visits' timestamp and enrich them with temporal semantic information, weather conditions, and crowding estimations. We also assume dealing with anonymous and occasional users for whom there is no previously available data. In this case, the similarity or clustering between users is performed only by considering that they plan to visit a specific region through the same city pass in a similar context. However, the proposed methodology is general enough to be extended to consider specific user preferences when available. This T-RS is not user-focused in the classical form but takes the point of view of tourism authorities, which need to dynamically suggest itineraries to users starting from a set of identified PoIs and trying to balance the maximization of user experience with the prevention of overcrowded situations in the current context. We can observe that once a tourist decides to buy a city pass, he/she implicitly expresses an interest or preference for the set of covered PoIs, or at least for the majority of them.
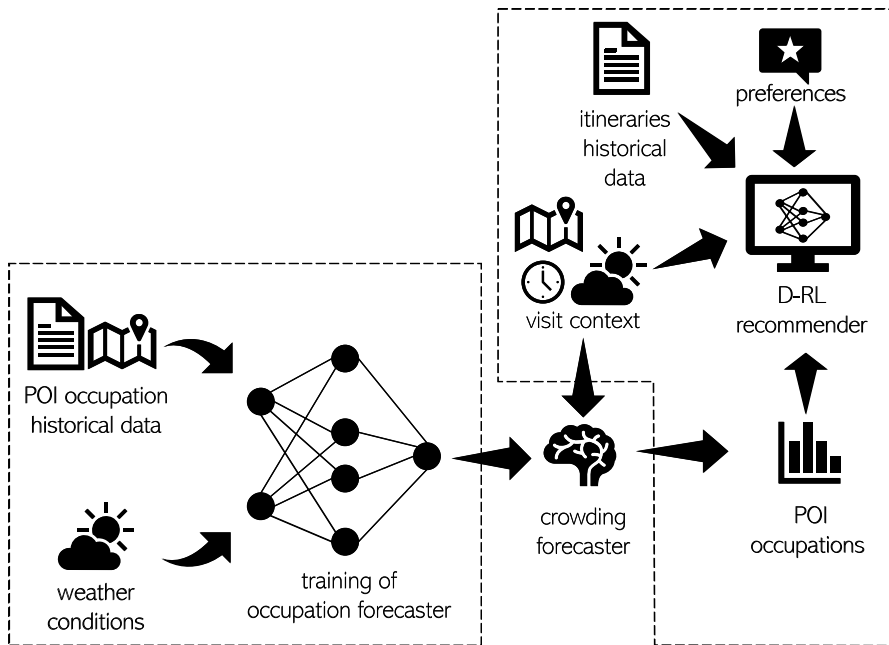
Fig. 1 Architecture of the proposed tourist itinerary Recommender System

The proposed methodology is based on an application of Deep Reinforcement Learning (D-RL) and takes care of the context in which the visits will be performed. The overall architecture of the proposed system is illustrated in Fig. 1: the two main components are the crowding forecaster and the D-RL recommender. Regarding the crowding forecaster, we consider (Belussi et al. 2022) as the state-of-the-art solution, as it uses historical data about past visits and other external contextual information, like weather conditions, to train a deep neural network representing the crowding forecaster itself. This component is used to produce one of the inputs of the D-RL recommender, which is the main contribution of this paper. As we will explain in more detail in the following sections, the D-RL recommender needs different kinds of information as input, like the context in which the itinerary is planned, the PoIs occupancy in such context, and so on, and produces the best recommendation for the final user. In particular, historical data about past user itineraries are also exploited to train the neural network, part of the RL techniques, through the experience buffer component, as we will describe in more detail in Sect. 4. Our solution has been experimented on a real-world dataset regarding the visits performed by tourists in Verona, a city in Northern Italy, from 2014 to 2023 and compared with three baselines: the user behavior without suggestions, which is typically exploratory rather than optimal (Massimo and Ricci 2022), and an approach considering popularity and proximity with and without the notion of context. The major contributions of this paper can be summarized as:

- We propose a context-aware T-RS for promoting more sustainable forms of tourism by avoiding the production of overcrowded situations. To do that, complete sequences of attractions (i.e., itineraries) covering a desired period of time or vacation are suggested by taking into consideration contextual estimations of the level of crowding in each attraction.
- We implement the described T-RS by using a D-RL technique, and we provide a reward function definition that formalizes the required objective of mitigating overcrowding and decreasing popularity bias. Information coming from historical data is also used to capture the typical tourists' behavior in different contexts and modulate the reward also with respect to this.
- We compare the proposed approach with some typical baselines and metrics, providing also a scalability experimentation of the technique.

The introduction of the notion of context, the provision of complete itinerary suggestions and the formulation of the reward function allow altogether to build a more effective T-RS for preventing over-tourism. As a further advantage, the use of a D-RL technique also provides a better explanation to tourists about the provided suggestions with respect to other machine or deep learning techniques, which are instead black box components. Explainability can have a great role in increasing the adoption of the proposed suggestions by tourists.

The remainder of the paper is organized as follows: Sect. 2 presents some existing work about the definition of RS in the touristic domain or regarding context-aware RS and RS for sequences of items. Section 3 formalizes the problem, while Sect. 4 describes its solution concerning the D-RL terminology. Section 5 illustrates the performed and obtained results. Finally, Sect. 6 concludes the work and discusses possible extensions.

2 Related work

This section summarizes the literature in the field of recommender systems, with particular attention to those developed in the tourist domain and classified based on their main features in Gavalas et al. (2014).

Tourist RS The development of RS for the tourist domain has received much attention in recent years; therefore, many surveys are available on the topic. In particular, Gavalas et al. (2014) provides a systematic categorization of the state-of-the-art in the field of mobile tourism RSs with an emphasis on typical recommendation tasks and support functions offered by existing mobile tourism RS applications.

In Islam et al. (2020), the authors summarize the proposed solutions using deep learning techniques to generate a suggestion about the next PoI to visit. In detail, the authors evaluate the performances of different solutions and study the factors that mainly influence the recommendations of a given PoI. The compared neural networks are the traditional feed-forward neural network, the Convolutional Neural Network (CNN), the Recurrent Neural Network (RNN), the Long-Short TermMemory (LSTM), the Gated Recurrent Unit (GRU), the Attention Mechanisms, and the Generative Adversarial Network. The datasets used are taken from the check-ins

collected through many different LBSNs (Location-based Social networks), like Instagram and Twitter. Similarly, in Borràs et al. (2014), the author compares many previous works about using Artificial Intelligence techniques in the tourist domain, considering not only the techniques and the algorithms used but also the user interfaces and the interaction with the final user.

The use of Reinforcement Learning (RL) techniques for producing recommendations in the touristic domain is proposed in Jiang et al. (2021), where the RL is used to determine the correlation between PoIs and satisfy some predefined constraints. Conversely, in Wang et al. (2020), the use of RL is proposed to study the behavior of tourists with the aim of replicating their choices and generating predictions about their future actions.

In Massimo and Ricci (2023), the authors consider the problem of suggesting relevant recommendations for new PoIs to new users within tourist recommender to augment the novelty and diversity of the provided recommendations. They propose an Inverse Reinforcement Learning (IRL) RS that leverages observed tourists' POI visit behavior and their inclination to explore and discover new POIs in their spatial proximity. The main aim is to discover the behavior of a cluster of tourists and identify the reasons behind their choices. Conversely, our setting is different because we want to provide suggestions that promote a sustainable form of tourism, namely the one that prevents overcrowded situations, but somehow also consider the usual tourist behavior for producing suggestions that still appeal to them.

The approach proposed in this paper differs from the ones already present in literature about the use of deep learning and RL techniques for two main aspects: the attention and importance given to the notion of spatio-temporal context in the determination of both the state and the reward, the aim to provide recommendations for sequences of activities (or itineraries), instead of next-item suggestions, and most importantly it does not only focus on the user preferences.

Contextual RS Context-aware recommender systems (CARSs) have gained popularity thanks to their ability to generate more relevant recommendations tailored for a specific context (Adomavicius et al. 2022). In Baltrunas et al. (2011), the authors propose a methodology for quantitatively assessing contextual factors, including crowdedness, worth considering.

In Yuan et al. (2013), the authors propose a methodology for suggesting the next PoI to visit, which considers both the specific moment inside the day when the visit will be performed and the location of such attraction. The used datasets are extracted from the main available LBSNs, and the obtained results provide an improvement of 37% in accuracy with respect to the base techniques, which do not consider the temporal context. However, the mentioned contextual dimensions are very limited with respect to the ones considered in this paper, where the temporal characterization is enriched with other semantic information, like the presence of holidays, and accompanied with additional knowledge, like weather conditions, the degree of crowding and the user spatial location.

In Zhou (2020), the authors deal with the problem of generating recommendations in situations where both users and items change dynamically and in a continuous way. Hence, recommendations must be adapted to the new conditions as new information arrives. The proposed solutions use a deep neural network, and some

insights are also used in this paper to deal with unknown users for whom we do not have much information about preferences.

The production of contextual recommendations for tourists is also treated in Migliorini et al. (2019), where the recommendation is formalized as a multi-objective optimization problem and solved with a MapReduce application of the Multi-Objective Simulated Annealing (MOSA) technique. This solution is further extended in Migliorini et al. (2022) for dealing with the dynamic evolution of groups of users and the balancing of preferences inside groups.

Sequence-based RS The itinerary recommendation problem has been investigated for suggesting POIs that tourists may be interested in visiting next, i.e., after they have already visited some other POIs. In He et al. (2016), the problem of the next POI recommendation under the influence of the user's latent behavior pattern has been introduced without considering the contextual dimensions. In Baral et al. (2018), the authors introduce the Contextualized Location Sequence Recommender (CLoSe) concept. They compare the use of a generic Recurrent Neural Network (CLoSe-RNN) with the use of a Long-Short Term Memory (CLoSe-LSTM) for the solution of the problem, showing that, as expected, the latter performs better than the former. They also analyze the main aspects that can influence the recommendation for sequences of PoI. They identify the following aspects: the sequential effect (how the previous PoI influences the choice of the next one), the spatial correlation between PoIs, the semantic correlation (the preferences of users towards specific categories of PoIs), the social influence, and the temporal dependency. Similarly, in Arentze et al. (2018), the authors underline the multi-criteria nature of tour preferences and present a method to estimate tourist preferences as a utility function considering various factors involved in trip planning. Such a model allows a T-RS to compose an optimal tour given personal information about the specific interests of an individual user. Many of these factors are considered also in our formalization of the notion of context.

A survey about RS for tourist itineraries is contained in Lim et al. (2019), where a taxonomy about the various formulations of the problem and some algorithms already presented in the literature are analyzed. One of the main dimensions of classification is the presence of temporal and spatial constraints, confirming that the notion of context is essential in such kinds of systems. Moreover, the problem related to the availability of data and the evaluation methodologies are also discussed.

The use of RL techniques for the construction of sequences of recommendations is also proposed in Gama and Fernandes (2020) and Chen et al. (2020); the obtained results confirm the goodness of this approach in the definition of recommendations for sequences of activities, but they do not consider the context in which the visits will be performed.

In Kotiloglu et al. (2017), the authors propose a solution based on the tabu-search methodology for generating personalized tour recommendations for tourists based on information from social media and other online data sources. The goal is to generate a tour containing a set of mandatory points and maximize the total score collected from optional points concerning several different constraints. Similarly, in Huang et al. (2021), the authors tackle the route planning problem with a flexible deep learning framework that integrates PoI attributes, user preferences, and

historical route data. The main difference between these techniques and our solution is the problem formulation and the assumption of having several different personal user information at our disposal. Conversely, in our hypothesis, users are occasional and anonymous, and our information about them is very limited. Therefore, the similarity notion needed to apply collaborative filtering approaches is limited to the similarity of the spatio-temporal context in which the visit will occur.

Crowd-aware RS The importance of crowdedness for providing relevant contextual recommendations has been highlighted in Baltrunas et al. (2011). In Migliorini et al. (2021), the authors study the instantaneous distribution of tourists to produce better recommendations about the next PoI to visit. The recommendation is formulated as an optimization problem and is concentrated on a single activity at a time. A similar problem about the contemporaneous visit of the same PoI by different users has been considered in Kong et al. (2022), where the author proposed an algorithm of multi-agent reinforcement learning with dynamic reward, which can distribute tourists in the various PoIs equally. Finally, in Belussi et al. (2022), the authors treat the problem of forecasting the level of crowding in a given or future spatio-temporal context. The proposed solution is based on a deep neural network, and it will be considered a building block in the overall architecture of our solution presented in Fig. 1.

In Kılıçarslan and Caber (2018), the authors recognize that tourist satisfaction is associated with the level of crowding in cultural heritage, while Luque-Gil et al. (2018) confirms the existence of an inverse relationship between tourist crowding and tourist satisfaction. However, despite the identification of this problem, none of the works provides a solution for effectively building a T-RS that produces more sustainable suggestions.

3 Problem formalization

This section formalizes the problem of producing a recommendation for a sequence of contextual tourist visits.

Definition 1 (*Tourist visit*) Given a set of PoI $\mathcal{P}$ and a set of users $\mathcal{U}$, a tourist visit performed by $u \in \mathcal{U}$ is a tuple

$$v = \langle u, p, t, g \rangle \tag{1}$$

where $u \in \mathcal{U}$ is a user identifier, $p \in \mathcal{P}$ identifies the PoI, t is a timestamp representing the start date and time of the visit, and g is the spatial position (e.g., latitude and longitude) where the PoI is located.

The set of all visits performed by users in $\mathcal{U}$ will be denoted as $\mathcal{V}$, while the visits performed by a specific user $u \in \mathcal{U}$ will be denoted as $\mathcal{V}_u$. Given the notion of visit, an itinerary can be defined as a sequence of visits performed by the same tourist $u \in \mathcal{U}$.

Definition 2 (*Tourist itinerary*) Given a set of PoI $\mathcal{P}$ and a set of users $\mathcal{U}$, a tourist itinerary performed by a user $u \in \mathcal{U}$ is an ordered set of visits

$$i = \{v_1, \ldots v_n\} \tag{2}$$

such that:

- $\forall h \in \{1 \ldots n\}, v_h \in \mathcal{V}_u$, and
- $\forall h \in \{1 \ldots n-1\}, v_h.t < v_{h+1}.t$, and
- $\forall h \in \{1 \ldots n-1\}, v_{h+1}.t - v_h.t < \tau$

where $v.t$ denotes the timestamp when the visit v begins, and τ is a predefined threshold.

The threshold τ can be properly set based on the problem at hand which identifies the right notion of itinerary. For instance, in some cases, we can consider a sequence of visits belonging to the same itinerary if performed on the same day, while in other cases, we can take the visits performed on an entire holiday. In the situation treated in this paper, where a specific duration characterizes the city pass, we can consider only the visits that span inside this validity period.

In the following, the set of all tourist itineraries performed by tourists will be denoted as $\mathcal{I}$. In contrast, the set of all itineraries a specific tourist $u \in \mathcal{U}$ performs will be denoted as $\mathcal{I}_u$.

Historical data about past tourist visits can be enriched with some contextual information better characterizing the conditions of the visit.

Definition 3 (*Contextual information*) Given a visit $v \in \mathcal{V}$, we define its context λ as a tuple of values for some relevant dimensions $\delta = \langle d_1, \ldots, d_n \rangle$ as follows:

$$\lambda = \langle c_1, \ldots, c_n \rangle \tag{3}$$

where each c_i is the value of a contextual dimension d_i characterizing the problem at hand. The set of all possible tuples λ representing a context with dimensions δ will be denoted as $\mathcal{C}$.

In our specific scenario, we consider as meaningful contextual dimensions the tuple:

$$\delta = \langle ts, doy, dow, hol, temp, prec, precType, orig, crow \rangle$$

where:

- *ts* is a predefined timeslot inside the day. Based on the problem at hand and the kind of available information, we can decide to extract from the timestamp t a timeslot at the desired granularity levels: e.g., any 10 min, any 1 h, or any 4 h, and so on. Clearly, a smaller value of *ts*, namely a greater granularity, could improve the precision of the obtained suggestions while increasing the need for more historical data. A good compromise between the level of detail

and the amount of required historical data can be inferred by a preliminary analysis and the extraction of context variations. In our specific scenario, the identification of three timeslots inside the day has emerged as the best value.

- *doy* is the day of the year;
- *dow* is the day of the week;
- *hol* is a boolean value representing the fact that the visit is performed on a public holiday and during a weekend or not;
- *temp*, *prec* and *precType* denotes the climatic conditions in terms of temperature, amount of precipitation, and kind of precipitation (like rain, snow, etc.);
- *orig* is the geographical position from which the user is coming;
- *crow* is the level of crowding of the tourist attraction.

The notion of context will be used to characterize the visits performed by users to specific PoIs. In particular, concerning the considered specific contextual dimensions, the first four elements allow us to better characterize the moment of the visit from a temporal point of view and abstract from the specific visit timestamp. In this case, two visits performed in the same period (day of the year) but in different years could be considered as occurring in the same context.

A context-aware tourist visit combines the information regarding a tourist visit with the context in which that visit has been performed.

Definition 4 (*Context-aware tourist visit*) Let $v = \langle u, p, t, g \rangle$ be a visit performed by a user u in a specific context $\lambda = \langle c_1, \dots, c_n \rangle$, where $\forall i \in \{1, \dots, n\}$ c_i is the actual value for the contextual dimension d_i, a context-aware tourist visit is defined as:

$$cv = \langle u, p, t, g, c_1, \dots, c_n \rangle \tag{4}$$

where cv can be seen as the tuple v, representing the tourist visit, enriched with the contextual values in λ.

The set of all context-aware tourist visits will be denoted as $\hat{\mathcal{V}}$, while $\hat{\mathcal{V}}_u$ will represent the set of context-aware tourist visits performed by $u \in \mathcal{U}$. In the same way, the concept of a tourist itinerary can be enriched with the notion of context, becoming an ordered sequence of context-aware tourist visits.

Definition 5 (*Context-aware itinerary*) Given a set of PoI $\mathcal{P}$ and a set of users $\mathcal{U}$, a context-aware tourist itinerary performed by a user $u \in \mathcal{U}$ is an ordered set of context-aware tourist visits:

$$\zeta = \{cv_1, \dots cv_n\} \tag{5}$$

such that $\forall h \in \{1 \dots n\}, cv_h \in \hat{\mathcal{V}}_u$ is a contextual tourist visit performed by $u \in \mathcal{U}$ and the same constraints stated in Definition 2 about the temporal components of the visits also hold for the contextual visits.

Given a context-aware tourist itinerary $\zeta = \{cv_1 \ldots cv_n\}$, $cv_1.orig$ denotes the user's position when he/she starts its journey, while $\forall h \in \{2, \ldots, n\}$, $cv_h.orig = cv_{h-1}.g$. In the following, all context-aware tourist itineraries will be denoted as $\hat{\mathcal{I}}$.

Given the notion of a context-aware tourist itinerary, it is possible to define a context-aware tourist recommendation as a suggestion that identifies the best contextual itinerary for the user that satisfies a given set of constraints. Let us notice that each visit composing the itinerary is characterized by its specific context. This means that in constructing the itinerary recommendation, we would consider the specific context in which each visit will be performed to provide the best suggestion in any situation. For instance, if weather conditions will likely change from rainy to sunny during an itinerary, we can suggest indoor attractions at the beginning, postponing the outdoor ones to the end.

Definition 6 (*Context-aware tourist recommendation*) Given a set of PoI $\mathcal{P}$ and a user $u \in \mathcal{U}$ which wants to perform an itinerary composed of attractions in $\mathcal{P}$ starting in an initial context $\lambda_0 \in \mathcal{C}$ and with a total amount of available time Δ_t. A context-aware tourist recommendation is a function:

$$\mathcal{R}_{\Delta_t} : \mathcal{U} \times \mathcal{P} \times \mathcal{C} \to \hat{\mathcal{I}} \tag{6}$$

which returns the best context-aware tourist itinerary, which complies with the available time Δ_t.

In the definition, the term *best* implies an ordering between the itineraries based on their goodness concerning specific criteria. The following section will clarify how the goodness concept is defined when the problem is formulated with D-RL terminology, and the concept of reward will be introduced.

In the following, the term *query* will be used to denote the set of parameters that characterizes a request for a context-aware recommendation given a set of PoI $\mathcal{P}$. More specifically, it is a tuple containing the current position of the user $p \in \mathcal{P}$, the initial context $\lambda_0 \in \mathcal{C}$ about the first visit, and the amount of available time Δ_t:

$$q = \langle p, \lambda_0, \Delta_t \rangle \tag{7}$$

4 Proposed solution

This section describes the proposed solution for generating recommendations about tourist itineraries in a given context to prevent overcrowded situations and promote a more sustainable form of tourism. As mentioned in the introduction, the solution is based on an application of Deep Reinforcement Learning (D-RL), more specifically of Deep Q-Learning (DQN), where the reward function reflects the desired characteristics of sustainable recommendations. The computation of this reward strongly correlates to the context in which the suggestion will be

Fig. 2 Basic interaction between an agent and the environment in RL

environment

state

reward

action

agent
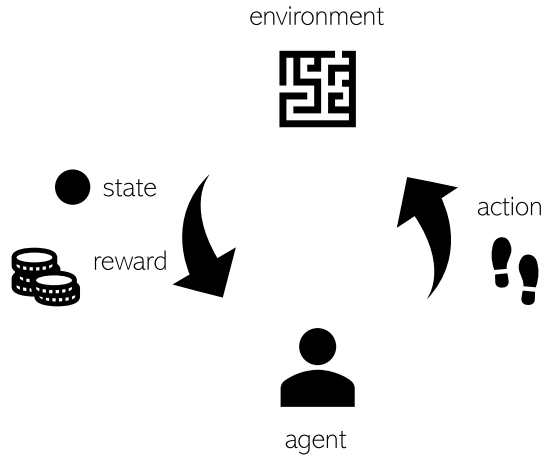
provided since each PoI's crowding level is contextually dependent. Section 4.1 introduces some background notions about D-RL, while Sect. 4.2 contextualizes the basic building blocks of D-RL with respect to the considered problem.

4.1 Deep reinforcement learning

Reinforcement Learning (RL) is a Machine Learning (ML) technique in which an *agent* makes some observations and, based on these, takes some decisions, namely performs *actions* in the surrounding *environment*, producing in turn a *state* change and obtaining a *reward*. This kind of framework is known as Markov Decision Process (MDP) (Sutton and Barto 2018). Through RL, an agent can understand the best behavior to adopt to maximize the cumulative reward achieved through a sequence of actions. In this case, the learning is due to the continuous interaction between the agent and the environment, as exemplified in Fig. 2. More specifically, in a given instant of time t, the agent and the environment are in a state s; in this state, the agent can decide to act a, moving to the state s' at time $t + 1$ and obtaining a reward r.

The agent aims to learn the best strategy to maximize the cumulative reward obtained through the actions taken in the various states. The strategy to determine the action to take in a given state is called *policy* and can be of several types. For instance, a policy is called *deterministic* if the agent will always take the same action a given the current state s, namely, the agent chooses an action in a deterministic way, among many possible actions. Conversely, it is said to be *stochastic* if the action to be taken in a given state s depends on a probability distribution function defined on the set of actions $\mathcal{A}$, given the state s.

The reward associated with each action a performed in a given state s denotes how a is good in the short term. However, in an RL problem, the agent is

interested in identifying the best sequence of actions to maximize the cumulative reward; namely, the agent wants to identify the right choice in the long term.

Definition 7 (*Cumulative reward*) The cumulative reward at a time step t, also known as *return value* and denoted as G_t, is defined as the weighted sum of the instantaneous rewards the agent achieves during its interaction with the environment starting from t up to the final time step.

$$G_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^n r_{t+n} = \sum_{k=0}^{n} \gamma^k r_{t+k} \qquad (8)$$

where $\gamma \in [0, 1)$ is called *discount factor* and enables to give a different weight to the current actions concerning the future ones.

The discount factor γ allows controlling how future rewards are discounted in comparison to the current one. In particular, $\gamma = 0$ means that only the current instantaneous reward is considered, while $\gamma = 1$ means that future rewards have the same importance as the current one. In real-world applications, γ is typically set between 0.9 and 0.99.

It follows that the choice of the best action a to be executed in a given state s does not only depend on the immediate reward associated with the transition (s, a, s'), but it also depends on the goodness of state s' which is reached from s by performing action a. The notion of the goodness of a state s' is given by the *value function*, which returns the maximum cumulative reward that the agent could expect to receive starting from the state s'.

Definition 8 (*Function value*) Given a state $s \in \mathcal{S}$, the value of s under a policy π, denoted as $V_\pi(s)$, is defined as the expected cumulative reward obtained by starting with state s and successively following policy π:

$$V_\pi(s) = E_\pi(G_t|s_t = s) = E_\pi\left(\sum_{k=0}^{n} \gamma^k r_{t+k}|s_t = s\right) \qquad (9)$$

where E_π is the expected value under policy π and $G_t|s_t = s$ denotes the cumulative reward G_t provided that the starting state s_t is s.

In the same way, it is possible to define the action-value function, known as Q-function, which considers not only the initial state s but also the action a to be performed.

Definition 9 (*Q-function*) Given a state $s \in \mathcal{S}$ and an action $a \in \mathcal{A}$, the action-value function $Q_\pi(s, a)$ is defined as the expected cumulative reward when the action a is performed in the state s and from here the policy π is followed:

$$Q_\pi(s,a) = E_\pi(G_t | s_t = s, a_t = a)$$
$$= E_\pi\left(\sum_{k=0}^{n} \gamma^k r_{t+k} | s_t = s, a_t = a\right) \tag{10}$$

We can now define that a policy π is better than a policy π', if the expected return (or cumulative reward) associated with π is greater than the one obtained with π' for all states $s \in \mathcal{S}$, namely if it holds the condition: $\forall s \in \mathcal{S} \ (V_\pi(s) \geq V_{\pi'}(s))$.

Given Eq. 10, it is possible to determine the best action-value function or Q-function recursively through the Bellman equation (Sutton and Barto 2018). The Bellman equation decomposes the Q-function into two parts: the immediate reward and discounted future values. In this way, the optimal solution can be found by solving simpler, recursive subproblems:

$$Q_*(s,a) = \sum_{s'} p(s'|s,a)[r(s,a,s') + \max_{a'} Q_*(s',a')] \tag{11}$$

where $p(s'|s,a)$ denotes the probability to reach the state s', starting from s and taking the action a, $r(s,a,s')$ is the reward associated to the transition from s to s' by performing a, and $\max_{a'}$ denotes the action a' reachable from s' which maximizes the value $Q_*(s',a')$. As you can notice, the use of $Q_*(s',a')$ in the definition of $Q_*(s,a)$ makes this definition recursive.

Given this set of shared definitions, different techniques have been defined to compute the optimal policy π^*, namely the one that, when followed, generates the highest expected reward or return G. Among these, one of the most important is represented by Q-Learning (Watkins and Dayan 1992), an implementation of the Temporal Difference (TD) technique that makes the proof of convergence of the recursive computation easier. TD and hence Q-Learning belong to the class of model-free reinforcement learning methods. Indeed, Q-Learning methods learn the optimal Q-value through an incremental exploration of the environment, like Monte Carlo (MC) methods, and perform updates based on current estimates, like Dynamic Programming (DP) methods (Sutton and Barto 2018). The exploration is done through an ϵ-greedy technique, which at the beginning encourages the choice of random actions instead of the better ones (i.e., the ones with a greater instantaneous reward), with the aim to enlarge the explored space (*exploration phase*); then, when the Q-values have been improved, the technique will prefer the best actions based on the experience made (*exploitation phase*). This model-free approach is particularly beneficial for scenarios where the underlying dynamics of an environment are difficult to model or completely unknown. This is very suitable for applications like T-RSs, where the transition probabilities are unknown or very difficult to estimate because they depend on several different intertwined aspects, like different contextual dimensions.

One of the problems of Q-Learning is the fact that it is a tabular RL technique. Namely, it requires storing the value $Q(s, a)$ inside a table for each pair of state s

and action *a*. This can induce scalability problems when the number of states and actions increases. For this reason, it has been proposed in the literature to substitute the storage of all these values with a regression function that can estimate the Q-value for each possible pair of states and actions. However, the relation between state and action tends to be nonlinear and often discontinuous. Therefore, an additional improvement is using a neural network instead of a linear regression function, leading to the so-called Deep Q-Learning (D-QL) (Mnih et al. 2013). The network approximating the Q-values is called Deep Q-Network (DQN). It can have different architectures, from simple, fully connected neural networks to convolutional neural networks or even recurrent neural networks. However, they are all characterized by a particular loss function that considers the square of the difference between the maximum expected Q-value and the Q-value estimated by the network:

$$loss = \left[(\underbrace{r_{t+1}}_{\text{immediate reward}} + \underbrace{\gamma \cdot max_a Q(s_{t+1}, a)}_{\text{discounted estimate optimal Q-value of the next state}}) - \underbrace{Q(s_t, a_t)}_{\text{former Q-value estimate}} \right]^2 \quad (12)$$

The training of the DNQ model estimating the Q-values is illustrated in Alg. 1. It exploits the use of an *experience replay*. Indeed, at each learning iteration the model has the possibility to exploit a set of samples taken from a historical dataset of past experiences in the form (s_t, a_t, r_t, s_{t+1}), where s_t is the current state, a_t is the action, r_t is the reward and s_{t+1} is the state reached after the action.

Algorithm 1 DQN

1: **procedure** TRAIN($target, prediction, replay, \gamma$)
2:     sample random minibatch $\mathcal{B}$ of transition (s_j, a_j, r_j, s_{j+1}) from $replay$
3:     **for** $(s_j, a_j, r_j, s_{j+1}) \in \mathcal{B}$ **do**
4:         $y = target.\text{PREDICT}(s_j)$ ▷ array of Q-values for any action a from s_j
5:         **if** s_{j+1} is terminal **then**   ▷ update the Q-values
6:             $y[a_j] = r_j$
7:         **else**
8:             $q\_future = max(target.\text{PREDICT}(s_{j+1}))$
9:             $y[a_j] = r_j + \gamma \cdot q\_future$ ▷ $r + \gamma \cdot max_a Q(s_{j+1}, a)$
10:        **end if**
11:        $prediction.\text{FIT}(state, y)$ ▷ gradient descent step on $(y_j - Q(s_j, a_j))^2$
12:    **end for**
13: **end procedure**

The experience replay is very important because each experience could be used in many weight updates, increasing data efficiency. Moreover, the use of

batches of previous experiences that are randomly chosen allows the break of correlations between data, reducing the variance of the updates (Mnih et al. 2013). Finally, the loss metric used during training is always the mean squared error due to what has been described above. At the same time, as an optimizer, the standard Adam optimizer is usually applied.

Algorithm 2 Deep Q-Learning

1: **procedure** $\text{DQL}(\mathcal{S}, \mathcal{A}, s_1 \in \mathcal{S}, \gamma, \varepsilon, \varepsilon\text{-decay}, m)$
2:     initialize $replay$ memory to a capacity n
3:     initialize $target$ model
4:     initialize $prediction$ model
5:     $\zeta = \{\}$
6:     **for** episode $e \in 1 \ldots m$ **do**
7:         initialize the sequence $s = \{s_1\}$
8:         **while** the final state has not been reached **do**
9:             $r = $ random value
10:             **if** $r < \varepsilon$ **then**
11:                 $a_t = $ random action from $\mathcal{A}$
12:             **else**
13:                 $a_t \in \mathcal{A}$ such that $max_a Q^*(s_t, a_t)$
14:             **end if**
15:             store transition (s_t, a_t, s_{t+1}, r_t) in $replay$
16:             $\text{TRAIN}(target, prediction, replay, \gamma)$
17:             every c steps updates $target$ with $predict$
18:             update ε using ε-decay
19:         **end while**
20:         update the best ζ
21:     **end for**
22: **end procedure**

Given the training of the DQN model, an iteration of D-QL is performed as illustrated in Alg. 2. D-QL is an offline policy method since it uses two distinct models with the same architecture: the *target* and the *prediction* networks. The estimations made by the target network are used as the ground truth for the prediction network (see line 4 and 8 in Alg. 1). The weights of the target network are updated only after c iterations (see line 17 in Alg. 2). Conversely, the weights of the prediction network are updated at each iteration (see line 11 in Alg. 1) to improve the stability of the Q-learning algorithm. Finally, as regards the policy used to choose the next action, the D-QL method applies an ε-greedy strategy (see lines 9–14 in Alg. 1), where at the beginning there is a greater probability of choosing a random action, while as the ε value decays, the next action to choose becomes the ones with the greatest estimated Q-value.

In the following section, we present the proposed solution by contextualizing the general D-QL technique to our specific problem.

4.2 Tourist RS as D-RL problem

Given the definitions provided in Sect. 3 about the problem of producing recommendations for tourist itineraries and the above description of the ingredients of a D-RL problem, it is necessary to customize the latter with respect to the considered scenario. In this case, the agent is represented by the tourist who moves from one PoI to another in a given context. Therefore, the notion of *action* is represented by the movement of the agent, which passes from one context-aware tourist visit to the subsequent one inside the sequence.

In RL a state represents the environment in which the agent is currently in. It shall include all the relevant information about the environment that the agent needs to know to make a decision. Therefore, in our case, it shall contain all the information necessary to define a contextualized visit. More specifically, the notion of state can be defined as follows:

Definition 10 (*State*) The state of a D-RL model for producing a context-aware tourist recommendation includes the following minimum set of information:

- *poi*: the ID of the PoI where the agent (the tourist) is located,
- *timestamp*: the timestamp, with the desired granularity, registering when the agent arrives at the PoI,
- *history*: a list representing the attractions already visited by the tourist.

The introduction of the *history* component inside the notion of state is necessary to prevent the inclusion of duplicated PoIs in a recommendation. Indeed, while in a traditional D-RL problem, the fact that an agent traverses the same state twice inside the same experience does not represent a problem, in this case, the provided recommendation must include any PoI only once. Therefore, this component limits the action space associated with a given state *s*. This list is essentially a list of PoI identifiers, and its representation can be properly optimized in the source code. Given the considered problem, its dimension cannot increase exponentially in real-world situations because the number of PoIs in an itinerary with reasonable duration is limited. With reference to the code in Alg. 2, this information is used in lines 9 and 11, where the set $\mathcal{A}$ will be replaced by a set $\mathcal{A}' \subseteq \mathcal{A}$ such that $\mathcal{A}'$ does not contain any action which lead to a previously visited PoI. This refinement of $\mathcal{A}$ could be even more sophisticated. For instance, in our case, we also remove the actions that lead to closed PoI at the expected arrival time.

The other two state components (*poi* and *timestamp*) are all the remaining pieces of information needed to reconstruct the notion of context-aware tourist visit in Definition 4. Indeed, we assume the presence of the following functions, which, through a set of external sources of information (as in Fig. 1), allow one to reconstruct the complete context of the visit:

- $loc(poi) \rightarrow g$: which returns the spatial position or location g of the attraction *poi*.

- $\sigma(timestamp) \rightarrow \{ts, doy, dow, hol\}$ which returns the semantic contextual characterization of the timestamp in terms of a timeslot inside the day ts, the day of the year doy, the day of the week dow, and the presence of holidays hol.
- $\omega(timestamp) \rightarrow \{temp, prec, precType\}$ which returns the weather conditions at timestamp $timestamp$ as the temperature $temp$, the amount of precipitation $prec$, and the kind of precipitation $precType$.
- $\rho(poi, timestamp) \rightarrow \{crow\}$ which returns the level of crowding at the attraction poi in the context derivable from the timestamp $timestamp$ through function $\sigma(timestamp)$ and $\omega(timestamp)$. This function represents the crowding forecasting component in Fig. 1.

Concerning Eq. 4, the only two missing components are the user u, the current agent, and the *orig*, which can be derived from the previous state visited by the agent during the exploration. As you can notice, this state representation allows the optimization of the required amount of storage since all the contextual dimensions can be easily derived using functions σ, ω, and ρ.

The last notion to be tailored is the reward, which has to consider two essential aspects: the level of crowding in the next PoI in the given context and the distance between the two subsequent PoIs.

Definition 11 (*Reward*) Given a set of states $\mathcal{S}$ representing the context-aware tourist visits, a set of actions $\mathcal{A}$ which determines the movement from a context-aware tourist visit to the subsequent one inside a context-aware tourist itinerary. The reward R associated with the tuple (s, a, s'), where $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ is defined as:

$$R(s, a, s') = b(s, s') + \frac{vt(s')}{(qt(s') + mt(s, s') + vt(s'))} \cdot w \tag{13}$$

where $b()$ denotes the popularity of the transition from $s.poi$ to $s'.poi$ in the given context $s.\lambda$, $vt()$ is a function returning the time needed to visit $s'.poi$, $qt()$ is the time wasted in the queue for entering $s'.poi$ in the context derived from $s'.timestamp$ due to the presence of a certain level of crowding $\rho(s'.poi, s'.timestamp)$, $mt()$ is a function returning the time required for moving from $s.poi$ to $s'.poi$, it depends on the distance between the two locations along the street network and the means of transport. Finally, the factor w is a weight that can also be used to balance the importance of the two components of the reward function.

Notice that the value returned by the function $b()$ for a given pair of PoIs $s.poi$ and $s'.poi$ in the specific context is computed starting from the available historical data and allows to take into consideration situations like "tourists usually visit the Juliette House after the Verona Amphitheater in sunny weekends". Without this component, the technique will conversely ignore the usual tourist behavior. This transition preference is contextual since, for instance, the usual behavior could change in different weather conditions or periods of the year. The function $vt()$ represents the suggested time to visit a specific attraction. The tourist office has provided this information, but it can also be established with an

educated guess. The weight w has been introduced to increase the role assigned to $vt()$ and mitigate the behavior of the D-RL technique, which otherwise gives preference to sequences composed of several PoIs with a short visiting time instead of sequences with fewer PoIs but with a longer visiting time. Without this component, several popular PoIs that require more time to be visited, like the Arena Amphitheater, were never included in a suggested itinerary in favor of minor attractions like gateways or inscriptions that require much less time to be enjoyed. Since the main aim of our RS is to minimize wasted time and maximize the amount of time spent inside the attractions, the sequence length in terms of the number of PoIs is not so relevant if the time is well spent anyway. More specifically, the weight w has been set equal to $vt(s')/x$, where x is a predefined time slot that depends on the given datasets. In our experiments, we set x equal to the minimum visiting time for all our attractions.

As regards the initialization of the DNQ network, this happens in lines 3–4 of Alg. 2. Generally, the DNQ network is initialized with random weights, and the training is performed incrementally as the agent explores the environment. However, we can improve and speed up this process by exploiting the available past historical data. Specifically, starting from the set of past itineraries, we select the ones that correspond to the current recommendation query (e.g., same starting point, same context, and similar duration), and from them, we compute tuples (s_t, a_t, r_t, s_{t+1}) to put in the initial experience buffer. Details about this process have been reported in Alg. 3.

Algorithm 3 DQN initialization

1: **procedure** INITIALIZEREPLAY($history$, $q = \langle p, \lambda, \Delta_t \rangle$, σ, ω, ρ)
2:     Extract from $history$ the set of itineraries I complying q
3:     $replay = \emptyset$
4:     **for** $\zeta = \langle cv_1, \ldots, cv_n \rangle \in I$ **do**
5:         **for** $i = 1, \ldots, n-1$ **do**
6:             Extract the state s_i from cv_i by using ω and ρ
7:             Extract the state s_j from cv_j by using ω and ρ
8:             Compute the transition $t = \langle s_i, a_i, r_i, s_j \rangle$
9:             $replay = replay \cup \{t\}$
10:         **end for**
11:     **end for**
12:     **return** $replay$
13: **end procedure**

Finally, the last customization needed in Alg. 2 is the exit condition for the inner cycle in line 8. In our case, reaching a final state is represented by the termination of the available time for the visit Δ_t specified in the recommendation query.
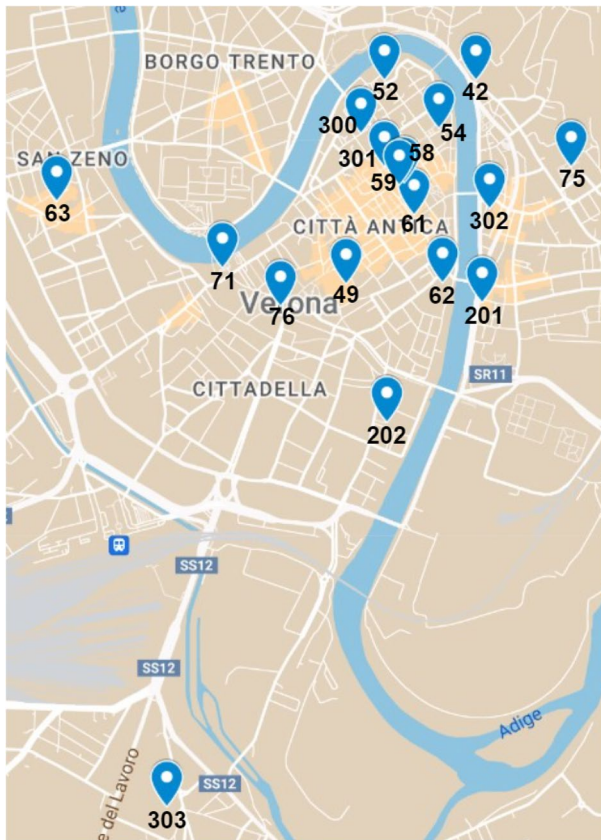
Fig. 3 Spatial position of the attractions covered by the VeronaCard city pass

5 Experiments

This section presents a set of experiments performed on a real-world dataset regarding touristic visits in the city of Verona in Italy. In particular, Sect. 5.1 describes the experimental setups and introduces both the considered baselines and evaluation metrics. Section 5.2 illustrates the behavior of the proposed T-RS in a specific scenario with a detailed comparison with the baselines. In Sect. 5.3, the results of several cumulative tests are reported, which describe the overall benefits of the proposed technique with respect to the baselines. Finally, Sect. 5.4 evaluates the scalability of the proposed approach in terms of both complexity and performance.

5.1 Experimental setup

We applied the proposed technique to a real-world dataset regarding the visits performed by tourists in Verona, a city in Northern Italy, with a city pass called VeronaCard from 2014 to 2023. This pass covers a set of 18 PoIs in Verona downtown, as

Table 1 PoIs contained in the VeronaCard dataset

Id	Name	Training		Query extraction	
		# visits	# $itiner_3$	# visits	# $itiner_3$
42	Archaeological Museum	143,017	140,524	3487	3252
49	Arena Amphitheatre	485,585	433,942	11,577	9115
52	The Cathedral	224,097	219,042	3841	3656
54	Church of St. Anastasia	251,276	243,950	4478	4231
58	Palazzo della Ragione	126,115	123,839	3630	3447
59	Lamberti Tower	330,628	317,754	7581	6842
61	Juliet's House	421,462	388,473	8430	7203
62	Church of St. Fermo	100,698	99,095	1945	1876
63	Church of St. Zeno	94,990	92,791	1429	1352
71	Castelvecchio Museum	303,924	288,595	6673	5843
75	Giusti's Garden	32,328	31,616	148	140
76	The Maffeiano Museum	43,183	42,475	1326	1250
201	Natural History Museum	17,909	17,514	1117	1020
202	Frescoes Museum Cavalcaselle	110,576	108,429	2383	2235
300	Miniscalchi Museum	58	58	19	17
301	Palazzo Maffei House Museum	887	837	481	418
302	National Museum	82	77	52	43
303	Eataly Verona	12	4	2	0

Column "# visits" contains the number of records regarding visits to this PoI, while "# $itiner_3$" contains the number of itineraries with length ≥ 3 involving it

illustrated in Fig. 3 and Table 1. The dataset contains about 2.7M visits performed by about 570K different tourists, with sequences or itineraries of average, minimum, and maximum length equal to 4.7, 1, and 15, respectively.

More specifically, we use the data from 2014 to 2019 to initialize the DQN network for Q-value estimations and train the crowding forecaster model in Belussi et al. (2022). Data from January to March 2023 are used to extract the queries and as the baseline for the method. Conversely, data from 2020 to 2022 have been discarded since, due to the COVID-19 Pandemic, they are not particularly significant. Notice that the last 4 PoIs (from 300 to 303) have been added only recently, so the number of visits regarding them is very limited. However, they have still been considered to allow a comparison with the baselines. Starting from the itineraries in the test set, we extract from them a set of recommendation queries in Eq. 7 in the following way: the starting point p is the location of the first PoI in the sequence, and the initial context λ_0 is extracted from the timestamp of the first visit by using the functions ω and ρ described in the previous section. Finally, the desired duration is given by the duration of the historical itinerary.

The initialization of the experience buffer is summarized in Alg. 3. Starting from recommendation query q and the set of past historical visits (*history*), we identify the set I of contextual itineraries that comply with q. Given such itineraries, all possible transactions are extracted and derived using the functions σ, ω, and ρ. Such

transactions are then added to the experience buffer *replay*. This initialization is used in lines 3–4 of Alg. 2. For the DQN model, we chose the following architecture: 5 sequential dense layers, Adam optimizer, and mean squared error as the loss function for the experiments.

With reference to Fig. 1, for the forecast of the occupation level of each PoI in a given context, we use the deep neural network proposed in Belussi et al. (2022). It is essentially a fully connected model with 2 hidden layers followed by 2 dropout layers. The model has been trained with contextual data about past levels of crowding in the various PoIs, and the average accuracy achieved is around 25%. This model is used to obtain the value of the $\rho()$ function, which, given a PoI and a timestamp, returns the level of crowding in such attraction in the context derived from the *timestamp* through $\sigma()$ and $\omega()$. Clearly, this model can be straightforwardly substituted with any more accurate or performing model without compromising the validity or generality of the proposed approach. For the computation of the distance between two PoIs along the road network, we use the library `OSMnx`[1]. It allows the exploitation of the geo-spatial data provided by OpenStreetMap for modeling, projecting, and visualizing road networks that can be traveled by foot, car, or bicycle. The library also allows adding custom infrastructures or personalized PoIs in the considered geographical area. Conversely, for the estimation of weather conditions, we use the API of Visual Crossing.[2] The generated datasets and the source code developed during the current study have been made available in a GitHub repository.[3]

As previously mentioned, in the experiments, we consider as baselines: (1) the original itinerary from which the recommendation query has been extracted (B-H), (2) a recommendation strategy based on their distance and popularity (B-DP), and (3) a variation of B-DP where popularity has been contextualized (B-CDP). In particular, for B-DP, the popularity of each PoI is computed from the training set as:

$$pop(p) = \frac{\#visits(p)}{|\mathcal{P}|} \tag{14}$$

where $\#visits(p)$ is the number of records regarding a visit of PoI p. Conversely, for B-CDP, the contextual popularity is defined as

$$cpop(p, \lambda) = \frac{\#visits(p, \lambda)}{|\mathcal{P}|} \tag{15}$$

where $\#visits(p, \lambda)$ is the number of records in the training set regarding the visit of PoI p in a specified context λ. Once the popularity (or its contextualized measure) is defined, it is possible to rank PoIs based on their popularity, where the most visited PoI occupies the first position, while the $|\mathcal{P}|$-th position is given to the one with the lowest number of visits. Similarly, this can also be done for distance, given the current PoI p for each possible destination PoI p', the raking is made from the nearest

---

PoI to the furthest. Alg. 4 summarized the recommendation strategy based on PoI distance and popularity (B-DP). At each step of the visit, the next PoI is chosen based on the combination of the popularity and the distance ranking. In this way, the next PoI is always the most popular among the nearest ones. Parameter *popRank* is a list ordered based on the popularity ranking computed as in Eq. 14, while *distRank* is a map returning for each PoI the list of other PoIs sorted by the distance between the two. To obtain the next PoI to visit, a new ranking that sums the popularity ranking and distance is calculated, and then the highest one is chosen. The recommendation strategy based on PoI distance and contextualized popularity (B-CDP) is computed similarly by substituting the *popRank* list with the one computed using Eq. 15.

Algorithm 4 Distance and popularity baseline

1: **procedure** B-DP($q = \langle p_0, \lambda_0, \Delta_t \rangle$, $popRank$, $distRank$)
2:     $t = \Delta_t$
3:     $\zeta = \{p_0\}$
4:     $i = 0$
5:     **while** $t > 0$ **do**
6:         $rank = \{\}$ ▷ List of PoIs sorted by the ranking
7:         **for** $q \in \mathcal{P}$ **do**
8:             $v = popRank(q) + distRank(p_i, q)$
9:             $rank = rank \cup \{\langle rank, next_p \rangle\}$
10:        **end for**
11:        p_{i+1} = extract from $rank$ the PoI with the smaller $rank$ value
12:        $\zeta = \zeta \cup \{p_{i+1}\}$
13:        update t according to the time needed to reach and visit p_{i+1}
14:     **end while**
15: **end procedure**

For the comparison between the proposed technique and the three baselines, besides analyzing the amount of time saved by using the proposed T-RS, we also analyze other standard metrics of RS that try to measure the popularity bias and diversity. *Popularity bias* refers to the phenomenon where the recommendations favor popular items over more diverse and new ones. As discussed in the introduction, the proposed T-RS wants to limit this problem and promote less known (and so less visited) PoIs. Conversely, *diversity* measures how varied the recommended items are for each user. It reflects the breadth of item types or categories to which each user is exposed. We use the coverage metric to analyze the popularity bias, while we use the Gini index for the diversity.

The *coverage* measures the shares of all items in the catalog presented in the recommendation (Ge et al. 2010). Since the amount of PoIs is limited in our experiment, we consider not only whether a PoI appears in a suggested itinerary but also how many times (i.e., in how many suggestions). This leads to a measure similar to the *Average Recommendation Popularity* (ARP) (Abdollahpouri et al. 2019; Yin et al. 2012). For the hypothesis given in this paper, where each PoI could be included

at most once in each itinerary and each itinerary is associated with a different user (since they are anonymous), the formulation of ARP can be adapted as:

$$arp = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \frac{1}{|I|} \sum_{\zeta \in I} \phi(p, \zeta) \tag{16}$$

where $\mathcal{P}$ is the set of considered PoIs and $|\mathcal{P}|$ is its cardinality, I is the set of considered suggestions, and $\phi(p, \zeta)$ returns 1 if ζ contains a contextual visit cv such that $cv.p = p$.

The *Gini* index is a measure mostly used in economics to quantify wealth or income inequality. It can be adapted for RS by considering the number of recommendations an item gets as its "wealth in the system" (Antikacioglu and Ravi 2017). Therefore, it defines the most equitable distribution as the one where every item is recommended an equal number of times.

$$G = \frac{\sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{P}} |\sum_{\zeta \in I} \phi(p, \zeta) - \sum_{\zeta \in I} \phi(q, \zeta)|}{|\mathcal{P}|^2} \cdot \frac{1}{|I|} \tag{17}$$

where the first term computes the mean absolute difference between the number of appearances of each pair of PoIs $p, q \in \mathcal{P}$, while the second term makes the measure relative normalizing the scale.

5.2 Illustrative test on a specific context

Let us consider a recommendation query for December 2nd, 2022, starting at 11 am from Miniscalchi Museum (PoI 300) with an available time of 4 h. Weather conditions are light rain, with an average temperature of 7°C, the day is a Friday, so very near the weekend. The query can be formulated by using Eq. 7 as follows:

$$q = \langle p = 300,$$
$$\lambda_0 = \langle \text{morning}, 336, 7, \text{no},$$
$$7°C, 0.1mm, \text{"light rain"}, \perp, \perp \rangle,$$
$$\Delta_t = 4 \rangle$$

where the symbol $\perp$ in the context λ_0 is used to denote the fact that this is the starting position in the sequence; namely, no previous PoI needs to be specified, and no crowding information is relevant here. Given such a query, we take the original itinerary as the first baseline (B-H), and we compute the other two: B-DP and B-CDP.

The RL technique provides the following sequence [300, 52, 76, 61]: starting from the Miniscalchi Museum (300), it then recommends to visit in order The Cathedral (52), The Maeffeiano Museum (76), and Juliet's House (61). The proposed itinerary is depicted in Fig. 4, where the color assigned to each PoI represents the degree of crowding: green means quite empty, yellow means a medium occupation level, red means overcrowded PoIs, and gray is used to identify the starting position. The letters placed along the path denote the order followed during the visit.

Fig. 4 Example of itinerary suggested by the D-RL technique. The color assigned to each PoI represents the degree of crowding, while the letters along the path denote the visit order. The overall length of the suggested path is 2.4 Km, which requires about 30 min of walking

Notice that all the PoIs included in the suggestion are quite empty, which means that no time has to be wasted in the queue before entering the attraction.

Conversely, an example of the sequence produced by the distance and popularity baseline B-DP is reported in Fig. 5. As one can notice, it comprises five PoIs; the first is quite empty, while the last three have an average occupation rate. Pretty similar is the sequence chosen by the distance and contextualized popularity baseline B-CDP in Fig. 6. The only difference concerns the second PoI visited, i.e., 58, which is more popular than PoI 59 in the specific context of the query. As one can notice, neither metrics care about the level of crowding, and the suggested PoIs present a medium occupancy level, inducing an additional waste of time in queue before entering the attractions. The overall path length is lower than the suggested one, with a small reduction in the movement time (i.e., 5 min). This is an inherent feature of the baseline policy favoring the closest PoI, independently from their level of crowding.
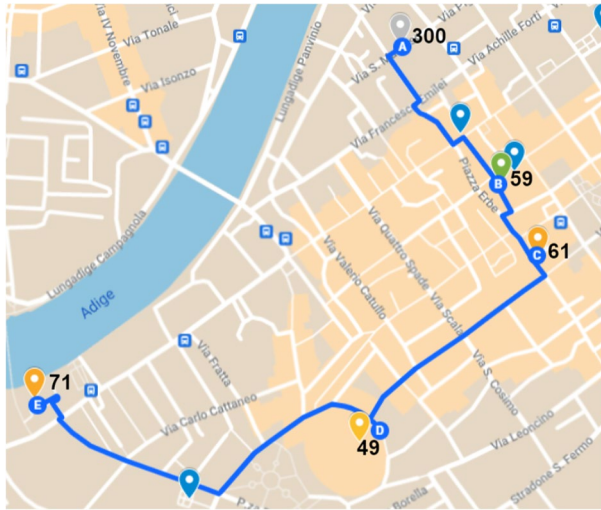
Fig. 5 Example of itinerary produced by the distance and popularity baseline. The color assigned to each PoI represents the degree of crowding, while the letters placed along the path denote the ordering followed during the visit. The overall length of the suggested path is 1.7 Km, which requires about 25 min of walking
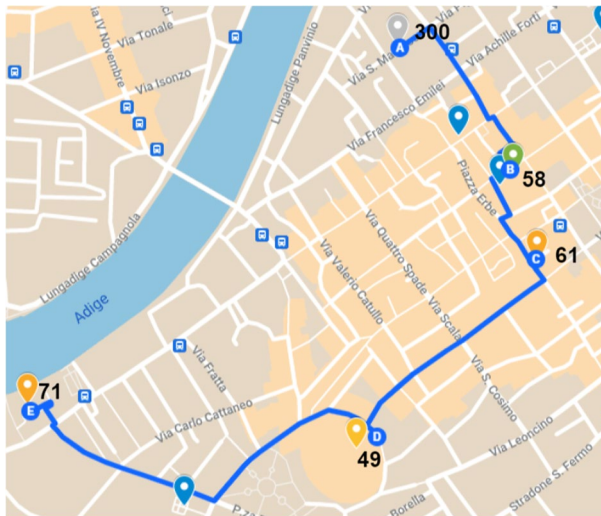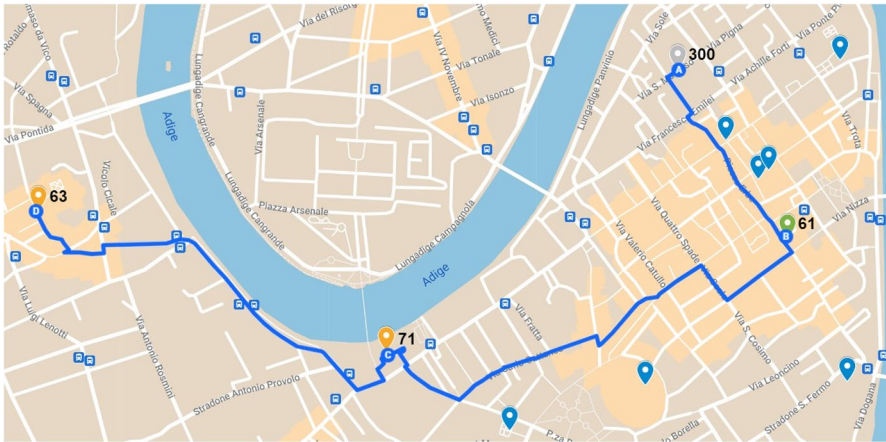

Fig. 6 Example of itinerary produced by the distance and contextualized popularity. The color assigned to each PoI represents the degree of crowding, while the letters placed along the path denote the ordering followed during the visit. The overall length of the suggested path is 1.7 Km, which requires about 25 min of walking

Fig. 7 Example of itinerary produced by the historical baseline. The color assigned to each PoI represents the degree of crowding, while the letters along the path denote the ordering followed during the visit. The overall length of the suggested path is 2.5 Km, which requires about 32 min of walking

Finally, the historical itinerary used to extract the query is reported in Fig. 7. In this case, the length of the traveled path is only a little bit longer than the suggested one (2.5Km in place of 2.4Km), but there are two PoIs with a medium level of crowding, meaning a little bit more time is wasted in the queue with respect to the provided suggestion.

5.3 Technique evaluation and performance comparisons

This section illustrates the overall results obtained by applying the proposed technique and the three baselines on batches of queries with different initial contexts and duration constraints. More specifically, as previously discussed, the test queries are retrieved from the historical data covering January to March 2023. Table 2 summarizes the average results obtained with the proposed technique (row **D-RL**), compared with the ones given by the three baselines: the original itinerary performed by the user and from which the query has been extracted (row **B-H**), the strategy based on distance and popularity (row **B-DP**) and the strategy based on distance and contextual popularity (row **B-CDP**). They are also categorized according to the hours available to complete an itinerary. Results are averaged and grouped by the desired duration of the itineraries, or in other words, the available time.

The average cumulative rewards are reported in the column **RW**. As expected, the D-RL technique is able to provide a greater average reward than all baselines in almost all the possible itinerary durations. The first four columns allow us to better analyze this behavior with respect to the cumulative reward value. They report the incidence of the various time components as a percentage with respect to the overall itinerary duration. Column **VT** reports the average cumulative visiting time, namely the time effectively spent enjoying the attractions, the itinerary proposed by our

Table 2 Comparison of the average results obtained with the proposed solution (D-RL) and the three baselines (B-H = historical, B-DP = distance and popularity, B-CDP = distance and contextualized popularity)

		VT (%)	MT (%)	QT (%)	RT (%)	RW	ARP	G
4 h	D-RL	69.58	11.67	**13.75**	5.00	**28.69**	**0.13**	**0.16**
	B-H	**70.73**	**8.79**	20.49	6.58	23.07	0.17	0.18
	B-DP	62.18	12.94	24.89	**0.00**	18.92	0.26	0.37
	B-CDP	62.18	12.94	24.89	**0.00**	18.92	0.26	0.37
5 h	D-RL	68.94	14.7	**15.0**	1.36	**35.91**	**0.17**	**0.22**
	B-H	**71.44**	**8.80**	19.76	5.46	29.63	0.21	0.23
	B-DP	66.21	10.35	20.0	3.44	27.05	0.26	0.38
	B-CDP	66.21	10.35	20.0	3.44	27.05	0.26	0.38
6 h	D-RL	**70.96**	12.12	**14.39**	2.53	**42.13**	**0.19**	**0.24**
	B-H	70.62	**9.76**	19.62	4.71	35.07	0.25	0.25
	B-DP	67.54	11.63	20.83	**0.00**	33.52	0.29	0.4
	B-CDP	67.54	11.63	20.83	**0.00**	33.52	0.29	0.4
7 h	D-RL	**73.81**	10.71	**15.0**	0.48	**46.09**	**0.23**	**0.24**
	B-H	71.18	**9.71**	19.11	3.55	41.93	0.29	0.29
	B-DP	65.46	12.39	20.71	1.44	37.62	0.33	0.44
	B-CDP	65.46	12.39	20.71	1.44	37.62	0.33	0.44
8 h	D-RL	70.23	12.41	**16.93**	**0.43**	**48.33**	**0.3**	**0.19**
	B-H	**70.54**	**10.06**	19.4	4.06	46.90	0.33	0.31
	B-DP	63.75	13.86	21.37	1.02	40.92	0.38	0.47
	B-CDP	63.75	13.86	21.37	1.02	40.92	0.38	0.47

The best values obtained for each metric (column) and experimental group (row group) are highlighted with bold

The statistics are computed by averaging the results obtained with 300 queries and grouping the results based on the available time. VT is the visiting time, MT is the moving time, QT is the queue time, RT is the remaining time, RW is the cumulative reward, ARP is the average recommendation popularity (Eq. 16), and G is the Gini index (Eq. 17)

methodology provides an overall visiting time similar to the history baseline, always higher than the baselines based on distance and popularity. The second column **MT** reports the average time required for moving from one attraction to the subsequent one. As one can notice, the provided suggestion is not always the one with the shortest path, meaning that sometimes a little more time is required to reach a less visited PoI. Overall, in the various cases, the average moving time is more similar to the last two baselines with respect to the historical behaviors. Column **QT** is the average overall time spent in the queue waiting to enter an attraction. The provided suggestion always outperforms all the baselines. As one can notice, B-H outperforms the other two baselines with respect to this measure, revealing that users tend to limit such quantity by themselves, eventually moving to a different attraction. Column **RT** reports the average remaining time after visiting the last attraction before the end of the available period; in this case, suggestions always perform better than the historical baseline, while the comparison with other B-DP and B-CDP improves as
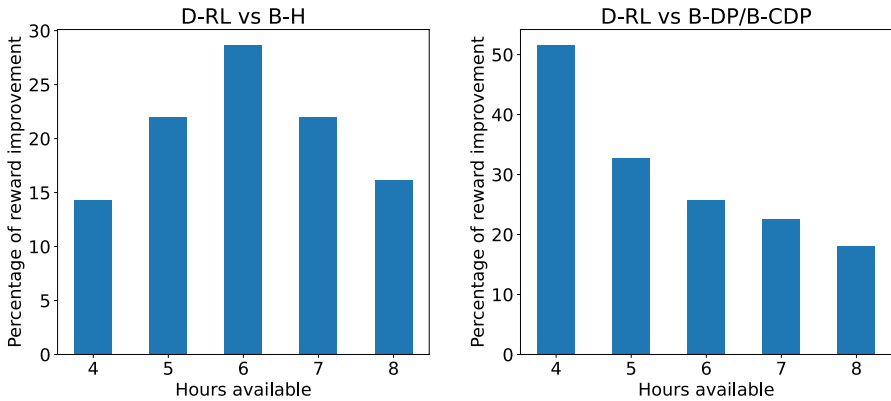
Fig. 8 Percentage of improvement of the reward value of our methodology compared to the considered baselines

the available time increases. Overall, B-DP and B-CDP baselines leave less time for tourists to visit the various attractions. Indeed, the **VT** component is always less than the other two techniques. Conversely, D-RL and historical behavior are much more similar in this regard. Overall, D-RL recommendations cause the tourist to spend more time moving from one PoI to another than waiting in a queue. This is appreciable in the considered touristic domain since, very frequently, the movement is done inside scenic routes that have their own tourist charm. Finally, the last two columns, **ARP** and **G**, report the *Average Recommendation Popularity* and the *Gini* index. The suggestions provided by the D-RL techniques always obtain itineraries with the lowest popularity bias and increase coverage. As expected, the B-DP and B-CDP always propose the most known PoIs, suffering from the popularity bias problem.

Figure 8 illustrates the average percentage of reward improvement of the D-RL technique with respect to the baselines as the amount of available time in the recommendation query increases. These percentages are greater for the comparison with B-DP and B-CDP than the comparison with B-H, demonstrating that tourists tend to avoid overcrowded situations by themselves while existing recommendation strategies usually do not take care of this aspect. Moreover, the comparison with B-H demonstrates that the provided suggestions become more useful (i.e., greater reward improvement) when enough time is available since when a small amount of time is available, the tourists are more careful not to waste time. Conversely, with respect to the other two baselines, the level of improvement decreases since, in our case, there is a limited number of PoIs and with a great amount of time at our disposal, even with these techniques also some less popular and hence less crowded PoIs are included in the suggested itinerary.
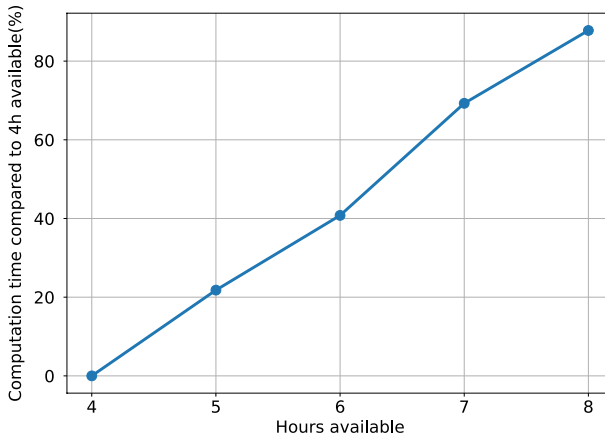
Fig. 9 Average increment of the computational time required by D-RL as the available time increases, with respect to the time required for a 4 h itinerary

5.4 Scalability tests

Besides evaluating the proposed technique with respect to the reward value and the coverage and popularity metrics, we also perform three experiments to evaluate the scalability of the approach. For this analysis, we synthetically augment the number of available PoIs with respect to the ones in Table 1 by using a tool like the one in Katiyar et al. (2020), and we perform some queries to check the effect of greater availability of attractions to choose from. We start from the initial set of 18 available PoIs and gradually increase it by 5 PoIs at a time. First of all, we examine how the execution time of our approach changes in two cases: (a) as the number of available PoIs increases and (b) as the amount of available time increases. Then, (c) we evaluate how the reward values and the quality metrics introduced in Sect. 5.1 change based on the number of available PoIs in our techniques and the two baselines based on distance and popularity (B-DP and B-CDP). Clearly, this comparison cannot be done with respect to the historical baseline B-H, because we do not have historical data about the synthetically generated PoIs.

As regards the first experiment about the execution time, we observe that the computation time remains essentially the same, with less than 5% of additional time required with 100 PoIs. This is also confirmed by the code reported in Alg. 2, since independently from the available number of PoIs, namely independently from the dimension of the action space, only an action is considered at each iteration. The two main nested loops depend only on the number of chosen episodes and the available time for the itinerary. Therefore, it is essential to consider the second experiment (b), namely how the execution time changes as the amount of available time increases. Figure 9 reports

Table 3 Comparison of the average results obtained with the proposed solution (D-RL) and the two baselines (B-DP = distance and popularity, B-CDP = distance and contextualized popularity)

		VT (%)	MT (%)	QT (%)	RT (%)	ARP	G
20 PoI	D-RL	**71.00**	10.0	**19.0**	**0.0**	**0.20**	**0.32**
	B-DP	68.60	**9.32**	21.19	0.89	0.31	0.36
	B-CDP	70.39	11.43	17.32	0.86	0.32	0.40
25 PoI	D-RL	72.00	12.33	**15.67**	**0.00**	**0.12**	**0.21**
	B-DP	**72.35**	**8.96**	18.33	0.36	0.24	0.31
	B-CDP	68.3	12.08	18.9	0.71	0.26	0.34
30 PoI	D-RL	**76.33**	**3.00**	16.67	4.00	**0.13**	**0.23**
	B-DP	71.31	9.73	18.96	**0.00**	0.20	0.27
	B-CDP	73.69	9.55	**16.34**	0.42	0.20	0.29
35 PoI	D-RL	75.67	8.67	**15.67**	**0.0**	**0.11**	**0.20**
	B-DP	69.02	9.70	20.62	0.65	0.18	0.27
	B-CDP	73.81	**7.23**	18.93	0.03	0.18	0.28
45 PoI	D-RL	**78.0**	8.67	**13.33**	**0.00**	**0.09**	**0.16**
	B-DP	74.85	**5.62**	19.49	0.03	0.11	0.18
	B-CDP	63.39	12.74	20.8	3.07	0.13	0.20

The best values obtained for each metric (column) and experimental group (row group) are highlighted with bold

The statistics are computed by averaging the results obtained and increasing the number of available PoIs. VT is the visiting time, MT is the moving time, QT is the queue time, RT is the remaining time, ARP is the average recommendation popularity (Eq. 16), and G is the Gini index (Eq. 17)

the percentage increment of the computational time with respect to the time required to produce an itinerary of 4 h. As one can notice, the increment is quite linear with respect to the increment of available time. Notice that 8 h is quite the maximum duration of a daily itinerary.

Finally, we compare how the performances of the proposed approach and of the two baselines B-DP and B-CDP change with respect to the reward function and the quality metrics, as the number of available PoIs is increased. Table 3 reports such values computed by using the same queries in Sect. 5.3 and by varying the number of available PoIs from 20 to 45. As you can notice, the proposed technique outperforms the two baselines in almost all cases as regards the ARP and Gini index. Relatively to the other parameters, as already observed in the previous section, our technique favors the augmentation of the visiting time (**VT**) and the reduction of the queue time (**QT**), eventually producing an increased moving time (**MT**). Moreover, as could be expected, the metrics ARP and the *Gini* index decrease as the set of PoIs increases.

6 Conclusion and future work

This paper deals with the problem of producing suggestions for sequences of tourist attractions or itineraries by considering the context in which the suggestion is required, and with the aim to prevent overcrowded situations. In contrast with

currently available T-RSs that are typically user-focused, the proposed solution provides suggestions that try to balance tourism's positive and negative impact in a given region from some points of view, including sustainability, by keeping in mind the usual tourist behavior. The choice to produce sequence-based recommendations in place of next-item ones greatly increases the complexity of the problem, but it allows the system to provide better and more valuable suggestions to users. Indeed, in the tourist domain, suggesting a complete itinerary covering a given period (e.g., an entire vacation) can have a helpful impact and amass great appreciation. Moreover, considering the context also allows for a more tailored suggestion. In particular, we consider as contextual dimensions the spatial-temporal characterization of the visits, the weather conditions, and, most importantly, the expected level of crowding in the various attractions within each specific situation.

A solution based on D-RL has been proposed: it considers the tourist as the agent and the action performed as his/her movement from one tourist attraction to another. The reward function is produced with the aim of taking care of the context of the visit and reducing the level of crowding in each attraction, subject to specific spatio-temporal constraints. The proposed technique has been evaluated with respect to a real-world dataset containing the tourist visits performed in Verona (Italy) through a city pass called VeronaCard. The obtained results have been compared with some baselines confirming the approach's efficacy in reducing wasted time and increasing the amount of time effectively spent visiting PoIs.

In future work, we plan to extend this approach by considering the interaction between agents moving inside the same environment and in the same context using a Cooperative Multi-Agent D-RL. The proposed technique will also be incorporated into a mobile app, which will provide suggestions to users, collect online feedback, as suggested in Mahmood et al. (2009), and consequently adapt itself, considering differences between the tourists' real behavior and the provided suggestions. This could be an optimal way to perform an evaluation from a user's perspective, with respect not only to the defined reward function but also to the user's feeling about sustainability concerns in the tourist domain.

Declarations

Conflict of interest The authors have no conflict of interest to declare that are relevant to the content of this article.

References

Abdollahpouri H, Burke R, Mobasher B (2019) Managing popularity bias in recommender systems with personalized re-ranking. CoRR arXiv:abs/1901.07555

Adomavicius G, Bauman K, Tuzhilin A, Unger M (2022). In: Ricci F, Rokach L, Shapira B (eds) Context-aware recommender systems: from foundations to recent developments context-aware recommender systems. Springer, New York, pp 211–250. https://doi.org/10.1007/978-1-0716-2197-4_6

Antikacioglu A, Ravi, R (2017) Post processing recommender systems for diversity. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. KDD '17. Association for Computing Machinery, New York, pp 707–716. https://doi.org/10.1145/3097983.3098173

Arentze TA, Kemperman A, Aksenov P (2018) Estimating a latent-class user model for travel recommender systems. J Inf Technol Tour 19(1–4):61–82. https://doi.org/10.1007/S40558-018-0105-Z

Baltrunas L, Ludwig B, Ricci F (2011) Context relevance assessment for recommender systems. In: Pu P, Pazzani MJ, André E, Riecken D (eds) Proceedings of the 16th international conference on intelligent user. (IUI 2011). ACM, New York, pp 287–290. https://doi.org/10.1145/1943403.1943447

Baral R, Iyengar S, Li T, Balakrishnan N (2018) CLoSe: contextualized location sequence recommender. In: Proceedings of the 12th ACM conference on recommender systems. RecSys'18, pp 470–47. https://doi.org/10.1145/3240323.3240410

Belussi A, Cinelli A, Dalla Vecchia A, Migliorini S, Quaresmini M, Quintarelli E (2022) Forecasting POI occupation with contextual machine learning. In: Advances in databases and information systems (ADBIS), pp 361–376

Borràs J, Moreno A, Valls A (2014) Intelligent tourism recommender systems: a survey. Expert Syst Appl 41(16):7370–7389. https://doi.org/10.1016/j.eswa.2014.06.007

Chen G, Chen L (2015) Augmenting service recommender systems by incorporating contextual opinions from user reviews. User Model User Adapt Interact 25(3):295–329. https://doi.org/10.1007/s11257-015-9157-3

Chen S, Chen BH, Chen Z, Wu Y (2020) Itinerary planning via deep reinforcement learning. In: Proceedings of the 2020 international conference on multimedia retrieval. ICMR'20, pp 286–290. https://doi.org/10.1145/3372278.3390727

Gama R, Fernandes HL (2020) A reinforcement learning approach to the orienteering problem with time windows. CoRR arXiv:abs/2011.03647

Gavalas D, Konstantopoulos C, Mastakas K, Pantziou G (2014) Mobile recommender systems in tourism. J Netw Comput Appl 39:319–333. https://doi.org/10.1016/j.jnca.2013.04.006

Ge M, Delgado-Battenfeld C, Jannach D (2010) Beyond accuracy: evaluating recommender systems by coverage and serendipity. In: Proceedings of the fourth ACM conference on recommender systems. RecSys '10. Association for Computing Machinery, New York, pp 257–260. https://doi.org/10.1145/1864708.1864761

He J, Li X, Liao L, Song D, Cheung WK (2016) Inferring a personalized next point-of-interest recommendation model with latent behavior patterns. In: Schuurmans D, Wellman MP (eds) Proceedings of the thirtieth AAAI conference on artificial intelligence, February 12–17, 2016, vol 30. AAAI Press, Phoenix, Arizona, pp 137–143 (2016). https://doi.org/10.1609/AAAI.V30I1.9994

Huang F, Xu J, Weng J (2021) Multi-task travel route planning with a flexible deep learning framework. IEEE Trans Intell Transp Syst 22(7):3907–3918. https://doi.org/10.1109/TITS.2020.2987645

Islam MA, Mohammad MM, Das SSS, Ali ME (2020) A survey on deep learning based point-of-interest (POI) recommendations. CoRR arXiv:abs/2011.10187

Jiang L, Zhou J, Xu T, Li Y, Chen H, Huang J, Xiong H (2021) Adversarial neural trip recommendation. CoRR arXiv:abs/2109.11731

Katiyar P, Vu T, Eldawy A, Migliorini S, Belussi A (2020) Spiderweb: a spatial data generator on the web. In: Proceedings of the 28th international conference on advances in geographic information systems. SIGSPATIAL '20. Association for Computing Machinery, New York, pp 465–468. https://doi.org/10.1145/3397536.3422351

Kılıçarslan D, Caber M (2018) The impacts of perceived crowding, and atmospherics on visitor satisfaction at cultural heritage sites: a comparison of Turkish and British visitors to Topkapi palace, Istanbul. J Tour Serv. https://doi.org/10.29036/jots.v9i17.25

Kong WK, Zheng SY, Nguyen ML, Ma Q (2022) A Multi-agent reinforcement learning approach towards congestion-aware route recommendation for tourists. DEIM2022

Kotiloglu S, Lappas T, Pelechrinis K, Repoussis PP (2017) Personalized multi-period tour recommendations. Tour Manage 62:76–88. https://doi.org/10.1016/j.tourman.2017.03.005

Lim KH, Chan J, Karunasekera S, Leckie C (2019) Tour recommendation and trip planning using location-based social media: a survey. Knowl Inf Syst. https://doi.org/10.1007/s10115-018-1297-4

Ludewig M, Mauro N, Latifi S, Jannach D (2021) Empirical analysis of session-based recommendation algorithms. User Model User-Adap Inter 31(1):149–181. https://doi.org/10.1007/s11257-020-09277-1

Luque-Gil AM, Gómez-Moreno ML, Peláez-Fernández MA (2018) Starting to enjoy nature in Mediterranean mountains: crowding perception and satisfaction. Tour Manag Perspect 25:93–103. https://doi.org/10.1016/j.tmp.2017.11.006

Mahmood T, Ricci F, Venturini A (2009) Improving recommendation effectiveness: adapting a dialogue strategy in online travel planning. Inf Technol Tour 11(4):285–302. https://doi.org/10.3727/109830510X12670455864203

Marchiori E, Cantoni L, Fesenmaier DR (2013) What did they say about us? message cues and destination reputation in social media. Information and communication technologies in tourism, vol 3013. Springer, Berlin, pp 170–182. https://doi.org/10.1007/978-3-642-36309-2_15

Mason P (2020) Tourism impacts, planning and management, 4th edn. Routledge, London. https://doi.org/10.4324/9780429273544

Massimo D, Ricci F (2022) Building effective recommender systems for tourists. AI Mag 43(2):209–224. https://doi.org/10.1002/aaai.12057

Massimo D, Ricci F (2023) Combining reinforcement learning and spatial proximity exploration for new user and new poi recommendations. In: Proceedings of the 31st ACM conference on user modeling, adaptation and personalization. UMAP '23. Association for Computing Machinery, New York, pp 164–174. https://doi.org/10.1145/3565472.3592966

Merinov, P.: Sustainability-oriented recommender systems. In: Proceedings of the 31st ACM conference on user modeling, adaptation and personalization. UMAP '23. Association for Computing Machinery, New York, pp 296–300 (2023). https://doi.org/10.1145/3565472.3595617

Merinov P, Massimo D, Ricci F (2022) Sustainability driven recommender systems. In: Pasi G, Cremonesi P, Orlando S, Zanker M, Massimo D, Turati G (eds) Proceedings of the 12th Italian information retrieval workshop. CEUR Workshop Proceedings, vol 3177. CEUR-WS.org, Milan, Italy

Migliorini S, Carra D, Belussi A (2018) Adaptive trip recommendation system: balancing travelers among POIs with MapReduce. In: IEEE international congress on big data. BigData Congress 2018, pp 255–259. https://doi.org/10.1109/BigDataCongress.2018.00045

Migliorini S, Quintarelli E, Carra D, Belussi A (2019) Sequences of recommendations for dynamic groups: what is the role of context? In: 2019 IEEE International congress on big data (BigDataCongress), pp 121–128. https://doi.org/10.1109/BigDataCongress.2019.00029

Migliorini S, Carra D, Belussi A (2021) Distributing tourists among POIs with an adaptive trip recommendation system. IEEE Trans Emerg Top Comput 9(4):1765–1779. https://doi.org/10.1109/TETC.2019.2920484

Migliorini S, Quintarelli E, Gambini M, Belussi A, Carra D (2022) Sequence recommendations for groups: a dynamic approach to balance preferences. Inf Syst 108:102023. https://doi.org/10.1016/j.is.2022.102023

Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller MA (2013) Playing atari with deep reinforcement learning. CoRR arXiv:abs/1312.5602

Patro GK, Chakraborty A, Banerjee A, Ganguly N (2020) Towards safety and sustainability: designing local recommendations for post-pandemic world. In: Proceedings of the 14th ACM conference on recommender systems. RecSys '20, pp. 358–367. Association for Computing Machinery, New York https://doi.org/10.1145/3383313.3412251

Sustainable development. https://www.unwto.org/sustainable-development. Accessed 3 Jan 2023

Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. Adaptive computation and machine learning. MIT Press, Cambridge. https://www.worldcat.org/oclc/37293240

Villegas NM, Sánchez C, Díaz-Cely J, Tamura G (2018) Characterizing context-aware recommender systems: A systematic literature review. Knowl Based Syst 140:173–200. https://doi.org/10.1016/j.knosys.2017.11.003

Wang P, Liu K, Jiang L, Li X, Fu Y (2020) Incremental mobile user profiling: Reinforcement learning with spatial knowledge graph for modeling event streams. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, pp 853–861. https://doi.org/10.1145/3394486.3403128

Watkins CJCH, Dayan P (1992) Q-learning. Mach Learn 8(3):279–292. https://doi.org/10.1007/BF00992698

Wörndl W, Hefele A, Herzog D (2017) Recommending a sequence of interesting places for tourist trips. J Inf Technol Tour 17(1):31–54. https://doi.org/10.1007/S40558-017-0076-5

Xiang Z, Fuchs M, Gretzel U, Wolfram H (2022) Handbook of e-Tourism. Springer, Berlin. https://doi.org/10.1007/978-3-030-05324-6

Yin H, Cui B, Li J, Yao J, Chen C (2012) Challenging the long tail recommendation. Proc VLDB Endow 5(9):896–907. https://doi.org/10.14778/2311906.2311916

Yu J, Egger R (2021) Tourist experiences at overcrowded attractions: a text analytics approach. In: Wörndl W, Koo C, Stienmetz JL (eds) Information and communication technologies in tourism 2021. Springer, Cham, pp 231–243

Yuan Q, Cong G, Ma Z, Sun A, Thalmann NM (2013) Time-aware point-of-interest recommendation. In: Proceedings of the 36th international ACM SIGIR conference on research and development in information retrieval. SIGIR'13, pp 363–372. https://doi.org/10.1145/2484028.2484030

Zhang W, Fesenmaier DR (2018) Assessing emotions in online stories: comparing self-report and text-based approaches. Inf Technol Tour 20(1):83–95. https://doi.org/10.1007/s40558-018-0122-y

Zhou Y (2020) A dynamically adding information recommendation system based on deep neural networks. In: 2020 IEEE international conference on artificial intelligence and information systems (ICAIIS), pp 1–4 . https://doi.org/10.1109/ICAIIS49377.2020.9194792

Authors and Affiliations

Anna Dalla Vecchia[1] · **Sara Migliorini**[1] · **Elisa Quintarelli**[1] · **Mauro Gambini**[1] · **Alberto Belussi**[1]

✉ Sara Migliorini
  sara.migliorini@univr.it

  Anna Dalla Vecchia
  anna.dallavecchia@univr.it

  Elisa Quintarelli
  elisa.quintarelli@univr.it

Mauro Gambini
mauro.gambini@univr.it

Alberto Belussi
alberto.belussi@univr.it

1   Department of Computer Science, University of Verona, Strada Le Grazie, 15, Verona, Italy