DKVMN-LB Based Personalized Learning Path Recommendation with Reinforcement Learning

Han Wan
State Key Laboratory of Virtual Reality Technology and Systems
School of Computer Science and Engineering
Beihang University
Beijing China
wanhan@buaa.edu.cn

Baoliang Che
State Key Laboratory of Virtual Reality Technology and Systems
School of Computer Science and Engineering
Beihang University
Beijing China
blche@buaa.edu.cn

Mengying Li
State Key Laboratory of Virtual Reality Technology and Systems
School of Computer Science and Engineering
Beihang University
Beijing China
mengyingli@buaa.edu.cn

ABSTRACT

Recently, researchers have developed advanced and flexible educational systems that use web-based platforms to automate adaptation based on student behavior, thus enabling personalized learning. Student interactions and knowledge states (KS) are crucial for facilitating the individualized experience in the context of such adaptive learning platforms. This study presents a highly efficient approach for recommending personalized learning paths to enhance students' academic progress. We improved the Dynamic Key-Value Memory Network (DKVMN) with a learning behavior module to productively trace students' knowledge states, resulting in the DKVMN-LB. The model was evaluated on various datasets, and the results demonstrated its effectiveness compared to the baseline models. To realize the recommendations, we employed the DKVMN-LB to simulate virtual students and trained a learning path recommendation algorithm through reinforcement learning (RL). Our RL-based recommendation approach outperformed collaborative filtering algorithms in simulation experiments on two public datasets, validating its feasibility. Extensive experiments on university course datasets investigated the usability of our recommendation method on real e-learning platforms. The proposed recommendation approach was also implemented in a real CS course to evaluate its usability. The findings from these experiments indicate that our personalized learning path recommendation method reduces recommendation time while increasing the students' average knowledge states, which is beneficial for adaptive learning.

CCS CONCEPTS

• Social and professional topics → Computer science education; Student assessment; • Applied computing → Education; Computer-assisted instruction.

KEYWORDS

Learning path recommendation, Knowledge tracing model, Reinforcement learning, Adaptive learning system

1 INTRODUCTION

Nowadays, education places great emphasis on students' initiative in the learning process. Adaptive learning system, as an extension of traditional classroom education, customize learning resources and provide personalized learning experiences through recommendation mechanisms [1,2]. However, most recommender systems are mainly applied to e-commerce and social media platforms, emphasizing the relationship and connection between users and items [3]. Such recommendation algorithms are not applicable enough to personalized learning platforms [4,5]. In personalized learning, the relationship between the student and each concept needs to be better demonstrated. Therefore, a more suitable learning path recommendation method for the learning platforms is required.

Learning path recommendations aim to help students master the required knowledge as much as possible [6], which requires assessing students' knowledge. Knowledge tracing determines the knowledge states of students based on their response sequence [7]. Existing knowledge tracing models primarily focus on analyzing the exercise sequence but often overlook the students' learning behaviors and lack interpretability. The DKVMN model utilizes a key matrix to store the relationship between questions and knowledge concepts, while a value matrix is employed to update the student's knowledge state [8]. Considering that the DKVMN does not account for student learning behaviors, which are essential for personalized learning, we could integrate a student learning behavior module into the DKVMN.

Influenced by collaborative filtering (CF), many online educational recommendation methods are limited to user-item similarity [9]. Researchers now have tried to apply reinforcement learning (RL) to education. RL is primarily employed to address problems involving sequential decision-making. It also involves scenarios with an environment and an agent, which makes it particularly suitable for learning path planning. Through a reinforcement learning algorithm, we could recommend appropriate learning paths to students based on their different knowledge states obtained from the knowledge tracing model.

The main goal of this study is to propose a learning path recommendation approach that combines enhanced knowledge tracing and reinforcement learning model, and to evaluate its usability on adaptive learning platforms. The rest of this paper is organized as follows: Section 2 conducts a literature review to explore existing research. Section 3 introduces the structure of DKVMN-LB and presents a novel personalized learning path recommendation approach, with the proposed DKVMN-LB used as a student simulator. The effectiveness of our approach is demonstrated in Section 4 through the experiments, followed by a discussion of the results. Section 5 summarizes our work and provides our future work.

2 RELATED WORK

2.1 Knowledgeable Tracing Models

Knowledge tracing is a general term for techniques that model students' knowledge states based on their response histories [7]. Existing knowledge tracing models frequently derive information from the past response records of students. The Bayesian Knowledge Tracing (BKT) model represents knowledge concepts and assumes that once students master a particular concept, they would retain it indefinitely [10]. However, on adaptive learning platforms, students may be required to remember concepts and such an assumption has obvious limitations. Researchers proposed the Deep Knowledge Tracing (DKT) model, which uses recurrent neural networks (RNN) to represent changes in students' knowledge states [11]. The RNN-based DKT can accurately predict the level of knowledge that students possess based on their recent learning performance. Additionally, it can effectively analyze and represent the complex relationship between challenging exercises and basic concepts. The DKT framework exhibits variable mastery of concepts among students, and there is a challenge in explaining student learning behaviors. The DKVMN incorporates the external memory module into the RNN network to enhance the interpretability of the DKT model. The read process of the DKVMN predicts the probability that students will answer the current question correctly. The write process of DKVMN updates the students' knowledge states. Based on this insight, we developed a DKVMN model integrating a module of student learning behavior.

2.2 Learning Path Recommendation

Learning path recommendation suggests personalized learning sequences based on students' needs, goals, and preferences [12]. RL mainly solves sequential decision problems that maximize long-term cumulative rewards, including environment and agent. Learning path planning also requires continuous decision-making to recommend resources based on the student's knowledge state to improve learning performance. Therefore, the learning path recommendation can be modeled as a reinforcement learning problem. Cai et al. [13] used knowledge tracing to model students' level as an RL environment. They deployed trust region policy optimization (TRPO) to learn exercise recommendation policy for assignment recommendations but lacked comparison experiments. Madani et al. [14] suggested a new RL approach that helps learners find the optimal learning path based on social filtering and collaborative filtering, which may have a cold start problem. Intayoad et al. [15] proposed a method based on contextual bandits and RL problems. It worked well in a dynamic online learning environment, but lacked analysis of learning behavior. Chen et al. [16] used the graph-based knowledge tracing and soft Actor-Critic algorithm to simulate the learning process with learners' goals, but the evaluation of the knowledge tracing network was neglected. Inspired by these studies, we modeled the learning path planning as an RL problem and employed two knowledge tracing models to simulate the environment and agent.

3 METHODOLOGY

3.1 The DKVMN-LB Model

DKVMN-LB is an improvement of DKVMN. We introduce a student learning behavior module, modify the original answer correctness module of DKVMN, and feed it with more features related to questions and students.

3.1.1 Input. The input for DKVMN-LB is derived from students and questions. Student ID, Level, and answer accuracy rate are the features of students. A student's Level refers to the rank of his number of correct answers among all students. We have divided it equally into five levels, represented by a one-hot code. The students' answer accuracy rate varies over time as they answer questions. A student's Wins and Fails represent the number of correctly and incorrectly answered questions when responding to the current query. Self-driven learners exhibit proactive engagement and invest considerable effort in tackling the question. Students' Wins and Fails indicate their participation and enthusiasm in learning and reflect their learning behaviors.

The features of questions include the question ID, type, and difficulty. Question types, such as multiple choice, cloze, etc., are represented using one-hot encoding. The difficulty of a question is represented by its correct rate. The correct rate of a question is based on the number of correct submissions and the total number of submissions, which vary over time.

3.1.2 Structure of DKVMN-LB. Figure 1 shows the structure of the DKVMN-LB. The gray blocks in the figure highlight the improvements made over the DKVMN. Along with the aforementioned input features, DKVMN-LB introduces a new learning behavior module and improves the original modules for modeling the questions and answer correctness.

The learning behavior module stores and updates student learning behavior throughout the learning process. Two matrices are utilized for modeling learning behaviors: key matrix S^K and value matrix S^V. S^K stores the embeddings of learning behavior patterns and is trained alongside the model training process. Once the training is completed, S^K remains unchanged. S^V records the students' performances in each learning behavior module. To model the features of the questions, we incorporate additional features based on the original structure of the DKVMN that effectively represent the question type and difficulty. The key matrix Q^K acts as an embedding that encapsulates the concepts learned by the model, and the value matrix Q^V stores students' knowledge states for each concept.

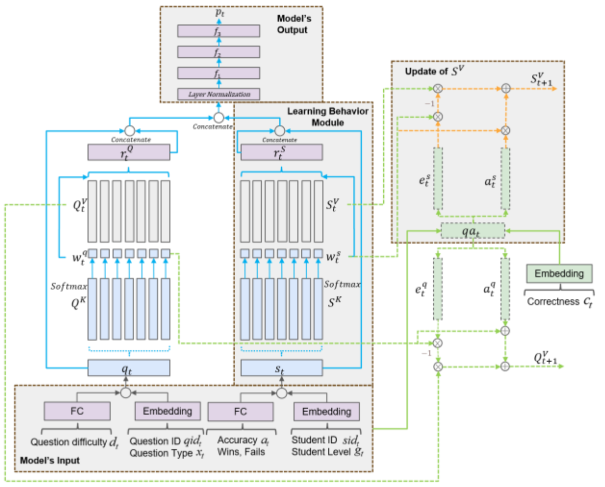


Figure 1: The structure of DKVMN-LB

In DKVMN, Q^V is updated based on the student's answer correctness. The DKVMN-LB enhances the correctness and value matrix portion of the original model, thereby facilitating the more effective modeling of student response patterns. First, question ID, question type, student ID, and Level are converted into one-hot encodings and multiplied by embedding matrices to get embedding vectors. After combining vectors from other features, an all-inclusive vector qa representing correctness, question features, and learning behaviors is created. Based on the qa, DKVMN-LB updates the value matrices, and then records the students' knowledge states.

After concatenating the results of the question modeling and the learning behavior modeling through layer normalization and fully connected layers, DKVMN-LB finally outputs the probability of students answering the current question correctly.

3.1.3 Read Process. The read process retrieves information from the memory modules based on the given key or query. DKVMN-LB obtains embedding s_t from students' features in each iteration. Discrete features, such as student ID sid_t and Level g_t, are encoded using one-hot encoding and multiplied by an embedding matrix to obtain embedding vectors. For student answer accuracy a_t, Wins and Fails, the fully connected layers with Tanh activation function are used to map them to vectors. E represents an embedding matrix, while W and b are the parameters for a fully connected layer. The student's embedding s_t is computed with sid_t, g_t, a_t, Wins and Fails. Similar to the processing of student-side features, the question embedding q_t is computed with qid_t, x_t and d_t as the same.

$$semb_t = E_1^{\ T} sid_t + E_2^{\ T} g_t \tag{1}$$

$$soh_t = Tanh(W_{soh}^T a_t + b_{soh}) \tag{2}$$

$$scon_t = Tanh(w_{scon}^T [wins,\ fails] + b_{scon}) \tag{3}$$

$$s_t = semb_t + soh_t + scon_t \tag{4}$$

The correlation weight w_t is calculated by taking the Softmax activation of the inner product between each input embedding and the key matrix. Sequentially, the read content r_t is extracted by the weighted sum of all memory slots in the value matrix using the correlation weight w_t.

$$w_t^S(i) = Softmax(s_t^T S^K(i)) \tag{5}$$

$$w_t^Q(i) = Softmax(q_t^T Q^K(i)) \tag{6}$$

At the output stage, the model concatenates the r_t^Q, q_t, r_t^S, s_t together and applies regularization through layer normalization. The DKVMN-LB now includes layer normalization. Layer normalization independently normalizes each vector without considering the sequence length. Thus, the enhancement improves regularization and handles variable-length sequences, reducing the training cost of the neural network.

The output of the layer normalization is passed through three fully connected layers with Tanh, Tanh, and Sigmoid activations, respectively. Finally, the predicted value p_t represents the probability that the student has mastered the question.

3.1.4 Write Process. The write process updates the memory modules with new information based on the given key and value. DKVMN-LB uses a structure that is more appropriate for processing answer correctness. It integrates question features, student features, and answer correctness to better model the student answer sequence. DKVMN-LB combines these different embeddings to further improve prediction accuracy, resulting in vector qa.

$$qa_t = s_t + q_t + E_3^{\ T} c_t \tag{7}$$

The write process updates the value matrices S^V and Q^V with erasure and append vectors to regenerate the student's state. Erasure vectors e_t^s, e_t^q and append vectors a_t^s, a_t^q are computed for the value matrices. Through these vectors, the write process updates memory slots in the end.

$$e_t^s = Sigmoid\left(W_{se}^T qa_t + b_{se}\right) \tag{8}$$

$$a_t^s = Tanh\left(W_{sa}^T qa_t + b_{sa}\right)^T \tag{9}$$

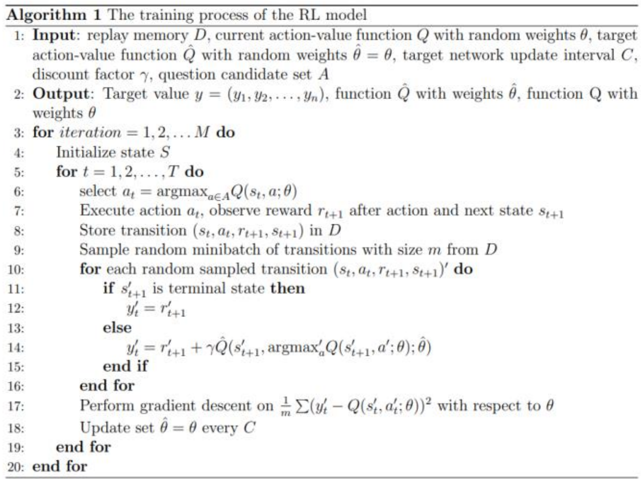
$$S_{t+1}^V(i) = S_t^V(i)\left[1 - w_t^s(i) e_t^s\right] + w_t(i) a_t^s \tag{10}$$

$$Q_{t+1}^V(i) = Q_t^V(i)\left[1 - w_t^q(i) e_t^q\right] + w_t(i) a_t^q \tag{11}$$

3.2 The RL-based Learning Path Recommendation

Based on the DKVMN-LB, we developed a student simulator to simulate student's learning behavior, providing an RL environment. The personalized learning path recommendation agent with a deep RL algorithm was then trained to recommend questions to students. Dueling Double DQN (D3QN) is chosen for reinforcement learning training because it combines the benefits of Double DQN [17] and Dueling DQN [18]. D3QN uses the current and target networks to enhance the learning process. They reduce action value overestimation and improve Q-value estimations.

This study proposes prioritized experience replay to enhance exploration by prioritizing and sampling experiences. The replay buffer stores the training experiences of D3QN. The student's knowledge state is defined as the model state, and a question recommendation is an action. The increase on the average knowledge mastery of the student is set as a reward. Thus, at time t, the agent observes the student's knowledge state S_t, and recommends a question A_t. After answering the question, the improvement of the student's average knowledge state is R_{t+1}, while the new state of the student is S_{t+1}.



Algorithm 1 The training process of the RL model

```
Input: replay memory $D$, current action-value function $Q$ with random weights $\theta$, target action-value function $\hat{Q}$ with random weights $\hat{\theta} = \theta$, target network update interval $C$, discount factor $\gamma$, question candidate set $A$
Output: Target value $y = (y_1, y_2, \dots, y_n)$, function $\hat{Q}$ with weights $\hat{\theta}$, function Q with weights $\theta$
for $iteration = 1, 2, \dots M$ do
    Initialize state $S$
    for $t = 1, 2, \dots, T$ do
        select $a_t = \mathrm{argmax}_{a \in A} Q(s_t, a; \theta)$
        Execute action $a_t$, observe reward $r_{t+1}$ after action and next state $s_{t+1}$
        Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in $D$
        Sample random minibatch of transitions with size $m$ from $D$
        for each random sampled transition $(s_t, a_t, r_{t+1}, s_{t+1})'$ do
            if $s'_{t+1}$ is terminal state then
                $y'_t = r'_{t+1}$
            else
                $y'_t = r'_{t+1} + \gamma \hat{Q}(s'_{t+1}, \mathrm{argmax}'_a Q(s'_{t+1}, a'; \theta); \hat{\theta})$
            end if
        end for
        Perform gradient descent on $\frac{1}{m} \Sigma (y'_t - Q(s'_t, a'_t; \theta))^2$ with respect to $\theta$
        Update set $\hat{\theta} = \theta$ every $C$
    end for
end for
```

Figure 2: The training process of the RL model

The current network uses the epsilon-greedy policy to choose an action in the current state during D3QN training. The model then performs the desired action in the environment and observes the next state and reward. The replay buffer stores the experience (state, action, reward, next state) for later sampling. The model addresses the problem of overestimation in action selection by utilizing both the current network and the target network to compute the target values:

$$y_t = r_{t+1} + \gamma \hat{Q}(s_{t+1}, argmax_a Q(s_{t+1}, a; \theta); \hat{\theta}) \tag{12}$$

Where y_t is the target value, γ is the discount factor, θ and $\hat{\theta}$ represents the current and target network parameters. To better learn the future rewards in reinforcement learning, the discount factor is usually 0.9 or 0.99. γ is set to be 0.9 in this study to consider the long-term reward fully. Algorithm 1 provides a detailed outline of the model's specific training process.

The RL-based learning path recommendation is revealed in Figure 3. We divide this method into the simulation and training. The knowledge tracing model acts as a student simulator in the simulation. At the beginning of the simulation, the value network rates all questions based on the virtual student's state. D3QN's current network is upgraded to a value network that handles student knowledge states. The top-rated question candidates are selected by the method. Question A_t is then recommended to the student. After receiving the question, the virtual student completes it, resulting in a change on his knowledge state from S_t to S_{t+1}, and a certain reward R_{t+1}. These data together form a transfer quadruple together: $(S_t, A_t, R_{t+1}, S_{t+1})$.

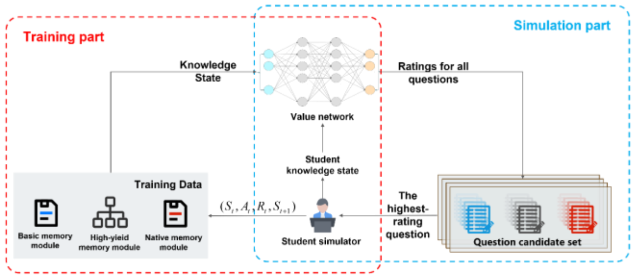


Figure 3: The process of learning path recommendation method based on RL

During training, the memory modules store quadruples continuously. After a round of simulation experiments, the training phase randomly selects quadruples from the three memory modules in a specific proportion to train the value network, as illustrated in Figure 3. This process repeats until the value network converges. The value network inputs the student's knowledge state and outputs the ratings for all available questions.

Transition quadruples $(S_t, A_t, R_{t+1}, S_{t+1})$. are observed and stored in memory module. Three memory modules are added to accelerate RL task convergence: basic memory module, high-yield memory module and native memory module. They are designed to preserve transitions and maximize student learning gains over time.

4 RESULTS AND DISCUSSION

This section presents the experimental results in detail and focuses the answers to our research questions. We also provide a detailed description of the datasets used, the evaluation procedures applied to the DKVMN-LB model, and the methodology underlying the simulation experiments in this section.

4.1 Datasets

To evaluate our knowledge tracing model and conduct recommendation experiments, we made use of two public datasets and university course datasets in this study. The *ASSISTment*

2009[1] (*ASSIST09*) dataset and *ASSISTment 2015*[2] (*ASSIST15*) are derived from the ASSISTments online learning platform. *CSCourseDatasets* were derived from a computer structure course offered by the computer science department of a university. The course is designed for sophomores and uses an online platform for exercises. The system supports multiple-choice, fill-in-the-blank, and programming assessments. The experiment utilized student response data from 2020–2022. A summary of dataset characteristics and statistics is presented in Table 1.

Table 1: Overview of the datasets

Datasets	Number of students	Number of questions	Number of records
ASSIST09	4151	16891	1325230
ASSIST15	19917	100	708632
CS 2020	334	182	17089
CS 2021	275	253	30439
CS 2022	241	249	26373

4.2 Evaluation of DKVMN-LB

To improve the quality of the recommendation simulation experiments, it is necessary to investigate the effectiveness of DKVMN-LB. The experiment was conducted on the above datasets to assess area under curve (AUC) of DKVMN-LB. For two datasets, 80% of the response sequences were kept as the training set, and the rest were kept as the testing set. We ran 30 experiments for each knowledge tracing model. For the purpose of comparison, the AUC values obtained from different experiments were averaged. According to the results presented in Table 2, DKVMN-LB has an advantage over other models, as evidenced by its higher average AUC across all datasets.

Table 2: AUC of knowledge tracing models

Datasets	Knowledge tracing models		
	DKT	DKVMN	DKVMN-LB
ASSIST09	80.53%	81.02%	**83.17%**
ASSIST15	72.38%	72.55%	**73.92%**
CS 2020	88.17%	88.93%	**89.81%**
CS 2021	82.89%	83.39%	**84.27%**
CS 2022	84.51%	85.42%	**87.09%**

4.3 The Process of Simulation Experiment

DKVMN-LB traces the student's knowledge state based on his past performance, which can be used to calculate the probability of his correct answer to a question. We used the knowledge state of a student to represent his average knowledge mastery degree in this study. Figure 4 illustrates the architecture diagram of the simulation experiment.

The DKVMN-LB model is designed to assess the knowledge state of the virtual student. Another separate DKVMN-LB model (the student simulator) generates the virtual student's responses and updates his knowledge state accordingly. Correspondingly, we divide the dataset into two parts to train the two different knowledge tracing models.

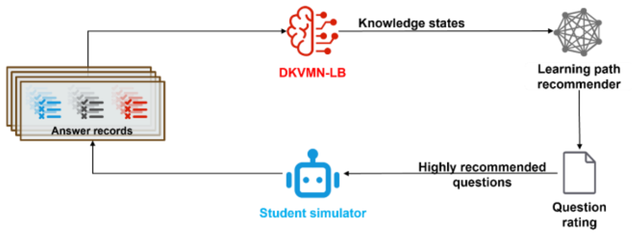


Figure 4: Schematic diagram of a simulation experiment

At first, DKVMN-LB processes the historical records of current virtual students to accurately compute their knowledge states. The learning path recommendation algorithm then suggests the highest rated questions to students. Subsequently, the student simulator generates the virtual student's answer based on the correct probability of his answer to the recommended question. This response is then inserted into the virtual student's historical response records, updating his knowledge state. Virtual students' knowledge states evolve as they answer more recommended questions by repeating this process.

4.4 Simulation Results

To better compare the effectiveness of the recommendation methods, we used two metrics to evaluate the usability of them.

Average Knowledge State Improvement: Students' knowledge states change after completing the recommended questions. The overall improvement in the average knowledge state of all students is considered the most critical evaluation metric.

Running time: Time is also an important metric for evaluating recommendation methods. To ensure the timeliness of recommendations, the average time it takes to recommend 500 questions to all students is considered a reference metric. The time for training a knowledge tracing model is not included.

To validate the effectiveness of our approach, we first performed identical validation experiments on two public datasets. Collaborative filtering based on user similarity (user-based CF) was used for comparison. User-based CF provides personalized recommendations by computing the similarity between students and exploiting the knowledge states of similar students. We chose the knowledge state at the time of students' complete answer records as the initial state. A total of 200 consecutive recommendation experiments were conducted, with a minimum of two questions recommended in each round. The final knowledge state of the students is recorded for comparison with the initial knowledge state. The results of the simulation experiments are presented in Table 3. The findings demonstrate the effectiveness of our learning path recommendation model in improving average knowledge state of students.

Before the implementation of the recommendation approaches, the knowledge states of the students in both groups were equivalent across all datasets. After recommendation, the mean values of knowledge states indicated that the RL recommendation

[1] https://sites.google.com/site/assistmentsdata/home/2009-2010-assistment-data/skill-builder-data-2009-2010.
[2] https://sites.google.com/site/assistmentsdata/datasets/2015-assistments-skill-builder-data.

algorithm outperformed the CF recommendation algorithm. The performance of both methods was comparatively better on ASSISTment 2009 than on ASSISTment 2015. The ASSISTment 2009 dataset has a greater variety of questions and a higher average number of student response records, which may help recommendation systems to anticipate student performance.

Table 3: Simulation results of different recommendation algorithms

Datasets	Original sequence	Collaborative filtering		Reinforcement learning	
	KS	KS	Running time(min)	KS	Running time(min)
ASSIST09	0.5933	0.6769	51.9	**0.7168**	**36.2**
ASSIST15	0.4089	0.4298	19.5	**0.4665**	**10.4**
CS 2020	0.5487	0.5651	15.1	**0.5776**	**8.5**
CS 2021	0.6053	0.6201	16.5	**0.6372**	**9.3**
CS 2022	0.4997	0.5180	15.9	**0.5294**	**8.6**

In terms of running time, the RL model takes less time to recommend questions than the CF model. User-based CF requires the maintenance of a user similarity matrix. As the number of users and response data increases, the complexity of computing the matrix also increases. The RL model leverages a trained neural network, which allows for the rapid calculation of ratings for all potential questions in the candidate set without incurring additional computational overhead. RL avoids the necessity of iterating through each question, so the cost time is shorter than that of CF.

For the CSCourseDatasets, we selected the answer records from students within one month after the course started to initialize their knowledge states. We then compared the recommendations using CF and RL with the student's original answer sequences. Since the questions are updated continuously during the courses, we performed daily recommendations for each student in the simulated recommendation process to observe the changes in their knowledge states. The maximum recommendation length is twenty.

Table 3 indicates that the RL recommendation method outperforms the CF recommendation method on CSCourseDatasets. Students who answered RL-recommended questions have the highest average knowledge states in these datasets. The average knowledge state progresses during the simulated recommendation process in Figure 5. Our method exhibits a more pronounced improvement in the average knowledge state as the number of answer records and recommendation iterations rises, in line with the objective of reinforcement learning algorithms to maximize long-term cumulative rewards.

The experimental results also show that RL recommends exercises to students much faster than CF. The RL model uses a trained neural network to quickly calculate ratings for all candidate questions without adding computational overhead. Thus, RL avoids iterating through each question. The CF model takes longer than RL because it requires traversing the question set and computing similarity metrics between each question. The results highlight the advantage of our approach, as our recommendation method can achieve better real-time recommendations and higher usability on adaptive learning systems.

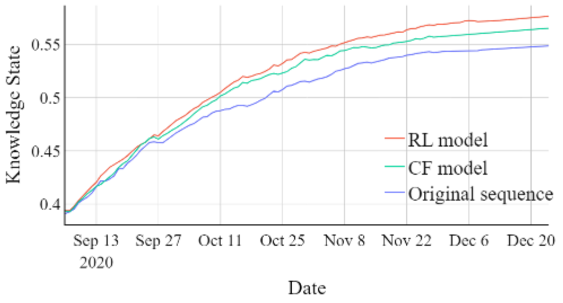


Figure 5a. Students' average KSs on CS2020 during recommendation

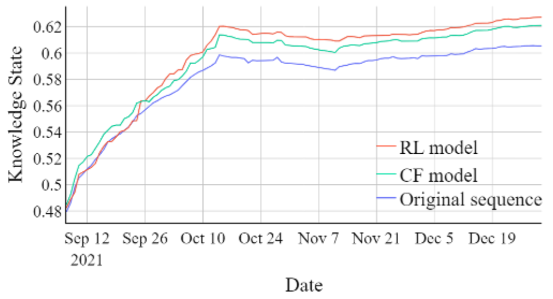


Figure 5b. Students' average KSs on CS2021 during recommendation

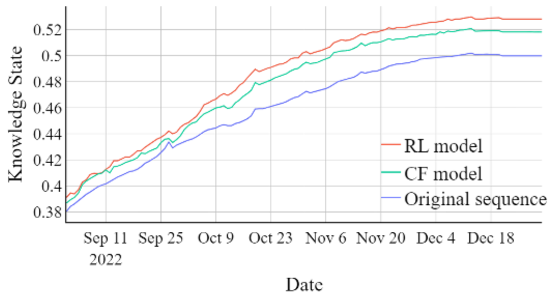


Figure 5c. Students' average KSs on CS2022 during recommendation

Figure 5: Students' average knowledge states on CSCourseDatasets during recommendation

In fall semester of 2023, we employed our RL recommendation method into a real CS course. Instead of virtual students, the actual students were considered in this experiment. The course was attended by 365 students. We randomly divided them into two groups: control group and treatment group. There were 183 and 182 students in the control and treatment groups, respectively. For the treatment group, we recommended exercises to them based on our RL recommendation method every week. Since collaborative filtering algorithms with high time complexity are not suitable for the actual class, we manually select a threshold and recommend all exercises with a pass probability below that threshold to control group students. There were 65 exercises on the recommendation list. Each week, we retrained the model, using all data from the beginning of the semester to the present. After that, we recommended the exercises using the most recent model.

The Kolmogorov-Smirnov test revealed that these distributions did not follow normal distributions ($p < 0.05$), so the Mann-Whitney U test was utilized to conduct additional analysis. After recommendation, the mean values of knowledge states indicated that the RL recommendation algorithm outperformed the other recommendation algorithm. The knowledge states of students who solved questions recommended manually were significantly lower than those who solved questions recommended by the RL-based method ($p < 0.05$). The results are in Table 4.

Table 4: Statistical analysis of two groups before and after the intervention and differences

	Control Group		**Treatment Group**		**p-value**
	Mean	SD	Mean	SD	
Before recommendation	0.427	0.046	0.431	0.042	0.433
After recommendation	0.524	0.106	0.547	0.101	0.055
Difference	0.097	0.084	0.116	0.079	0.044

5 CONCLUSION AND FUTURE WORK

This study introduces DKVMN-LB, an extension of DKVMN with a new student behavior module and an updated answer correctness structure, which outperforms the original model on five datasets. We developed a personalized learning path recommendation model using RL and incorporated three memory modules to simulate the student learning process. A simulation experiment was conducted to evaluate the recommendation performance between our method and a CF-based method. The results indicated that our approach improved students' average knowledge state more effectively on various datasets. The method is also productive for real students in actual courses, demonstrating its excellent performance and efficiency in adaptive learning systems. The proposed approach integrates answer sequences and student behavior to achieve more effective personalized recommendations, providing new perspectives for adaptive learning systems.

In the future, we will continue to apply our personalized learning path recommendation algorithm proposed to actual courses. The proposed learning path recommendation approach can potentially recommend other resources, such as reading materials and videos. We will further improve the extensibility of our approach to enhance it in an authentic learning environment.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (61907002).

REFERENCES

[1] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2014. Engaging with massive online courses. In *Proceedings of the 23rd international conference on World wide web*. Association for Computing Machinery, Seoul Korea, 687-698. https://doi.org/10.1145/2566486.2568042

[2] Hassan Khosravi, Shazia Sadiq, and Dragan Gasevic. 2020. Development and Adoption of an Adaptive Learning System: Reflections and Lessons Learned. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 58–64. https://doi.org/10.1145/3328778.3366900

[3] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. 2013. Recommender systems survey. *Knowledge-Based Systems* 46, (July 2013), 109-132. https://doi.org/10.1016/j.knosys.2013.03.012

[4] Ling Zhang, James D. Basham, and Sohyun Yang. 2020. Understanding the implementation of personalized learning: A research synthesis. *Educational Research Review* 31, (November 2020), 100339. https://doi.org/10.1016/j.edurev.2020.100339

[5] Jesus Bobadilla, Antonio Hernando, and Angel Arroyo. 2011. E-learning experience using recommender systems. In *Proceedings of the 42nd ACM technical symposium on Computer science education (SIGCSE '11)*. Association for Computing Machinery, New York, NY, USA, 477–482. https://doi.org/10.1145/1953163.1953300

[6] Amir Hossein Nabizadeh, José Paulo Leal, Hamed N. Rafsanjani, and Rajiv Ratn Shah. 2020. Learning path personalization and recommendation methods: A survey of the state-of-the-art. *Expert Systems with Applications* 159, (November 2020), 113596. https://doi.org/10.1016/j.eswa.2020.113596

[7] Ghodai Abdelrahman, Qing Wang, and Bernardo Nunes. 2023. Knowledge Tracing: A Survey. *ACM Comput. Surv*. 55, 11 (November 2023), 1-37. https://doi.org/10.1145/3569576

[8] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic Key-Value Memory Networks for Knowledge Tracing. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, Perth Australia, 765-774. https://doi.org/10.1145/3038912.3052580

[9] Zhenhai Wang, Zhiru Wang, Yuhao Xu, Xing Wang, and Hongyu Tian. 2023. Online course recommendation algorithm based on multilevel fusion of user features and item features. *Comp Applic In Engineering* 31, 3 (May 2023), 469-479. https://doi.org/10.1002/cae.22592

[10] Albert T. Corbett and John R. Anderson. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Model User-Adap Inter* 4, 4 (1995), 253-278. https://doi.org/10.1007/BF01099821

[11] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas and Jascha Sohl-Dickstein. 2015. Deep Knowledge Tracing. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*. MIT Press, Montreal Canada, 505-513.

[12] Nur W. Rahayu, Ridi Ferdiana, and Sri S. Kusumawardani. 2023. A systematic review of learning path recommender systems. *Educ Inf Technol* 28, 6 (June 2023), 7437–7460. https://doi.org/10.1007/s10639-022-11460-3

[13] Dejun Cai, Yuan Zhang, and Bintao Dai. 2019. Learning Path Recommendation Based on Knowledge Tracing Model and Reinforcement Learning. In *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*. IEEE, Chengdu, China, 1881-1885. https://doi.org/10.1109/ICCC47050.2019.9064104

[14] Youness Madani, Hanane Ezzikouri, Mohammed Erritali, and Badr Hssina. 2020. Finding optimal pedagogical content in an adaptive e-learning platform using a new recommendation approach and reinforcement learning. *J Ambient Intell Human Comput* 11, 10 (October 2020), 3921-3936. https://doi.org/10.1007/s12652-019-01627-1

[15] Wacharawan Intayoad, Chayapol Kamyod, and Punnarumol Temdee. 2020. Reinforcement Learning Based on Contextual Bandits for Personalized Online Learning Recommendation Systems. *Wireless Pers Commun* 115, 4 (December 2020), 2917-2932. https://doi.org/10.1007/s11277-020-07199-0

[16] Zhanxuan Chen, Zhengyang Wu, Yong Tang, and Jinwei Zhou. 2023. TGKT-Based Personalized Learning Path Recommendation with Reinforcement Learning. In *International Conference on Knowledge Science, Engineering and Management*. Springer Nature Switzerland, Cham, 332-346. https://doi.org/10.1007/978-3-031-40289-0_27

[17] Hado van Hasselt, Arthur Guez, and David Silver. 2015. Deep Reinforcement Learning with Double Q-learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, AAAI Press, Phoenix Arizona USA, 2094-2100. http://arxiv.org/abs/1509.06461

[18] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling Network Architectures for Deep Reinforcement Learning. In *Proceedings of the 33rd International Conference on Machine Learning*. JMLR, New York NY USA, 1995-2003. https://proceedings.mlr.press/v48/wangf16.html

[19] Yujia Huo, Derek F. Wong, Lionel M. Ni, Lidia S. Chao, and Jing Zhang. 2020. Knowledge modeling via contextualized representations for LSTM-based personalized exercise recommendation. *Information Sciences* 523, (June 2020), 266-278. https://doi.org/10.1016/j.ins.2020.03.014