An Actor-Critic Hierarchical Reinforcement Learning Model for Course Recommendation

Kun Liang , Guoqiang Zhang * , Jinhui Guo , Wentao Li

Posted Date: 23 October 2023

doi: 10.20944/preprints202310.1367.v1

Keywords: course recommendation; actor-critic method; hierarchical reinforcement learning method; policy gradient method

Article

An Actor–Critic Hierarchical Reinforcement Learning Model for Course Recommendation

Kun Liang, Guoqiang Zhang *, Jinhui Guo and Wentao Li

College of Artificial Intelligence, Tianjin University of Science and Technology, Tianjin 300457, China
* Correspondence: zhangguoqiang@mail.tust.edu.cn

Abstract: Online learning platforms provide diverse course resources, but this often result in the issue of information overload. Learners always want to learn courses that are appropriate for their knowledge level and preferences quickly and accurately. Effective course recommendation plays a key role in helping learners select appropriate courses and improving the efficiency of online learning. However, when a user is enrolled in multiple courses, Existing course recommendation methods face the challenge in accurately recommending the target course that is most relevant to the user, because of the noise courses. In this paper, we propose a novel reinforcement learning model named Actor-Critic Hierarchical Reinforcement Learning (ACHRL). The model incorporates the Actor-Critic method to construct the profile reviser. This can remove noise courses and make personalized course recommendation effectively. Furthermore, we propose a policy gradient based on temporal difference error to reduce the variance in the training process, to speed up the convergence of the model, and improves the accuracy of the recommendation. We evaluate the proposed model on two real datasets, and the experimental results show that the proposed model is significantly outperforms the existing recommendation models (improving 3.77% to 13.66% in terms of HR@5).

Keywords: course recommendation; actor-critic method; hierarchical reinforcement learning method; policy gradient method

1. Introduction

With the reform of traditional education methods, online course education has been developing rapidly. Massive open online courses (MOOCs) have become the most important platform for online education [1]. Worldwide, various MOOC platforms such as Coursera and edX have emerged, extending access to courses from prestigious universities to millions of students. In China, XuetangX has risen as one of the largest MOOC platforms, offering thousands of courses and attracting a substantial user base [2]. The evolution of MOOC platforms has disrupted the conventional offline teaching approach of traditional education [3–6]. This transformation not only addresses the issue of unequal distribution of educational resources but also offers users the convenience of flexible learning. Nevertheless, this proliferation of online courses and the constant evolution of knowledge pose a new challenge in the form of information overload for MOOC platforms [7–9]. In a new domain, users are often confused when facing the massive content, and they cannot find the courses that are really suitable for them quickly. Now, The primary challenge is whether the learners' records of their learning can be analyzed and utilized effectively, recommend suitable target courses to them.

Course recommendation [10–14] is an effective approach to tackle this challenge [15]. The formalized definition of the course recommendation is: When presented with a set of historical courses that users enrolled in before time t, our goal is to recommend the most relevant courses that the user will enroll in at time t+1 [16]. Clearly, it is particularly important to describe and model the user profile accurately for the recommendation model. To this end, the factor item similarity model (FISM) [17] can be used to extract implicit feedback from the learning behaviors of users, or the neural attentive item similarity (NAIS) [18] can be used to distinguish the importance of different historical courses to target courses by assigning different weight coefficients to historical courses. However, different from many other recommendation domains [19–23], As users' improvement of cognition level and the change of interests , they will enroll diversified courses. Under the circumstances, the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

recommendation effect of courses with high contribution will be weakened by other courses with low contribution, which may result in recommending a course that do not align with the user's preferences. These low contribution courses are called noise courses.

In order to solve the problem of noise courses, Hierarchical Reinforcement Learning (HRL) algorithm [16]can be used, acting on the user profile to remove noise courses in the historical courses sequence through a two-level sequential decision process. Then, joint training with the recommendation model (NAIS).

While substantial advancements have been achieved in course recommendation using the HRL model, there are still some issues with the model. Specifically, The profile reviser of the model contains a high- level task and a low-level task, when the high-level task decides not to revise the user profile, even if there are some noise courses at this time， the model cannot remove them.

Inappropriate decisions will diminish the overall performance of the model and may converge to a local optimum state. For the low-level task, because the sampling of the policy function and the state transition function are both random probability values in the reinforcement learning (RL) method [24]. Removing some noise courses randomly will lead to instability of model prediction. When recommending target courses to users, some of the more relevant courses may be ignored. As shown in Figure 1, the HRL model recommended the target course "Monetary Finance" due to the noise course "Principles of Economics". But according to the historical courses, the user has studied "C++" and "Principles of Databases", so he or she should prefer the computer programming courses.

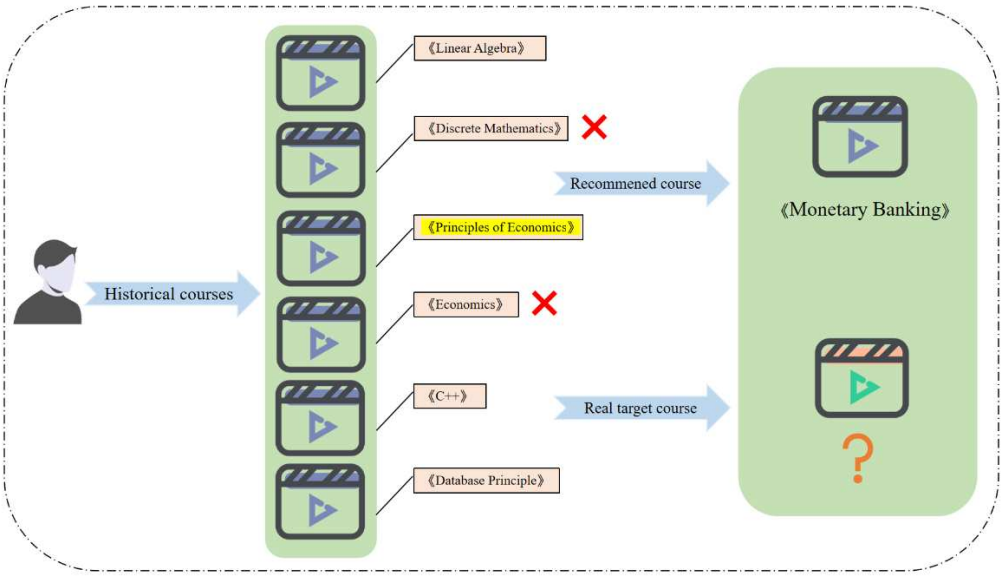

Figure 1. An example of HRL model for course recommendation.

In the above case, The user profile reviser does not guarantee the accuracy of decisions. In this work, we aim to reconstruct the profile reviser to improve the accuracy of course recommendation. To this end, we propose the new course recommendation model: actor-critic hierarchical reinforcement learning (ACHRL) model. The Actor-Critic method (AC) embedded in the model can improve the accuracy of decision making efficiently in the process of revising user profile. In particular, we adopt the policy gradient method [25] based on temporal difference error [26].This can reduce variance and improves robustness. The profile reviser can be updated in a timely manner, improving the accuracy of decisions adaptively. In summary, the contributions of our research are as follows:

- We propose an actor-critic hierarchical reinforcement learning model (ACHRL) to optimize the user's profile reviser of HRL model and improve the accuracy of course recommendations.
- We propose a policy gradient method based on temporal difference error, which updating the policy adopted by agent of reinforcement learning model with rewards at the current moment and states of the moments before and after. This can speed up model convergence and improve

 doi:10.20944/preprints202310.1367.v1

the accuracy of the recommendation model.

- We performed experiments on two MOOC datasets, with the majority of users enrolling in courses from various categories, and our proposed model demonstrated a significant improvement over the baseline model on different evaluation metrics.

2. Related Work

A. Course recommendation

The rapid development of Massive Open Online Course (MOOC) platforms has led to increased research and application of course recommendation systems. Traditional course recommendations often rely on feature engineering, where valuable insights are extracted from user characteristics and historical learning data. Existing course recommendation methods can be classified into four primary categories:

Course Recommendation Based on Collaborative Filtering. Collaborative filtering methods encompass user-based and item-based collaborative filtering. These methods typically recommend courses to users based on the preferences of users with similar interests or the characteristics of courses. For example, Ray et al. [27] extended the collaborative filtering approach to develop a course recommendation system that provides accurate grade predictions, which aids in elective course selection. Li et al. [28] designed a personalized online education platform based on collaborative filtering, resulting in more accurate and efficient course recommendations aligned with users' interests. **Course Recommendation Based on Content.** Content-based recommendation methods leverage information about the features of the courses to make recommendations. These methods are suitable for addressing cold start problems as they don't rely on user interaction data. Ghauth et al. [29] proposed an e-learning recommendation system based on content-based filtering, which enhanced users' course outcomes by increasing the accuracy of the learning system. **Course Recommendation Based on Knowledge Graph.** Knowledge graph-based recommendation methods utilize graph structures to represent the relationships between courses. They leverage domain knowledge and semantic relationships within the knowledge graph to model connections between a user's interests and courses, resulting in personalized recommendations. Xu et al. [30] applied knowledge graph representation learning to embed semantic information of items into a low-dimensional semantic space and calculated semantic similarity between recommended items. This approach enhanced recommendation performance at the semantic level. **Course Recommendations Based on a Hybrid Approach.** Hybrid recommendation methods combine multiple recommendation techniques, leveraging the strengths of each to provide a combined set of recommendations that aim to enhance accuracy and diversity. Hybrid approaches have become a popular solution for addressing various recommendation challenges simultaneously, offering greater adaptability compared to single-method approaches. Emon et al. [31] used a hybrid approach that combines association rule mining and user-based collaborative filtering to develop a recommendation system that identifies the unique interests of different students in learning materials more effectively, resulting in more personalized recommendations. Gulza ret al. [32] proposed a hybrid method that uses ontology in combination with ontology to retrieve valuable information and make accurate recommendations, with the potential to improve learners' performance and satisfaction.

B. RL-based Course recommendation

In recent years, there have been significant breakthroughs in Reinforcement Learning (RL) within the field of recommendation systems [33]. RL has achieved remarkable success in various domains, including e-commerce, video, and gaming [34–36]. These advancements have also opened up new opportunities in the realm of course recommendations within the education sector. As an interactive recommendation approach, RL-based recommendation models can continuously update their recommendation policies by receiving real-time feedback from users. This dynamic approach aligns more closely with real-world recommendation scenarios compared to traditional static methods. Particularly, Deep Reinforcement Learning (DRL) [37–40] stands out for its capacity to process large-scale data, extract underlying features, and accurately achieve specific goals through end-to-end learning.-One of the challenges in traditional reinforcement learning methods [41–43] is

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

the issue of the "dimensional disaster." When the environment is complex or the task is intricate, the state space of the agent becomes too extensive. This results in a rapid increase in the number of parameters to be learned and the memory space required. Consequently, achieving the desired results becomes difficult. To address the dimensional disaster problem, researchers have introduced Hierarchical Reinforcement Learning (HRL) [44–46]. The objective is to break down complex problems into smaller, more manageable subproblems and solve the original task by addressing these subproblems individually. In complex recommendation tasks, optimizing the policy directly according to the final goal can be inefficient. Hierarchical methods offer an effective means of enhancing recommendation efficiency by breaking down complex tasks into more manageable components.

HRL has made great progress in MOOC course recommendation. The accurate construction of the user profile model is the foundation of a recommendation system. Zhang et al. [16] applied HRL technology to course recommendation for the first time. The author believed that the noise courses in the sequence of historical courses will affect the weight of the courses with real contributions. The noise courses are removed by the reinforcement learning method. Utilizing the revised data can indeed enhance the accuracy of course recommendations. On the basis of the above work, Yang et al. [47] focused on improving the performance of the profile reviser, making the agent to obtain the cumulative rewards of the context and adopt better policies Lin et al. [48] used a recommendation model can capture user preferences by historical course and improve the accuracy of course recommendation. Inspired by the above methods, we proposed the ACHRL model to optimized the profile reviser, which ensures more accurate decisions on whether to delete noise courses. Finally, the accuracy of the recommendation results is improved.

3. Preliminaries

C. Policy gradient method

We need to learn a target policy explicitly to defining the policy-based method [49] in reinforcement learning. The policy gradient is the basis of the policy-based method. It can parameterize the policy. Our objective is to discover an optimal policy that maximizes the expected return in the given environment. Therefore, the update of the objective function approximates the gradient ascent in $J(\theta)$ as follows:

$$\theta_{t+1} = \theta_t + \alpha\widehat{\nabla J(\theta_t)} \tag{1}$$

where θ_t is the policy parameter at time t, α is the learning rate, which controls the step size of parameter updates, $J(\theta_t)$ is a random estimate, and its expected value can be approximated as the gradient of $J(\theta_t)$ according to its parameter θ_t The method that follows Eq. (1) is referred to as the policy gradient method.

D. Actor-critic method

In reinforcement learning, the policy-based method needs to learn a policy function, the value-based method needs to learn a value function explicitly. Actor-Critic(AC) method [50] combines both aspects, enabling the simultaneous learning of both the policy function and the value function. This combination allows the value function to assist in the more effective learning of the policy function. In the context of policy gradients, we can express the gradient formula in a more generalized form as follows:

$$g = \mathbb{E}\left[\sum_{t=0}^{T} \psi_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right] \tag{2}$$

$$\psi_t = r_t + \gamma\, V_\omega(s_{t+1}) - V_\omega\,(s_t)$$

where $\mathbb{E}$ represents the expected value of a random variable based on the policy function π, In this paper, ψ_t is the temporal difference error to guide the learning of policy function. $\pi_\theta(a_t \mid s_t)$ represents policy function, We could call it "Actor".

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

In this paper, we employ the actor-critic method, as illustrated in Figure 2. The value function in this method plays a crucial role in guiding the learning of the policy function and optimizing the agent's action selection. We update the value function using the Temporal Difference error (TD error). Notably, the actor-critic method is essentially a policy-based approach, because its goal is also to optimize a policy with parameters. In summary, we utilize a policy gradient method based on the temporal difference error to enhance the objective function.

4. Proposed ACHRL Framework

In this section we first present the overview of the proposed model. Then, we elaborate the two modules of the model and two separable components of ACHRL, Ultimately, we will provide a detailed description of the model's training process.

Overview. The ACHRL model is depicted in Figure 3. It consists of two parts: a profile reviser module optimized by AC method and a recommendation module based on attention mechanism. The profile reviser is a hierarchical reinforcement learning framework, consisting of a two-layer Markov decision process [51] (MDP). In this framework, the high-level task is responsible for determining whether to revise the user profile, while the low-level task is tasked with deciding which noise course should be removed. When the high-level task determines to revise the user profile, the delay reward can be obtained after removing the last noise course in the low-level task. Two temporal difference errors from the AC method: TD-ERROR_H and TD-ERROR_L can guide the updating of high-level policy and low-level policy respectively After completing the task of the profile reviser, the embedding vector of the courses is fed into the recommendation module generating the recommended probability value. In the ACHRL model, we optimize the two-layer tasks of the profile reviser so that the agent(a^h, a^l) can take more accurate actions. This can provide more accurate data for recommendation modules.(The variables involved in the figure are explained in the introduction to the two modules of the model).

E. Profile reviser module

In the profile reviser, the agent selects the high-level action decision according to the high-level policy and removes the noise course according to the low-level policy. The key challenge is how the two-level task can revise the user profile more accurately.

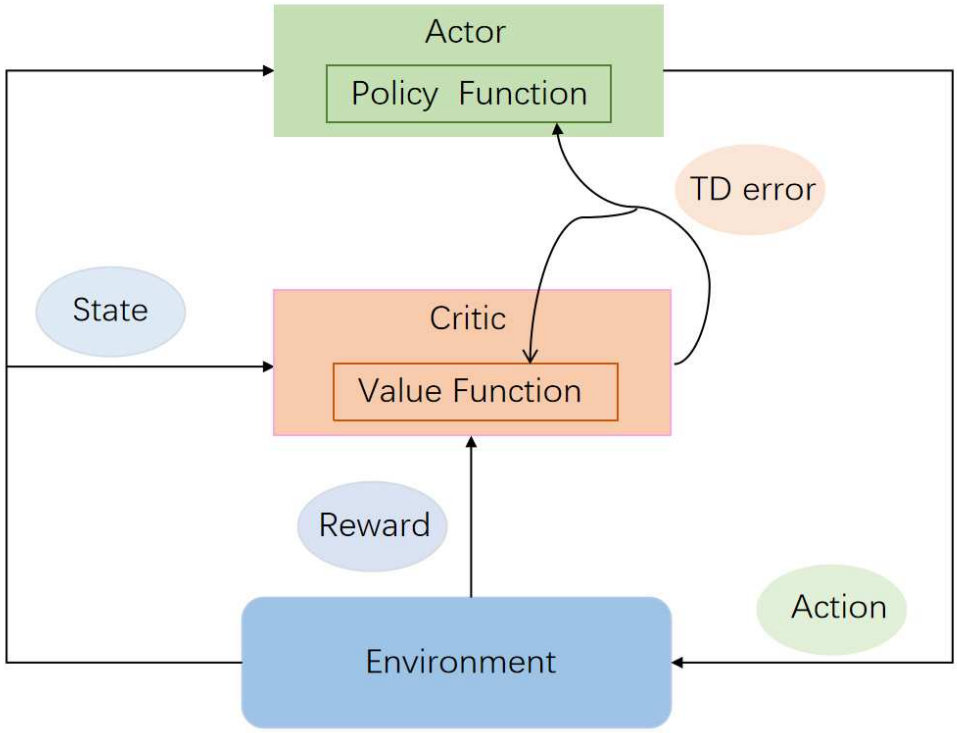

Figure 2. The execution process of AC method.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

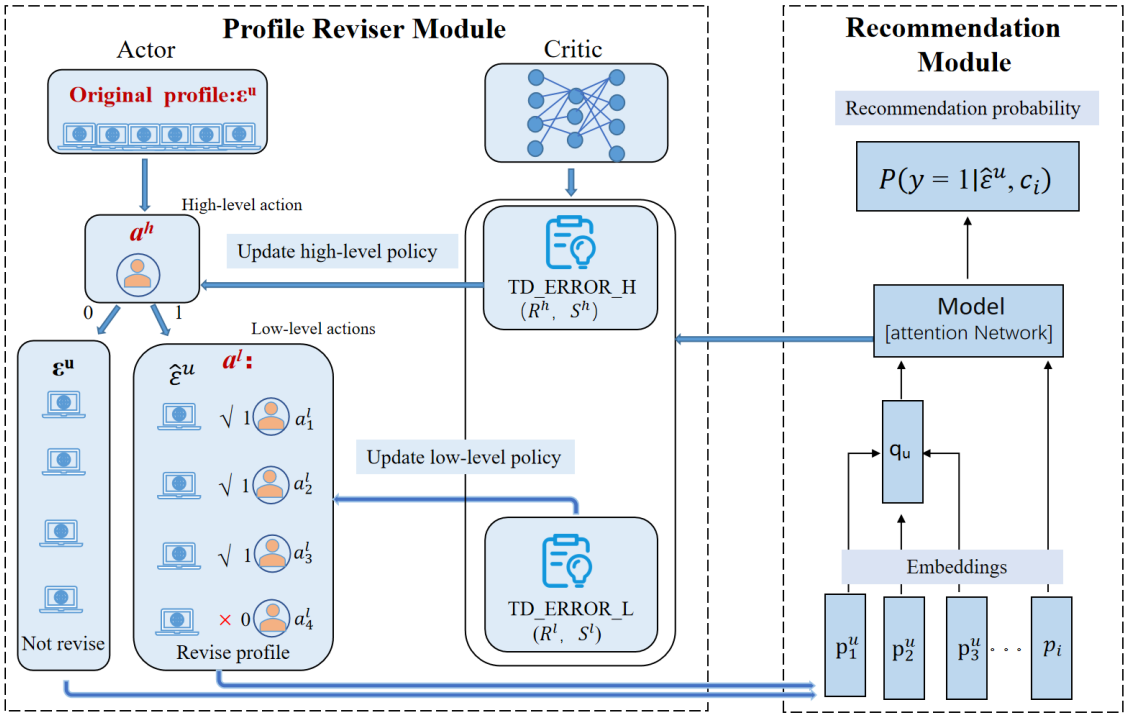

Figure 3. The overall framework of the proposed model (ACHRL) We use AC method to optimize the HRL model to revise the user profile. Next, we will introduce the specific composition of the profile reviser based on HRL and its optimization by AC method in detail.

1) HRL for profile reviser module

The modification task of the user profile can be divided into a two-layer Markov decision process (MDP). The first layer is a high-level task that decides whether to revise the user profile, and the second layer is a low-level task responsible for determining which historical course to remove. They can be transformed into a 5-tuple Markov Decision Process (MDP) denoted as $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \beta \rangle$. $\mathcal{S}$ represent a set of states, and $\mathcal{A}$ represent a set of actions. $\mathcal{P}$ is the state transition function. That $\mathcal{S} \times \mathcal{A} \times \mathcal{S}$ converts to a probability value of [0,1] by mapping. $\mathcal{R}$ is a function used to represent rewards. $\beta \in \{0,1\}$ signifies the discount factor applied to the reward at each moment. The following paragraph describes the content come from [16].

State. In the high-level task, we define the state features s^h, which is characterized as the average cosine similarity and the average element-wise product between the embedding vectors [52] of each historical course e_t^u and the target course c_i. In the low-level task, we define the state feature s_t^l, which is characterized as the cosine similarity [53], the element-wise product and the average of the two previous features between the embedding vector of the current historical course e_t^u and the target course c_i.

Action. Action is a discrete variable. The action $a^h \in \{0,1\}$ in the high-level task determines whether to revise the profile of a user or no, and the action $a^l \in \{0,1\}$ in the low-level task represent whether to remove the historical course e_t^u or not.

Reward. Reward is used to indicate whether the performed actions are reasonable or not. When deciding to revise the user profile, the reward can be obtained after the last action is executed in the low-level task. Before that, the reward is zero. Here $R_h(s_t^h, a_t^h)$ is defined as the difference between the log-likelihood after and before the profile is revised. The reward for the high-level task is defined as follows:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

$$R_h(s_t^h, a_t^h) = \log P(\hat{\varepsilon}^u, c_i) - \log P(\varepsilon^u, c_i), \quad if\ t = t_u$$
$$R_h(s_t^h, a_t^h) = 0, \quad if\ t \neq t_u \tag{3}$$

where s_t^h denotes the state of high-level task at time t, a_t^h denotes the action of the high-level task at time t, and $\hat{\varepsilon}^u$ is the revised profile, which is a subset of ε^u.The ε^u is the original profile. c_i denotes a target course, $P(\hat{\varepsilon}^u, c_i)$ is an abbreviation of $P(y = 1 \mid \hat{\varepsilon}^u, c_i)$ The logarithmic function is used in the formula to have better convergence for the training of the model.

For the reward $R_l(s_t^l, a_t^l)$ in the lower-level task, aside from the common component shared with the high-level task reward, there is an internal reward in the low-level task, which is defined as the difference of the average cosine similarity between each historical course e_t^u after and before the user profile is revised and the target course c_i, Internal reward is used to optimize the efficiency of the policy execution of the agent in the low-level task. The reward of the low-level task is defined as follows:

$$R_l(s_t^l, a_t^l) = (\log P(\hat{\varepsilon}^u, c_i) - \log P(\varepsilon^u, c_i)) + \left(\frac{\sum_{t=1}^{\hat{t}_u} (e_t^{uT} c_i)}{\hat{t}_u} - \frac{\sum_{t=1}^{t_u} (e_t^{uT} c_i)}{t_u}\right) \quad if\ t = t_u$$
$$R_l(s_t^l, a_t^l) = 0 \quad if\ t \neq t_u \tag{4}$$

where s_t^l represents the state of the low-level task at time t, a_t^l represents the action of low-level task at time t, $e_t^{uT} c_i$ represents the cosine similarity between the historical courses e_t^u and the target course c_i.

Policy. The policy function of the high-level task:

$$\pi_\theta(s_t^h, a_t^h) = \text{softmax}\left(a_t^h \sigma(W_2^h z_t^h) + (1 - a_t^h)\left(1 - \sigma(W_2^h z_t^h)\right)\right)$$
$$z_t^h = \text{ReLU}(W_1^h s_t^h + b^h) \tag{5}$$

where $\mathbf{W}_1^h \in \mathbb{R}^{d_1^h \times d_2^h}, \mathbf{W}_2^h \in \mathbb{R}^{d_2^h \times 1}$ and bias vector $\mathbf{b}^h \in \mathbb{R}^{d_2^l}$ are the parameters to be learned in the neural network, d_1 is the size of the hidden layer, d_2 is the number of state features, z_t^h is the size of the hidden layer of the neural network at time t in the high-level task, and σ is the non-linear activation function used to transform the state into a probability value. ReLU is a activation function. Therefore, the policy parameter of the high-level task can be defined as follows $\boldsymbol{\theta}^h = \{\mathbf{W}_1^h, \mathbf{W}_2^h, \mathbf{b}^h\}$, Similarly, we can get the low-level task policy parameter as $\boldsymbol{\theta}^l = \{\mathbf{W}_1^l, \mathbf{W}_2^l, \mathbf{b}^l\}$.

2) AC method optimize profile reviser

Different from the decision made by the agent relying on the policy function, the action selection of the agent in the AC method has higher accuracy. Based on this, we use the AC method in the hierarchical task of the profile reviser. In this case, instead of relying on the cumulative rewards obtained in each training round for policy updates, the agent performs policy updates through the temporal differential error $\delta(t)$. The $\delta(t)$ is estimated using the critic network of AC method. In the calculation process of $\delta(t)$, we use the rewards of the current moment and the states of the moments before and after. It is more instructive for the policy update. The formula is as follows:

$$\delta(t) = r_t + \gamma V_\omega(s_{t+1}) - V_\omega(s_t) \tag{6}$$

where V_ω represents the state value network in reinforcement learning, ω is the parameter of the value network, s_t and s_{t+1} represent the state of the current moment and the next moment respectively, r_t is the reward of the current moment. $\gamma \in (0,1)$ is a discount factor for cumulative rewards of the agent.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

Algorithm 1 Actor-Critic Method

Input: a derivable policy parameterization $\pi_\theta(s,a)$, a derivable actor-value a derivable parameterization $Q^{\pi_\theta}(s_t,a_t)$, a state-value parameterization $V_\omega(s_t)$

Initialize: parameters $\theta = \theta^0$, $W = W^0$

1: ***for*** *sequence* $e = 1$ *to* E ***do***:
2: Generate a sampling sequence {s1,a1,r1,s2,a2,r2…} following π_θ
3: for data at each step **do**
4: $\delta_t = r_t + \gamma V_\omega(s_{t+1}) - V_\omega(s_t)$
5: $w = w + \alpha_\omega \sum_t \delta_t \nabla_\omega V_\omega(s_t)$
6: $\theta = \theta + \alpha_\theta \sum_t \delta_t \nabla_\theta \log \pi_\theta(s_t,a_t)$
7: ***end for***

3) Policy gradient based on temporal differential error

The previous section illustrates that the value function in the AC method can guide the policy function enable the agent make more accurate decisions. We use the temporal difference error in the gradient update process, and the gradient formula is as follows:

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a \nabla_\theta \pi(a \mid s,\theta)(r_t + \gamma V_\omega(s_{t+1}) - V_\omega(s_t)) \quad (7)$$

The $\propto$ in the formula represents the two sides of the formula are proportional, $\mu(s)$ represents a policy distribution in the policy function, $r_t + \gamma V_\omega(s_{t+1}) - V_\omega(s_t)$ is temporal differential error (δ(t)).

4) Objective function

The ACHRL model improves the accuracy of course recommendations. For the hierarchical tasks, the temporal difference error is used in the policy gradient method to optimize policy function. It is calculated by the value of rewards and states. For our model, the objective function is to maximize the expectation of expected rewards:

$$\theta^* = \arg\max_\theta \sum_\tau P_\theta(\tau) T(\tau) \quad (8)$$

The θ represents the policy parameter: θ^h or t θ^l.τ represents the sampling sequence of the actions and transition states. For the high-level task, it can be represented as $\{s^h, a^h\}$, for the low-level task, it can be represented as $\{s_1^l, a_1^l \ldots s_t^l, a_t^l\}$, $P_\theta(\tau)$ is the probability of the state transition in the high-level task or the low-level task, $T(\tau)$ is the temporal differential error δ(t).

The policy function for high-level task are as follows:

$$\nabla_\theta = \frac{1}{n} \sum_{n=1}^{N} \sum_{t=1}^{t_u} \nabla_\theta \log \pi_\theta(s^n,a^n) r_t^h + \gamma V_\omega^h(s_{t+1}) - V_\omega^h(s_t) \quad (9)$$

where t_u represents the quantity of historical courses. r_t^h is the real reward obtained by the high-level task at time t. V_ω^h is the representation of the state value network $V_\omega$in the high-level task, ω is the updated parameter in the state value network V. $r_t + \gamma V_\omega(s_{t+1}) - V_\omega(s_t)$ is the temporal difference error.

In the low-level task, the delayed reward can be obtained after removing the last noise course. The temporal difference error can be obtained through the calculation of the rewards and states. The policy function of the low-level task as follows:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

$$\nabla_\theta = \frac{1}{n}\sum_{n=1}^{N}\sum_{t=1}^{t_u} \nabla_\theta \log \pi_\theta(s_t^n, a_t^n) r_t^l + \gamma V_\omega^l(s_{t+1}) - V_\omega^l(s_t) \tag{10}$$

where r_t^l is the real reward which obtained at the time t of the low-level task. V_ω^l is the representation of the state value network V_ω in the low-level task. The meanings of other terms in the formula can be analogous to the relevant definitions in the above high-level policy network.

F. Attention-based recommendation module

In our model, the recommendation module is the NAIS model [18]. It is a recommendation model that relies on an attention mechanism, and it excels at efficiently converting the course embedding vectors obtained from the profile reviser into recommended probability values. Besides, this can ensure provide a fair and reasonable verification for our ACHRL model and the HRL model. The following is a basic introduction of the NAIS model.

$$\mathbf{q}_u = \sum_{t=1}^{t_u} a_{it}^u \mathbf{p}_t^u, a_{it}^u = f(\mathbf{p}_t^u, \mathbf{p}_i) \tag{11}$$

where $\mathbf{q}_u$ represents the embedding vector of profile, $\mathbf{p}_t^u$ is a embedding vector of historical courses e_t^u. $\mathbf{p}_i$ is an embedding vector of target course c_i, a_{it}^u is an attention weight coefficient of the historical course e_t^u, during the recommendation process. $f(\mathbf{p}_t^u, \mathbf{p}_i)$ is an attention function . Its essence is one multi-layer perceptron (MLP) [54].

$$f(\mathbf{p}_t^u, \mathbf{p}_i) = \mathbf{h}^\top \mathrm{ReLU}(\mathbf{W}^{at}(\mathbf{p}_t^u \odot \mathbf{p}_i) + \mathbf{b}^{at}) \tag{12}$$

where $\mathbf{h}^\top$ is the attention weight vector mapped by the hidden layer of the neural network, $\mathbf{W}^{at}$ is the weight matrix, $\mathbf{b}^{at}$ is a bias vector. ReLU is an activation function.

According to $\mathbf{q}_\mathrm{u}$ and $\mathbf{p}_i$ the attention-based recommendation module generates a recommendation probability $\rho_{u,i}$.

$$\rho_{u,i} = P(y = 1 \mid \hat{\varepsilon}^u, c_i) = \sigma(\mathbf{p}_i^T \mathbf{q}_u) \tag{13}$$

If the value of y is 1, the recommendation is successful, σ is an activation function used to convert the input embedding to the probability value in the recommendation model.

G. Separable two components of ACHRL

The model proposed in this paper is called ACHRL to indicate that we optimize the two-layer structure of the profile reviser by using the AC method. We use the term ACHRL_H to represent the optimization of the high-level task. The term ACHRL_L represents the optimization of the low-level task. Next, we will provide a comprehensive description of both of them. In the ablation experiment , we will compare both model methods with HRL and ACHRL.

1) High-level task optimization: ACHRL_H

As has been introduced above, a major problem of the HRL model is the action of agent is random. The high-level task cannot decide whether to revise the user profile accurately. The ACHRL_H model enhances the precision of decision-making for high-level task. To ensure a fair performance comparison with the HRL model, we employ the REINFORCE algorithm [55] to calculate the gradient of the low-level task policy function too:

$$\nabla_\theta = \frac{1}{n}\sum_{n=1}^{N}\sum_{t=1}^{t_u} \nabla_\theta \log \pi_\theta(s_t^n, a_t^n) R_l(s_t^n, a_t^n) \tag{14}$$

where $R_l(s_t^n, a_t^n)$ represents the cumulative delay reward of the low-level task obtained by each sampling sequence τ.

2) Low-level task optimization: ACHRL_L

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

In the HRL model, the key to the low-level task is How to identify the noise course and remove it accurately. Due to the limitation of the policy function itself, the low-level task removes noise courses with randomness, which makes the execution of the policy less accurate, and may even remove the valid course mistakenly, affecting the recommendation performance. To solve this problem, we use ACHRL_L to assist low-level task to remove noise courses improving the accuracy of low-level action selection. While removing noise courses, we reserve the target course for each user as far as possible. We still use the REINFORCE algorithm to calculate the gradient of the high-level task policy function:

$$\nabla_\theta = \frac{1}{n}\sum_{n=1}^{N}\sum_{t=1}^{t_u}\nabla_\theta \log \pi_\theta(s^n, a^n) R_h(s_t^n, a_t^n) \tag{15}$$

where $R_h(s_t^n, a_t^n)$ represents the cumulative delay reward for each sampling sequence in the high-level task. Note here: Some element representations in formulas (9), (10), (14), and (15) are simplified, and we omit h and l representing high and low levels.

D. Model training

The previous section outlines that the entire model can be split into two components. The training process involves three key steps: initially, pre-training the recommendation module [56], followed by pre-training the profiler reviser module, and finally, jointly train the two models together.

Next, we illustrate the details of its interaction by combining Figure 4 and Algorithm 2. First, pre-train our recommendation module (i.e., $\emptyset^0 = \{\mathrm{h}^\top, \mathrm{W}^{at}, \mathrm{b}^{at}\}$) using the original dataset (ε^u). Then we conduct the pre-training of the profile reviser module (i.e., $\theta^0 = \{w_1, w_2, b\}$), In this process, the update of the high -level policy and the low-level policy depend on the optimization of the temporal difference error (TD-ERROR_H and TD-ERROR_L) calculated by the AC method. Finally, we conduct the joint training of the two models. If the output: $P(y = 1 \mid \hat{\varepsilon}^u, c_i)$ of the recommendation model is greater than the threshold set in the experiment (e.g., 0.6) The recommendation model has got a reasonable recommendation result. At this time, the high-level task makes the decision corresponding to "0" in Figure 4, which means that the user's profile is not modified. No reward will be given in this case and the δ^h is zero(corresponding to the fifth line in Algorithm 2) The course embedding vector of the unrevised profile are input into the recommendation model directly. On the contrary, the high-level task takes the decision corresponding to "1" in Figure 4, which means revising the user's profile is required. At this time, noise courses need to be removed .In low-level task, "0" means remove the course, "1" means keep the course. When the low-level task is completed, both the high-level task and the low-level task obtain the reward. The corresponding values are also obtained for both δ^h and δ_t^l.Under the optimization of them, the removing work of all noise courses is completed, and the user profile is revised (lines 7-9 of Algorithm 2). The revised profile is input into the recommendation model as an embedding vector. The next round of joint training begins until the end of the training. The parameters are updated during the training process (lines 12-13 of Algorithm 2).

Algorithm 2 Actor-Critic -Based Hierarchical Reinforcement Learning

Input: training data: ε^u; pre-train recommendation model parameterized by: $\emptyset^0 = \{\mathrm{h}^\top, \mathrm{W}^{at}, \mathrm{b}^{at}\}$; pre-train profile reviser parameterized by: $\theta^0 = \{w_1, w_2, b\}$; Derivable state value functions: $v_h(s,w)$, $v_l(s,w)$

Initialize: $\emptyset = \emptyset^0$, $\theta = \theta^0$, $\delta^h(r,s)=0$, $\delta^l(r,s)=0$;

1: **for** sequence k=1 to k **do**

2: **for each** ε^u: = $(e^u_1, \ldots e^u_{t_u})$ and c_i **do**

3: Sample a high-level action a^h with θ^h, in the high-level task;

4: **if** $P(y = 1 \mid \hat{\varepsilon}^u, c_i) > 0.6$ **then**

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

5: $\delta_h^k(r_t^h, s_t^h) = 0$

6: **else**

7: Sample a sequence of states and actions with θ^l, in the low-level task;

8: Calculate: $\delta_h^k(r_t^h, s_t^h)$, $\delta_l^k(r_t^h, s_t^h)$;

9: Calculate the gradients by Eq. (9) and (10) according to Algorithm 1;

10: **end if**

11: **end for**

12: Update parameters θ^h, θ^l, w^h, w^l by the gradients;

13: Update parameter$\emptyset$ by the recommendation module;

14: Output the recommendation probability$P(y = 1 |$ $\hat{\varepsilon}^u, c_i)$ by Eq. (13)

15: **end for**

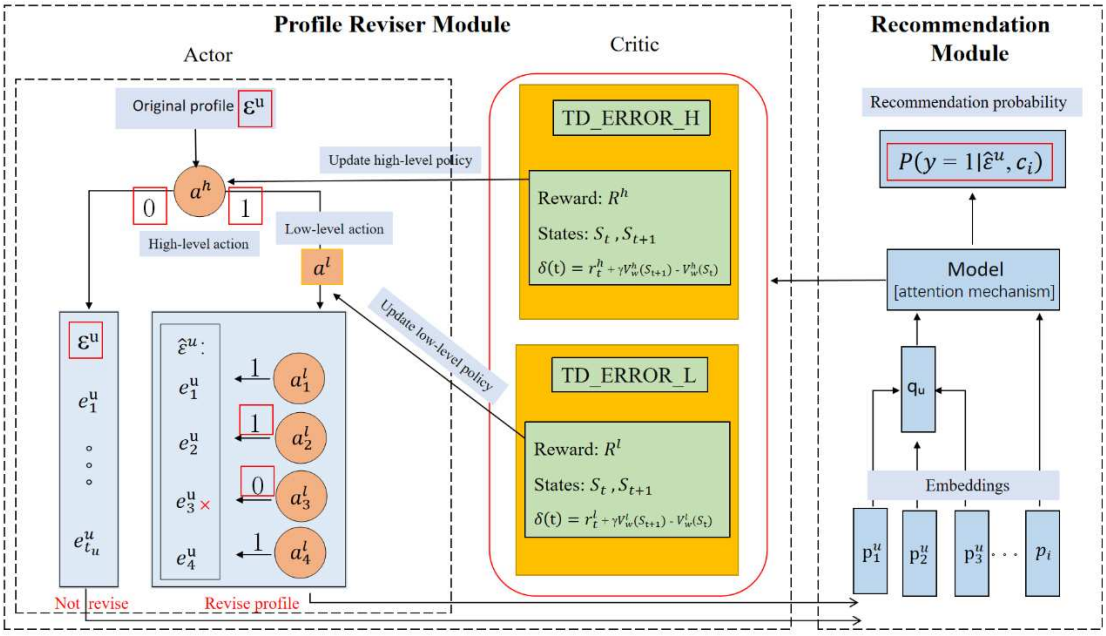

Figure 4. Interaction details of recommendation module and profile reviser.

5. Experiment and Analyze

In this section, we'll begin by introducing two datasets and explaining the experimental setup. Subsequently, we'll delve into the results and findings of the experiment.

E. Dataset

We conduct the experiment using the real-world datasets, MOOCCourse[1] and MOOCCube. Both of these datasets are from the domestic MOOC platform "XuetangX[2]". Table 1 shows the details of the two datasets. Each user registered for at least three courses in the MOOCCourse dataset, and each user registered for at least two courses in the MOOCCube dataset. To ensure the fairness of the experiment, we adopted the same data preprocessing method as the HRL mode [16]. We structured the user enrollment behavior in such a way that the actions in the training phase, actions were taken in the training set before the test set. Each entry in both training and test sets includes a series of previous courses alongside the user's target courses. For training, we identified the final course in each sequence as the target course, with all the previous courses treated as historical. In the testing

[1] http://moocdata.cn/data/course-recommendation.

[2] http://www.xuetangx.com.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

phase, every historical course in the test set was regarded as the target course, and the corresponding course from the training set for the same user was considered the historical course.

Table 1. The experimental datasets.

Dataset	#Courses	#Users	#Interactions
MOOCCourse	1302	82535	458453
MOOCCube	706	55203	190049

F. Experimental setup

1) Compared method

To evaluate model performance, we compared the ACHRL model with some baseline models, as follows:

- **MLP [54]**: A powerful recommendation system model that leverages deep learning techniques to provide personalized recommendations based on complex user-item interactions.
- **FISM [17]**: An important model in the field of recommendation systems, particularly suitable for collaborative filtering recommendation problems where user historical behavior data is the primary input.
- **NeuMF [57]**: A model that combines matrix decomposition techniques and MLP methods to mine the potential information of user courses for modelling and recommend relevant courses to users.
- **NARM** [58] [58]: An optimized gated recursive model that evaluates the attention factor based on the behavior and primary purpose of users.
- **NAIS [18]**: The model built by an item-based collaborative filtering method combined with an attention mechanism neural network that can distinguish between different historical course weights for course recommendation.
- **HRRL [47]**: An HRL-based method using time-context rewards can optimize strategy learning in reinforcement learning for course recommendation.
- **DARL [48]**: a novel course recommendation framework can capture user preferences by historical data improving the effectiveness of course recommendations.

We investigate recommendation performance of the HRL model and two variants of the ACHRL model (ACHRL_H and ACHRL_L).

- HRL [16]: Recommendation model and profile reviser joint training.
- ACHRL_H: A Simplified version of ACHRL model and profile reviser joint training, and only adopts optimization of AC method in the high-level task of profile reviser.◎
- ACHRL_L: A simplified version of ACHRL model and profile reviser joint training, and only adopts optimization of AC method in the low-level task of profile reviser.

2) Evaluation Metrics

We employ the most authoritative evaluation metrics used in the recommended field [59] :

- HR@K (Hit Ratio at K) : A measure of how many of the relevant items were successfully included in the top K recommendation.

The following formula represents a successful recommendation to the user

$$HR@K = \frac{\sum_{u=1}^{U} H_{its_u@K}}{|GT|} \tag{17}$$

where GT represents the total number of items corresponding to users in all test sets, $H_{its_u@K}$ represents the count of items that users interact with or find relevant in the previous K recommendations.

- NDCG@K (Normalized Discounted Cumulative Gain) : An accumulative performance measure that takes into account both the relevance and position of ranked items It can be defined as follows:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

$$\text{NDCG@K} = \frac{1}{U}\sum_{u=1}^{U}\frac{DCG_u@K}{IDCG_u@K}$$
$$DCG_u@K = \sum_{i=1}^{K}\frac{2^{\text{reli}}-1}{\log_2(i+1)} \quad (18)$$

where $IDCG_u@K$ represents the discount cumulative gain of the optimal TOP-K recommendation list, and $"reli"$ represents a measure of the correlation between the recommended result and the target.

3) Parameters and Environment

Experimental Environment: We implement the model by Tensorflow2 and deployed it on a Linux server equipped with an NVIDIA RTX 3090 GPU featuring 20 GB of video memory.

Parameter Settings: Recommendation module: The size of both course embedding vector D_e and the hidden layer D_h are configured as 16. The batch size is 256. The learning rate is 0.02. Profile reviser: The sampling times N is set to 3, the learning rate for the pre-training model and the joint training model is set to 0.001, 0.0005. For the policy network, the hidden layer (d^{h_1},d^{h_2}) is configured as 8, the state size (d^{h_2}) of the high-level layer is 18, and the state size(d^{l_2}) of the low-level layer is 34. The discount factor of reward β is 0.5. ACHRL model parameter: the sampling times N is also 3, and the hiding layer size of the recommendation module is configured as 16. For the value network, the hidden layer size (d^{h_1},d^{h_2}) is 20. The state (d^{h_2}) size of the high-level layer is 18, The state (d^{l_2}) size of the low-level layer is configured as 34

G. Comparison of experimental results

Table 2 demonstrates that our three proposed models achieved better results in terms of recommendation performance compared to the baseline models. For the MOOCCourse dataset, the HR was improved of 0.13% to 6.35% and the NDCG was improved of 0.59% to 5.34%. For the MOOCCube dataset, the HR was improved of 5.58% to 6.35% and the NDCG was improved of 1.48% to 2.12%. In summary, the above results provide a proof of the effectiveness of ACHRL model.

Table 2. Recommendation performance (%).

	MOOCCourse				MOOCCube			
	HR@5	**HR@10**	**NDCG@5**	**NDCG@10**	**HR@5**	**HR@10**	**NDCG@5**	**NDCG@10**
MLP	52.53	66.74	40.61	40.96	51.62	66.55	40.00	43.58
FISM	53.12	65.89	40.63	45.13	52.85	65.80	40.50	45.52
NeuMF	54.20	67.25	42.06	46.05	54.25	67.50	41.72	46.00
NARM	54.23	69.37	42.54	47.24	54.12	69.50	41.85	47.20
NAIS	56.05	68.98	43.58	47.69	56.02	69.53	43.50	47.23
HRL	59.84	75.00	44.50	50.95	58.45	72.05	44.87	49.28
HRRL	61.36	78.29	45.82	51.70	-	-	-	-
DARL	63.12	77.63	48.53	53.25	-	-	-	-
ACHRL_H	63.61	77.21	46.07	51.18	61.95	76.98	45.42	50.67
ACHRL_L	64.96	78.04	48.58	52.86	62.91	77.13	46.33	51.07
ACHRL	**66.19**	**78.42**	**49.84**	**53.84**	**64.03**	**78.40**	**46.35**	**51.40**

In Table 2, of all the baseline models, namely NeuMF, FISM, and MLP, they exhibit worst performance. This is primarily due to their inability to account for the contribution of various historical courses when making recommendations for target courses. NARM and NAIS underperform in comparison to all HRL-based models This performance gap arises from their limited

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

ability to differentiate the impact of historical courses when users are enrolled in a larger variety of courses. All models based on HRL surpassed the performance of other baseline models. The is because that the HRL-based models optimize the user profile. It makes the updated data represent the preferences of users more accurately and improves the accuracy of the recommendation. Comparing the performance of ACHRL model on the two datasets in Figure 5. It's clear that the ACHRL model performs better on the MOOCCourse dataset. Those users in MOOCCourse dataset enrolled in more courses. This proves that the ACHRL model has a good recommendation effect to those users who are interested in multiple courses. More courses help the model to train adequately and to removing noise courses accurately. This can recommend target courses to users more accurately.

In addition, to further verify the effectiveness of the ACHRL model, we compare it with the two newest models: HRRL and DARL in the past two years, that ACHRL model exhibits the highest performance. since the MOOCCube dataset of this thesis is slightly different from the MOOCCube dataset used by the two models mentioned above, the experimental data of the comparison is not given in the corresponding position in Table 2.

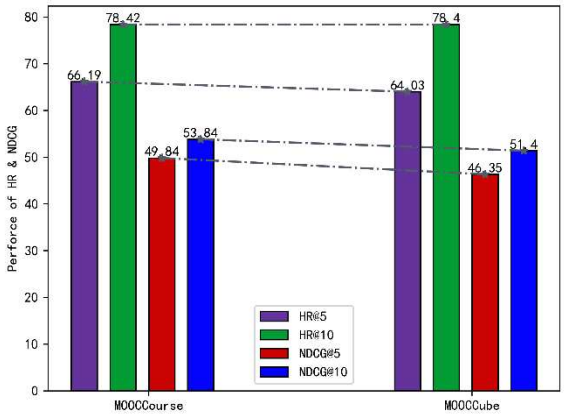

Figure 5. Recommendation performance of ACHRL model on MOOCCourse and MOOCCube(HR,NDCG).

H. Ablation experiment

Figures 6 and 7 illustrate the performance of HRL-based models concerning different top N values on two datasets. Within the category of HRL-based models, the ACHRL, ACHRL_H, and ACHRL_L models demonstrated superior performance in HR on two datasets. This proves the effectiveness of our proposed model.

The ACHRL_H model outperforms the HRL model. This because the high-level task modified the user profile more accurately after being optimized by the AC method. The ACHRL_L model also outperforms the HRL model because the noise courses in the course sequence are removed more accurately in the lower-level task. To sum up, The AC method optimizes two layers of tasks well.

As can be seen from the two figures, the ACHRL model attains the highest level of performance. This superiority is observed when compared to the other two variations of the method. Clearly, the combination of ACHRL_H and ACHRL_L Maximize the accuracy of decision making for high -lever and low-level tasks. The ACHRL model, which optimizes both the two layers of tasks of the model, improves the accuracy of the decision task, and achieves a more accurate course recommendation task.

Figures 8 and 9, showing the variation in reward values of ACHRL, ACHRL_H, ACHRL_L models during the course of the experiment. It can be seen from the change of reward value in the figures that the ACHRL model performed best, which further proves the effectiveness of the model for course recommendation.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

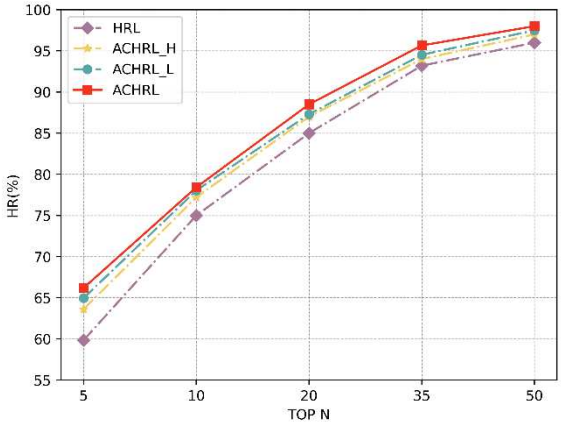

Figure 6. The performance of HRL-based models, assessed in terms of HR (%) at different top N HR values, on MOOCCourse.

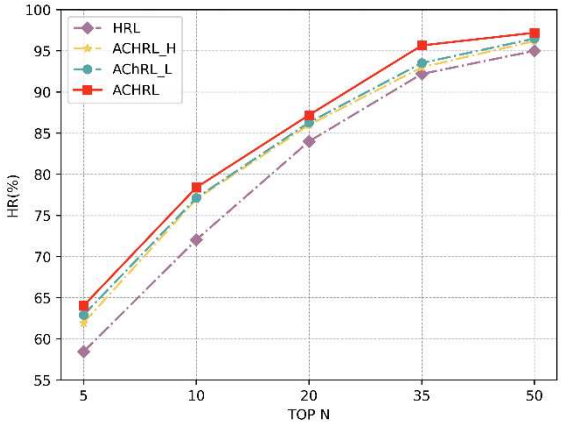

Figure 7. The performance of HRL-based models, assessed in terms of HR (%) at different top N HR values, on the MOOCCube.

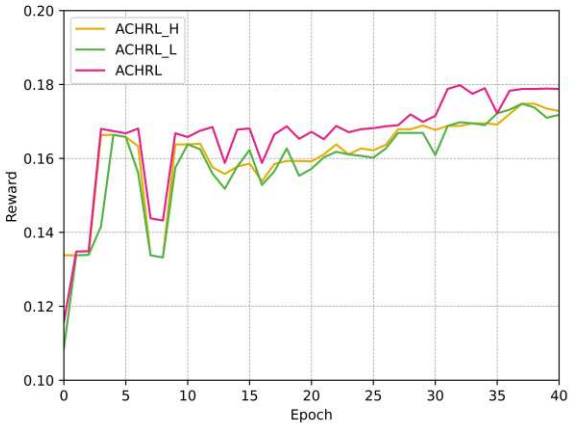

Figure 8. Variation of reward value with epoch of ACHRL, ACHRL_H, ACHRL_L models on MOOCCourse.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

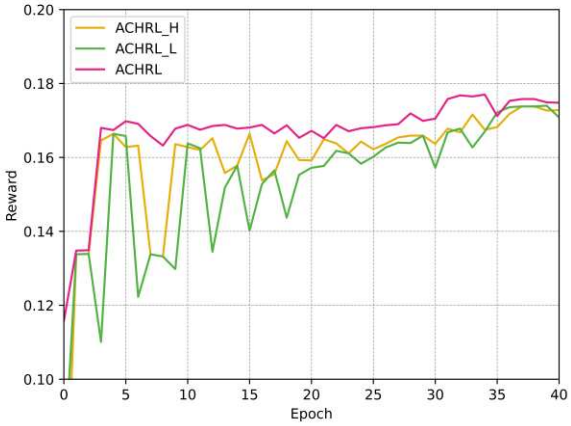

Figure 9. Variation of reward value with epoch of ACHRL, ACHRL_H, ACHRL_L models on MOOCCube.

I. Influence of hyper-parameters

we examine the impact of two crucial hyperparameters. (the size of the hidden layer of the attention mechanism network and the course embedding layer) on the model performance.

Figures 10 and 11 show the performance of the five models at various embedding layer sizes. In practice, The embedding layer size is specified as 8, 16, 32, 64, and 128. First of all, it can be seen from the two figures that The HRL-based model notably outperforms the NAIS model in HR. The ACHRL model achieves the best results on two datasets. The ACHRL model handles historical courses more accurately and improves recommendation performance. In addition, Our conclusion is that the recommendation performance of the five models in the experiment improves as the embedding layer size increases. This is because with an increase in the dimension of the embedding layer, the attention mechanism's capacity for representation is enhanced., and the model can provide more useful information for recommendation learning.

Figures 12 and 13 display the performance of the five models across various hidden layer sizes. In our experiment, we empirically chose 4, 8, 16, 32, and 64 as the hidden layer sizes, respectively. As shown in the two figures, the ACHRL model performed best relative to the other models on two datasets. As can be seen from the figures that our model is robust.

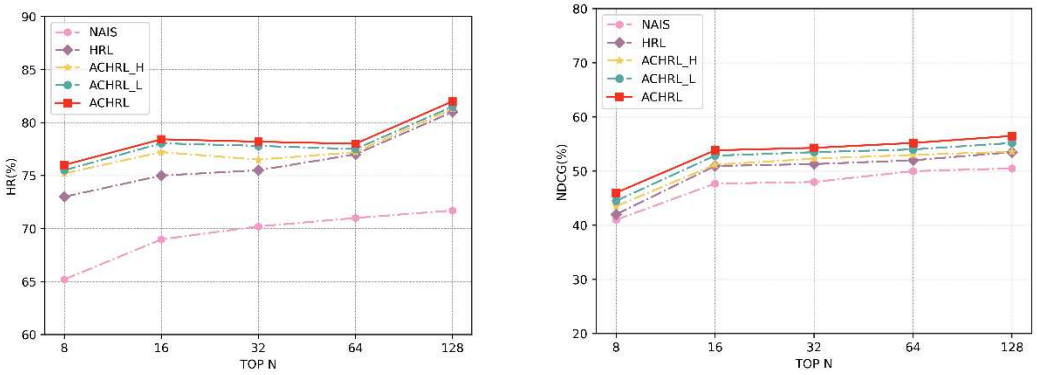

Figure 10. The performance of ACHRL, ACHRL_H, ACHRL_L, HRL, and NAIS, concerning different embedding size 8/16/32/64/128, are evaluated with respect to HR@10, and NDCG@10 on MOOCCourse.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

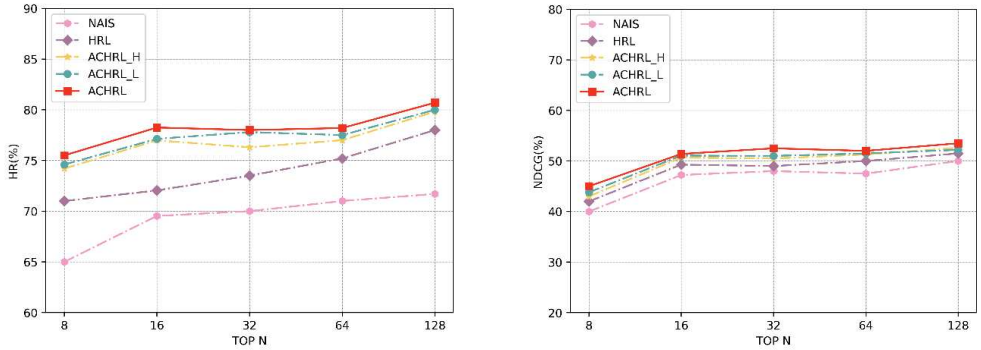

Figure 11. The performance of ACHRL, ACHRL_H, ACHRL_L, HRL, and NAIS, concerning different embedding size 8/16/32/64/128, are evaluated with respect to HR@10, and NDCG@10 on MOOCCube.

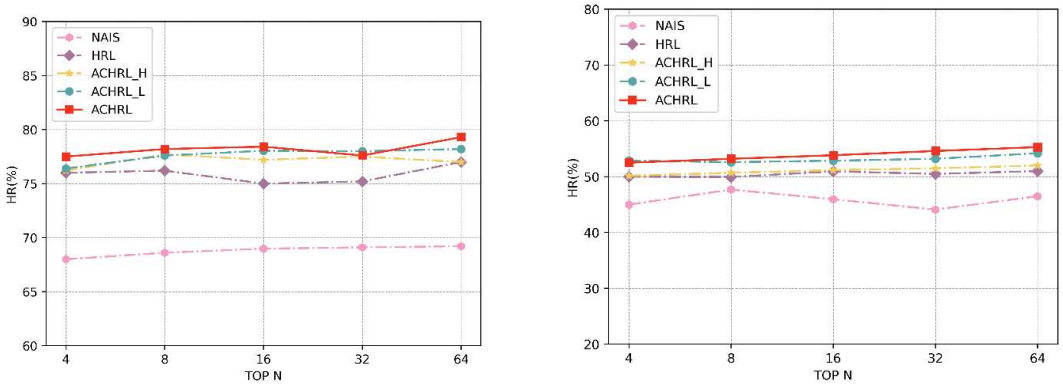

Figure 12. The performance of ACHRL, ACHRL_H, ACHRL_L, HRL, and NAIS, concerning different hidden layer sizes (4/8/16/32/64) are evaluated with respect to HR@10, and NDCG@10 on MOOCCourse.

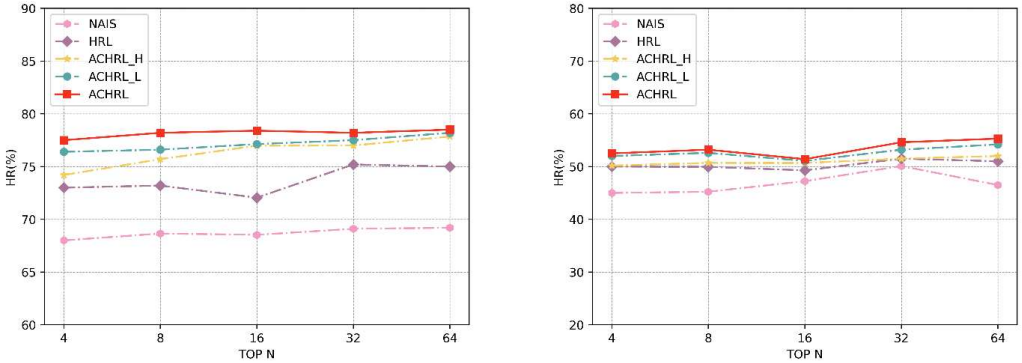

Figure 13. The performance of ACHRL, ACHRL_H, ACHRL_L, HRL, and NAIS, concerning different hidden layer sizes (4/8/16/32/64) are evaluated with respect to HR@10, and NDCG@10 on MOOCCube.

J. Performance analysis

To visually illustrate the effectiveness of the HRL model, we provide specific instances of course recommendations for qualitative analysis. Table 3 displays the performance of the ACHRL and HRL model. In the first case, the ACHRL model removes the noise course "Economics" accurately and recommends the target course correctly. In the second case the ACHRL model removes noise

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 October 2023 doi:10.20944/preprints202310.1367.v1

courses " Operating Systems ", and recommends the target course correctly. while the HRL model did not accurately remove the noise course.

This indicates the ACHRL model's capacity to effectively filter out irrelevant courses and offer recommendations that align better with the user's preferences.

Table 3. Two cases of the recommendation performance of ACHRL and HRL.

	Model	Performance	Recommended Result
(1)	ACHRL	Data Structure, Java, Assembly Language, Software Engineering	Software Engineering (√)
	HRL	Data Structure, Java, Economics, Data Structure, Software Engineering	Organic Chemistry (×)
(2)	ACHRL	Monetary and Financial Studies, Investment Studies, Corporate Finance	Principles of Economics (√)
	HRL	Operating Systems, Monetary and Financial Studies, Investment Studies	Software Engineering (×)

6. Conclusions

In this paper, we introduce the ACHRL model for course recommendation. We are the first to apply the AC method to hierarchical reinforcement learning model, reconstructing the user profile effectively. We applied the AC method to hierarchical tasks of profile reviser, improving the accuracy of action selection at each layer respectively. In addition, the use of the policy gradient method, which relies on temporal difference error, leads to an enhancement in the recommendation performance. This gradient method can be applied not just in episodic scenarios but also the continuing situations. This allows the model to be used in more scenarios. The model with good expansibility. For example, it can be applied to music, film, radio and other fields. The historical data formed by the historical information of user interaction can be modelled well by the model method in this paper. Through the construction of user profile, users can be recommended to their favorite items.

While the ACHRL model aligns with the primary objectives of many existing recommendation models, which prioritize recommendation accuracy, the field of recommendation embraces a multitude of user satisfaction metrics. Users often appreciate diverse offerings. For example: ranging from popular courses at different universities to those from various majors. Consequently, a recommendation model that strikes a balance between precision and variety can significantly enhance user satisfaction.

In our future research, we plan to explore the ACHRL model's performance across multiple evaluation metrics, particularly focusing on recommendation accuracy and diversity. To achieve this, we intend to employ a multi-objective evolutionary approach for optimizing the evaluation process [60]. Moreover, we will focus on the interpretability of the model. This can help users get the best recommended courses and at the same time understand the motivation of learning the target course. In terms of methods, we learned the Advantage Actor-Critic(A2C) method [61] is an improvement of the Actor-Critic method designed to improve training efficiency and stability, we will consider embedding A2C method into our model to ensure the accuracy of model recommendations.

Author Contributions: Conceptualization, K.L.; Methodology and Writing, G.Z.; Formal analysis, J.G.; Data curation, W.L. All authors have read and agreed to the published version of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated during the current study are available from the corresponding author on reasonable request.

Acknowledgments: This work is supported by the National Natural Science Foundation of China (No.62377036, No.61807024) and Science and Technology Program of Tianjin (No. 22YDTPJC00940).

Conflict of Interest: The authors declare that they have no conflicts of interest.

References

1. Saadatdoost, R., Sim, A. T. H., Jafarkarimi, H., & Mei Hee, J. (2015). Exploring MOOC from education and Information Systems perspectives: a short literature review. Educational Review, 67(4), 505-518.
2. Cheng, J., Yuen, A. H., & Chiu, D. K. (2022). Systematic review of MOOC research in mainland China. Library Hi Tech, (ahead-of-print).
3. Tucker, S. (2001). Distance education: Better, worse, or as good as traditional education. Online journal of distance learning administration, 4(4), 1-6.
4. He, L., Yang, N., Xu, L., Ping, F., Li, W., Sun, Q., ... & Zhang, H. (2021). Synchronous distance education vs traditional education for health science students: A systematic review and meta-analysis. Medical education, 55(3), 293-308.
5. Mazoue, J. G. (2013). The MOOC model: Challenging traditional education. EDUCAUSE review online, 28.
6. He, L., Yang, N., Xu, L., Ping, F., Li, W., Sun, Q., ... & Zhang, H. (2021). Synchronous distance education vs traditional education for health science students: A systematic review and meta-analysis. Medical education, 55(3), 293-308.
7. Atiaja, L. A., & Proenza, R. (2016). The MOOCs: origin, characterization, principal problems and challenges in Higher Education. Journal of e-learning and Knowledge Society, 12(1).
8. Laurillard, D. (2016). The educational problem that MOOCs could solve: professional development for teachers of disadvantaged students. Research in Learning Technology, 24.
9. Xu, M., Deng, J., & Zhao, T. (2020). On Status Quo, Problems, and Future Development of Translation and Interpreting MOOCs in China--A Mixed Methods Approach. Journal of Interactive Media in Education, 2020(1).
10. Parameswaran, A., Venetis, P., & Garcia-Molina, H. (2011). Recommendation systems with complex constraints: A course recommendation perspective. ACM Transactions on Information Systems (TOIS), 29(4), 1-33.
11. Zhang, H., Huang, T., Lv, Z., Liu, S., & Zhou, Z. (2018). MCRS: A course recommendation system for MOOCs. Multimedia Tools and Applications, 77, 7051-7069.
12. Jiang, W., Pardos, Z. A., & Wei, Q. (2019, March). Goal-based course recommendation. In Proceedings of the 9th international conference on learning analytics & knowledge (pp. 36-45).
13. Ma, B., Lu, M., Taniguchi, Y., & Konomi, S. I. (2021). CourseQ: the impact of visual and interactive course recommendation in university environments. Research and practice in technology enhanced learning, 16, 1-24.
14. Thanh-Nhan, H. L., Nguyen, H. H., & Thai-Nghe, N. (2016, October). Methods for building course recommendation systems. In 2016 Eighth international conference on knowledge and systems engineering (KSE) (pp. 163-168). IEEE.
15. Khalid, A., Lundqvist, K., & Yates, A. (2022). A literature review of implemented recommendation techniques used in Massive Open online Courses. Expert Systems with Applications, 187, 115926.
16. Zhang, J., Hao, B., Chen, B., Li, C., Chen, H., & Sun, J. (2019, July). Hierarchical reinforcement learning for course recommendation in MOOCs. In Proceedings of the AAAI conference on artificial intelligence (Vol. 33, No. 01, pp. 435-442).
17. Kabbur, S., Ning, X., & Karypis, G. (2013, August). Fism: factored item similarity models for top-n recommender systems. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 659-667).
18. He, X., He, Z., Song, J., Liu, Z., Jiang, Y. G., & Chua, T. S. (2018). Nais: Neural attentive item similarity model for recommendation. IEEE Transactions on Knowledge and Data Engineering, 30(12), 2354-2366.
19. Zhang, L., & Zhang, L. (2021). Top-N recommendation algorithm integrated neural network. Neural Computing and Applications, 33, 3881-3889.
20. Zhao, X., Zhang, Z., Bi, X., & Sun, Y. (2020). A new point-of-interest group recommendation method in location-based social networks. Neural Computing and Applications, 1-12.
21. Jiang, X., Sun, H., Zhang, B., He, L., & Jia, X. (2022). A novel meta-graph-based attention model for event recommendation. Neural Computing and Applications, 34(17), 14659-14682.
22. Liu, H., Wang, Y., Lin, H., Xu, B., & Zhao, N. (2022). Mitigating sensitive data exposure with adversarial learning for fairness recommendation systems. Neural Computing and Applications, 34(20), 18097-18111.
23. Ren, Y., Liang, K., Shang, Y., & Zhang, Y. (2023). MulOER-SAN: 2-layer multi-objective framework for exercise recommendation with self-attention networks. Knowledge-Based Systems, 260, 110117.
24. Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.
25. Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. Advances in neural information processing systems, 12.
26. O'Doherty, J. P., Dayan, P., Friston, K., Critchley, H., & Dolan, R. J. (2003). Temporal difference models and reward-related learning in the human brain. Neuron, 38(2), 329-337.

27. Ray, S., & Sharma, A. (2011). A collaborative filtering based approach for recommending elective courses. In Information Intelligence, Systems, Technology and Management: 5th International Conference, ICISTM 2011, Gurgaon, India, March 10-12, 2011. Proceedings 5 (pp. 330-339). Springer Berlin Heidelberg.
28. Li, J., & Ye, Z. (2020). Course recommendations in online education based on collaborative filtering recommendation algorithm. Complexity, 2020, 1-10.
29. Ghauth, K. I., & Abdullah, N. A. (2011). The effect of incorporating good learners' ratings in e-Learning content-based recommender System. Journal of Educational Technology & Society, 14(2), 248-257.
30. Xu, G., Jia, G., Shi, L., & Zhang, Z. (2021). Personalized course recommendation system fusing with knowledge graph and collaborative filtering. Computational Intelligence and Neuroscience, 2021, 1-8.
31. Emon, M. I., Shahiduzzaman, M., Rakib, M. R. H., Shathee, M. S. A., Saha, S., Kamran, M. N., & Fahim, J. H. (2021, August). Profile Based Course Recommendation System Using Association Rule Mining and Collaborative Filtering. In 2021 International Conference on Science & Contemporary Technologies (ICSCT) (pp. 1-5). IEEE.
32. Gulzar, Z., Leema, A. A., & Deepak, G. (2018). Pcrs: Personalized course recommender system based on hybrid approach. Procedia Computer Science, 125, 518-524.
33. Moerland, T. M., Broekens, J., Plaat, A., & Jonker, C. M. (2023). Model-based reinforcement learning: A survey. Foundations and Trends® in Machine Learning, 16(1), 1-118.
34. Rohde, D., Bonner, S., Dunlop, T., Vasile, F., & Karatzoglou, A. (2018). Recogym: A reinforcement learning environment for the problem of product recommendation in online advertising. arXiv preprint arXiv:1808.00720
35. Wang, X., Chen, W., Wu, J., Wang, Y. F., & Wang, W. Y. (2018). Video captioning via hierarchical reinforcement learning. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4213-4222).
36. Lanctot, M., Zambaldi, V., Gruslys, A., Lazaridou, A., Tuyls, K., Pérolat, J., ... & Graepel, T. (2017). A unified game-theoretic approach to multiagent reinforcement learning. Advances in neural information processing systems, 30.
37. François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018). An introduction to deep reinforcement learning. Foundations and Trends® in Machine Learning, 11(3-4), 219-354.
38. Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018, April). Deep reinforcement learning that matters. In Proceedings of the AAAI conference on artificial intelligence (Vol. 32, No. 1).
39. Li, S. E. (2023). Deep reinforcement learning. In Reinforcement Learning for Sequential Decision and Optimal Control (pp. 365-402). Singapore: Springer Nature Singapore.
40. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. nature, 518(7540), 529-533.
41. Wiering, M. A., & Van Otterlo, M. (2012). Reinforcement learning. Adaptation, learning, and optimization, 12(3), 729.
42. Mo, S., Pei, X., & Wu, C. (2021). Safe reinforcement learning for autonomous vehicle using monte carlo tree search. IEEE Transactions on Intelligent Transportation Systems, 23(7), 6766-6773.
43. Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., ... & Botvinick, M. (2016). Learning to reinforcement learn. arXiv preprint arXiv:1611.05763.
44. Pateria, S., Subagdja, B., Tan, A. H., & Quek, C. (2021). Hierarchical reinforcement learning: A comprehensive survey. ACM Computing Surveys (CSUR), 54(5), 1-35.
45. Nachum, O., Gu, S. S., Lee, H., & Levine, S. (2018). Data-efficient hierarchical reinforcement learning. Advances in neural information processing systems,31.
46. Botvinick, M. M. (2012). Hierarchical reinforcement learning and decision making. Current opinion in neurobiology, 22(6), 956-962.
47. Lin, Y., Lin, F., Yang, L., Zeng, W., Liu, Y., & Wu, P. (2022). Context-aware reinforcement learning for course recommendation. Applied Soft Computing, 125, 109189
48. Lin, Y., Feng, S., Lin, F., Zeng, W., Liu, Y., & Wu, P. (2021). Adaptive course recommendation in MOOCs. Knowledge-Based Systems, 224, 107085.
49. Nachum, O., Norouzi, M., Xu, K., & Schuurmans, D. (2017). Bridging the gap between value and policy based reinforcement learning. Advances in neural information processing systems, 30.
50. Howard, R. A. (1960). Dynamic programming and markov processes.
51. Garcia, F., & Rachelson, E. (2013). Markov decision processes. Markov Decision Processes in Artificial Intelligence, 1-38.
52. Frome, A., Corrado, G. S., Shlens, J., Bengio, S., Dean, J., Ranzato, M. A., & Mikolov, T. (2013). Devise: A deep visual-semantic embedding model. Advances in neural information processing systems, 26.
53. Lahitani, A. R., Permanasari, A. E., & Setiawan, N. A. (2016, April). Cosine similarity to determine similarity measure: Study case in online essay assessment. In 2016 4th International Conference on Cyber and IT Service Management (pp. 1-6). IEEE.

54. Taud, H., & Mas, J. F. (2018). Multilayer perceptron (MLP). Geomatic approaches for modeling land change scenarios, 451-455.
55. Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8, 229-256.
56. Poth, C., Pfeiffer, J., Rücklé, A., & Gurevych, I. (2021). What to pre-train on efficient intermediate task selection. arXiv preprint arXiv:2104.08247.
57. Liu, H., Yu, J., Chen, X., & Zhang, L. (2022). NeuMF: Predicting Anti-cancer Drug Response Through a Neural Matrix Factorization Model. Current Bioinformatics, 17(9), 835-847.
58. Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., & Ma, J. (2017, November). Neural attentive session-based recommendation. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (pp. 1419-1428).
59. Dalianis, H., & Dalianis, H. (2018). Evaluation metrics and evaluation. Clinical text mining: secondary use of electronic patient records, 45-53.
60. Von Lücken, C., Barán, B., & Brizuela, C. (2014). A survey on multi-objective evolutionary algorithms for many-objective problems. Computational optimization and applications, 58, 707-756.
61. Alibabaei, K., Gaspar, P. D., Assunção, E., Alirezazadeh, S., Lima, T. M., Soares, V. N., & Caldeira, J. M. (2022). Comparison of on-policy deep reinforcement learning A2C with off-policy DQN in irrigation optimization: A case study at a site in Portugal. Computers, 11(7), 104.