Received 25 July 2023, accepted 11 August 2023, date of publication 15 August 2023, date of current version 25 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3305584

RESEARCH ARTICLE

Smart E-Learning Framework for Personalized Adaptive Learning and Sequential Path Recommendations Using Reinforcement Learning

SAMINA AMIN[1], M. IRFAN UDDIN[1], (Member, IEEE), ALA ABDULSALAM ALAROOD[2], WALI KHAN MASHWANI[3], ABDULRAHMAN ALZAHRANI[4], AND AHMED OMAR ALZAHRANI[4]

[1]Institute of Computing, Kohat University of Science and Technology (KUST), Kohat 26000, Pakistan
[2]College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia
[3]Institute of Numerical Sciences, Kohat University of Science and Technology (KUST), Kohat 26000, Pakistan
[4]Department of Information System and Technology, College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia

Corresponding author: M. Irfan Uddin (irfanuddin@kust.edu.pk)

This work was supported by the University of Jeddah, Jeddah, Saudi Arabia, under Grant UJ-22-DR-110.

ABSTRACT Learning activities are considerably supported and improved by the rapid advancement of e-learning systems. This gives students a tremendous chance to participate in learning activities worldwide. The Massive Open Online Courses (MOOCs) platform has emerged as one of the most significant platforms for e-learning as a result of the rapid growth of network information technologies. Due to the increased number of online courses available, it is harder for individual learners to choose the appropriate courses, activities, and learning paths for the actual necessities they want, which reduces their learning performances. Moreover, a sequential Recommender System (RS) can identify the learner's future interest and suggest the subsequent item or learning content given a sequence of past interactions. This is in contrast to interactive recommendation methods that can create recommendations based on the learner's feedback via constant interactions. To address these challenges, the goal of this paper is to propose a Reinforcement Learning (RL) based smart e-learning framework with Markov Decision Process (MDP) that has the potential to enhance the learning experience for each student by providing them with a personalized and effective learning path. Applying the MDP and RL-based techniques such as Q-learning for Sequential Path Recommendation (SPR) and learning development is more achievable. This is because the MDP allows for adjusting the recommendations method to find new activities and learning paths based on the learners' feedback on recommendations results. Experimental findings reveal that the suggested model obtains significant improvements and provides viable performance under different parameters optimization. Furthermore, we also show that the proposed method outperforms a long session (long-term rewards such that they maximize learning progress while minimizing frustration and disengagement). This demonstrates the model's improvements in simulating the learner's sequential behavior, learning activities, various learning materials, and learning paths simultaneously. These promising initial results provide a possible solution to assess this challenge further in future work.

INDEX TERMS Markov decision process, reinforcement learning, Q-learning, e-learning, adaptive learning, sequential behavior, MOOC, recommender systems.

I. INTRODUCTION

Personalized learning is an approach to education that emphasizes tailoring instruction to the unique needs, interests, and abilities of each student. This method tries to give students a personalized learning experience that caters to their individual needs while acknowledging that they have diverse learning preferences, methods, and goals. Personalized learning can be implemented in a variety of educational settings, from traditional classrooms to online learning environments. It has

The associate editor coordinating the review of this manuscript and approving it for publication was Fu Lee Wang.

the potential to improve student engagement, motivation, and learning outcomes by providing students with a more personalized and relevant learning experience. However, it requires careful planning, ongoing assessment, and skilled facilitation to be successful. While adaptive learning is a method of education that uses technology to personalize the learning experience for each student. It involves using data analytics and machine learning techniques to gather information about each student's learning style, pace, and performance and then adjust the content and difficulty level of the course material to match their individual needs in the online learning platform.

MOOC is one of the most significant approaches of online learning platforms as a result of the quick advancements in network information technologies. Due to the increased number of online courses available, it is harder for leaners to choose the courses they want. This reduces their learning performance. Since RSs [1], [2] can handle the issue of information overload [3], [4], personalized course recommendation [5], [6], [7] has emerged as the primary research area to tackle the aforementioned challenges in recent years. MOOC platforms offer a variety of different courses. It is essential to recommend someone on the best course to take by developing the abilities required for the learner's ideal future career. For instance, the course's learning achievement can indicate the degree to which a student possesses a particular credential or expertise. Based on the learning results of the courses, the competencies, abilities, and knowledge can be compared required by the workforce with those offered by online courses. Moreover, sequential RSs [8], identify the learner's future interest and suggest the subsequent item or learning content given a sequence of past interactions. This is in contrast to interactive recommendation methods that can create recommendations based on the learner's feedback via constant interactions. However, the application of AI in delivering distance learning education will become more common in the future as academic institutions place a greater emphasis on personalized and adaptable learning. With the help of AI-powered technologies, students can enrol/register in any program (course) anywhere in the world.

To analyze the learning activities of students in e-learning [9], a few techniques take into consideration the adaptability of the RS. For example, Pang et al. [10] designed a method termed adaptive recommendation for MOOC. Adaptive RS uses grades and study timeframes as parameters for a recommendation. Parameters must be present for an item to be recommended since the learning activity has not yet occurred. In the opinion of a similar leaner, one is with collaborative-based filtering's recommendation. The second involves the time series in the adaptive perspective. It takes just as much creativity to suggest integrating the target learner's time series of learning behaviors with the grades and study timeframes of similar learners. With the development of information technology, MOOC has quickly turned into essential forums for knowledge development. Although it is regarded as a multi-criteria challenge, finding acceptable learning resources from a vast variety of academic tools has become more challenging for learners as a result of the growing amount of information available. Utilizing a Personalized Recommender System (PRS) based on RL is one approach to get around this problem.

PRS [11], [12] are capable of delivering interesting content that corresponds to users' interests, hence reducing the issue of information overload. The majority of the time, recommendation algorithms use a variety of data to present users with potential things. In real-world circumstances, the RS makes suggestions for things based on the history of user-item interactions and then collects input from the user to refine those recommendations [13]. In other ways, the RS seeks to learn about users' interests through interactions and make suggestions for products they would find interesting. In order to achieve this, the initial recommendation study mainly concentrates on designing content-based filtering and collaborative-based filtering techniques [14], [15].

Traditional course RSs [16], [17] employed collaborative-based filtering methods [10], [18] to collect implicit feedback that indirectly indicates a learner's preferences. Recently, deep learning-based neural recommendation algorithms have surpassed these methods [18], [19]. One such model is the neural attentive session-based RS [20], which replicates users' sequential actions and derives users' primary goals from their learning patterns. In addition, the fundamental RS based on the attention network and the profile reviser simultaneously developed by the hierarchical RL method [21] reduces noisy courses. However, when students are registered in multiple courses, the course recommendation performance can be enhanced. Since hierarchical RL does not take into account the student's explicit demands or implicit preferences, it could also yield poor recommendation outcomes.

The aforementioned techniques are competent in terms of course recommendations to some point, however, they all have the flaw of ignoring the users' shifting preferences in sequential learning ability/activities. Moreover, these techniques might not accurately reflect the user's preferences for each content, particularly if the users' preferences are being changed over time across a variety of courses. In this instance, these techniques fall short of offering the RS adaptivity efficiently; specifically, they are weak at tracking the variations in users' preferences adaptively. Traditional RS suffers from the data sparsity challenge in real-world applications. In other words, only a small part of all course material in the RS can be found in a user's list of highly-rated/studied courses. Nevertheless, all potential candidate courses should be explored in order to retrieve learning materials that are relevant to the learners' interests and preferences. With the help of the users' sequential interaction data, sequential recommendations aim to determine their subsequent decision. Markov chains can be used to model sequential behaviors efficiently [22]. Particularly, a variant of the Markov chain known as the Markov Decision Process (MDP) (covered in more detail in section III-B), offers a

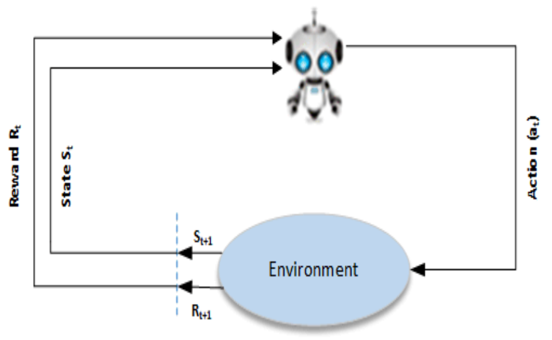

FIGURE 1. Overview of RL's structure.

mathematical formulation for simulating scenarios involving decision-making. Nearly all RL issues may be formulated as MDP issues. For this, five key components, including state, action, reward, transition, and discount factor can be used to represent MDP. The ability to deliver a personalized experience to the diverse demographics that MOOC aspires to serve has been constrained, which has impeded the development of a more democratic learner management framework.

To solve the aforementioned problems, this paper develops a personalized adaptive and SPR framework in e-learning and proposes a framework based on RL with MDP techniques. Specifically, a significant problem in the learning process is adapting elements, including reading, listening, quizzes, assignments, entertainment, playing games, etc. to possible variations in learners' states and desires, incorporating earlier study or learning process. RL is frequently used to create RS when the user's behavior is dynamic. Therefore, deploying an RL agent can be quite helpful in those situations because the agent continuously learns by interacting with the environment. The agents should dynamically modify information based on the user's preferences and temporal variations as well as from time to time in online course recommendations.

RL is a type of machine learning that involves an agent (in our case, a computer program in the e-learning platform that decides what to display next) taking actions in an environment (the e-learning content) to maximize a reward signal (the student's learning progress). A general structure of RL is shown in Figure 1. Where *Agent* is a program (algorithm) that decides what to display next in a collection of e-learning; *Environment* is a learning framework; *Action* is used to recommend a new class tutorial or reading notes, do an assignment, take a quiz, exams, or an advertisement, etc.; *State*: a learner's interaction features are depicted as a state. The state-value can be defined to evaluate the goodness of the current state (e.g., the current position of the learner) and *Rewards* are the feedback signals provided by the environment to the agent after it takes an action. Rewards indicate the desirability of the agent's action in a particular state. In our context, it is positive if the learner decides to watch the class video; the reward is more positive if the learner chooses to take exams; if the learner exits, plays a game, or becomes bored, the result can be negative. In the context of personalized adaptive sequential learning, the goal is to use RL to recommend the most effective sequence of learning activities for each student, such that they maximize their learning progress while minimizing their frustration and disengagement.

For example, if a student is struggling with a particular concept, the system may present additional examples or resources to help them master the material. Alternatively, if a student is progressing quickly through the material, the system may challenge them with more advanced content. Overall, personalized adaptive sequential learning and SPR is a powerful approach to e-learning that can help learners achieve their goals more effectively and efficiently. By tailoring the learning experience to the individual needs and preferences of each learner, this approach can help increase engagement, motivation, and learning outcomes.

The following list highlights the primary contributions of the proposed model.

1. To propose a novel virtual environment using RL with MDP terminologies that will make sequential decisions adaptively in multipath navigation and recommend the most suitable learning content based on the unique characteristics of each learner.
2. To create a personalized adaptive sequential learning and SPR method that effectively tracks a user's changing preferences. During each iteration/cycle of recommendation, it dynamically modifies the need for the relevant learning paths.
3. To signify a long-term student outcome by maximizing long-term rewards, accounting for possibly slow or average learning effects and influences on future learning choices.
4. The learner's feedback is collected by the agent. The agents should dynamically modify information based on the learner's preferences as well as from time to time in online learning and SPR.
5. To adapt elements such as reading, listening, assignments, quizzes, games, etc. to possibly dynamic changes in learners' states and preferences, including earlier study or learning experiences.

The remainder of this article is systematized in the following manner. The relevant work on literature is discussed in Section II, and the proposed methodology is presented in Section III. Section IV presents the experimental outcomes. The conclusions and prospective improvements of the proposed methodology are covered in Section V.

II. LITERATURE REVIEW

In this section, the current relevant studies are reviewed in the following research topics: (A) personalized recommendations, (B) Sequential recommendations, (C) RL for recommendations, and (D) Sequential personalized course recommendations.

A. PERSONALIZED RECOMMENDATIONS

Personalized recommendations [11], [12] are capable of delivering interesting content that corresponds to users' interests, hence reducing the issue of information overload. Various practical RS are developed based on users' preferences [23], [24], improving community membership information [25], deep neural network method with item embedding [26], hybrid RS for patron-driven library collection and pruning [27], [28], and a travel planning system using clustering to provide personalized point-of-interest recommendations [29]. However, the main technical issues that restrict the widespread deployment of the RS are still issues with data sparsity, cold start, and interpretability [30], [31].

B. SEQUENTIAL RECOMMENDATION

In RS, users develop a vast number of interaction behaviors over time. The process of sequential recommendations identifies the user's subsequent interaction item by extracting information from these behavioral patterns. The user's historical behaviors are crucial in RS for simulating the user's interests. Numerous widely used recommendation techniques, such as collaborative filtering [32], [33], develop the model using samples of user behavior.

Based on Markov chains, a sequential RS predicts users' subsequent actions based on their previous behavior [34]. The MDP presented by Moling et al., [36], is designed to deliver recommendations in a session-based way, and the most basic MDP is the first-order Markov chains in which the following recommendation can be easily calculated using the transition probabilities between items. Contiguous sequential patterns are better suited for sequential prediction tasks than generic sequential patterns.

In the context of tailored sequential pattern mining for next-item recommendations, Yap et al., [35] present a new competence score measure. As Markov chains, playlists are modelled by Shambour [3] who also suggest logistic Markov embeddings to learn song representations for playlist prediction. When trying to incorporate all conceivable sequences of probable user selections over all objects, the state space quickly becomes unmanageable, which is a significant problem when employing Markov chains to the session-based recommendation task.

Markov chains are a useful tool for modelling sequential behaviors [8], [22]. For instance, Moling et al., [36] designed a channel RS using implicit feedback from user hearing behavior. Given that the learning process is comparable to the Markov chain of sequential actions, sequential recommendation techniques can likewise be successfully applied to e-learning environments [37], [38]. To find the learner's context and sequential access behavior, Tarus et al., [39] suggested a sequential recommendation architecture that includes context awareness, sequential pattern mining, and the collaborative-based filtering technique. In order to update the user profile, Zhang et al., [21] proposed a hierarchical sequential decision mechanism. This aids the sequential RS in making more appropriate course recommendations for the users.

C. REINFORCEMENT LEARNING FOR RECOMMENDATIONS

RL has been widely used in developing various domains. By offering actual paths with the RL method over a knowledge graph, Xian et al., [40] designed an RL-based knowledge graph reasoning method that combines recommendation and interpretability.

When developing an RS where the user's behavior is dynamic, RL is frequently utilized. For instance, the user's preferences or behavior may occasionally alter while using a music recommendation engine [41], [42]. Although RL agents constantly learn by interacting with their environment, using RL in those situations can be highly helpful. Recurrent RL is superior in stock trading [43] and investing decision-making [44]. For example, Ji et al., [50] proposed a prototype RL structure to produce sentence interpretations with a customizable attention-based neural network, which dynamically regulates the explanatory performance.

Since a machine learning technique emphasizes how an intelligent agent engages with its surroundings, RL [45], [46], develops the policy through trial and error exploration, which is advantageous to sequential making decisions. As a result, it may offer methods for modelling the interactions between the user and the agent. A recent research trend in RS is the use of RL to overcome recommendation challenges [47], [48], [49], [50], [51]. To determine the best recommendation strategy, the recommender agent can frequently interact with the environment (such as users or collected data). In real-world applications, RL-based RS has been used in numerous specific contexts [52], [53], [54], such as health care [55], [56], [57], movie recommendation [58], [59], personalized music recommendation [41], e-commerce [60], [61], and e-learning [6], [21], [62].

D. SEQUENTIAL PERSONALIZED COURSE RECOMMENDATION

In order to recommend courses, traditional approaches [16], [63] typically use feature engineering. The following categories apply to the sophisticated course recommendation techniques currently in use [64]; collaborative-based filtering [65], content-based filtering [19], [66], hybrid RS [67], [68], semi-supervised [69], ontology-based [70], [71], and sequence mining-based [72], etc. Additionally, Bousbahi and Chorfi [16] proposed a model by using a case-based reasoning method and a distinctive information retrieval algorithm, the system suggests the most suitable MOOC for the learners.

Hybrid course recommendation techniques have emerged as a standard response to difficult problems in order to meet the demands of individualized learning. To address the issues of sparsity and the cold start problem in course RS, Jing and Tang [17] integrated collaborative-based filtering

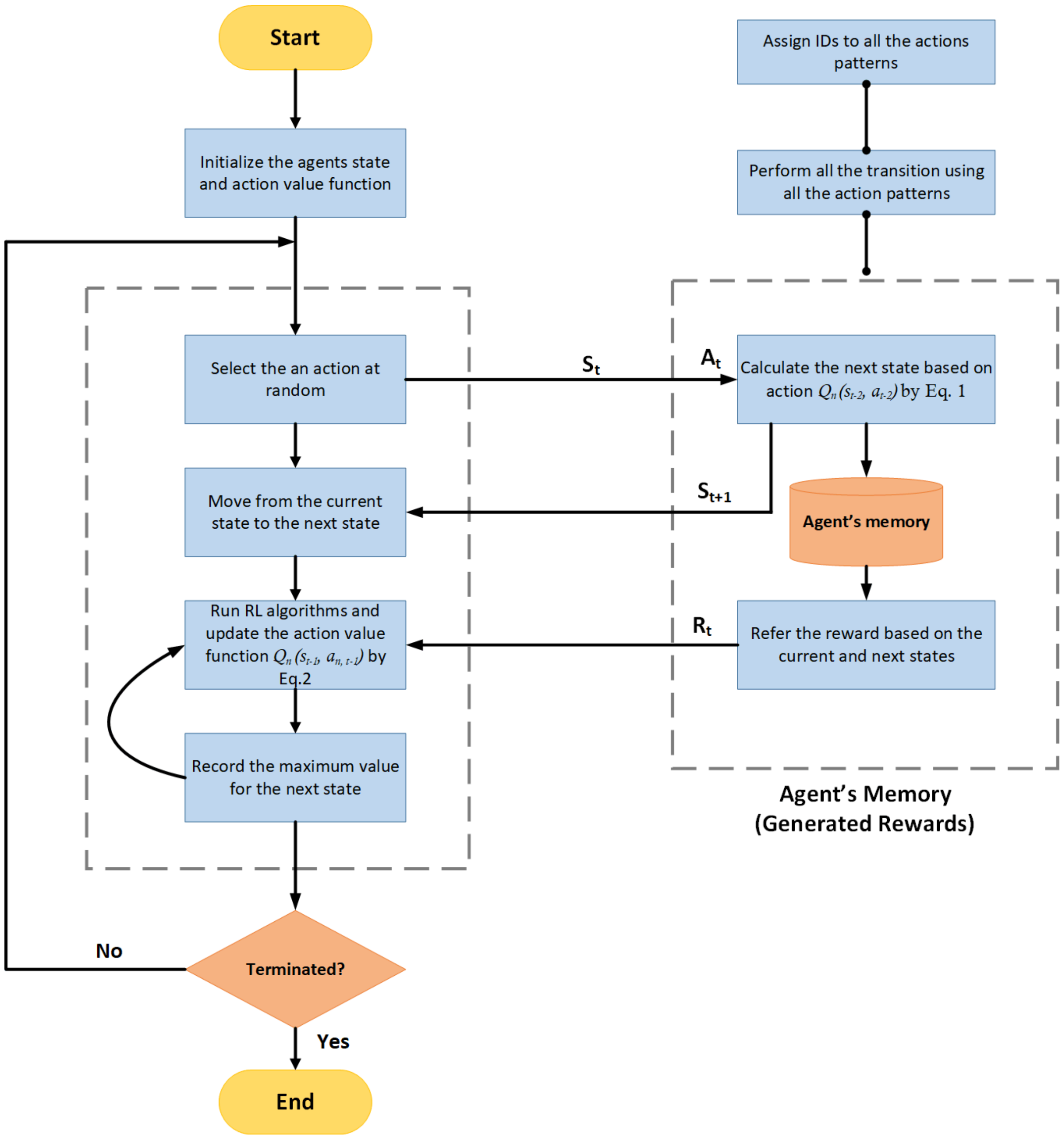
Start
Initialize the agents state and action value function
Select the an action at random
Move from the current state to the next state
Run RL algorithms and update the action value function $Q_n(s_{t-1}, a_{n,t-1})$ by Eq.2
Record the maximum value for the next state
Terminated?
No
Yes
End
Assign IDs to all the actions patterns
Perform all the transition using all the action patterns
S_t
A_t
Calculate the next state based on action $Q_n(s_{t-2}, a_{t-2})$ by Eq. 1
S_{t+1}
Agent's memory
R_t
Refer the reward based on the current and next states
Agent's Memory (Generated Rewards)

FIGURE 2. Flowchart of the proposed Q – Learning algorithm.

and content-based techniques. To explain multisource heterogeneous data, Zhu et al., [64] suggested a hybrid RS that made use of a graph-structured teaching assessment network. To understand the user's relationship structure, a neural network made up of random walks was used, and a Bayesian probabilistic tensor factorization was used for course recommendation.

In this proposed work, we offer a personalized adaptive learning and SPR framework that uses an RL method with MDP terminologies and a dynamic attention mechanism as inspiration from the aforementioned research. As opposed to earlier dynamic attention techniques, the suggested framework automatically records the user's dynamic preferences during each session and adaptively modifies the attention weights of associated items at each session in the RS.

To the best of our knowledge, this is the novel approach to overcoming the difficulty of designing a real-time, personalized adaptive solution in an e-learning environment. The proposed approach tackles the requirement to discover a means to support students in improving their effectiveness in navigating the learning materials. Since previous research demonstrated, there is a significant correlation between

TABLE 1. List of states, actions, and rewards assigned to each action.

List of states	List of actions and reward assigned to each action
State 0 – reflects "start"	
State 1 – reflects "reading"	1: Stay – '10'
State 2 – reflects "watching (video lesson)"	2: Previous course – '20'
State 3 – reflects "entertaining"	3: Additional content – '50'
State 4 – reflects "getting bored or frustration"	4: Do assignment/quiz – '60'
State 5 – reflects "rest/sleep"	5: Move to exams – '100'
State 6 – reflects "writing"	6: Move to the high-level course – '100'
State 7 – reflects "playing a game"	7: Move to the low-level course – '70'
State 8 – reflects "clicking on an ad"	8: Move to social media – '-10'
State 9 – reflects " course completion"	
State 10 – reflects "quit study or disengagement"	

e-learning learner's success (as measured by course completion) and the qualities of their learning journey through the course. As a result, a new method for helping e-learning students is introduced in the proposed SPR. Furthermore, the adaptive learning and sequential learning framework uses RL based model (Q-learning) to track each student's progress and adapt the content and pace of instruction in real-time based on their responses. The goal of this approach is to use RL techniques to recommend the most effective learning path for each student, based on their learning history, goals, and preferences.

III. PROPOSED METHOD

This section discusses the use of dynamic programming to resolve real-world issues where the environment is perfectly modelled. Dynamic programming is a generic strategy for solving issues by decomposing them into smaller issues that can be resolved independently, processed, and then merged to solve the larger issue. The flowchart of the proposed Q-learning algorithm is depicted in Figure 2, which is covered in detail in the following subsections.

A. STUDY FRAMEWORK (SIMULATION FRAMEWORK)

For simulations, a virtual environment is built to make/produce states and outcomes based on certain actions. The virtual framework is formed based on the learning states (activities) of the learner in an online learning scenario, including watching a video lesson, "reading", "writing", "getting bored or frustration", "resting/sleeping", "entertaining", "playing a game", "clicking on ads", "course completion", and "quit study or disengagement". Figure 3 reveals the different stages of the proposed framework. Due to the randomness in various transitions, designing a virtual SPR approach makes the problem more challenging. The proposed framework is developed in a 2D network where the students can adaptively change their learning path (see Figure 3).

The framework is deliberately developed to grasp the best policy for personalized adaptive learning and recommendations, as depicted in Figure 3. Here, states are represented as S ∈ {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, where each number of states reflects the position of each learner during the learning process. Then, rewards are assigned to each state-action pair based on the problem and learning context. The proposed states and their corresponding reward values are listed in Table 1. *Higher rewards* encourage the agent to priorities certain actions or states that are more valued and attractive than others. *Average rewards* can be utilized to offer modest rewards for actions that are generally positive but not necessary. Without overly favoring or ignoring certain actions, *medium rewards* can assist the agent in exploring and learning a balanced policy. *Negative/Lower rewards* can assist the agent in learning to ignore the actions that result in undesirable outcomes or learning in the incorrect direction. The problem is solved by leveraging one-shot policy recommendations for the modelling of SPR and personalized learning. The set of actions followed in the development of the proposed framework are recommending the next course, next video, recommending a game or ad, suggesting future influence for personalized learning, improvement in recommending suitable content, influence on future learning choices, and attempting to maximize the learner's satisfaction and minimize learner interactions to learn in light the learner's performance features and design personalized adaptive paths by plummeting negative experiences.

B. MARKOV DECISION PROCESSES (MDP)

Figure 4 illustrates an MDP, which is a fundamental framework of RL and fulfills a Markov property. A Markov property is one where the agent just cares about the current state of the process and has no curiosity about the entire history [73]. Mathematically, it is defined in Equation (1).

$$P\left(s_{t+1}|s_0, a_0, s_1, a_1, \ldots\ldots\ldots\ldots, s_t, a_t\right) = P\left(s_{t+1}|s_t, a_t\right) \quad (1)$$

Here P is the probability of a state transition, s shows the state, a represents action and t shows time. Every epoch, the agent performs an action that modifies its surroundings and yields a reward. Value functions and the best policy are suggested as additional processing methods for the reward value.

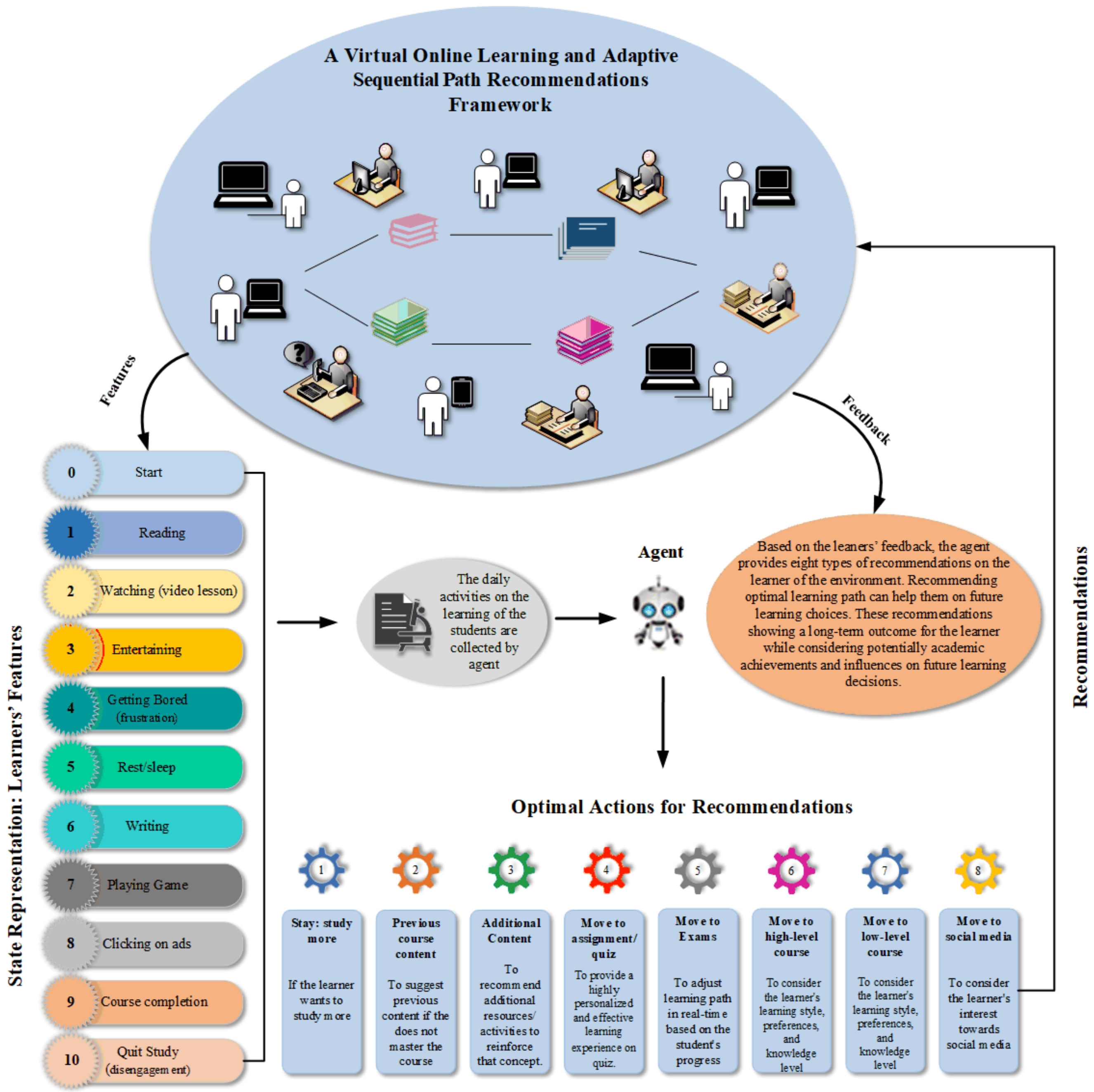
A Virtual Online Learning and Adaptive Sequential Path Recommendations Framework
Features
Feedback
Recommendations
State Representation: Learners' Features
0 Start
1 Reading
2 Watching (video lesson)
3 Entertaining
4 Getting Bored (frustration)
5 Rest/sleep
6 Writing
7 Playing Game
8 Clicking on ads
9 Course completion
10 Quit Study (disengagement)
The daily activities on the learning of the students are collected by agent
Agent
Based on the leaners' feedback, the agent provides eight types of recommendations on the learner of the environment. Recommending optimal learning path can help them on future learning choices. These recommendations showing a long-term outcome for the learner while considering potentially academic achievements and influences on future learning decisions.
Optimal Actions for Recommendations
1 Stay: study more — If the learner wants to study more
2 Previous course content — To suggest previous content if the does not master the course
3 Additional Content — To recommend additional resources/ activities to reinforce that concept.
4 Move to assignment/ quiz — To provide a highly personalized and effective learning experience on quiz.
5 Move to Exams — To adjust learning path in real-time based on the student's progress
6 Move to high-level course — To consider the learner's learning style, preferences, and knowledge level
7 Move to low-level course — To consider the learner's learning style, preferences, and knowledge level
8 Move to social media — To consider the learner's interest towards social media

FIGURE 3. **Proposed framework – personalized adaptive learning and SPR.**

In this paper, a discrete time stochastic control method known as MDP is employed for simulation. It offers a mathematical formulation for simulating decision-making in contexts where results are partially determined by chance and partially operated by the decision-maker. When a decision-maker engages sequentially with the environment, MDP is a useful framework for simulating those sequential decision-making challenges. An MDP presents a framework to the RL agent as depicted in Figure 4. First, we study the foundations of MDP to understand the framework.

According to the Markov property, the present is the only factor that affects the future and not the past. The Markov chain is a probabilistic technique in which the future is conditionally independent of the past and only relies on the current state, not on any prior states. Transition is the act of changing from one state to another, and its probability is referred to as a transition probability, where the next state solely depends on the current state.

Moreover, Markov property only considers the current state and not previous states (e.g., it makes the supposition

Algorithm 1 Pseudocode for Calculating Transition Probability Using MDP

```
initialize: states, actions, transition probability
define: transition probability matrix
                trans_prob = {}
initialize: transition probabilities
        for state in states:
            trans_prob[state] = {}
        for action in actions:
            trans_prob[state][action] = {}
assign: transition probability
        trans_prob['reading']['stay'][' Course completion '] = 0.05
        trans_prob['reading']['Additional content ']['Course completion '] = 0.1
        trans_prob['reading']['Do assignment/quiz ']['Course completion '] = 0.3
        trans_prob['reading']['Move to exams ']['Course completion '] = 0.5
        trans_prob['reading'][' Move to social media']['Course completion '] = 0.05
compute: transition probability for a state-action pair
                def comp_trans_prob(state, action, next_state):
        if next state in trans_prob[state][action]
                return trans_prob[state][action][next_state]
        else
                return0
calculate:
    result ← comp_trans_prob(state, action, next_state)
end
```

that the past is completely represented in the present.). In other terms, considering present circumstances mean that the future is independent of the past. A Markov Process is a process that possesses such a feature. A set of states with a Markov property, such as S_1, S_2, $S_3 \ldots$, S_n is referred to as a process in this context. The transition function P, or the transition probability to move from one state to another, and the state S are the two parameters used to define it. The reward collected for a Markov process is defined as a Markov reward process. It is formulated as State S, Transition Function P, Reward R, and Discount factor γ. Considering a reward in the present, the discount factor explains the rewards in the future. If γ is 0, the agent simply considers the subsequent reward. If γ is 1, the agent is concerned with all potential future rewards.

In MDP [73], a state S, transition function P, reward R, discount factor γ, and a collection of actions a serve as its representation. One could imagine an MDP as the interaction between an agent and its surroundings. An environment responds to an agent's specific set of activities by rewarding it and changing its state. Only the prior state and action have any bearing on the subsequent state and reward. The mathematical formulation of the RL framework is shown in Figure 4 provided by MDP. The MDP environment comprises Markov states, which follow the Markov Property: the state contains all the data necessary for predicting the future from the past. The property of MDP is formally stated as follows (see Figure 1):

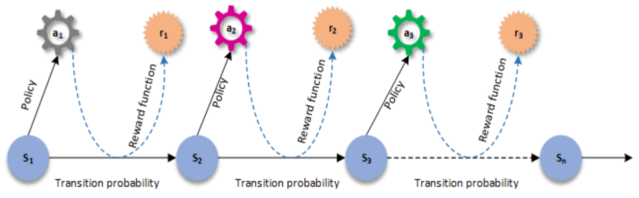

FIGURE 4. Representation of the MDP architecture.

Agent: A program (algorithm) that decides what to display next in a collection of e-learning.

Environment: The learning framework.

Action: Recommending a new class tutorial or reading notes, doing an assignment, taking a quiz, exams, or an advertisement, etc.

State: A learner's interaction features are depicted as a state. The state-value v (S) can be defined to evaluate the goodness of the current state (current position of the learner).

Reward: Positive if the learner decides to watch the class video; the reward is more positive if the learner chooses to take exams; if the learner exits, plays a game, or becomes bored, the result can be negative.

Transition: is the process of moving from one state to the next.

Transition Probability: the likelihood that the agent will switch from one state to another. Mathematically it can be specified as follows (Equation. 2).

$$P\left[S_{t+1}|S_t\right] = P\left[S_{t+1}|S_1, \ldots.., S_t\right] \tag{2}$$

The agent's present state is indicated by S_t, while the subsequent state is shown by S_{t+1}. According to this formula, the change from state S_t to S_{t+1} is completely independent of the previous state. P shows the notation for transition probability. Therefore, if the model has a Markov Property, the right-hand side of the equation signifies the same as the left-hand side. It follows logically that the current state has knowledge about previous ones.

State Transition Probability: The state transition probability for a Markov State from S_t to S_{t+1}, or any additional beneficiary state, is mentioned in Equation (3).

$$\mathcal{P}_{ss'} = P\left[S_{t+1} = s' | S_t = s\right] \tag{3}$$

The state transition probabilities can be represented in a state transition probability matrix as given below:

$$P = \begin{pmatrix} p11 & p12 & p13 & . & . & . & p1n \\ p21 & p22 & p23 & . & . & . & p2n \\ p31 & p32 & p33 & . & . & . & p3n \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ pn1 & pn2 & pn3 & . & . & . & pnn \end{pmatrix}$$

It can be noted that $p_{ij} \geq 0$, and for all i, as stated in Equation (4).

$$\sum_{k=1}^{r} p_{ik} = \sum_{k=1}^{r} P\left(S_{m+1} = k | S_m = i\right)$$
$$\sum_{k=1}^{r} p_{ik} = 1 \tag{4}$$

Each row in the matrix represents the probability of transitioning from the initial state i to any subsequent one k. $\sum$ is the sum of each row in the transition matrix. The aggregate of each row is equal to 1. Algorithm 1 illustrates the proposed pseudo-code of MDP for calculating transition probability.

C. Q - LEARNING

Q-learning benefits from its previous behaviour, as well as those it does in the future to draw lessons from the past and choose the optimal course of action [73], [74].

In the aforementioned hypothetical context, all the transitions and their accompanying actions from one state to another are possible. In the simulations, the learner can allot a positive reward or a negative reward (penalty/punishment) to each action by employing the Q-table matrix also known as the brain of the Q-table. Using the Q-table or R-matrix, where each component shows the reward of a transition from one state to another (action). Moreover, the rows of the Q-table are represented as states (in our case learners' features) and the columns of the Q-table are represented as actions. These actions consider measurable positive effects, for instance, an increase in watching tutorials, doing assignments, or taking exams as well as an increase in reading or writing and so forth. In these situations, the actions get a positive reward that is proportional to the positive reward

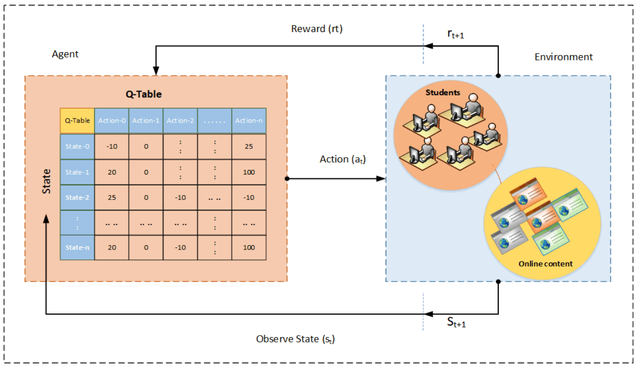
Reward (rt)
r_{t+1}
Agent
Environment
Q-Table
Q-Table | Action-0 | Action-1 | Action-2 | ...... | Action-n
State-0 | -10 | 0 | : | : | 25
State-1 | 20 | 0 | : | : | 100
State-2 | 25 | 0 | -10 | .. .. | -10
: | .. .. | .. .. | .. .. | : | .. ..
State-n | 20 | 0 | -10 | : | 100
State
Action (a_t)
Students
Online content
S_{t+1}
Observe State (s_t)

FIGURE 5. Representation of Q-learning structure for the proposed adaptive sequential learning path recommendations.

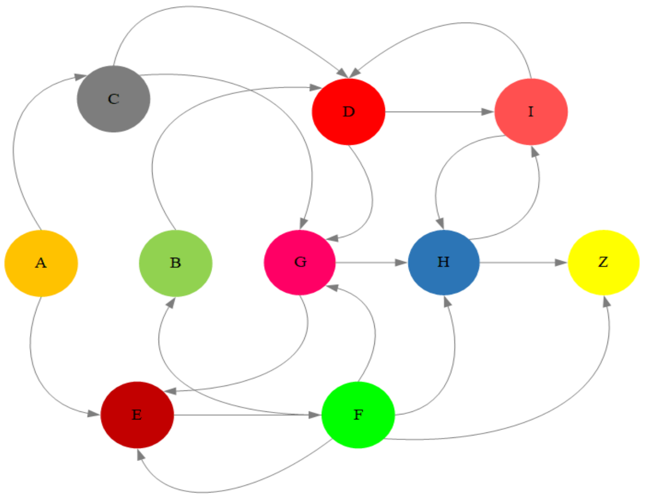

FIGURE 6. Example of a state diagram of student behavior (the sequence of the changing state in the form of student preferences).

perceptible consequences that have been generated. In other way, transitions that generate negative consequences (such as being idle, clicking on an ad, or switching to social media applications during studying time) get negative rewards, as in the case of a decrease in study time. Thus, if an action generates positive results, but an increase in the use of social media and playing games or entertainment at the same time then the reward is still positive, but it can be reduced by the collateral negative effects. The elements used in Q-learning are determined by Equation (5).

$$\begin{aligned} \mathcal{Q}\left(s_t, a_t\right) = \mathcal{Q}\left(s_t, a_t\right) + \alpha \\ \times \left[R_t + \gamma max \mathcal{Q}'\left(s_{t+1}, a_{t+1}\right) - \mathcal{Q}\left(s_t, a_t\right)\right] \end{aligned} \tag{5}$$

where α denotes learning rate $(0 \leq \alpha \leq 1)$; $R\left(s_t, a_t\right) +$ denotes the observed reward, s_{t+1} denotes the new state, $\gamma < 1$ denotes a discounted factor for the future rewards attained as a result of the action chosen. The highest reward that the system can quantify by performing some future action in the state s_{t+1} is estimated as $\mathcal{Q}\left(s_{t+1}, a_{t+1}\right)$.

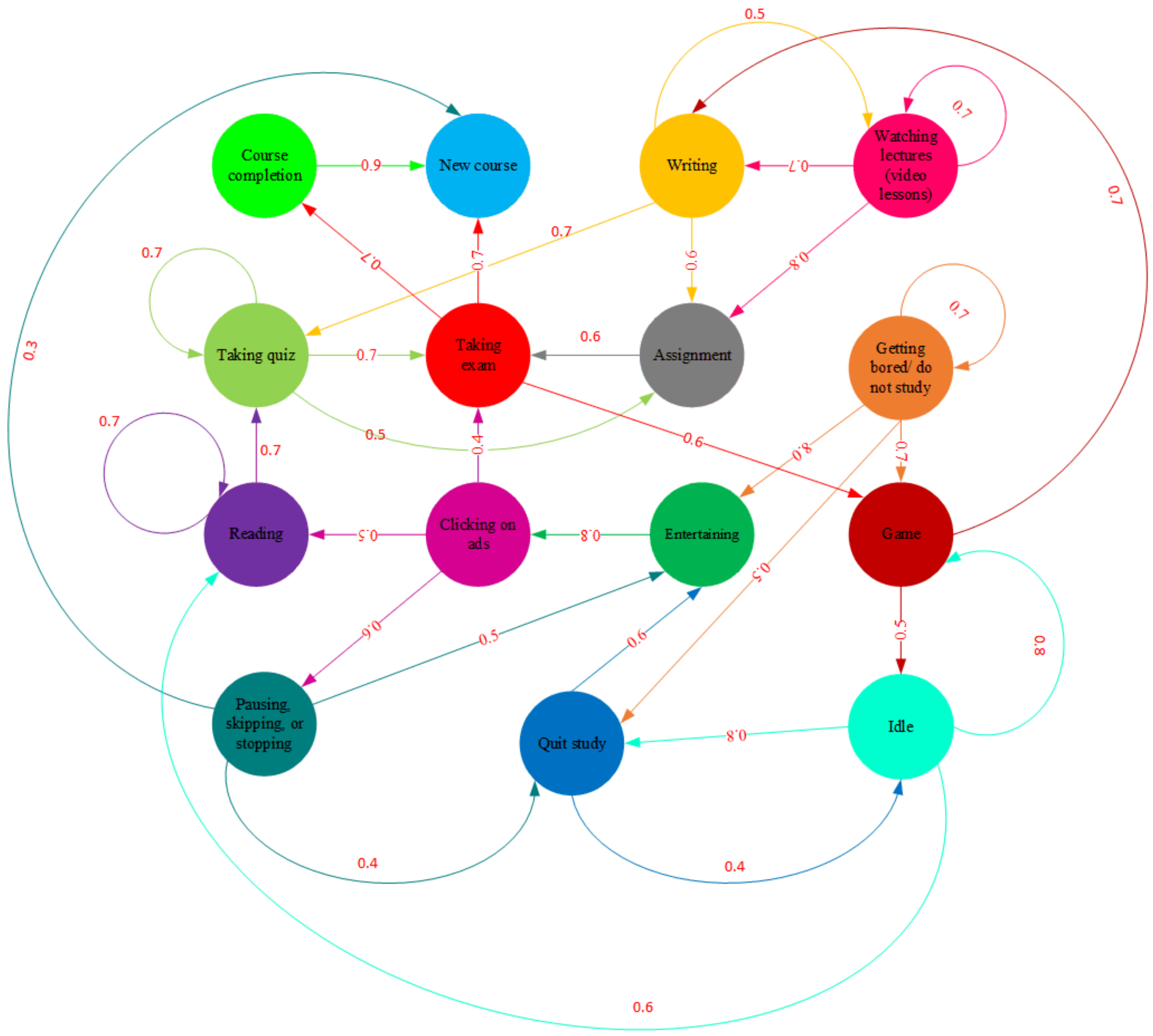

FIGURE 7. State diagram of MDP for personalized adaptive learning and SPR.

The suggested method can exhibit better efficiency and can enhance the learning condition for SPR inside the learning framework. Figure 5 depicts the framework of Q-learning while algorithm 2 illustrates the proposed pseudo-code.

Example: Let us suppose a moment when a student/learner logs into a system and loads a page. Links on a page can be seen as actions of states, and the pages themselves can be seen as the system's states. The system's positions/sets are the changing value between states in the Q-matrix as indicated in Table 2, and the state diagram (course-to-course in our case) is illustrated in Figure 6. Figure 6 depicts the sequence of the changing state in the form of student preferences as: A→C, A→E, B→D, C→D, C→G, D→G, D→I, E→F, F→B, F→E, F→G, F→H, F→Z, G→E, G→H, H→I, H→Z, I→D, and I→H. Table 2 reports a Q-matrix of a student who has changing state as in Figure 6. The student logs into a website and selects course A. Then the values of the sequence of changing state e.g., A→C, A→E, B→D, C→D, C→G, D→G, D→I, E→F, F→B, F→E, F→G,

TABLE 2. Q-matrix of student: Highlight cell indicates the chosen/selected course 'G'.

State	Action										
	A	B	C	D	E	F	G	H	I	..	Z
A			1		1				1	..	
B				1			1			..	
C		1			1		1+1+5	1		..	1
D					1			1		..	
:	:	:	:	:	:	:	:	:	:	..	:
Z	1			1			1			..	1

TABLE 3. An example of a Markov table with the current state, next state, and transition probability for the design of personalized adaptive SPR.

Current State	Next State	Transition Probability
Course completion	Recommend new course	0.8
Taking exam	Course completion	0.7
Reading	Quiz	0.7
Writing	Quiz	0.7
Watching lectures (video lessons)	Assignment	0.8
Taking quiz	Assignment	0.5
Assignment	Exam	0.6
Taking exam	Game	0.3
Writing	Assignment	0.6
Writing	Watching lectures (video lessons)	0.5
Watching lectures (video lessons)	Writing	0.7
Taking exam	New course	0.7
Getting bored/ do not study	Entertaining	0.8
Taking quiz	Exam	0.7
Idle	Quit study	0.8
Clicking on ads	Exam	0.4
Entertaining	Ads	0.8
Game	New course	0.5
Pausing, skipping, or stopping (video lesson, quiz, assignment, writing)	New course	0.3
Quit study/disengagement	Entertaining	0.6
Getting bored/ frustration	Game	0.7
Getting bored/ frustration	Quit study/disengagement	0.5
Pausing, skipping, or stopping (video lesson, quiz, assignment, writing)	Entertaining	0.5
Quit study	Idle	0.4
Pausing, skipping, or stopping (video lesson, quiz, assignment, writing)	Quit study/disengagement	0.5
Game	Idle	0.5
Idle	Game	0.8
Idle	Reading	0.6

F→H, F→Z, G→E, G→H, H→I, H→Z, I→D, I→H and select course G. The model will update the Q-matrix by plus 1 when a student clicks on each designed course and by plus 5 when they select the course's materials (see table2).

D. MARKOV DECISION PROCESS (MDP) FOR SEQUENTIAL LEARNING PATH AND SPR

The issue we are facing is as follows: we have a framework of 11 states, one of which is an impediment initial state (state 5), and two of which are end states (states 10, 11). We want to determine the optimal policy to apply for reward collection for each state as reported in Table 3. Table 3 lists an example of a Markov table with the current state, next state, and transition probability for the design of the proposed framework. The transition probability is determined using MDP mechanisms as illustrated in Algorithm 1.

It should be determined the optimum course of action for each state that the students are in, including whether they should continue to the previous or subsequent course, move on to tests or assignments, skip classes/stop studying, use a social media application, or stop completely. In other words, the model aims to reach state 10 (course completion) as soon as possible. Here and foremost, we must design a student class that will serve as the learning environment for the problem. The student class should be designed as follows.

The elements of MDP used in the proposed framework are defined as follows $(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{R}, \boldsymbol{P}, \boldsymbol{\gamma})$:

State $(\boldsymbol{s})$ $:< s_0, s_1, s_2, \ldots\ldots, s_{10} >$

Action $(\boldsymbol{A})$ $:< A_1, A_2, \ldots.., A_{10} >$

Rewad $(\boldsymbol{R})$ $:< 10, 20, 50, 60, 100, 100, 70, -10 >$

Probability $(\boldsymbol{P})$ $:< 0.1, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1 >$

Discount Factor $(\boldsymbol{\gamma})$: 1; determines how important current and future rewards are.

E. STATE DIAGRAM OF MDP FOR LEARNING PATH RECOMMENDATION

In this section, an intuition about MDP for personalized adaptive learning and SPR to pick up suitable course content and learning path is illustrated as follows.

Using the terminologies of MDP, we first define the MDP states and actions. Next, we initialize an empty transition probability matrix. Then, we manually assign the transition

Algorithm 2 Q-Learning Algorithm for Adaptive Sequential Learning Path Recommendations

```
initialize: $\mathcal{Q}$ -value function $\mathcal{Q}$(s, a), discount factor ($\gamma$), learning rate ($\alpha$), exploration function $\epsilon$
output: optimal sequential path recommendation in e-learning
initialize: $\mathcal{Q}$-table with 0
initialize: states $s_t$
while learning do
for $t \rightarrow 1$ to $T$ do
choose an action $a$ at max$\mathcal{Q}'(s_{t+1}, a_{t+1})$ with probability 1 - $\varepsilon$ or explore using
recommendsequential learning path or learning course content ($a_t$) as per estimated failed competencies
choose learning content ($a_t$) as per their desire: Adaptability selection
move into the subsequent state ($s_{t+1}$):
compute reward $R_t$
compute $\mathcal{Q}$-value using Bellman technique: $\mathcal{Q}(s_t, a_t) = \mathcal{Q}(s_t, a_t) + \alpha\left[R_t + \gamma max_a \mathcal{Q}'(s_{t+1}, a_{t+1}) - \mathcal{Q}(s_t, a_t)\right]$
if $\mathbf{a_u}$ gives a positive response to $a_c$ then
        set $s_{t+1} \leftarrow$
    else
        set $s_t \leftarrow s_{t+1}$
update $Q-$table : update(initial_state, action, $\gamma$)
end for
end while
```

probabilities for each state-action pair based on the problem's dynamics and domain knowledge (see Algorithm 1). Algorithm 1 demonstrates the assignment of transition probability and their computations using MDP mechanisms. Additionally, following the transition probability, Figure 7 exhibits the scenario of learning paths. The circles illustrate the states in which the learner can be and the values in the red are the transitions probability that the agent can take depending on the state the learners are in. For example, in the state 0, learners can choose whether they want to study (e.g., taking an exam, reading, writing, watching lectures, taking a quiz, doing assignments), click on ads, or get bored/not study, idle, or switch to social media applications (Facebook, Twitter, Instagram, LinkedIn, etc.) or playing games, or refreshment/entertaining by opening different entertainment applications, and depending on what actions the learner performs, a reward is assigned to each action. There is also an action node (end node) from where a learner can end up in different states depending on the transition probability; for instance, after deciding to go to writing from reading, or watching tutorials, the learner has a 0.4 probability of getting into an assignment or taking exams. This node shows the randomness of the environment over which learners have no control. In all other cases, the transition probability is 1 and if the discount factor is 1 then MDP can be defined as; since maximizing the total of rewards is the aim of solving RL challenges, now MDP is implemented to choose the best learning path. Formally, the best possible policy can be determined that will increase an agent's potential cumulative reward.

Now, to signify a random process, the edges of the tree represent the likelihood of a transition. Take a sample from this chain into consideration. Now it can be supposed that while a learner was watching a tutorial, there is a 0.7 percent chance that he/she would do an assignment, a 0.1 percent chance that he/she would watch longer, 0.1 percent chance that they would write something, 0.05 probability that they quit the study, and a further 0.05 probability that they would switch to social media. According to this, we can come up with other sequences from this chain to sample (Figure 7).

F. SETTING UP THE TRANSITION MATRIX OF THE STATE DIAGRAM FOR PERSONALIZED ADAPTIVE LEARNING AND SEQUENTIAL PATH RECOMMENDATIONS

Afterward, the reward mechanism is formed, which was set to a maximum of 100. If learning proceeds without interruption during all iterations, the maximum reward will be 100, as depicted in the matrix produced (Matrix 1). The starting component of the initialized probabilities is also this matrix. The matrix's columns represent actions, while its rows represent states. We can create the matrix of the ideal learning path and compute the required rewards by setting the feedback value at 0.75 and running 100, 200, and 500 simulations of iterations, (as illustrated in Figures 11-13).

Matrix 1: Reward table of assumed state-action combinations

$$P = \begin{array}{c} \\ S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \\ S_6 \\ S_7 \\ S_8 \\ S_9 \\ S_{10} \end{array} \left(\begin{array}{cccccccc} A_1 & A_2 & A_3 & A_4 & A_5 & A_6 & A_7 & A_8 \\ 10 & 0 & 50 & 0 & 0 & 0 & 70 & -10 \\ 10 & 0 & 50 & 60 & 100 & 0 & 0 & 0 \\ 10 & 0 & 50 & 0 & 0 & 0 & 0 & -10 \\ 0 & 20 & 0 & 60 & 100 & 0 & 0 & -10 \\ 0 & 20 & 50 & 60 & 100 & 100 & 70 & 0 \\ 0 & 20 & 50 & 60 & 0 & 0 & 0 & -10 \\ 10 & 0 & 50 & 0 & 0 & 0 & 0 & 0 \\ 10 & 0 & 50 & 60 & 100 & 100 & 70 & 0 \\ 0 & 20 & 0 & 60 & 0 & 0 & 0 & -10 \\ 0 & 20 & 0 & 60 & 0 & 0 & 70 & 0 \\ 10 & 20 & 50 & 60 & 100 & 100 & 70 & -10 \end{array}\right)$$

TABLE 4. Formulation of parameters settings.

Parameter	Assigned value
Learning rate: α	0.5
Discount factor: γ	0.8
States: s	11
Actions: a	8
Total episodes	100, 200, 500
Maximum iterations	100

G. PARAMETERS SETTINGS

Following the aforementioned strategy, the Q-learning algorithm is trained against 100, 200, and 500 iterations. The formulation of the parameters is listed in Table 4. The training of the Q-learning algorithm uses the Belman technique (discussed above). We will alter epsilon throughout training to strike a balance between exploration and exploitation. To progressively transition from pure exploration to exploitation, we will start with epsilon=1 (pure exploration) and let it decrease to 0.8 with each episode as we progress from pure exploration to exploitation.

IV. EMPIRICAL SIMULATIONS AND EVALUATIONS

The proposed method has been designed in a Python programming language and can be used to compare various simulating policies. The simulations that are produced can then be used to assess well-known policies and contrast them with potential alternatives.

From the simulations, we obtained an optimal actions workflow that starts from state 0 and goes through states 1, 5, and 6. We can understand the meaning of the output, just by looking at the definitions assigned above to these States' labels. Briefly, we moved from watching video lessons (State 2) towards course completion (State 9), going through states 1, 4, and 6. That path corresponds to the focus on study policy in e-learning at the beginning, followed by a very aggressive policy to contain the quitting study. It can be imagined that the proposed decisional strategy is changed for accelerating the solution of the problem. This is what can happen in some real cases: at the beginning, we hope that the learning difficulties evolve spontaneously towards a natural solution, as happens for many students learning every time. However, if this does not happen, then we apply optimal actions to reduce negative experiences during training and with boring situations and quit studying. To optimize personalization of the learning path that tries to improve the learner's performance by leveraging the RL technique (Q-learning) while decreasing the demand for actual interaction with the learning mechanism throughout training. This is what happened in online learning, for example for the average learning student. Of course, the output of the model depends on the rewards that are assigned to each possible action. If the values in the R Matrix are modified, the output can be very different. Consequently, the crucial point is how to define, properly, the correct rewards and punishments/penalties in the R-Matrix, to optimize the sequential decisional process. This is the first fundamental question to be solved when moving from simplistic simulations to real-world applications.

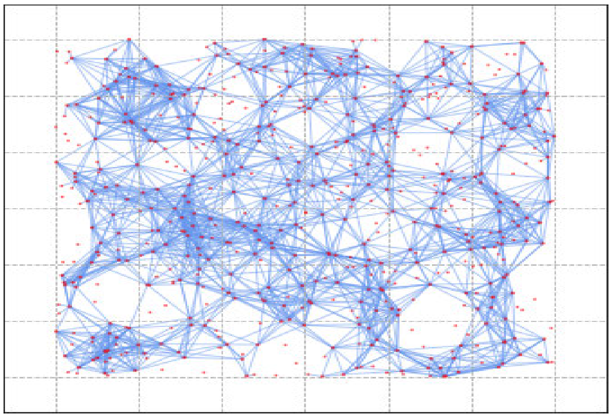
FIGURE 8. The students-course network – Number of 6,000 students and a default assumed daily learning rate of 600 content (the virtual environment comprises 6,000 students).

Figure 8 exhibits the students-course network with a virtual number of 6,000 assumed students and a default assumed daily learning rate of 600 content. This network is randomly generated with 600 pieces of content and 6,000 students, distributed randomly across those contents. Table 5 reports the generated rewards value over 100 iterations. The best reward in Q-learning is the maximum possible reward that can be obtained in an environment. In other words, it is the reward that the agent would receive if it always chose the optimal action in each state. The optimal action is the one that maximizes the Q-value for a given state. Over 100 iterations, Table 6 reveals the trained Q-table (matrix) – a value of the state-action pair. Additionally, the outcomes of the Q-Learning simulations are reported in Table 7 (Trained Q-table) and Table 8 (Generated rewards) and Table 9 (Trained Q-table), and Table 10 (Generated rewards) over 200 and 500 iterations, respectively. The best reward achieved is 85.743 and 100.00 for tables 7 and 9 over 200 and 500 iterations, respectively.

The Q-table is acquired as output after training. Nonetheless, it is challenging to determine whether it is optimum. As a result, we will need to conduct some further analysis. An excerpt from a post-training Q-table is exhibited in Tables 5-10.

The outcomes of implementing the algorithm are shown in tables 5-10. The output displays the possible decisions the learner could make in a given situation. A value of zero (0) indicates that this state cannot be affected by the taken action. Performing this activity now when the value is low is preferable to pursuing other actions. The best path for the active learner is 0-1-2-6-9. The algorithm shows us a student's preferred path of study for a given course. Moreover, the Q-learning algorithm also considers the student's learning style, preferences, and knowledge level when making

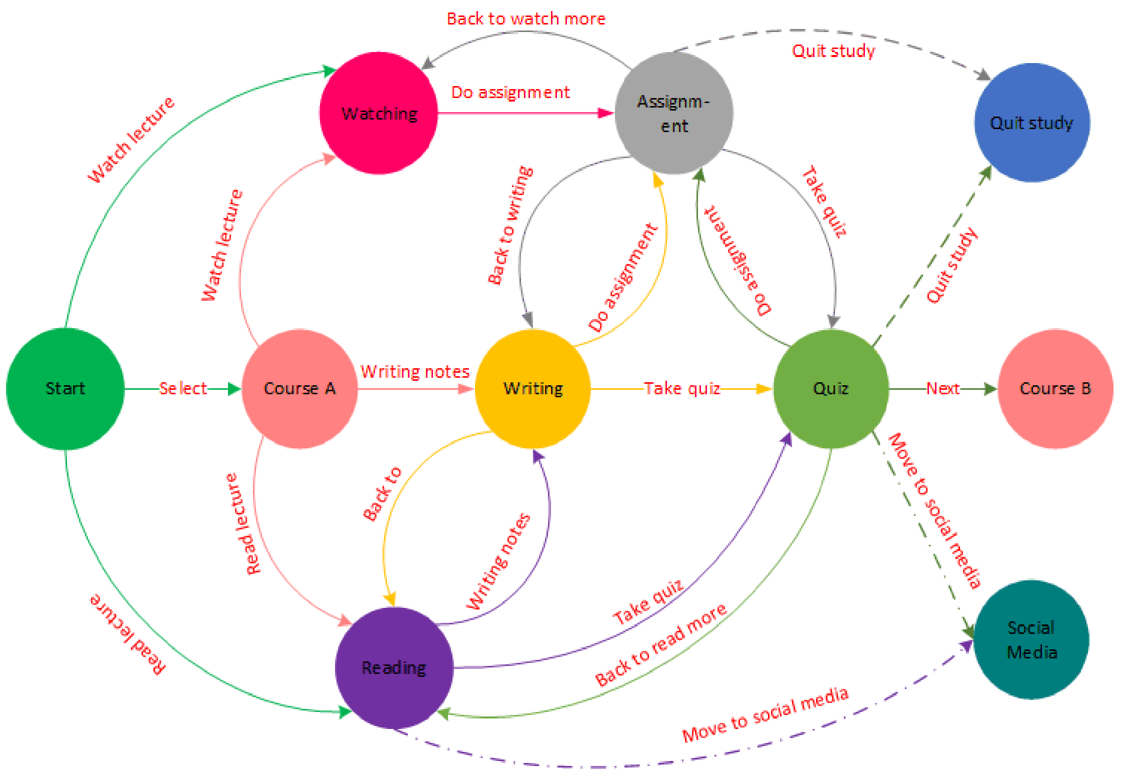

FIGURE 9. Example of personalized adaptive learning and suggesting sequential learning path to a learner/student.

recommendations. Figure 9 exhibits the scenario of learning paths. The circles illustrate the states in which the learner can be and the words in red are the actions that the agent can take depending on the state the learner is in. For instance, if a student prefers visual learning, the algorithm may recommend videos or interactive visual content. If the student is struggling with a particular concept, the algorithm may recommend additional resources or activities to reinforce that concept (see Figures 7 and 9).

The framework also uses an adaptive learning approach, which means that the learning path can be adjusted in real-time-based on the student's progress (Figure 9). For instance, if the student is struggling with a particular concept, the framework may recommend additional resources or activities to help them master that concept (see Figure 7).

In conclusion, an RL-based personalized adaptive learning and SPR framework can provide a highly personalized and effective learning experience for students. By tailoring the learning path to each student's needs and preferences, the framework can improve engagement, motivation, and knowledge retention, leading to better overall learning outcomes. Now let us execute 500 episodes to gauge the agent's effectiveness because it is challenging to assess the outcome by looking at the Q-table. Together with the distribution graph, we will use metrics such as mean, standard deviation, and min/max of the rewards.

The statistical performance outcomes of the Q-learning in executing and recommending the sequential learning path over 100, 200, and 500 times are reported in Tables 11, 12, and 13, respectively. To investigate the impact of the learning environment on Q-learning performance, simulation experiments were carried out over 100, 200, and 500 iterations, and the corresponding experimental results are displayed in tables 11, 12, and 13. According to Table 11, the mean reward is 4.50 ± 2.50. It suggests some fluctuation in rewards, but this is because of the positions of a learner's study, and a drop-off change at the beginning of each iteration. As a result, each time the recommendation is performed, it takes the model longer and does not always get the same reward (each move is minus 1 point). The Q-learning's performance metrics increased as the number of iterations increased, as illustrated in Tables 11-13. During this period, the Q-learning algorithm over 500 iterations outperformed the Q-learning with 100 and 200 iterations. The performances of Q-learning with 100 iterations were lower than those of Q-learning with 200 and 500 iterations. In terms of training time, the Q-learning with 100 iterations was always better than the other two characteristics. The other performance indices of the Q-learning with 500 were all better than those of the other two iterations, despite the training time not being as outstanding as that of the Q-learning with 100 and 200 iterations. The proposed method exhibited the lowest average number of turning times over 500 iterations. In terms of average success rate, Q-learning with 500 iterations outperformed Q-learning over 100 and 200 iterations by 11.66 and 7.7, respectively.

TABLE 5. Generated rewards value over 100 iterations.

State	Action							
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
S_0	0.	0.	0.	0.	68.845	0.	72.176	0
S_1	0.	52.461	0.	5.731	0.	59.979	68.594	0.
S_2	0.	53.356	55.939	0.	0.	16.836	77.549	69.55
S_3	0.	51.565	0.	0.	46.023	45.650	7.164	64.103
S_4	0.	0.	38.486	0.	67.054	47.284	70.385	0.
S_5	42.908	0.	0.	0.	47.814	49.233	52.586	65.969
S_6	0.	0.	72.059	53.073	37.827	0.	36.466	85.743
S_7	0.	7.164	52.815	53.073	56.308	077.890	86.505	80.223
S_8	0.	30.663	0.	0.	56.308	0.	0.	80.223
S_9	0.	0.	72.059	0.	56.308	61.731	52.586	66.041
S_{10}	0.	32.454	44.705	7.522	0.	0.	0.	87.114

TABLE 6. Trained Q-table (matrix) – value of state-action pair over 100 iterations.

State	Action							
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
S_0	434.238	0.	0.	0.	530.298	0.	188.000	0.
S_1	0.	276.52	0.	220.172	0.	314.4	536.238	0.
S_2	0.	0.	0.	209.216	117.52	0.	0.	281.52
S_3	0.	154.4	275.216	0.	371.52	0.	0.	261.52
S_4	428.990	26.000	0	0.	0.	575.372	0.	0.
S_5	26	134.4	0.	0.	50	415.372	469.216	261.52
S_6	0.	154.4	325.216	219.216	107.52	575.372	379.216	670.298
S_7	0.	468.990	325.216	42.	107.52	575.372	636.238	314.4
S_8	0.	174.4	235.216	0.	107.52	475.372	0.	314.4
S_9	0.	0.	325.216	42.	107.52	0.	268.	0.
S_{10}	0.	184.4	425.216	0.	107.52	425.372	268.	0.

TABLE 7. Generated rewards value over 200 iterations.

State	Action							
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
S_0	68.448	0.	0.	0.	84.145	0.	81.142	72.213
S_1	0.	70.339	0.	67.6239	0.	84.383	80.00	0.
S_2	0.	70.905	69.662	67.6239	77.350	76.457	85.662	79.615
S_3	0.	68.738	74.192	0.	80.747	74.887	84.529	77.350
S_4	68.448	67.507	70.794	0.	83.012	95.271	81.132	0.
S_5	68.448	67.509	0.	0.	81.880	77.589	91.324	77.350
S_6	0.	68.5303	69.772	68.756	76.217	95.271	81.132	100.
S_7	0.	71.003	79.854	68.756	76.217	95.272	89.708	87.542
S_8	0.	72.037	69.662	67.623	76.217	84.383	91.324	87.542
S_9	67.021	69.530	70.772	68.216	75.203	95.450	80.302	89.863
S_{10}	0.	71.869	79.709	68.682	76.370	95.462	91.252	87.622

The proposed method with 100, 200, and 500 iterations may miss the efficient learning content in terms of average step size and the number of times it is encountered, however, the Q-learning approach with 500 iterations outperformed the other two episodes.

The fact that a positive reward is always received and that the minimum reward is -10. As depicted in Figure 10, it can be quickly determined that it requires 7 steps (11 transitions plus 1 recommendation action) if the learner is in an opposing situation (i.e., switching to social media or quitting the study)

TABLE 8. Trained Q-table (matrix) – value of state-action pair over 200 iterations.

State	Action							
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
S_o	609.445	0.	0.	0.	749.306	0.	731.416	0.
S_1	0.	624.306	0.	601.133	0.	749.133	711.416	0.
S_2	0.	629.306	619.133	601.133	689.306	679.133	761.416	709.270
S_3	0.	619.270	659.133	0.	719.306	669.133	751.416	689.270
S_4	609.416	599.306	629.133	0.	739.306	849.133	721.416	0.
S_5	609.416	599.270	0.	0.	729.270	689.133	811.416	689.270
S_6	0.	619.306	709.133	611.133	679.306	849.087	721.416	889.270
S_7	0.	639.270	709.133	611.133	679.306	849.087	811.416	779.270
S_8	0.639.306	618.999	601.133	679.306	749.133	811.416	779.270	0.
S_9	609.320	599.122	0.	0.	699.245	679.103	801.321	689.342
S_{10}	0.	638.708	708.385	610.385	678.708	848.385	810.966	778.708

TABLE 9. Generated rewards value over 500 iterations.

State	Action							
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
S_o	68.534	0.	0.	0.	84.2585	0.	82.2483	0.
S_1	0.	70.203	0.	67.589	0.	84.234	80.000	0.
S_2	0.	70.765	69.633	67.589	77.511	76.369	85.622	79.769
S_3	0.	69.640	74.119	0.	80.885	75.244	84.498	77.511
S_4	0.	84.497	77.511	0.	80.884	75.244	84.	0.
S_5	68.5303	67.391	70.746	0.	83.1336	95.483	81.124	0.
S_6	68.5303	67.391	0.	0.	82.009	77.498	91.244	77.511
S_7	0.	69.640	79.741	68.722	76.387	95.483	100.00	81.124
S_8	0.	71.889	79.741	68.722	76.387	95.493	91.244	87.631
S_9	65.435	71.889	69.622	67.598	76.387	84.239	91.244	87.631
S_{10}	0.	639.344	709.180	611.180	679.344	849.180	811.475	779.344

TABLE 10. Trained Q-table (matrix) – value of state-action pair over 500 iterations.

State	Action							
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
S_o	609.475	0.	0.	0.	749.344	0.	624.344	0.
S_1	0.	624.344	0.	601.180	0.	749.10	711.475	0.
S_2	0.	629.344	619.180	601.180	689.344	679.180	761.475	709.344
S_3	0.	619.344	659.180	0.	719.344	669.180	751.475	689.344
S_4	607.475	599.344	629.180	0.	739.344	849.180	721.475	0.
S_5	609.475	599.344	0.	0.	729.344	689.180	811.475	689.344
S_6	0.	619.344	709.180	611.180	679.344	849.180	721.475	889.344
S_7	0.	639.344	709.180	611.180	679.344	849.243	811.475	779..344
S_8	0.	639.344	619.180	601.180	667.320	749.180	811.472	779.344
S_9	609.475	598.231	0.	0.	728.221	688.850	812.575	699.100
S_{10}	0.	639.344	709.180	611.180	678.321	846.231	811.180	799.248

being in the same state. The action for selecting the ideal path in the farthest case would thus be 2 (11 - 9).

To check why any greater rewards cannot be expected since the limit is 7. When a learner is started in the exam state, the episode terminates right away (i.e., not a valid scenario, so there is no hope for scoring 20). So, initializing the student and course in the same state is the best-case scenario that we can aim for. Exams and quizzes are the two closest state

TABLE 11. Performance of Q-learning (100 episodes).

Evaluation indexes	Achieved results
Training time (s)	5.125
Mean reward	4.50 +/- 2.50
Standard deviation	2.12101
Minimum reward	2.0
Maximum reward	7.0
Optimum action taken	3
Average success rate (%)	60.86
Average step range (action)	12.50
Number of turning times of the optimal path (course selection)	10
Best reward (cumulative reward)	9400.400471514848

TABLE 12. Performance of Q-learning (200 episodes).

Evaluation indexes	Achieved results
Training time (s)	5.211
Mean reward	5.00+/- 2.16
Standard deviation	2.16024
Minimum reward	2.0
Maximum reward	7.0
Optimum action taken	3
Average success rate (%)	70.82
Average step range (action)	13.20
Number of turning times of the optimal path (course selection)	8
Best reward (cumulative reward)	22704.01361296278

TABLE 13. Performance of Q-learning (500 episodes).

Evaluation indexes	Achieved results
Training time (s)	5.231
Mean reward	7.96 +/- 2.59
Standard deviation	3.23012
Minimum reward	2.0
Maximum reward	7.0
Optimum action taken	5
Average success rate (%)	78.52
Average step range (action)	15.00
Number of turning times of the optimal path (course selection)	6
Best reward (cumulative reward)	35831.24957762145

transitions; therefore, the agent would need to make 5 moves (1 pick-up and 4 moves) to transfer the learner from the exam state to the high-level course, giving the student the maximum reward for state 7 (11 - 4).

A nearly normal distribution of rewards is seen in the reward distribution graph (Figure 10). As such, it shows that the agent is acting logically, even though it does not demonstrate that we have an ideal policy. The reward for each episode is represented by Figures 11, 12, and 13. The proposed model's rapid convergence may be seen in these Figures and that is one of the principal benefits of the Q-learning method. With a smoothing threshold of 20, Figure 11 shows the smoothed reward for the Q-learning algorithm across the first 100 episodes. As can be observed in Figure 13, the reward converges after 100 episodes, indicating that the best learning strategy has been discovered after the Q-learning has been trained with 500 episodes.

Figures 14, 15, and 16 show the cumulative rewards for 100, 200, and 500 iterations, respectively. The cumulative reward is the total reward accumulated by the agent over a sequence of actions taken in an environment. The expected cumulative reward is the sum of all future rewards discounted by a factor gamma. The discount factor gamma is used to weigh future rewards less than immediate rewards, to account for the fact that the agent may receive delayed feedback. Figure 16 exhibits the highest cumulative reward (35831.24957762145) achieved over 500 iterations and Figure 15 exhibits the total cumulative (22704.01361296278)

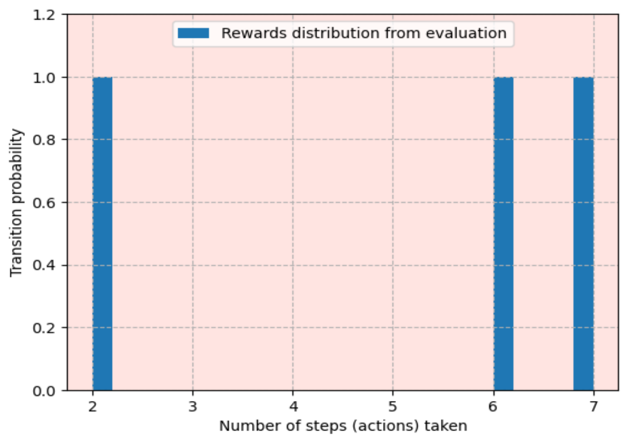

FIGURE 10. Path length – reward distribution from evaluation.

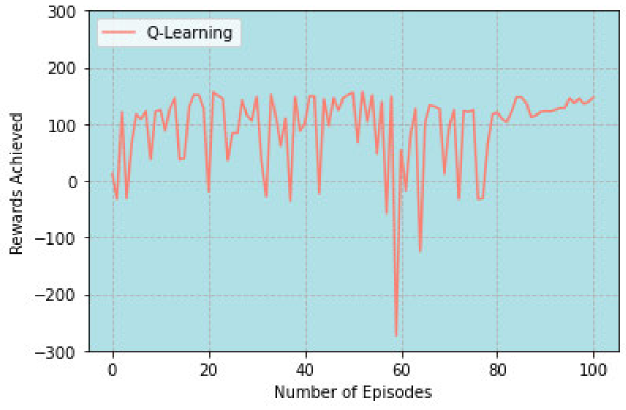

FIGURE 11. Rewards achieved for each episode (over 100 iterations).

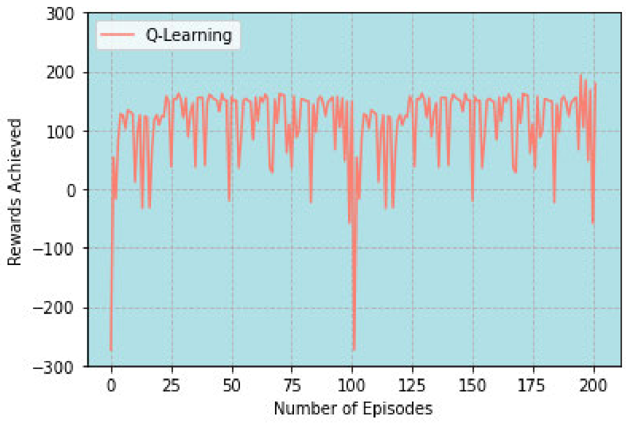

FIGURE 12. Rewards achieved for each episode (over 200 iterations).

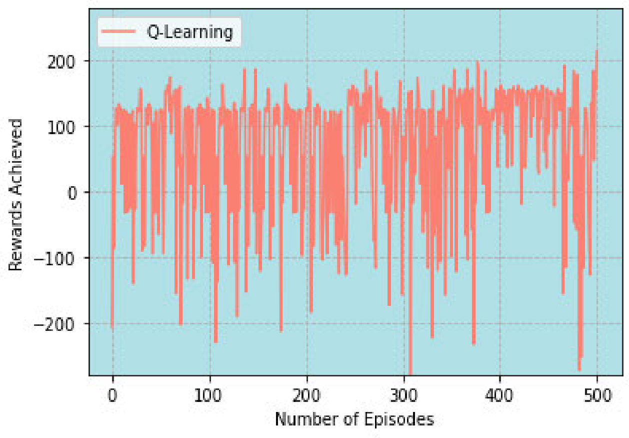

FIGURE 13. Rewards achieved for each episode (over 500 iterations).

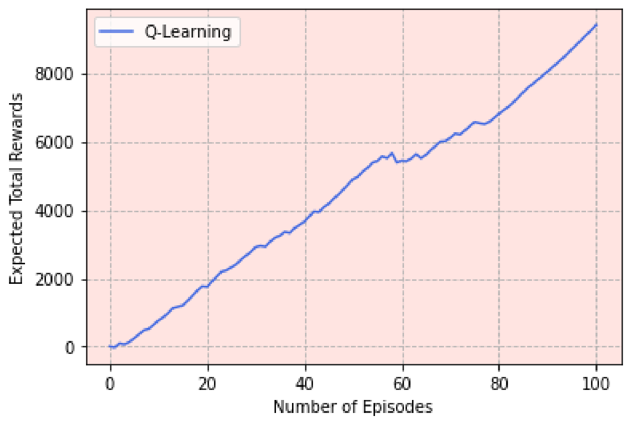

FIGURE 14. Expected total (9400.400471514848) rewards achieved over 100 iterations.

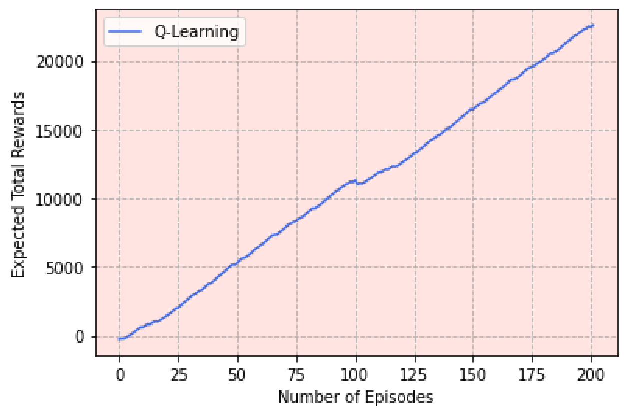

FIGURE 15. Expected total (22704.01361296278) rewards achieved over 200 iterations.

rewards achieved over 200 iterations and the total cumulative reward for 100 iterations is 9400.400471514848 and can be seen in Figure 14.

The optimal policy is the one that maximizes the expected cumulative reward over time. The Q-learning algorithm is one way to learn the optimal policy in a given environment. The Q-value function learned by Q-learning estimates the expected cumulative reward for each state-action pair as exhibited in Figures 14-16. The optimal policy can be derived from the Q-value function by choosing the action that maximizes the Q-value for each state.

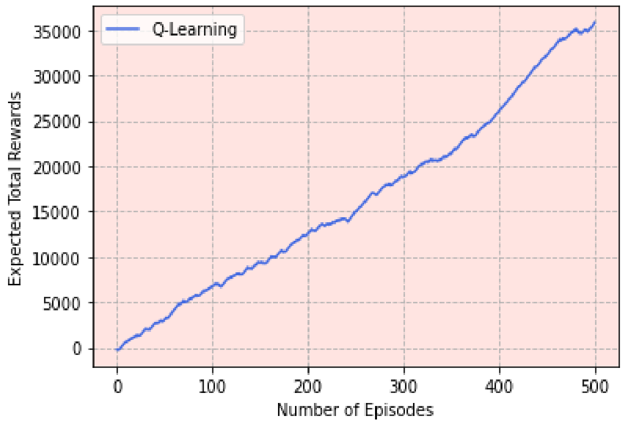

FIGURE 16. Expected total (35831.24957762145) rewards achieved over 500 iterations.

The state transition path exhibited in Tables 6, 8, and 10 illustrates how the hidden qualities change as a result of a sequence of decisions (actions) made using the Q-learning strategy produced without taking estimating error into account. As an illustration, consider the assignment and writing task. To start, the learner's watching skills are improved by continually choosing the initial learning resource. The third writing and assignment-related resource are then chosen. The second learning resource is picked in the final few rounds to further develop the learner's writing skills/abilities.

Furthermore, the learning curve of the rewards (Figures 11, 12, and 13) and expected cumulative rewards (Figures 14, 15, and 16) show how the agent's performance improves over time. These figures are based on metrics such as the average reward per episode or the success rate of achieving specific goals. As the learning curves show an upward trend at a high level and suggest that the agent is effectively learning and improving its decision-making.

The best reward is important because it provides an upper bound on the performance of the agent. If the agent is able to achieve the best reward, it means that it has learned the optimal policy for the environment. However, in many environments, it may not be possible for the agent to achieve the best reward due to the stochastic nature of the environment or the limitations of the agent's actions. Therefore, in practice, the goal of Q-learning is not to achieve the best reward but rather to learn a policy that maximizes the expected cumulative reward over time. Figures 14-16 exhibit the expected cumulative rewards achieved over 100, 200, and 500 iterations, correspondingly. This requires balancing the exploration of new actions with the exploitation of the current knowledge to obtain the maximum possible reward. It can be observed that the Q-learning algorithm over 500 iterations outperformed the Q-learning with 100 and 200 iterations. Also, the performances of Q-learning with 100 iterations were lower than those of Q-learning with 200 and 500 iterations.

TABLE 14. List of acronyms used in this paper.

Acronym	Full Form
MOOC	Massive Open Online Courses
RS	Recommender System
RL	Reinforcement Learning
MDP	Markov Decision Process
PRS	Personalized Recommender System
SPR	Sequential Path Recommendations

TABLE 15. List of notations used in this paper.

Notation	Description
P	Transition probability
S	M number of states
s	Current state
$s^{'}$	Next state
α	Action
A	N number of actions
γ	Discount factor
$Q(s_t, a_t)$	Current state-action
$Q'(s_{t+1}, a_{t+1})$	Next state-action
α	Learning rate
R	Rewards
Σ	Sum/aggregation
t	Time
ϵ	Exploration function

V. CONCLUSION AND FUTURE RESEARCH DIRECTION

RL can be used in developing adaptive SPR for academia by training an agent to make recommendations based on the actions and feedback of users. The agent can learn to optimize recommendations over time by adjusting its behavior based on the rewards or penalties it receives for each recommendation. In this paper, we proposed a framework that could be used to create personalized and adaptive recommendations for students based on their individual goals and preferences, as well as the overall context of their academic journey. By continuously learning and adapting, the agent can improve the quality and effectiveness of its recommendations, ultimately leading to better outcomes for students.

The proposed framework's primary functionality is to recommend learning paths based on sequential behavior, learning style/paths, learning activities, various learning materials, adaptive difficulty levels, personalized feedback, preferences, competency, and knowledge level simultaneously using the Q-learning algorithm. It can be observed that the Q-learning algorithm over 500 iterations outperformed the Q-learning with 100 and 200 iterations. The proposed method exhibited the lowest average number of turning times over 500 iterations. In terms of average success rate, Q-learning with 500 iterations outperformed Q-learning over 100 and 200 iterations by 11.66 and 7.7, respectively. The framework allows for the personalization of the learning experience and offers learning objects that are tailored to the demands and characteristics of the students. It is a distributed framework comprising autonomous agents that interact continually to address learners' requests. Furthermore, Any learning management system can be incorporated with the suggested solution. This system can easily interface and interact with other systems due to the functions it has implemented.

Future activities shall include numerous actions for each state and as numerous states as necessary to enable the exploration of the best course of action for each learner. The wide range of potential states or action values of the state, however, is the fundamental issue. Precisely if the approach is to be applied online by employing conventional RL, it would result in complexity and convergence problems.

For possible future research work, the following research gaps are suggested.

i. For problems with complexity, convergence, and model efficiency, employing deep Q-learning as an alternative would be a nice idea.
ii. Traditional RL approaches have many difficulties, including the risk of algorithmic inefficiency if the action space is too big because the algorithm evaluates all actions as a whole. To address this problem, Deep Deterministic Policy Gradient seems to be a good solution.
iii. For future research, we also aim to consider more states and actions to find the optimal learning paths on the learner's adaptive sequential behaviours, learning style/paths, learning activities, various learning materials, adaptive difficulty levels, optimal learning paths, personalized feedback, preferences, competency, knowledge level, etc. simultaneously when making recommendations by using multi-agent RL techniques.

ACKNOWLEDGMENT

The authors would like to thank the University of Jeddah for its technical and financial support.

REFERENCES

[1] Y.-C. Chen, L. Hui, and T. Thaipisutikul, "A collaborative filtering recommendation system with dynamic time decay," *J. Supercomput.*, vol. 77, no. 1, pp. 244–262, Jan. 2021, doi: 10.1007/s11227-020-03266-2.

[2] Y. Zhou, C. Huang, Q. Hu, J. Zhu, and Y. Tang, "Personalized learning full-path recommendation model based on LSTM neural networks," *Inf. Sci.*, vol. 444, pp. 135–152, May 2018, doi: 10.1016/j.ins.2018.02.053.

[3] Q. Shambour, "A deep learning based algorithm for multi-criteria recommender systems," *Knowl.-Based Syst.*, vol. 211, Jan. 2021, Art. no. 106545, doi: 10.1016/j.knosys.2020.106545.

[4] Q. Cui, S. Wu, Q. Liu, W. Zhong, and L. Wang, "MV-RNN: A multi-view recurrent neural network for sequential recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 2, pp. 317–331, Feb. 2020, doi: 10.1109/TKDE.2018.2881260.

[5] P. Symeonidis and D. Malakoudis, "Multi-modal matrix factorization with side information for recommending massive open online courses," *Expert Syst. Appl.*, vol. 118, pp. 261–271, Mar. 2019.

[6] Z. A. Pardos, S. Tang, D. Davis, and C. V. Le, "Enabling real-time adaptivity in MOOCs with a personalized next-step recommendation framework," in *Proc. 4th ACM Conf. Learn. Scale*, Apr. 2017, pp. 23–32, doi: 10.1145/3051457.3051471.

[7] P.-C. Chang, C.-H. Lin, and M.-H. Chen, "A hybrid course recommendation system by integrating collaborative filtering and artificial immune systems," *Algorithms*, vol. 9, no. 3, p. 47, Jul. 2016, doi: 10.3390/a9030047.

[8] J. Tang and K. Wang, "Personalized top-*N* sequential recommendation via convolutional sequence embedding," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, Feb. 2018, pp. 565–573.

[9] D. Wu, J. Lu, and G. Zhang, "A fuzzy tree matching-based personalized e-learning recommender system," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 6, pp. 2412–2426, Dec. 2015.

[10] Y. Pang, W. Liu, Y. Jin, and H. Peng, "Adaptive recommendation for MOOC with collaborative filtering and time series," *Comput. Appl. Eng. Educ.*, vol. 1, no. 13, pp. 1–13, 2018, doi: 10.1002/cae.21995.

[11] Z. Huang, X. Xu, H. Zhu, and M. Zhou, "An efficient group recommendation model with multiattention-based neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4461–4474, Nov. 2020, doi: 10.1109/TNNLS.2019.2955567.

[12] S. Deng, L. Huang, G. Xu, X. Wu, and Z. Wu, "On deep learning for trust-aware recommendations in social networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1164–1177, May 2017, doi: 10.1109/TNNLS.2016.2514368.

[13] Y. Lin, Y. Liu, F. Lin, L. Zou, P. Wu, W. Zeng, H. Chen, and C. Miao, "A survey on reinforcement learning for recommender systems," 2021, *arXiv:2109.10665*.

[14] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, Jan. 2020.

[15] Z. Y. Khan, Z. Niu, S. Sandiwarno, and R. Prince, "Deep learning techniques for rating prediction: A survey of the state-of-the-art," *Artif. Intell. Rev.*, vol. 54, no. 1, pp. 95–135, Jan. 2021, doi: 10.1007/s10462-020-09892-9.

[16] F. Bousbahi and H. Chorfi, "MOOC-Rec: A case based recommender system for MOOCs," *Proc. Social Behav. Sci.*, vol. 195, pp. 1813–1822, Jul. 2015, doi: 10.1016/j.sbspro.2015.06.395.

[17] X. Jing and J. Tang, "Guess you like: Course recommendation in MOOCs," in *Proc. Int. Conf. Web Intell.*, Aug. 2017, pp. 783–789.

[18] X. He, Z. He, J. Song, Z. Liu, Y.-G. Jiang, and T.-S. Chua, "NAIS: Neural attentive item similarity model for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 12, pp. 2354–2366, Dec. 2018, doi: 10.1109/TKDE.2018.2831682.

[19] X. Li, X. Li, J. Tang, T. Wang, Y. Zhang, and H. Chen, "Improving deep item-based collaborative filtering with Bayesian personalized ranking for MOOC course recommendation," in *Proc. Int. Conf. Knowl. Sci., Eng. Manage.*, Dec. 2020, pp. 247–258.

[20] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 1419–1428.

[21] J. Zhang, B. Hao, B. Chen, C. Li, H. Chen, and J. Sun, "Hierarchical reinforcement learning for course recommendation in MOOCs," in *Proc. AAAI Conf. Artif. Intell.*, Dec. 2019, vol. 33, no. 1, pp. 435–442.

[22] X. Xin, A. Karatzoglou, I. Arapakis, and J. M. Jose, "Self-supervised reinforcement learning for recommender systems," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 931–940.

[23] R. Logesh and V. Subramaniyaswamy, "Exploring hybrid recommender systems for personalized travel applications," in *Cognitive Informatics and Soft Computing*. Cham, Switzerland: Springer, 2019, pp. 535–544.

[24] S.-S. Chen, B. Choubey, and V. Singh, "A neural network based price sensitive recommender model to predict customer choices based on price effect," *J. Retailing Consum. Services*, vol. 61, Jul. 2021, Art. no. 102573, doi: 10.1016/j.jretconser.2021.102573.

[25] D. H. Lee and P. Brusilovsky, "Improving personalized recommendations using community membership information," *Inf. Process. Manage.*, vol. 53, no. 5, pp. 1201–1214, Sep. 2017, doi: 10.1016/j.ipm.2017.05.005.

[26] W. Zhang, Y. Du, T. Yoshida, and Y. Yang, "DeepRec: A deep neural network approach to recommendation with item embedding and weighted loss function," *Inf. Sci.*, vol. 470, pp. 121–140, Jan. 2019, doi: 10.1016/j.ins.2018.08.039.

[27] M. Rhanoui, M. Mikram, S. Yousfi, A. Kasmi, and N. Zoubeidi, "A hybrid recommender system for patron driven library acquisition and weeding," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 34, no. 6, pp. 2809–2819, Jun. 2022, doi: 10.1016/j.jksuci.2020.10.017.

[28] C. Chen, Z. Liao, Y. Ju, C. He, K. Yu, and S. Wan, "Hierarchical domain-based multicontroller deployment strategy in SDN-enabled space–air–ground integrated network," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 58, no. 6, pp. 4864–4879, Dec. 2022.

[29] L. Ravi, V. Subramaniyaswamy, V. Vijayakumar, R. H. Jhaveri, and J. Shah, "Hybrid user clustering-based travel planning system for personalized point of interest recommendation," in *Mathematical Modeling, Computational Intelligence Techniques and Renewable Energy*. Cham, Switzerland: Springer, 2021, pp. 311–321.

[30] L. Ji, Q. Qin, B. Han, and H. Yang, "Reinforcement learning to optimize lifetime value in cold-start recommendation," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 782–791, doi: 10.1145/3459637.3482292.

[31] L. Zou, L. Xia, Y. Gu, X. Zhao, W. Liu, J. X. Huang, and D. Yin, "Neural interactive collaborative filtering," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 749–758.

[32] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, "Variational autoencoders for collaborative filtering," in *Proc. World Wide Web Conf. (WWW)*, 2018, pp. 689–698.

[33] N. Sachdeva, G. Manco, E. Ritacco, and V. Pudi, "Sequential variational autoencoders for collaborative filtering," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, Jan. 2019, pp. 600–608.

[34] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders meet collaborative filtering," in *Proc. 24th Int. Conf. World Wide Web*, May 2015, pp. 111–112.

[35] G.-E. Yap, X.-L. Li, and P. S. Yu, "Effective next-items recommendation via personalized sequential pattern mining," in *Proc. Int. Conf. Database Syst. Adv. Appl.*, Dec. 2012, pp. 48–64.

[36] O. Moling, L. Baltrunas, and F. Ricci, "Optimal radio channel recommendations with explicit and implicit feedback," in *Proc. 6th ACM Conf. Recommender Syst.*, Sep. 2012, pp. 75–82.

[37] P. Wang, Y. Fan, L. Xia, W. X. Zhao, S. Niu, and J. Huang, "KERL: A knowledge-guided reinforcement learning model for sequential recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 209–218.

[38] J. K. Tarus, Z. Niu, and A. Yousif, "A hybrid knowledge-based recommender system for e-learning based on ontology and sequential pattern mining," *Future Gener. Comput. Syst.*, vol. 72, pp. 37–48, Jul. 2017.

[39] J. K. Tarus, Z. Niu, and D. Kalui, "A hybrid recommender system for e-learning based on context awareness and sequential pattern mining," *Soft Comput.*, vol. 22, no. 8, pp. 2449–2461, Apr. 2018.

[40] Y. Xian, Z. Fu, S. Muthukrishnan, G. de Melo, and Y. Zhang, "Reinforcement knowledge graph reasoning for explainable recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2019, pp. 285–294.

[41] X. Wang, Y. Wang, D. Hsu, and Y. Wang, "Exploration in interactive personalized music recommendation: A reinforcement learning approach," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 11, no. 1, pp. 1–22, Aug. 2014.

[42] L. Yang, Y. Li, S. X. Yang, Y. Lu, T. Guo, and K. Yu, "Generative adversarial learning for intelligent trust management in 6G wireless networks," *IEEE Netw.*, vol. 36, no. 4, pp. 134–140, Jul. 2022.

[43] S. Almahdi and S. Y. Yang, "An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown," *Expert Syst. Appl.*, vol. 87, pp. 267–279, Nov. 2017.

[44] P. Gabrielsson and U. Johansson, "High-frequency equity index futures trading using recurrent reinforcement learning with candlesticks," in *Proc. IEEE Symp. Ser. Comput. Intell.*, Dec. 2015, pp. 734–741.

[45] X. Chen, L. Yao, J. McAuley, G. Zhou, and X. Wang, "Deep reinforcement learning in recommender systems: A survey and new perspectives," *Knowl.-Based Syst.*, vol. 264, Mar. 2023, Art. no. 110335, doi: 10.1016/j.knosys.2023.110335.

[46] S. Amin, A. Alharbi, M. I. Uddin, and H. Alyami, "Adapting recurrent neural networks for classifying public discourse on COVID-19 symptoms in Twitter content," *Soft Comput.*, vol. 26, no. 20, pp. 11077–11089, Oct. 2022, doi: 10.1007/s00500-022-07405-0.

[47] W. Song, Z. Duan, Z. Yang, H. Zhu, M. Zhang, and J. Tang, "Explainable knowledge graph-based recommendation via deep reinforcement learning," 2019, *arXiv:1906.09506*.

[48] X. Zhao, L. Xia, L. Zhang, Z. Ding, D. Yin, and J. Tang, "Deep reinforcement learning for page-wise recommendations," in *Proc. 12th ACM Conf. Recommender Syst.*, Sep. 2018, pp. 95–103.

[49] R. Han, K. Chen, and C. Tan, "Curiosity-driven recommendation strategy for adaptive learning via deep reinforcement learning," *Brit. J. Math. Stat. Psychol.*, vol. 73, no. 3, pp. 522–540, Nov. 2020.

[50] S. Ji, Z. Wang, T. Li, and Y. Zheng, "Spatio-temporal feature fusion for dynamic taxi route recommendation via deep reinforcement learning," *Knowl.-Based Syst.*, vol. 205, Oct. 2020, Art. no. 106302.

[51] Q. Liu, S. Tong, C. Liu, H. Zhao, E. Chen, H. Ma, and S. Wang, "Exploiting cognitive structure for adaptive learning," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 627–635.

[52] P. Wei, S. Xia, R. Chen, J. Qian, C. Li, and X. Jiang, "A deep-reinforcement-learning-based recommender system for occupant-driven energy optimization in commercial buildings," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6402–6413, Jul. 2020.

[53] Y. Sun, F. Zhuang, H. Zhu, Q. He, and H. Xiong, "Cost-effective and interpretable job skill recommendation with deep reinforcement learning," in *Proc. Web Conf.*, Apr. 2021, pp. 3827–3838.

[54] Z. Fu, L. Yu, and X. Niu, "TRACE: Travel reinforcement recommendation based on location-aware context extraction," *ACM Trans. Knowl. Discovery Data*, vol. 16, no. 4, pp. 1–22, Aug. 2022.

[55] L. Wang, W. Zhang, X. He, and H. Zha, "Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2447–2456.

[56] Z. Zheng, C. Wang, T. Xu, D. Shen, P. Qin, X. Zhao, B. Huai, X. Wu, and E. Chen, "Interaction-aware drug package recommendation via policy gradient," *ACM Trans. Inf. Syst.*, vol. 41, no. 1, pp. 1–32, Jan. 2023.

[57] M. Liu, X. Shen, and W. Pan, "Deep reinforcement learning for personalized treatment recommendation," *Stat. Med.*, vol. 1, no. 23, pp. 1–23, 2022.

[58] Z. Yuyan, S. Xiayao, and L. Yong, "A novel movie recommendation system based on deep reinforcement learning with prioritized experience replay," in *Proc. IEEE 19th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2019, pp. 1496–1500, doi: 10.1109/ICCT46805.2019.8947012.

[59] Z. Zhao, X. Chen, Z. Xu, and L. Cao, "Tag-aware recommender system based on deep reinforcement learning," *Math. Problems Eng.*, vol. 2021, pp. 1–12, May 2021, doi: 10.1155/2021/5564234.

[60] X. He, B. An, Y. Li, H. Chen, R. Wang, X. Wang, R. Yu, X. Li, and Z. Wang, "Learning to collaborate in multi-module recommendation via multi-agent reinforcement learning without communication," in *Proc. 14th ACM Conf. Recommender Syst.*, Sep. 2020, pp. 210–219.

[61] G. Ke, H.-L. Du, and Y.-C. Chen, "Cross-platform dynamic goods recommendation system based on reinforcement learning and social networks," *Appl. Soft Comput.*, vol. 104, Jun. 2021, Art. no. 107213.

[62] Y. Lin, S. Feng, F. Lin, W. Zeng, Y. Liu, and P. Wu, "Adaptive course recommendation in MOOCs," *Knowl.-Based Syst.*, vol. 224, Jul. 2021, Art. no. 107085.

[63] A. Elbadrawy and G. Karypis, "Domain-aware grade prediction and top-*N* course recommendation," in *Proc. 10th ACM Conf. Recommender Syst.*, Sep. 2016, pp. 183–190.

[64] Y. Zhu, H. Lu, P. Qiu, K. Shi, J. Chambua, and Z. Niu, "Heterogeneous teaching evaluation network based offline course recommendation with graph learning and tensor factorization," *Neurocomputing*, vol. 415, pp. 84–95, Nov. 2020.

[65] Y. Dai, Y. Asano, and M. Yoshikawa, "Course content analysis: An initiative step toward learning object recommendation systems for MOOC learners," in *Proc. Int. Educ. Data Min. Soc.*, 2016, pp. 103–105.

[66] Q. Zhang, J. Lu, and G. Zhang, "Recommender systems in e-learning," *J. Smart Environ. Green Comput.*, vol. 3, pp. 76–89, Feb. 2022, doi: 10.20517/jsegc.2020.06.

[67] O. Bourkoukou and E. E. Bachari, "Toward a hybrid recommender system for e-learning personnalization based on data mining techniques," *Int. J. Informat. Vis.*, vol. 2, no. 4, pp. 271–278, 2018.

[68] S. Wan and Z. Niu, "A hybrid e-learning recommendation approach based on learners' influence propagation," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 5, pp. 827–840, May 2020.

[69] W. Hoiles and M. Schaar, "Bounded off-policy evaluation with missing data for course recommendation and curriculum design," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1596–1604.

[70] M. E. Ibrahim, Y. Yang, D. L. Ndzi, G. Yang, and M. Al-Maliki, "Ontology-based personalized course recommendation framework," *IEEE Access*, vol. 7, pp. 5180–5199, 2019.

[71] J. Xu, T. Xing, and M. van der Schaar, "Personalized course sequence recommendations," *IEEE Trans. Signal Process.*, vol. 64, no. 20, pp. 5340–5352, Oct. 2016.

[72] S. Muruganandam and N. Srininvasan, ''Personalised e-learning system using learner profile ontology and sequential pattern mining-based recommendation,'' *Int. J. Bus. Intell. Data Min.*, vol. 12, no. 1, pp. 78–93, 2017.

[73] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[74] J. Zhang, Q. Yan, X. Zhu, and K. Yu, ''Smart industrial IoT empowered crowd sensing for safety monitoring in coal mine,'' *Digit. Commun. Netw.*, vol. 9, no. 2, pp. 296–305, Apr. 2023.

SAMINA AMIN received the M.S. degree in computer science from the Institute of Computing, Kohat University of Science and Technology (KUST), Kohat, Pakistan, in 2020, where she is currently pursuing the Ph.D. degree. She has published several articles in reputed journals. Her research interests include machine learning, deep learning, data science, data mining, natural language processing, social media analysis, neural network, recurrent neural networks, convolutional neural networks, image processing, health informatics, recommender systems, and reinforcement learning. She has received the Gold Medal from KUST. She serves as a reviewer for different journals.

M. IRFAN UDDIN (Member, IEEE) received the degree in computer science. He is currently a Faculty Member with the Institute of Computing, Kohat University of Science and Technology (KUST), Kohat, Pakistan. He has been involved in teaching and research activities related to different diverse topics of computer science and has more than 18 years of teaching plus research experience. He was a researcher in European Union-funded projects. He has published more than 100 research papers in reputed international journals and conferences. His research interests include machine learning, data science, artificial neural networks, deep learning, convolutional neural networks, recurrent neural networks, attention models, reinforcement learning, generative adversarial networks, computer vision, image processing, machine translation, natural language processing, speech recognition, big data analytics, parallel programming, multi-core, many-core, and GPUs. He is a member of ACM and HiPEAC. He has been actively involved in organizing national and international seminars, workshops, and conferences.

ALA ABDULSALAM ALAROOD received the bachelor's and master's degrees in computer science and the Ph.D. degree in computer science from the University of Technology Malaysia (UTM), in 2017. He is currently an Associate Professor with the Department of Data Science, College of Computer Science and Engineering, University of Jeddah. His research interests include data science, artificial intelligence, the Internet of Things (IoT), and cybersecurity.

WALI KHAN MASHWANI received the M.Sc. degree in mathematics from the University of Peshawar, Khyber Pakhtunkhwa, Pakistan, in 1996, and the Ph.D. degree in mathematics from the University of Essex, U.K., in 2012. He is currently a Professor in mathematics and the Director of the Institute of Numerical Sciences, Kohat University of Science & Technology (KUST), Khyber Pakhtunkhwa. He is also the Dean of Physical and Numerical Sciences with KUST. He has published more than 100 academic papers in peer-reviewed international journals and conference proceedings. His research interests include evolutionary computation, hybrid evolutionary multi-objective algorithms, and decomposition-based evolutionary methods for multi-objective optimization, mathematical programming, numerical analysis, and artificial neural networks.

ABDULRAHMAN ALZAHRANI is currently an Assistant Professor with the Department of Information System and Technology, College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia. He is also the Head of the Department of Computer Network. He teaches e-commerce and data visualization classes. He also teaches several courses to bachelor's and master's students. His research interests include information technology and innovations in the area of data science, machine learning, and health informatics.

AHMED OMAR ALZAHRANI received the first master's degree in computer science from California Lutheran University and the second master's and Ph.D. degrees in information systems and technology from Claremont Graduate University. He is currently an Assistant Professor with the Faculty of Computer Science and Engineering, University of Jeddah. His research interests include geographic information systems (GIS), energy informatics, remote senescing, and machine learning.