Hierarchical Reinforcement Learning on Multi-Channel Hypergraph Neural Network for Course Recommendation

Lu Jiang\(^1,4\), Yanan Xiao\(^1,4\), Xinxin Zhao\(^1,4\), Yuanbo Xu\(^2,5\), Shuli Hu\(^1,4\), Pengyang Wang\(^3,6\) and Minghao Yin\(^1,4\)*

\(^1\)School of Computer Science and Information Technology, Northeast Normal University, China
\(^2\)College of Computer Science and Technology, Jilin University, China
\(^3\)Department of Computer and Information Science, University of Macau, China
\(^4\)Key Laboratory of Applied Statistics of MOE, Northeast Normal University, China
\(^5\)Mobile Intelligent Computing (MIC) Lab, Jilin University, China
\(^6\)The State Key Laboratory of Internet of Things for Smart City, University of Macau, China

{jiangl761,xiaoyn117, zhaoxx767, husl903, ymh}@nenu.edu.cn, yuanbox@jlu.edu.cn, pywang@um.edu.mo

Abstract

With the widespread popularity of massive open online courses, personalized course recommendation has become increasingly important due to enhancing users’ learning efficiency. While achieving promising performances, current works suffering from the vary across the users and other MOOC entities. To address this problem, we propose Hierarchical reinforcement learning with a multi-channel Hypergraphs neural network for Course Recommendation (called HHCOR). Specifically, we first construct an online course hypergraph as the environment to capture the complex relationships and historical information by considering all entities. Then, we design a multi-channel propagation mechanism to aggregate embeddings in the online course hypergraph and extract user interest through an attention layer. Besides, we employ two-level decision-making: the low-level focuses on the rating courses, while the high-level integrates these considerations to finalize the decision. Finally, we conducted extensive experiments on two real-world datasets and the quantitative results have demonstrated the effectiveness of the proposed method.

1 Introduction

The prosperity of massive open online courses (MOOCs) is due to the rapid development of online education. The overwhelming and spotty learning materials in MOOC platforms undermine users’ efficiency. Against this background, accurately modeling user preference for learning materials offers valuable insights with course recommender system [Zhang et al., 2019]. The selection of the next course by users is influenced by the interplay between network interactions, which echo user needs and vary. Therefore, in this paper, we propose to develop an effective recommender system with hypergraph learning for course recommendation in MOOCs.

Prior literature in an online course recommendation method can be categorized into three aspects: (1) Collaborative filtering (CF) method [Yang and Cai, 2022] relies on user-item interaction data to predict course preferences; (2) Sequence-based method [Shao et al., 2021; Hou et al., 2018] uses the sequence of courses to recommend future learning paths; (3) Graph-based method [Wang et al., 2021; Xu et al., 2022] uses a complex network structure to model the relationship between users and courses. There are two main challenges: (1) the interactions among users are very complex and the relationships can be high-order; and (2) traditional recommendations cannot model real-time online study behavior in a continuously updated manner. Below we formally introduce each challenge and how we address them in our proposed framework.

First, graph neural network (GNN)-based [Wang et al., 2021] models have shown promising performance in course recommendation, due to the powerful capability in modeling relationships. A limitation of these GNN-based recommendation methods is that exploit the pairwise relations and ig-
nore the high-order relations among the entities. Although
which the long dependencies of relations are considered high-order,
only permit a maximum of two entities per relationship, as
shown in Figure 1(a). These heterogeneous graph structures
are unable to formulate complex high-order relations be-
more than two nodes. As shown in Figure 1(b), it is natu-
real to think that two users who are studying the same course
have a stronger relationship, we employ hypergraph to make
it connect more than two nodes, to model complex high-order
relations among users. We define the MOOC hypergraph to
organize the multiple to multiple relationships. We utilize hy-
peredges to mine high-order semantic information between
various types entity to form multiple channels. And incor-
porates an attention mechanism in the information transmis-
sion process to ensure semantic integrity during cross per-
spectives information propagation. By aggregating multiple
embeddings learned through multiple channels, we can ob-
tain comprehensive user representations that are considered
to contain multiple types of high-order relations.

Second, it is natural and promising to exploit reinfor-
cement learning, a real-time learning paradigm optimized with long-
term reward, to develop a course recommender system for
MOOCs. To achieve this goal, we reformulate the course rec-
ommendation problem in MOOC as a hierarchical reinforce-
ment learning task. HHCoR is built following the two-layer
decision-making process: (1) the low level focuses on the rat-
ing courses, and (2) the high level integrates these considera-
tions to finalize the decision. To facilitate our framework with
a proper environment, we propose a MOOC hypergraph to
organize the multi-channel semantics of study records. The
hyperedge embeddings from this MOOC hypergraph serve
as the state to support the decision-making process in our
method. In summary, we formulate the online course recom-
mendation problem as Hierarchical reinforcement learning
with multi-channel Hypergraphs neural network for Course
Recommendation (called HHCoR).

The main contributions are as follows:

- We reformulate the problem of personalized course rec-
  ommendation as a task based on hierarchical reinforce-
  ment learning.

- We construct a MOOC hypergraph, which effectively
  handles the heterogeneous nature of courses and utilizes
  an attention mechanism to capture user preferences from
  multi-channel semantics.

- We design a policy optimization framework based on hi-
  erarchical reinforcement learning and introduce reward
  function guidance mechanism to optimize the two-level
  agent’s policy.

- We validate our method on two real datasets and the
  results demonstrate the excellent performance of our
  method on the task of course recommendation.

2 Definitions and Problem Formulation

2.1 MOOC Hypergraph

In order to capture the complex relationships between the par-
ticipation of multiple entities on the MOOC platform, we pro-
pose to construct a hypergraph to represent historical records,
called MOOC Hypergraph. Formally, MOOC Hypergraph \( G \)
is defined as \( G = (V, E) \), where \( V \) and \( E \) represents the ver-
tex set and hyperedge set respectively. Each hyperedge \( e \in E \)
connects two or more vertices.

Vertices. MOOC hypergraphs aim to organize MOOC ele-
ments while preserving multi-aspect semantics. Specifically,
we categorize MOOC elements into three semantic channels,
including (1) the course channel, denoted as \( c \); (2) the con-
cept channel, denoted as \( k \); (3) the video channel, denoted
as \( v \). In this work, we consider three types of vertices corre-
sponding to three semantic channels. Then, the vertex set can
be denoted as \( V = c \cup k \cup v \).

Hyperedge. We define four types of hyperedges: (1) Course
hyperedge, which connects to all course nodes that the user
has been enrolled in; (2) Concept hyperedge, which connects
all learned concept nodes; (3) Video hyperedge, which con-
nects the video nodes that the user has watched; (4) Feature
hyperedge, connecting user, concept, and video nodes to each
other. We learn user perspectives from multiple sources, and
user perspectives consist of four types of hyperedge embed-
dings. We utilize the Parallel Aggregated Ordered Hyper-
graph [Valdivia et al., 2021] (PAOH) model to construct our
proposed MOOC hypergraph and hyperedges.

2.2 Problem Formulation

In this work, we formulate course recommendation as a
Markov Decision Process [Feinberg and Shwartz, 2012]
(MDP). Users decide which course to enroll in next based on
a history that reflects their personal preferences under a par-
ticular MOOC platform. The main components of the MDP
are defined as (1) States \( S \). Each state \( s \in S \) represents a spe-
cific user context derived from the MOOC platform history,
which is organized into a MOOC hypergraph. (2) Actions \( A \).
Each action \( a \in A \) corresponds to a potential next enrollment
course. (3) Transition Probabilities \( \Gamma \). \( \Gamma(s'|s, a) \) denotes
the probability of transitioning from state \( s \) to state \( s' \) when
action \( a \) is taken. This probability can be estimated from the
user’s platform history and reflects how often the user transi-
tions from one learning environment to another after selecting
a particular course. (4) Rewards \( R \). \( R(s, a, s') \) denotes the
reward received after transitioning from state \( s \) to state \( s' \) due
to action \( a \). The reward can be designed to reflect user satis-
faction or any other metric of interest. We will introduce the
reward design later. (5) Environment \( E \). The environment
consists of all participants of study events. It responds to the
user’s action by providing a new state and a reward. The en-
vironment’s dynamics are governed by the transition proba-
bilities \( \Gamma \) and the reward function \( R \). (6) Policy \( \pi \). A policy
\( \pi \) defines how users take action. Specifically, \( \pi(s) \) gives the
probability distribution over actions in state \( s \). The goal of
the MDP is to find an optimal policy \( \pi^* \) that maximizes the
expected cumulative reward over time.
In view of the course being studied the form of the MDP is recommended, our goal is to develop a hierarchical reinforcement learning framework to find the optimal policy $\pi^*$ that guides the user’s decision to register for the next course.

3 Method

In this section, we introduce the core architecture of our method HHCoR, including hypergraph representation learning, low-level policy, and high-level policy.

3.1 Framework Overview

The proposed HHCoR is illustrated in Figure 2. First, we learn the state of the environment by constructing a MOOC hypergraph, we propose a multi-channel aggregating mechanism to propagate various information among nodes in three channels. Then, we utilize the attention layer to extract the user preferences based on different hyperedges. After that, the low-level agents take the environment state as input, and the low-level agents model the multidimensional preference representation by analyzing the importance of each historical course to the target course. Finally, the high-level agents formulate a course recommendation policy by receiving learning insights from the low-level agents. The two-layer agents reinforce each other through iterative updates.

3.2 Hypergraph Representation Learning

Vertex Embedding. We denote the raw features of vertex $v_i \in V$ as $x_i \in \mathbb{R}^d$, and $N_i$ represents vertex $v_i$’s neighbors that are within the hyperedges. We employ the attention mechanism to capture the interrelationship between vertices and the respective neighbors in the same channel. Specifically, for the vertex $v_i$ and its neighbor $v_j (j \in N_i)$, the attention coefficient $\alpha_{ij}$ can be represented as

$$\alpha_{ij} = \frac{\exp(v_i, v_j)}{\sum_{j \in N_i} \exp(v_i, v_j)}. \quad (1)$$

Then, the embedding $h_i$ of the node $v_i$ can be represented by aggregate the neighbors’ define as

$$h_i = \sum_{j \in N_i} \alpha_{ij} v_i. \quad (2)$$

Hyperedge Embedding. In our study, we defined four types of hyperedges, including courses, videos, concepts, and features. Among them, course, video, and concept hyperedges are homogeneous (connecting vertices within the same semantic channel) and feature hyperedges are heterogeneous (connecting vertices across all semantic channels). For the homogeneous hyperedge $c_i \in \mathcal{E}$, we denote the hyperedge embedding by the set of all node embeddings within the hyperedge. The hyperedge embedding $q_i$ can be represented as

$$q_i = \sigma(\sum_{j \in |e_i|} h_j), \quad (3)$$

where $|e_i|$ denotes all the linked nodes in $e_i$.

The feature hyperedges serve as a bridge to link the semantics from different perspectives through the hypergraph topology. We then update the hyperedge embedding $q_i$ by
aggregating information from hyperedges on other perspectives that are interlinked by the same feature hyperedge:

$$q_t = \sigma \left( \sum_{k \in \Phi(e_t)} W_{\Phi(e_t)} Q_k \right),$$

(4)

Where \(\sigma\) is the sigmoid function [Elfwing et al., 2018], \(\Phi(e_t)\) denotes the query function that retrieves hyperedges from alternative perspectives that are interlinked by the same feature hyperedge as the given hyperedge, \(\Psi(\cdot)\) is the function to return the type of the given hyperedge, and \(W_{\Phi(e_t)}\) denotes the aggregation weights for the given the type \(\Psi(e_t)\).

### 3.3 Low-level Policy

In the initial phase of the HHCoR system, the low-level agent is responsible for meticulously rating historical courses, and this rating process is a key foundation for understanding and recognizing user decision-making patterns. Subsequent sections will detail the core components and operational mechanisms that make up low-level decision-making.

**State.** We use hyperedge embeddings as a representation of states. Specifically, for the low-level agent, states aim to capture the preferences and interactions of multiple aspects of the MOOC platform. Therefore, we connect relevant hyperedge embeddings to represent the state. Formally, let \(s^l\) denote the low-level agent state define as

$$s^l = \text{CONCATENATE}(q_c, q_h, q_v)$$

$$c = \Theta_c(u) \quad \text{and} \quad k = \Theta_k(u) \quad \text{and} \quad o = \Theta_o(u),$$

(5)

where \(\Theta_c(u), \Theta_k(u)\) and \(\Theta_o(u)\) denote the indexes of associated course hyperedge, concept hyperedge, and video hyperedge for the user \(u\), respectively.

**Low-Level Agent with DDPG.** In the HHCoR framework, the low-level agent comprises two parts: the ‘critic’, which assesses historical courses by computing the value function \(Q(s^l, a^l | \theta^Q)\) for each action, and the ‘actor’, which refines strategies based on these evaluations. This process involves scoring predictions to reflect the effectiveness of course actions, with the output—a weight between 0 and 1—indicating each course’s significance for user representation. The value function is defined as

$$Q(s^l, a^l | \theta^Q) \approx Q^*(s^l, a^l),$$

(6)

Where \(Q^*(s^l, a^l)\) represents the optimal action-value function. The critic network is trained by minimizing a defined loss function defined as

$$L(\theta^Q) = \mathbb{E}_{s^l, a^l, r, s^{l'}} \left[ (Q(s^l, a^l | \theta^Q) - y)^2 \right],$$

(7)

Where \(y = r + \gamma Q(s^{l'}, a^{l'} | \theta^Q)\) is the target value, \(\gamma\) denotes discount factor emphasizing the importance of future rewards and \(s^l\) and \(a^l\) represent the next state and action respectively.

In the actor component, another neural network is used to approximate the policy with parameters \(\theta^\mu\) defined as

$$\mu(s^l | \theta^\mu) \approx \pi^*(s^l),$$

(8)

where \(\pi^*(s^l)\) is the optimal policy.

The actor-network is trained by applying the policy gradient [Kakade, 2001] defined as

$$\nabla_{\theta^\mu} J \approx \mathbb{E}_{s^l, a^l} \left[ \nabla_{\theta^\mu} \mu(s^l, a^l | \theta^\mu) \nabla_{a^l} Q(s^l, a^l | \theta^Q) \right].$$

(9)

Then, to enhance the exploration capabilities of our model, we introduce the controllable stochasticity [Lapan, 2018] to promote exploration. Specifically, we use the Ornstein-Uhlenbeck [Lillicrap et al., 2016] process to generate temporally correlated noise.

**Low-level Reward Function.** The reward function for low-level agents is intended to guide learning. The reward \(r^l\) is computed as the change in correlation between the target predicted value and the real enrolled course before and after the action \(a^l\), defined as

$$r^l = Q(s^l, a^l | \theta^Q) - Q(s^l, a^l | \theta^Q),$$

(10)

where the agent’s action \(a^l\) outputs a probability ranging from 0 to 1, indicating the current course’s relevance to the user’s historical preferences.

If a low-level agent’s action \(a^l\) improves the relevance of a target course’s prediction, it earns a positive reward; otherwise, a negative reward is given for reduced relevance. This incentivizes the agent to adjust the importance weights of historical courses, enhancing predictive accuracy. Continuous interaction with the environment and corresponding rewards enable the agent to develop effective course rating strategies, thus aiding the decision-making of high-level agents.

### 3.4 High-level Policy

The high-level decision-making process employs a specialized agent to amalgamate insights garnered from lower-level agents, effectively merging these insights with platform factors within the MOOC hypergraph framework. This integration facilitates the formulation of a comprehensive course recommendation decision. This section delineates the principal components of the high-level agent and provides an overview of its operational workflow.

**State.** In order to encapsulate the low-level agent’s understanding of the user’s preference, the state of the high-level agent is defined by the updated low-level agent state defined as

$$s^h = \text{CONCATENATE}(q'_c, q'_h, q'_v)$$

$$c = \Theta'_c(u) \quad \text{and} \quad k = \Theta'_k(u) \quad \text{and} \quad o = \Theta'_o(u),$$

(11)

where \(q'_c, q'_h\) and \(q'_v\) denote the relevant course hyperedge, concept hyperedge, and video hyperedge embeddings of user \(u\) after the low-level agent update.

**High-Level Agent with REINFORCE.** The high-level agent implements the REINFORCE algorithm [Williams, 1992], utilizing feedback from the low-level agent and environmental data for prediction guidance. This agent adopts a stochastic policy \(\pi^h(s^h, a^h | \theta^\pi^h)\), with \(s^h\) and \(a^h\) denoting the state and action, respectively, aimed at forecasting the user’s next likely course selection. The policy parameters \(\theta^\pi^h\) are refined through gradient ascent define as

$$\nabla_{\theta^\pi^h} J \approx \mathbb{E}_{s^h, a^h} \left[ \nabla_{\theta^\pi^h} \log \pi^h(s^h, a^h | \theta^\pi^h) \right] \cdot (Q^h(s^h, a^h) - b(s^h)),$$

(12)
Where $Q^h(s^h, a^h)$ is the action-value function as estimated by the high-level agent and $b(s^h)$ is a baseline function for variance reduction. We adopt the mean of the action-value function as this baseline function. The high-level agent processes the output of the low-level agent along with the environmental information to make its decisions.

Exploring Deterministic and Stochastic Policies. We explore two policies for the high-level agent: a deterministic policy and a stochastic policy.

- **Stochastic Policy**: By employing the REINFORCE algorithm, Advanced Agents adopt a random strategy to introduce a certain degree of randomness in course selection. This approach facilitates deeper exploration of the course catalog to uncover hidden preferences or interests of users.

- **Deterministic Policy**: Conversely, we implement a deterministic policy for the high-level agent where it consistently recommends the same courses in response to specific user profiles or behaviors. This approach ensures stability and efficiency, focusing on optimizing user satisfaction with highly relevant courses, although it may limit the variety of courses explored.

High-level Reward Function. We developed a reward function $r^h$ for the high-level agent, aimed at enhancing its decision-making accuracy. This function comprises three components: (1) Concept similarity $r_k$ between the target and predicted courses, (2) Video content similarity $r_o$, between the target and predicted courses; and (3) The probability of recommending the target course $r_p$. The overall reward is a combination of these elements defined as

$$r^h = w_k \cdot r_k + w_o \cdot r_o + w_p \cdot r_p,$$

where $w_k$, $w_o$, and $w_p$ denote the weights for balancing the influence of $r_k$, $r_o$, $r_p$, respectively.

This weighting allows for fine-tuning of the recommendation process, ensuring that each aspect of the user’s preferences is appropriately considered, leading to highly personalized and effective course recommendations.

### Table 1: Overall Performance Comparison.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>MOOCube</th>
<th>MOOCourse</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR @5 @10 @20</td>
<td>NDCG @5 @10 @20</td>
<td>HR @5 @10 @20</td>
</tr>
<tr>
<td>FISM</td>
<td>0.1254 0.2001 0.3187</td>
<td>0.1336 0.1956 0.3123</td>
</tr>
<tr>
<td>MLP</td>
<td>0.1939 0.3006 0.4948</td>
<td>0.1951 0.2416 0.3139</td>
</tr>
<tr>
<td>NAIS</td>
<td>0.1194 0.1956 0.3123</td>
<td>0.1279 0.1856 0.2508</td>
</tr>
<tr>
<td>HRL</td>
<td>0.2580 0.4027 0.6116</td>
<td>0.2642 0.4042 0.5875</td>
</tr>
<tr>
<td>SR-GNN</td>
<td>0.0881 0.1360 0.2386</td>
<td>0.1279 0.1856 0.2508</td>
</tr>
<tr>
<td>LightGCN</td>
<td>0.1488 0.2024 0.3411</td>
<td>0.2046 0.2623 0.3392</td>
</tr>
<tr>
<td>COHIN</td>
<td>0.1272 0.1856 0.2508</td>
<td>0.1973 0.2416 0.3139</td>
</tr>
<tr>
<td>HHCoR</td>
<td>0.3477 0.5140 0.7420</td>
<td>0.3135 0.2241 0.2816</td>
</tr>
</tbody>
</table>

4 Experiment

In our study, we carried out a comprehensive series of experiments on two real-world MOOC datasets to address five key research questions:

- **Q1**: How is the performance of our proposed HHCoR in the course recommendation task?
- **Q2**: How does the MOOC hypergraph affect HHCoR recommendation performance?
- **Q3**: How does the MOOC hyperedge affect HHCoR recommendation performance?
- **Q4**: How do different components of the agent contribute to decision-making in our model?
- **Q5**: How do different reward designs impact course recommendation performance?

### 4.1 Experiment Settings

**Datasets.** We evaluate the model performance using two datasets: the MOOCube [Yu et al., 2020] and the MOOCourse [Zhang et al., 2019; Lin et al., 2022]. The samples in the training and test sets consist of a sequence of historical courses with the target course. For training, the last course in the sequence is the target course and the rest are history courses. Each positive sample corresponds to the construction of four negative samples that replace the target course. For testing, the course in the test set was used as the target and paired with 99 random negative samples.

**Baselines.** We compare our proposed method with the following baseline algorithms, including (1) FISM [Kabbur et al., 2013]; (2) MLP [He et al., 2017]; (3) NAIS [He et al., 2018]; (4) HRL [Zhang et al., 2019]; (5) SR-GNN [Wu et al., 2019]; (6) LightGCN [He et al., 2020]; (7) DHCN [Xia et al., 2021b]; (8) COTREC [Xia et al., 2021a] and (9) CoHIN [Zhang et al., 2022].

**Evaluation Metrics.** We evaluate the course recommendation accuracy in terms of the widely used metrics, including hit ratio (HR@N) and normalized discounted cumulative gain (NDCG@N). Evaluation was performed with $N = 5, 10, 20$. 

where $Q^h(s^h, a^h)$ is the action-value function as estimated by the high-level agent and $b(s^h)$ is a baseline function for variance reduction. We adopt the mean of the action-value function as this baseline function. The high-level agent processes the output of the low-level agent along with the environmental information to make its decisions.
Hyperparameter Settings. For the hypergraph representation, the dimensionality of the node embeddings was set to 64 and we utilized 8 attention heads in the attention mechanism. The DDPG agent and the REINFORCE agent were optimized with a discount factor ($\gamma$) set to 0.99. Both the agents employed Adam optimizers, with the learning rates set to 0.001.

4.2 Overall Performance (Q1)

In this section, we compare the overall performance of all models on real datasets. Overall, as Table 1 indicates, our model outperforms other baselines in HR and NDCG metrics. Compared with MLP representing node attributes, item-based collaborative filtering methods (FISM, NAIS), reinforcement learning-based methods (HRL), and graph neural network-based methods (SR-GNN, LightGCN, COTREC, DHCN, CoHHN), our proposed method incorporates course-related auxiliary information, which is more comprehensive and performs better in capturing users’ interests. Compared with item-based collaborative filtering methods and reinforcement learning-based methods, our proposed framework also considers heterogeneous hypergraph embeddings and high-order semantic relations between heterogeneous information. Compared to graph neural network-based methods, our proposed method analyzes the degree to which each historical course of a user represents that user’s interests. In conclusion, the results validate that our model is beneficial for course recommendation, which can help to better infer users’ interests and improve recommendation accuracy.

4.3 The Study of MOOC Hypergraph (Q2)

We conducted an experiment to verify the necessity of the hypergraph structure. In this experiment, we designed a variant of HHCoR, called (HHCoR'), which directly takes the user’s sequence as input without using the hypergraph structure. Beyond that, we replaced the hypergraph representation with other well-known graph representations such as Graph Convolutional Network (GCN) and Graph Attention Network (GAT). As shown in Figure 3, HHCoR exhibits a significant performance advantage. The superiority of HHCoR over its variants underscores the unique ability of hypergraph architectures to model complex relationships and higher-order interactions among data points, which standard graph models like GCN and GAT might miss.

4.4 The Study of MOOC Hyperedges (Q3)

In the MOOCCube dataset, we conducted experiments to assess hyperedge types’ impact, including the removal of concept ($e_1$), video ($e_2$), feature ($e_3$) hyperedges individually, and removing all except the course hyperedge ($e_4$). For the MOOCCourse dataset, experiments involved removing field ($e_1$) and feature ($e_2$) hyperedges, and a combined removal of field and feature ($e_3$). As shown in Figure 4, the performance of different hyperedge combinations varies, highlighting their importance in capturing the multi-semantics of users on MOOC platforms. HHCoR achieves optimal performance when it incorporates all types of hyperedges.

4.5 The Study of Agent Architecture (Q4)

The design of Low-level Agent. The results from HHCoR-L, where the low-level agent is omitted, indicate a marked reduction in the system’s capacity for precise user preference analysis, highlighting the agent’s integral role in processing course-related data. In the case of HHCoR-S, restricting the agent’s exploration scope leads to a diminished ability to generate innovative recommendations, crucial for adaptive learning. As Figure 5 shows, these outcomes not only validate the essential role of the low-level agent in the HHCoR framework but also underscore its contribution to the sophistication and reliability of the course recommendation process.

The design of High-level Agent. Our study examined the significance of the high-level agent in our hierarchical reinforcement learning model through two experiments: HHCoR-H, which omits the high-level agent’s explicit predictive function, and HHCoR-D, employing a deterministic policy for the high-level agent. These tests, results of which are depicted in Figure 6, aimed to assess the influence of the high-level agent’s predictive capacity and policy randomness.
4.6 The Study of Reward Design (Q5)

We consider combinations of weight settings for high-level agents and different reward functions to test the performance of HHCoR. The low-level reward is automatically learned and cannot be manually adjusted. Therefore, we only analyze the reward settings of the high-level agent. In our analysis, MOOCCube considers three components: $w_k$, $w_o$, and $w_p$, while MOOCourse involves two components: $w_t$ and $w_p$.

We mapped the performance of various combinations (where $w_k + w_o + w_p = 1$ for MOOCCube, and $w_t + w_p = 1$ for MOOCourse) onto 3D and 2D spaces, respectively. As shown in Figure 7, the better the performance, the darker the color.

5 Related Work

5.1 Personalized Course Recommendation

Personalized course recommendation has advanced from traditional content-based and collaborative filtering, which struggle with scalability and capturing dynamic user preferences, to more sophisticated machine learning techniques. These include matrix factorization, factorization machines, and deep learning methods like autoencoders and RNNs, which better handle complex user-course interactions. Studies like [Hou et al., 2018; Xu et al., 2024; Xu et al., 2022; Yang and Jiang, 2019] have made notable contributions, utilizing course clusters and combined user-course networks, respectively. Despite improvements, these methods still face challenges in adapting to the evolving and varied preferences of online learning users.

5.2 Graph-based Methods in Course Recommendation

Graph-based methods like GCN have been increasingly applied in personalized course recommendation to address its challenges. Studies like [Wang et al., 2021; Zhu et al., 2023a; Wang et al., 2022] effectively utilize these techniques for capturing intricate user-course relationships, with the latter viewing user embeddings as hyperedges in a learning hypergraph. Such methods excel in identifying complex, high-order relationships, a feat traditional methods often miss. However, they typically assume a homogeneous graph structure, which doesn’t align with the heterogeneous nature of MOOCs. To address this, [Fan et al., 2021; Xia et al., 2022] have explored the use of heterogeneous hypergraphs and hypergraph transformer networks, respectively, offering a more fitting solution for modeling the diverse and complex relationships prevalent in MOOC platform.

5.3 Reinforcement Learning in Course Recommendation

Reinforcement learning (RL) in course recommendation treats it as a sequential decision-making problem, adept at handling dynamic user behavior for optimized long-term suggestions. [Gong et al., 2022; Zhu et al., 2020; Zhu et al., 2023b] approached MOOC recommendations using RL, employing meta-paths on HIN and a heterogeneous graph attention network. Similarly, [Jiang et al., 2023] used a MOOC knowledge graph to guide interpretable recommendation paths. Traditional RL, however, struggles with large, complex action spaces typical in course recommendations, necessitating the use of Hierarchical Reinforcement Learning (HRL). [Xie et al., 2021; Zhang et al., 2024; Zhao et al., 2020] tackled this by dividing the recommendation process into multiple tasks, with agents operating at different abstraction levels, thereby effectively managing personalized and multi-objective recommendations.

6 Conclusion

In this paper, we study the problem of personalized course recommendation with a MOOC hypergraph and propose a hierarchical reinforcement learning framework for multi-channel hypergraph neural networks for online course recommendation. Specifically, we first formulate the MOOC personalized recommendation problem as a task based on hierarchical reinforcement learning. Secondly, we construct a MOOC hypergraph and propose to use the attention mechanism to extract the multi-channel semantics of MOOC entity relationships in different channels and capture user preferences. Third, we design a policy optimization framework based on hierarchical reinforcement learning and introduce a reward function guidance mechanism to optimize the two-level agent’s policy. Finally, we conduct extensive experiments on two real-world MOOC datasets to verify the effectiveness of our proposed method.
References


