Meta Multi-Agent Exercise Recommendation: A Game Application Perspective

Fei Liu
feiliu@mail.hfut.edu.cn
Hefei University of Technology
Hefei, Anhui, China

Xuegang Hu*
jsjxhuxg@hfut.edu.cn
Hefei University of Technology
Hefei, Anhui, China

Shuochen Liu
shuochenliu@mail.hfut.edu.cn
Hefei University of Technology
Hefei, Anhui, China

Chenyang Bu
chenyangbu@hfut.edu.cn
Hefei University of Technology
Hefei, Anhui, China

Le Wu*
lewu.ustc@gmail.com
Hefei University of Technology
Hefei, Anhui, China

ABSTRACT

Exercise recommendation is a fundamental and important task in the E-learning system, facilitating students' personalized learning. Most existing exercise recommendation algorithms design a scoring criterion (e.g., weakest mastery, lowest historical correctness) in conjunction with experience, and then recommend the recommended knowledge concepts (KCs). These algorithms rely entirely on the scoring criteria by treating exercise recommendations as a centralized system. However, it is a complex problem for the centralized system to choose a limited number of exercises in a period of time to consolidate and learn the KCs efficiently. Moreover, different groups of students (e.g., different countries, schools, or classes) have different solutions for the same group of KCs according to their own situations, in the spirit of competency-based instructing. Therefore, we propose *Meta Multi-Agent Exercise Recommendation (MMER)*. Specifically, we design the multi-agent exercise recommendation module, in which the KCs involved in exercises are considered agents with competition and cooperation among them. And the meta-training stage is designed to learn a robust recommendation module for new student groups. Extensive experiments on real-world datasets validate the satisfactory performance of the proposed model. Furthermore, the effectiveness of the multi-agent and meta-training part is demonstrated for the model in recommendation applications.

CCS CONCEPTS

• **Information systems** → **Information systems applications**;
• **Applied computing** → **Education**.

*Corresponding authors.

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08…$15.00
https://doi.org/10.1145/3580305.3599429

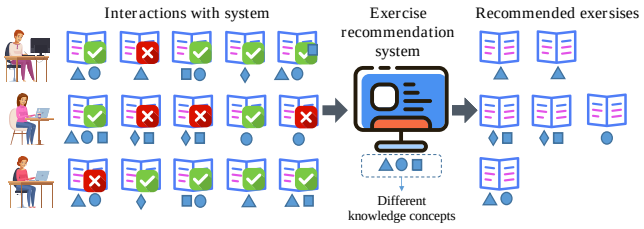

Figure 1: We show an example of an exercise recommendation system. Given users' historical interactions with exercises, as well as the annotated KCs (denoted as the blue icons in this figure), the systems recommends exercises to students that involve specific KCs.

KEYWORDS

educational data mining, exercise recommendation, multi agents, meta learning, reinforcement learning

ACM Reference Format:
Fei Liu, Xuegang Hu, Shuochen Liu, Chenyang Bu, and Le Wu. 2023. Meta Multi-Agent Exercise Recommendation: A Game Application Perspective. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3580305.3599429

1 INTRODUCTION

With the progressive enrichment of learning resources on the Internet, exercise recommendation [30] becomes a fundamental task in E-learning systems, facilitating students' personalized learning [15, 20]. By suggesting suitable exercises, students acquire knowledge instead of searching by themselves [27]. Through an open environment, students interact individually with the recommendation system agent to adapt their learning [7].

We show an example of an exercise recommendation system, presented in Fig. 1. The exercises involve different knowledge concepts (KCs). Given users' historical interactions with exercises, as well as the annotated KCs (denoted as the blue icons in this figure), the systems recommends exercises to students that involve specific KCs. Based on the above analysis, it can be found that the core of

the exercise recommendation is to learn an effective recommendation intelligence that enables users to achieve an efficient learning effect [16].

Most exercise recommendation models combine experience in the field of education to design a reasonable scoring criterion (e.g., weakest mastery of KCs, lowest correct answers to history exercises). Thus, they find the KCs where students are weak in learning and recommend exercises related to these KCs. These models can be categorized from both an educational psychology and data mining perspective [16]: 1) In a generic data mining perspective, collaborative filtering is a classical recommendation method that can also be applied to exercise recommendation systems [27]. 2) From the perspective of cognitive diagnosis in education [11], cognitive diagnostic models can predict students' performance in answering exercises involving these KCs by assessing their cognitive state. As a result, recommendations can be achieved based on the cognitive diagnostic model to discover the KCs that students do not master. Huang et al. [16] proposed a multi-objective exercise recommendation algorithm, arguing that the goals of review and exploration, as well as smoothness of exercise difficulty, also need to be considered when making recommendations.

However, most of the existing models face the limitation of local greedy optimality. The rational selection of a limited number of exercises to achieve efficient consolidation and learning of KCs within a period of time is a complex issue. For example, if each recommended exercise is associated with the KCs that are currently considered to be the least mastered, then after a period of time, the KCs that are better mastered will also be forgotten; on the whole, this period of learning is not efficient. Moreover, different groups of students (e.g., different countries, schools, or classes) have different learning solutions for the same group of KCs according to their own situations, in the spirit of competency-based instructing. These existing models require a large amount of training data about a certain group of students if they are to achieve good recommendation results for them. Their recommendations become less effective if this group of students lacks a large amount of historical data.

From the perspective of efficient learning of KCs for long-term gains, a decentralized multi-agent approach is better to find a good solution to this problem due to the complexity of the exercise recommendation problem. Therefore, we propose *Meta Multi-Agent Exercise Recommendation (MMER)*, in which the KCs are regarded as the agents. Specifically: 1) Each KC as an agent decides whether it needs to be recommended with reference to its current state of being learned and those of the others. They have a competitive and cooperative relationship: Each KC wants to make itself learn better and thus compete for the limited recommendation quota (competitive relationship). And it is through the cooperation between KCs that the overall learning situation is good (cooperative relationship). 2) Moreover, the meta-training stage is designed for various agents. In this way, our model can quickly achieve good results on new tasks with only a few shots (new group students' exercise recommendations) by learning the robust model with more tasks (different groups of students' exercise recommendations). It is worth noting that there are usually dozens or hundreds of KCs of a course. They act as a system of agents in multi-agent Reinforcement Learning (RL) when a large-scale game is applied since applications

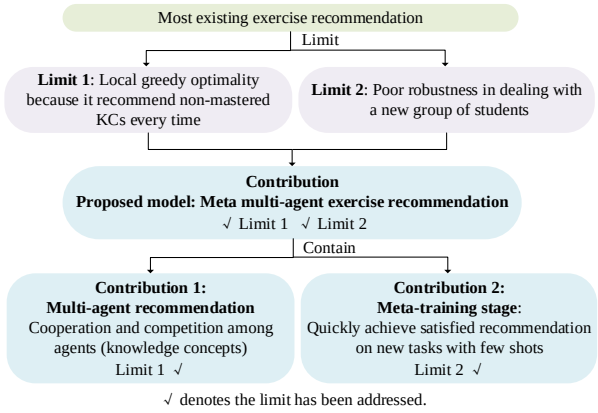

Figure 2: Relationship between limits of existing models and contribution of this study.

in this domain usually have only a few (at most two) agents. The main contributions of this study are summarized as follows:

- We propose MMER to achieve efficient learning of KCs for long-term gains. Dozens or hundreds of KCs are regarded as the agents, which is the application of a large-scale game.
- We design the meta-training stage for the various agents in the recommendation task. Thus our model can quickly achieve good results on new tasks with only few shots.
- Experiments on real-world datasets illustrate the superior performance of MMER, especially on recommendation for different student groups. In addition, the ablation experiments demonstrate the effectiveness of the modules in MMER.

2 RELATED WORK

In this section, related work is presented regarding knowledge tracing and recommendation systems in education.

2.1 Knowledge Tracing

Learning materials that are coherent with students' knowledge states are essential for E-learning systems to help students acquire more necessary KCs [4, 6, 33].

Knowledge tracing aims to model the dynamic cognitive states of students considering the time information of students' exercise logs [1, 9, 14]. As far as we know, BKT [8] is the first knowledge tracing model that uses hidden variables in the Hidden Markov Model to represent a student's cognitive state. DKT [26] and DKVMN [32] respectively use neural network technologies such as Recurrent Neural Networks and Key-Value Memory Networks to model knowledge tracing, improving the accuracy of knowledge tracing models. IEKT [21] optimizes students' knowledge states during the read and write stages. This process deep models individual cognition and acquisition for enhancing the accuracy of knowledge tracing. DSC_DKT [25] assigns students to various groups with similar ability at regular time intervals and then traces the cognitive states of students facing the student groups. AKT [13] used a monotonic attention mechanism that relates learners' future responses to their past responses to achieve better performance and explainability.

Other representative models improve tracing accuracy by exploring other rich information in response records. For example, FBKT [18] explore the fuzzy relation between students' cognitive states and exercise scores. EERNN [28] and EKT [19] use textual information in the knowledge tracing process. CT-NCM [23] is valuable work as it better models students' learning and forgetting processes through continuous-time information.

The above knowledge tracing model can track changes in a student's cognitive state over time and predict the student's score on exercises given their current state. Therefore, we can recommend exercises that the student may not have mastered according to the predictions of the knowledge tracing models.

2.2 Recommendation Systems in Education

Recommender systems are widely used in education [17], and there is a wide variety of recommended elements, including courses, learning resources, papers, programming problems, and universities [11]. The techniques used in these recommender systems mainly contain traditional collaborative filtering, knowledge-based, and machine learning-based approaches [17].

Most of the exercises recommended assessing the cognitive state of students in relation to their knowledge in the field of education. Thus the above cognitive diagnoses are one of the main supporting models for exercise recommendations. They tend to recommend exercises that are associated with KCs that students do not master. As the research progressed, researchers realized that recommending exercises based only on mastery or non-mastery may lead to the problem of recommending exercises that are too difficult for students to complete at all. Therefore, Huang et al. [16] proposed a multi-objective exercise recommendation algorithm, arguing that the goals of review and exploration, as well as smoothness of exercise difficulty, also need to be considered when making recommendations.

3 PROBLEM DEFINITION AND PRELIMINARIES

The problem definition of this study is presented. And then, mean-field multi-agent RL is introduced to solve the problem of the large-scale game, which is an RL process involving a larger number of agents.

3.1 Problem Definition

In E-learning systems, there are M groups of students with numbers $(Ns_1, Ns_2, ..., Ns_M)$. Each student group $m(m \in \{1, 2, ..., M\})$ has historical response logs, denoted as $\boldsymbol{L}_m = (\boldsymbol{L}_m^1, \boldsymbol{L}_m^2, ..., \boldsymbol{L}_m^{Ns_m})$. Each log sequence is denoted as $\boldsymbol{L}_m^i = (\langle k_1^i, c_1^i \rangle, \langle k_2^i, c_2^i \rangle, ..., \langle k_{T_i}^i, c_{T_i}^i \rangle)$, where T_i is the total response time steps of student i. It indicated that student i answered an exercise examining KC $k_{T_i}^i$ and the score is $c_{T_i}^i$ at time step $t(t \in \{1, 2, ..., T_i\})$. $c_{T_i}^i = 0$ if student i conducted a wrong answer and $c_{T_i}^i = 1$ if he conducted a right answer. Then, the problem we study in this paper is described as follows: Given response logs $\{\boldsymbol{L}_1, \boldsymbol{L}_2, ...\boldsymbol{L}_{M-1}\}$ of $(M-1)$ groups and few logs $\boldsymbol{L}_M$ of the new group M, our goal is to recommend exercises $\boldsymbol{e}_M = (\boldsymbol{e}_{M,1}, \boldsymbol{e}_{M,2}, ..., \boldsymbol{e}_{M,D})$ for the student group M, where D is the number of recommendations at different decision time steps.

3.2 Preliminaries

Preliminaries regarding RL and meta-learning are presented.

3.2.1 Reinforcement Learning. In RL [2, 29], the agent interacts with the environment and iteratively optimizes based on the reward it receives. Each state-action pair (s, a) corresponds to a Q-value $Q(s, a)$, based on which the action in a particular state is selected in the learning process. Q-matrix $\boldsymbol{Q}$ is updated according to Eq. (1). The goal of RL is to maximize the cumulative returns.

$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha \left[r + \gamma \max Q(s', a')\right], \tag{1}$$

where (s', a') is the state-action pair after state transition from state s when conducting action a. r is the reward that conducting action a at state s. α and γ are the learning and discount factors, respectively.

Multi-agent RL. Multi-agent RL [5] means that multiple agents interact with the environment at the same time. Each agent still follows the goal of RL. The difference is that the change in the global state of the environment is related to the joint action of all agents. Therefore, the impact of joint action needs to be considered during the update process of the Q-matrix, shown in Eq. (2).

$$Q(s_j, a_j, a_{-j}) \leftarrow (1-\alpha)Q(s_j, a_j, a_{-j}) + \alpha \left[r + \gamma \max Q(s'_j, a'_j, a'_{-j})\right], \tag{2}$$

where j and $-j$ represent the j-th agent and the other agents, respectively.

Mean field multi-agent RL. Most existing multi-agent RL methods are usually limited to situations where the number of agents is small. When the number of agents increases significantly, the learning process becomes difficult due to the large increase in dimensionality and the increased complexity of interactions between agents, e.g., the optimization process of $\max Q(s', a'_j, a'_{-j})$ in Eq. (2). To address the above problem, Yang et al. [31] proposed the mean field multi-agent RL. It considers only the interactions of each agent with its neighboring agents and redefines the Q-value function as Eq. (3) [31].

$$Q(s_j, a_j, a_{-j}) = \frac{1}{|N_j|} \sum_{k \in N_j} Q(s_j, a_j, a_k), \tag{3}$$

where N_j is the neighbor set of agent j and $|N_j|$ is the number of neighbor agents. This design still maintains global interaction between each pair of agents [3]. $Q_j(s, a_j, a_k)$ is estimated using the theory of mean fields as Eq. (4) and was derived through Taylor's theorem in [31].

$$Q(s_j, a_j, a_{-j}) = Q(s_j, a_j, \overline{a}_j), \tag{4}$$

where $\overline{a}_j = \frac{1}{N_j} \sum_{k \in N(j)} a_k$. Based on the above definition, the Q-value is updated as shown in Eq. (5) [31].

$$Q(s_j, a_j, \overline{a}_j) \leftarrow (1 - \alpha)Q(s_j, a_j, \overline{a}_j) + \alpha \left[r_j + \gamma v_j(s'_j)\right], \tag{5}$$

where α and γ are learning and discount factors, respectively. $v_j(s'_j)$ is the mean filed value function for agent j with state s'_j at the next time step.

3.2.2 Meta Learning. The essence of meta learning, learning to learn, is the tendency to learn knowledge and experience from existing tasks so that the model is not ignorant when faced with an unknown task [12]. Finn et al. [12] proposed model-agnostic meta-learning (MAML), which can be applied to most gradient-based models.

4 PROPOSED MODEL

In this section, the proposed model MMER is detailed. First, the overall architecture (Section 4.1) is introduced. Then, the three stages (Section 4.2) in the model and the exercise recommendation module (Section 4.3) in the stages are presented. Notation is described in Table 1.

Table 1: Notations used in this paper.

Notation	Description
M	Number of student groups (tasks)
N	Number of KCs (agents)
D	Total decision time steps
$\boldsymbol{L}_m$	Response logs of the m-th student group
$\boldsymbol{e}_m$	Recommended exercises for the m-th student group
$\boldsymbol{P}$	Parameters in the exercise recommendation module
$\mathcal{L}_m, \mathcal{G}_m$	Loss and gradient for the m-th task
MF	Mean-filed network
$RankNet$	Rank-score network
$Online\_Q$	Online Q-network
$Target\_Q$	Target Q-network
$\boldsymbol{Online\_a}$	Online action vectors
$\boldsymbol{Ramdom\_a}$	Random vectors
$\boldsymbol{s}$	State vectors
$\boldsymbol{a}$	Action vectors
$\boldsymbol{r}$	Reward vectors
$\overline{\boldsymbol{a}}$	Mean-field action vectors

4.1 Overall Architecture

The overall architecture of the proposed model is illustrated in Fig. 3. The model constructs a meta multi-agent exercise recommendation system with N agents for student groups. The pseudo-code of MMER in the model is presented in Algorithm 1.

As shown in Fig. 3-(a), it includes three stages, i.e., the meta-training, the fine-tuning, and the testing stages (detailed in Section 4.2). The meta-training stage is designed to optimize the parameters in the exercise recommendation module, which is detailed in Section 4.2. The multi-agent recommendation module is designed to model the recommended process for KCs based on their states, which is detailed in Section 4.3. Specifically, suppose there are M tasks (corresponding to the recommendation tasks for M student groups). In the meta-training stage, the response logs $\{\boldsymbol{L}_1, \boldsymbol{L}_2, ...\boldsymbol{L}_{M-1}\}$ of the former $(M-1)$ groups are the experience from existing tasks. Then, in the fine-tuning stage, facing a new student group M, the model is tuned through the few logs $\boldsymbol{L}_M$. Finally, in the testing stage, we recommend exercises $\boldsymbol{e}_M = (\boldsymbol{e}_{M,1}, \boldsymbol{e}_{M,2}, ..., \boldsymbol{e}_{M,D})$ for the group M (testing stage).

Algorithm 1 The proposed MMER model.

Input: Response logs $\{\boldsymbol{L}_1, \boldsymbol{L}_2, ...\boldsymbol{L}_{M-1}\}$; few logs $\{\boldsymbol{L}_M\}$; total decision time steps D;
Output: Recommended exercises $\boldsymbol{e}_M = (\boldsymbol{e}_{M,1}, \boldsymbol{e}_{M,2}, ..., \boldsymbol{e}_{M,D})$;
1: Initialize the exercise recommendation module;
2: —**Meta-training Stage**—
3: **while** $epoch \leqslant Epoch$ **do**
4: Optimize the learnable parameters in the exercise recommendation module using $\{\boldsymbol{L}_1, \boldsymbol{L}_2, ...\boldsymbol{L}_{M-1}\}$ (Algorithm 2);
5: **end while**
6: —**Fine-tuning Stage**—
7: Optimize the learnable parameters in the exercise recommendation module using $\{\boldsymbol{L}_M\}$;
8: —**Testing Stage**—
9: Output the recommended exercises $\boldsymbol{e}_M = (\boldsymbol{e}_{M,1}, \boldsymbol{e}_{M,2}, ..., \boldsymbol{e}_{M,D})$ for task M at D decision time steps (Algorithm 3).

Furthermore, the architecture of the exercise recommendation module in a task is shown in Fig. 3-(b) (detailed in Section 4.3). For each task $m, m \in \{1, 2, ..., M\}$, there are N agents (corresponding to N KCs involved in the exercises). They make action decisions based on their own state and other agents. The list $\boldsymbol{e}$ of recommended exercises for the task is obtained according to the joint actions of the agents. The task will give rewards (or penalties) to these agents for their actions. Specifically, each agent selects actions based on the policy that depend on the initial action, the state, and the mean-filed actions (the actions of the other agents).

4.2 Meta Optimization Module

The meta-training, the fine-tuning, and the testing stages are introduced. These three stages are for the optimization of the parameters in the exercise recommendation module.

4.2.1 Meta-training Stage. The meta-training stage is to pre-train the model through the existing response logs of the other student groups. In particular, different student groups may be served different learning programs or learning paths, so our model is not expected to fall into the local optimal solution of any other student group; instead, we expect to learn a robust model that adapts more quickly with new tasks. The robust population can be obtained by the design of this stage because it does not seek to achieve optimal on every task to preserve species diversity. There are the response logs $\{\boldsymbol{L}_1, \boldsymbol{L}_2, ...\boldsymbol{L}_{M-1}\}$ of the former $(M-1)$ groups are the experience from existing tasks.

Desired by the idea of MAML [12], in the meta-training stage, the logs $\boldsymbol{L}_m$ for each task m are split into the training set $\boldsymbol{L}_{mtr}$ and the validation set $\boldsymbol{L}_{mva}$. The learnable parameters in the exercise recommendation module $\boldsymbol{P}$ are copied to $\boldsymbol{P1}$ so that subsequent operations do not act directly on $\boldsymbol{P}$. First, $\boldsymbol{P1}$ is updated to $\boldsymbol{P}_2$ by the gradient $\mathcal{G}_{mtr}$, which is derived from the loss $\mathcal{L}_{mtr}$ obtained by performing the recommendation task on $\boldsymbol{L}_{mtr}$.

$$\boldsymbol{P}_2 \leftarrow \boldsymbol{P}_1 - lr * \mathcal{G}_{mtr}, \tag{6}$$

where lr is the learning rate in the optimization process.

Then, the loss $\mathcal{L}_{mva}$ is computed by performing the exercise recommendation task on $\boldsymbol{L}_{mtr}$ using the parameters $\boldsymbol{P}_2$. In this

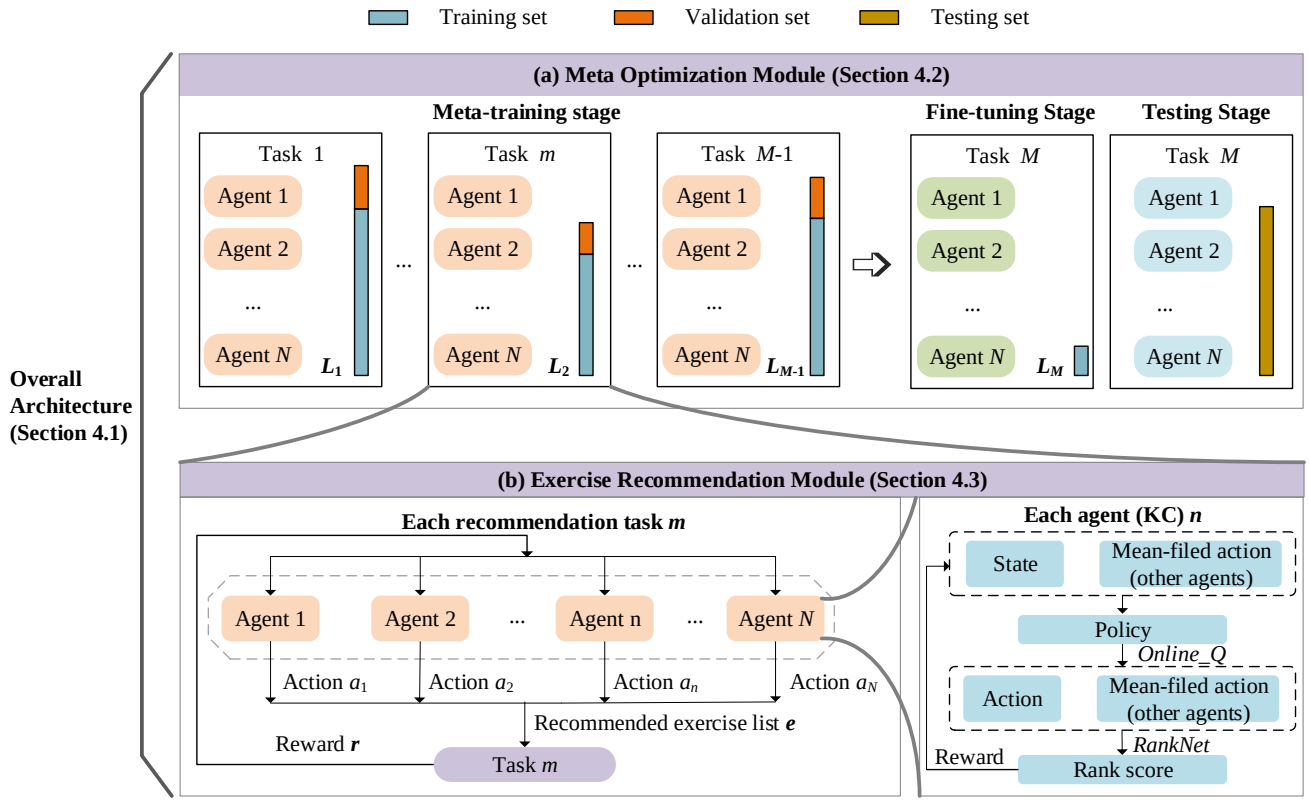

Figure 3: Overall architecture of the proposed model MMER. It contains the meta optimization module (detailed in Section 4.2) to optimize the exercise recommendation module (detailed in Section 4.3). Specifically, the exercise recommendation module is a multi-agent reinforcement learning system, regarding KCs as the agents for recommendation.

way, the parameters $\boldsymbol{P}$ in the exercise recommendation module are optimized by $\mathcal{G}_{mva}$, to keep the parameters from falling completely into fitting the local optimal solution of this task.

$$\boldsymbol{P} \leftarrow \boldsymbol{P} - lr * \mathcal{G}_{mva}. \tag{7}$$

The pseudo-code of the meta-training stage is presented in Algorithm 2 (Appendix A).

4.2.2 Fine-tuning Stage. After the meta-training stage, the parameters in the exercise recommendation module are updated from the existing $(M-1)$ tasks. The parameters learned in the meta-training stage are not fully adapted to the new task M because the tasks are different. However, there are only a few-shots $\boldsymbol{L}_M$ in the new task M. Therefore, in this stage, the population P needs to be fine-tuned.

$$\boldsymbol{P} \leftarrow \boldsymbol{P} - lr * \mathcal{G}_M, \tag{8}$$

where $\mathcal{G}_M$ is derived from the loss $\mathcal{L}_M$ obtained by performing the recommendation task on $\boldsymbol{L}_M$. The fine-tuning stage is designed to accommodate this new task, so the parameters of the process are optimized in line with the normal flow. The fine-tuning epoch can be set to 1, and it is also easy to think of multiple times, inspired by [12].

4.2.3 Testing Stage. In the testing stage, there is no response logs for the model. The optimized exercise recommendation module (obtained in the fine-tuning stage) is conducted for recommendation.

4.3 Exercise Recommendation Module

The exercise recommendation module works in all three stages (i.e., the meta-training, fine-tuning, and testing stages). It is formulated as a tuple $\langle \mathcal{S}, \mathcal{A}, MF, \mathcal{P}, \mathcal{R}, Online\_Q, target\_Q \rangle$, representing, in turn, the state space, action space, mean-filed-network, policy, reward, online Q-network, and target Q-network of the agents. The proposed module makes recommendations according to the competition and cooperation among KCs. Each KC's decision on actions depends on the states and joint actions of the other KCs to achieve competition and cooperation. In the module, the agents are designed to be homogeneous, that is, their above-mentioned factors are similarly designed. The pseudo-code of the exercise recommendation module is presented in Algorithm 3 (Appendix A).

4.3.1 State and Action Spaces. The state of the agents is described by the students' historical correctness in the exercises regarding the KCs. It is one of the bases for the agent's choice of action. $\mathcal{S} = \{S_1, S_2, ..., S_n, ..., S_N\}$ is the set of all agents' state spaces, where S_n is the state of the n-th agent and N is the number of agents (i.e.,

KCs). Each S_n is denoted as a continuous space. Denote the current state of agent n as $s_n \in S_n$. When an exercise in the recommendation list is answered, the state of the corresponding KCs tested in the exercise is updated as $s'_n = \beta s_n + (1-\beta) score_{ave}$, while the states of other KCs remain unchanged. $score_{ave}$ denotes the average score of exercises that test the KC. β is a hyper-parameter which is set to 0.5 in this paper.

The action of agents is described by the level to recommend exercises involving these KCs. $\mathcal{A} = \{A_1, A_2, ..., A_n, ..., A_N\}$ is the set of all agents' action spaces, where A_n is denoted as a continuous space. The simplest implementation is to set any A_n to contain two elements, which involves two actions: recommending and not recommending.

4.3.2 Mean-filed-network. The mean-filed-network MF is to obtain the mean-filed actions $\overline{\boldsymbol{a}} = (\overline{a}_1, \overline{a}_2, ..., \overline{a}_n, ..., \overline{a}_N)$, which is one of the bases for agent's choice of action. Each $\overline{a}_n$ is within a continuous space, denoting the mean actions of the other agents except the n-th agent. The mean-filed actions at the d-th decision time step are obtained by feeding those at the $(d-1)$-th decision time step into the MF network, shown in Eq. (9).

$$\overline{\boldsymbol{a}} \leftarrow MF(\overline{\boldsymbol{a}}), \tag{9}$$

where the MF consists of two fully-connected layers and a Sigmoid activation function.

4.3.3 Policy. The policy $\mathcal{P}$ of agents is that they select actions and finally, the resulting joint actions form a recommendation ranking. Moreover, stochastic exploration is conducted to avoid getting trapped in a local optimum when the agents select actions. The action selection satisfies Eq. (10).

$$\boldsymbol{a} = \varsigma * \boldsymbol{Random\_a} + (\neg\varsigma) * \boldsymbol{Online\_a}, \tag{10}$$

where $\boldsymbol{Random\_a}$ denotes the random action vector. $\boldsymbol{Online\_a}$ denotes the output action vector of the online Q-network, shown in Eq. (14). ς is an exploration flag, where any dimension is true if the random sample is not greater than the epsilon. ς is an exploration flag. If ς is True, the agents execute random actions; if ς is False, the agents execute online actions. $epsilon$ is the threshold for exploration. The generated random number $sample$ is compared with $epsilon$. If $sample < epsilon$, then $\varsigma = True$, else $\varsigma = False$.

Then, the rank score vector $\boldsymbol{rs} = \{rs_1, rs_2, ..., rs_{EK}\}$ of exercises is obtained by a network $RankNet$, where EK is the number of exercises in the list. The rank score rs_{ek} for each exercise is shown as Eq. (11).

$$rs_{ek} = RankNet(\boldsymbol{a}, \boldsymbol{k}_{ek}), ek \in \{1, 2, ..., EK\} \tag{11}$$

where $RankNet()$ consists of two fully-connected layers and a Sigmoid activation function. $\boldsymbol{a}$ is the action vector obtained in Eq. (10). $\boldsymbol{k}_{ek}$ is the vector of dimension N denoting the relation between the ek-th exercise and the KCs. N is the number of agents. The n-th dimension of $\boldsymbol{k}_{ek}$ is 1 represents the ek-th exercise is related to the n-th KC. Otherwise, vice versa. Then, the recommendation list of exercises $\boldsymbol{e}$ is obtained as Eq. (12).

$$\boldsymbol{e} = \arg top_K(\boldsymbol{rs}, K), \tag{12}$$

where K is the number of exercises to recommend.

4.3.4 Reward. After the recommended exercises are obtained in Eq. (12). The actions chosen by the agents are rewarded (or punished) by the student's answers to the recommended exercises. The reward $\boldsymbol{r} = \{r_1, r_2, ..., r_N\}$ for the agents is presented as Eq. (13).

$$r_n = \frac{\sum_{ek=1}^{EK} \varsigma_{ek} * d(rs_{ek}, ts_{ek})}{\sum_{ek=1}^{EK} \varsigma_{ek}}, n \in \{1, 2, ..., N\} \tag{13}$$

where $d(rs_{ek}, ts_{ek}) = \sqrt{(rs_{ek} - ts_{ek})^2}$ represents the distance between rs_{ek} and ts_{ek}. $\varsigma_{ek} = 1$ when the n-th dimension of $\boldsymbol{k}_{ek}$ is 1, i.e., the ek-th exercise is related to the n-th KC. rs_{ek} is computed in Eq. (11). $ts_{ek} \in \{0, 1\}$ is the true score of the recommended exercises of the student, representing the incorrect and correct answers, respectively. The higher the rank score of the exercise that the student answered incorrectly, the greater the reward for the KC related to that exercise. This is because the model wants to recommend KCs that students do not master as much as possible.

4.3.5 Online Q-network. The online Q-network is designed to obtain agents' actions based on their states and the joint action of the other agents. From the perspective of KCs, each KC wants to be learned better. Each KC decides whether it needs to be recommended or not by referring to its current state of being learned and the current state of other KCs being learned. Therefore, there is competition and cooperation among KCs: Each KC wants to be learned better and thus competes for a limited number of referrals (competition); Cooperation between KCs is the only way to achieve a good overall learning situation (cooperation). Therefore, the actions $\boldsymbol{Online\_a}$ is obtained according to the online Q-network, shown as Eq. (14).

$$\boldsymbol{Online\_a} = \arg\max(Online\_Q(\boldsymbol{s}, \overline{\boldsymbol{a}}), dim = 1) \tag{14}$$

where the $Online\_Q$ consists of two fully-connected layers and a Sigmoid activation function. $\boldsymbol{s}$ represents the current states of the agents. $\overline{\boldsymbol{a}}$ represents the joint actions, which is obtained in Eq. (9). Then, $\boldsymbol{pred}$ is calculated as Eq. (15).

$$\boldsymbol{pred} = \max(Online\_Q(\boldsymbol{s}, \overline{\boldsymbol{a}}), dim = 1) \tag{15}$$

4.3.6 Target Q-network. The target Q-network is designed to guide the online Q-network in the direction of optimization that maximizes future returns. Therefore, the target Q-network $\hat{Q}$ is shown as Eq. (16).

$$\boldsymbol{target} = \boldsymbol{r} + \gamma \max(Target\_Q(\boldsymbol{s}', \overline{\boldsymbol{a}}'), dim = 1), \tag{16}$$

where the structure of $Target\_Q$ is the same as that of $Online\_Q$. $\boldsymbol{s}'$ and $\overline{\boldsymbol{a}}'$ represent the next states and joint actions of the agents after they conduct $\boldsymbol{a}$, respectively. γ is the learning factor. $\boldsymbol{r}$ is the reward, shown in Eq. (13).

4.4 Optimization

The exercise recommendation module is optimized by the three stages, to minimize the loss $\mathcal{L}$ between $\boldsymbol{pred}$ and $\boldsymbol{target}$, i. e.,

$$\mathcal{L} = Loss(\boldsymbol{pred}, \boldsymbol{target}), \tag{17}$$

where $Loss$ represents the MSE_Loss function.

Table 2: Dataset information (Scenarios 1 and 2 indicate the training tasks and testing task are from different continents and the same continent, respectively).

Scenario	Training task 1	Training task 2	Testing task
Scenario 1	Asia	Europe	America
Scenario 2	Europe	Europe	Europe

5 EXPERIMENTS

Experiments on multiple real data sets are performed to validate the effectiveness of the proposed model in this section.

5.1 Setup

Setup is introduced including the datasets, baselines, metrics, and implementation details.

5.1.1 Datasets. The PISA 2015 (Programme for International Student Assessment) database[1] is a famous real-world database, containing the full set of responses to student groups in various areas in the world. The database contains the response logs from student groups in a total of 73 countries or regions. To validate the effectiveness of the proposed model, we have classified these regions according to the continent they belong to. Different continents are labeled as different datasets. For each continent, we randomly selected two of them as training tasks and one as a testing task. For each task, we selected 1000 student response logs, because the numbers of students for different tasks widely vary. The task information of the datasets is presented in Table 2. It is noticed that the regions are from different continents in the former dataset and the regions are from the same continent in the latter one. These two cases can verify the effect of whether the testing task belongs to the same continent as the training task on the recommendation performance of the models.

5.1.2 Baselines. The cognitive diagnosis models recommend exercises to students that involve the KCs they do not master by tracing their mastery status [24]. Using a cognitive diagnostic approach to complete the exercise recommendation task is the most common situation in practice [16]. At each decision time step, the baselines predict the future student's response score for exercises and execute the strategy that the smaller the predicted score, the more likely the exercise is to be recommended. This is fair for comparing the baselines with our model because the goal of baselines is to minimize the error between their predicted scores and their true scores; the smaller the error in baselines' predictions, the greater the hit rate of the exercises they are recommended. Since the students' response logs in the dataset are in temporal order, we compare our model with the knowledge tracing models [22]. Moreover, the representative exercise recommendation model was compared. The baselines are introduced as follows.

- BKT [8] first built a knowledge tracing model based on the hidden Markov model.
- DKT [26] first applied the recurrent neural network into the knowledge tracing field and performed satisfied prediction results.
- DKVMN [32] exploited the correlation between the KCs and students' cognitive levels through keys and values for knowledge tracing and performance prediction.
- DSC_DKT [25] clusters students to various groups with similar ability at regular time intervals and then traces the cognitive states of students facing the student groups.
- AKT [13] used a monotonic attention mechanism that relates learners' future responses to their past responses to achieve better performance and explainability.
- DRE [16] models the exercise recommendation task through a single-agent system. It considers the multi-objective optimization of review & exploration, difficulty, and engagement in the long-term learning process.

[1] https://www.oecd.org/pisa/data/

Table 3: Parameter settings

Parameters	Settings
Epoch	300
γ (Eq. (16))	0.85
Hidden size	200
Learning rate in optimization	0.005
Learning rate in innerstep	0.001
Epoch for innerstep	5
Recommendation interval	6
Validation ratio	0.5
Fine-tuning ratio	0.2
Fine-tuning epoch	2
Split way	student

5.1.3 Metrics. To evaluate the recommendation accuracy of the proposed model, three widely used metrics, NDCG, Recall, and F1, are used in the experiments. The larger the value of these metrics, the better the recommended effect. Due to the recommendation interval being set to 6, which means a recommendation decision is made every 6 time steps, we calculated Metrics@K (where K ranges from 1 to 5).

5.1.4 Implementation Details. The implementation details for the overall comparison are as follows. According to the three stages in Section 4.2, the training tasks are split into the training and validation sets; and the testing tasks are split into the fine-tuning and testing sets. There are two ways to split the datasets, i.e., by time steps and by students. In the overall comparison, the datasets are split by students (Section 5.2). And we also analyze the performance of the proposed model by the two split ways in the ablation studies (Section 5.3). The parameter settings are presented in Table 3.

5.2 Performance Comparison

The proposed MMER model is compared with the several knowledge tracing models. As for the compared models, they evaluate the cognitive states of students and thus predict their future performance of exercises. Therefore, the compared models can recommend exercises for students that they have not mastered.

Table 4: Comparison experiments for tasks (from different continents). MMER outperforms the baselines in most cases, especially when K is small. This is attributed to the meta-training module of the model.

Models	$K=1$			$K=2$			$K=3$			$K=4$			$K=5$		
	NDCG@1	Recall@1	F1@1	NDCG@2	Recall@2	F1@2	NDCG@3	Recall@3	F1@3	NDCG@4	Recall@4	F1@4	NDCG@5	Recall@5	F1@5
BKT [8]	0.611	0.611	0.758	0.743	0.612	0.763	0.776	0.612	0.759	0.792	0.607	0.755	0.799	0.605	0.754
DKT [26]	0.747	0.747	0.855	0.831	0.706	0.828	0.849	0.677	0.807	0.852	0.651	0.788	0.853	0.620	0.765
DKVMN [32]	0.749	0.749	0.857	0.837	0.713	0.832	0.857	0.675	0.806	0.860	0.644	0.783	0.859	0.622	0.767
DKT_DSC [25]	0.738	0.738	0.849	0.835	0.708	0.829	0.852	0.676	0.807	0.855	**0.652**	**0.789**	0.856	0.629	0.772
AKT [13]	0.735	0.735	0.847	0.831	0.707	0.828	0.851	**0.679**	**0.809**	0.857	0.650	0.788	0.855	0.629	0.772
DRE [16]	0.745	0.745	0.851	0.835	0.708	0.829	0.852	0.670	0.802	0.848	0.642	0.782	0.849	0.620	0.766
MMER	**0.755**	**0.755**	**0.860**	**0.844**	**0.714**	**0.833**	**0.861**	0.675	0.806	**0.864**	0.647	0.786	**0.860**	**0.630**	**0.773**

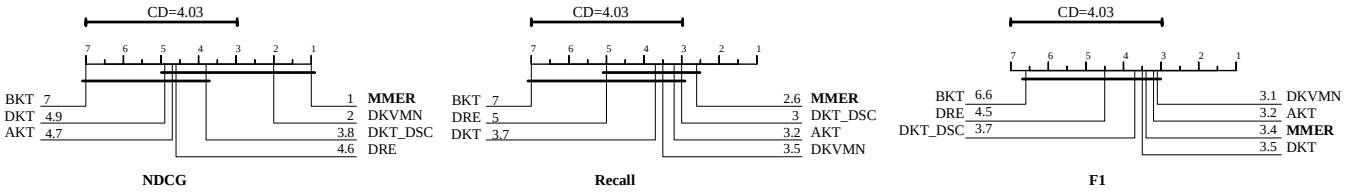

Figure 4: Nemenyi tests for the comparison in which tasks from different continents. It demonstrate that our model has a significant advantage in recommending exercises for the student groups from different regions.

Performance on Scenario 1. The overall comparison results conducted on Scenario 1 are presented in Tables 4. Since the recommendation interval is set to 5 (Table 3), we demonstrated the values of three metrics for the cases that $K = 1$ to 5. The higher the values, the better the metrics. As illustrated in Table 4, the proposed model outperforms the compared models in most cases, especially when K is small. This is attributed to the meta-training module of the model. The performance of these metrics on different real-world datasets indicates the effective recommendation performance of our recommendation algorithm. This demonstrates the superiority of the proposed algorithm on a new task (a new group of students).

The Nemenyi [10] test is conducted to present the comparison between MMER and the baselines. The Nemenyi test results are presented in Figs. 4. In the Nemenyi tests, it is considered that a significant difference exists if the average ranks of two models differ by at least one critical difference (CD), which is calculated using a 5% significance level. The smaller the ranking score, the better the model metric. In the scenario of tasks from different continents, MMER has a significant advantage in performing the task of recommending exercises for the student groups from different regions, especially for the NDCG and Recall metrics. Hence, the results of the Nemenyi test verify that MMER model can significantly optimize the exercise recommendation for different groups of students.

Performance on Scenario 2. According to the results in Table 5, MMER performs best in all metrics when $K = 1$, compared to all comparison models. As for the other cases, MMER can obtain comparable results compared with the state-of-the-art baselines. This illustrates that the proposed model is more effective than the comparison models when the number of recommended exercises becomes smaller.

5.3 Ablation Studies

The ablation studies are introduced to demonstrate the effectiveness of the modules in our model. As shown in Fig. 3, the meta-learning and multi-agent reinforcement learning part contribute to our model. Therefore, several versions are introduced in this section, as shown in Table 8. We divide the fine-tuning set according to both time (V1-V3) and student (V4-V6) divisions. Specifically, the full versions of the proposed model are V3 (split by time) and V6 (split by student). '✗' in 'meta learning' represents the models with the normal optimization process, without differentiating the gradients on the training and validation sets. '✗' in 'multi agents' represents the single agent reinforcement learning. The results of the ablation studies are illustrated in Tables 6 and 7.

5.3.1 Multi-agent Analysis. The effectiveness of the multi-agent part is demonstrated by comparing V1 to V2 and V4 to V5. The results of the three metrics with different values of K are shown in Tables 6 and 7. It is found that all the results of V2 and V5 are better than those of V1 and V6, respectively. This illustrates the feasibility and validity of the idea of considering KCs as multi-agents in recommendation applications.

5.3.2 Meta-training Analysis. The effectiveness of the multi-agent part is demonstrated by comparing V2 to V3 and V5 to V6. V2, V3, V5, and V6 are all versions of multi-agent exercises recommendation. The difference is that V2 and V5 do not use the meta-training stage in Fig 3. Instead, V2 and V5 use the regular network optimization process. As shown in Tables 6 and 7, it is found that all the

Table 5: Comparison experiments for tasks (from the same continent). MMER performs best when $K = 1$ compared to all the baselines, demonstrated its effectiveness in single-item recommendation. As for the other cases, MMER can obtain comparable results compared with the state-of-the-art baselines.

Models	$K = 1$			$K = 2$			$K = 3$			$K = 4$			$K = 5$		
	NDCG@1	Recall@1	F1@1	NDCG@2	Recall@2	F1@2	NDCG@3	Recall@3	F1@3	NDCG@4	Recall@4	F1@4	NDCG@5	Recall@5	F1@5
BKT [8]	0.650	0.651	0.788	0.785	0.662	0.796	0.812	0.654	0.791	0.822	0.651	0.789	0.826	0.652	0.790
DKT [26]	0.767	0.767	0.868	0.857	0.741	0.851	0.873	0.717	0.835	0.876	0.695	0.820	0.876	0.677	0.807
DKVMN [32]	0.779	0.779	0.876	**0.869**	0.754	0.860	0.884	0.724	0.840	**0.888**	0.700	0.824	**0.885**	0.679	0.809
DKT_DSC [25]	0.776	0.776	0.874	0.865	0.748	0.856	0.883	0.724	0.840	0.885	0.698	0.822	0.884	0.678	0.808
AKT [13]	0.780	0.780	0.876	0.860	**0.759**	**0.863**	**0.885**	**0.731**	**0.845**	0.886	**0.702**	**0.825**	0.883	**0.681**	**0.810**
DRE [16]	0.779	0.779	0.873	0.865	0.740	0.852	0.878	0.713	0.835	0.884	0.694	0.814	0.873	0.673	0.804
MMER	**0.783**	**0.783**	**0.878**	0.864	0.737	0.849	0.882	0.715	0.834	0.883	0.692	0.818	0.881	0.670	0.803

Table 6: Ablation experiments for various versions of MMER (split by time steps).

Versions	$K = 1$			$K = 2$			$K = 3$			$K = 4$			$K = 5$		
	NDCG@1	Recall@1	F1@1	NDCG@2	Recall@2	F1@2	NDCG@3	Recall@3	F1@3	NDCG@4	Recall@4	F1@4	NDCG@5	Recall@5	F1@5
V1	0.671	0.671	0.803	0.780	0.654	0.791	0.812	0.644	0.783	0.821	0.628	0.772	0.825	0.612	0.760
V2	0.692	0.692	0.818	0.792	0.666	0.799	0.823	0.653	0.790	0.832	0.638	0.779	0.834	0.615	0.762
V3 (**MMER**)	**0.725**	**0.725**	**0.840**	**0.822**	**0.696**	**0.821**	**0.844**	**0.667**	**0.800**	**0.851**	**0.642**	**0.782**	**0.852**	**0.623**	**0.768**

Table 7: Ablation experiments for various versions of MMER (split by students).

Versions	$K = 1$			$K = 2$			$K = 3$			$K = 4$			$K = 5$		
	NDCG@1	Recall@1	F1@1	NDCG@2	Recall@2	F1@2	NDCG@3	Recall@3	F1@3	NDCG@4	Recall@4	F1@4	NDCG@5	Recall@5	F1@5
V4	0.699	0.699	0.823	0.801	0.664	0.798	0.825	0.650	0.788	0.834	0.636	0.777	0.837	0.618	0.764
V5	0.740	0.740	0.841	0.813	0.691	0.817	0.838	0.666	0.799	0.845	0.644	0.784	0.847	0.620	0.766
V6 (**MMER**)	**0.755**	**0.755**	**0.860**	**0.844**	**0.714**	**0.833**	**0.861**	**0.675**	**0.806**	**0.864**	**0.647**	**0.786**	**0.860**	**0.629**	**0.772**

Table 8: Version description.

Versions	Meta-training	Multi-agents	Split way
V1	✗	✗	time step
V2	✗	✓	time step
V3	✓	✓	time step
V4	✗	✗	student
V5	✗	✓	student
V6	✓	✓	student

results of V3 and V6 are better than those of V2 and V5, respectively. Therefore, the meta-training stage is verified to be effective for optimizing the parameters of the exercise recommendation module in the face of exercise recommendation application scenarios with various student groups.

6 CONCLUSION AND FUTURE WORK

We propose *Meta Multi-Agent Exercise Recommendation (MMER)* to achieve efficient learning of KCs for long-term gains. Specifically, the KCs with competing and cooperative relationships are treated as multiple agents in the model; the meta-training stage is designed for the exercise recommendation module and thus it can quickly achieve good results on new tasks with only a few shots. Experiments on real-world datasets illustrate the superior performance of MMER. In addition, the ablation experiments demonstrate the effectiveness of meta-training and multi-agent parts in MMER. Avenues for future work include: 1) Incorporating rich features, such as difficulty of KCs, forgetting factor of students, text of exercises, to help agents make the decision; 2) Designing recommendation strategy that target a broader range of student groups, such as classrooms, schools, regions, and other similar scenarios.

ACKNOWLEDGMENTS

This work was supported in part by grants from the National Key Research and Development Program of China under grant 2021ZD0111802, the National Natural Science Foundation of China under grants 61806065, 62076085, 72188101, and 62120106008, the Fundamental Research Funds for the Central Universities under grant JZ2022HGTB0239, and the Open Fund of Infrared and Low Temperature Plasma Key Laboratory of Anhui Province, NUDT under grants IRKL2022KF06 and KYWX2023002. The authors also appreciate the partial support from Hefei AI Computing Center (Project Team).

REFERENCES

[1] Ghodai Abdelrahman, Qing Wang, and Bernardo Pereira Nunes. 2022. Knowledge tracing: A survey. *arXiv preprint arXiv:2201.06953* (2022). doi: 10.48550/arXiv.2201.06953.

[2] M Mehdi Afsar, Trafford Crump, and Behrouz Far. 2022. Reinforcement learning based recommender systems: A survey. *Comput. Surveys* 55, 7 (2022), 1–38.

[3] Lawrence E Blume. 1993. The statistical mechanics of strategic interaction. *Games and Economic Behavior* 5, 3 (1993), 387–424.

[4] Chenyang Bu, Fei Liu, Zhiyong Cao, Lei Li, Yuhong Zhang, Xuegang Hu, and Wenjian Luo. 2022. Cognitive diagnostic model made more practical by genetic algorithm. *IEEE Transactions on Emerging Topics in Computational Intelligence (TETCI)* 7, 2 (2022), 447–461. doi: 10.1109/TETCI.2022.3182692.

[5] Lorenzo Canese, Gian Carlo Cardarilli, Luca Di Nunzio, Rocco Fazzolari, Daniele Giardino, Marco Re, and Sergio Spanò. 2021. Multi-agent reinforcement learning: A review of challenges and applications. *Applied Sciences* 11, 11 (2021), 4948.

[6] Yunxiao Chen, Xiaoou Li, Jingchen Liu, and Zhiliang Ying. 2018. Recommendation system for adaptive learning. *Applied Psychological Measurement* 42, 1 (2018), 24–41.

[7] Yuying Chen, Qi Liu, Zhenya Huang, Le Wu, Enhong Chen, Run-ze Wu, Yu Su, and Guoping Hu. 2017. Tracking knowledge proficiency of students with educational priors. In *Proceedings of the ACM on Conference on Information and Knowledge Management (CIKM)*. ACM, 989–998.

[8] Albert T Corbett and John R Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction (UMUAI)* 4, 4 (1994), 253–278.

[9] Miao Dai, Jui-Long Hung, Xu Du, Hengtao Tang, and Hao Li. 2021. Knowledge tracing: A review of available technologies. *Journal of Educational Technology Development and Exchange (JETDE)* 14, 2 (2021), 1–19.

[10] Janez Demšar. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *The Journal of Machine Learning Research* 7 (2006), 1–30.

[11] LV DiBello, LA Roussos, and W Stout. 2007. Review of cognitively diagnostic assessment and a summary of psychometric models. 26 (2007), 970–1030.

[12] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of International Conference on Machine Learning (ICML)*. PMLR, 1126–1135.

[13] Aritra Ghosh, Neil Heffernan, and Andrew S Lan. 2020. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD)*. 2330–2339.

[14] Xuegang Hu, Fei Liu, and Chenyang Bu. 2020. Research advances on knowledge tracing models in educational big data. *Journal of Computer Research and Development* 57, 12 (2020), 2523–2546.

[15] Zhenya Huang, Qi Liu, Yuying Chen, Le Wu, Keli Xiao, Enhong Chen, Haiping Ma, and Guoping Hu. 2020. Learning or Forgetting? A Dynamic Approach for Tracking the Knowledge Proficiency of Students. *ACM Transactions on Information Systems (TOIS)* 38, 2, Article 19 (2020), 33 pages. https://doi.org/10.1145/3379507

[16] Zhenya Huang, Qi Liu, Chengxiang Zhai, Yu Yin, Enhong Chen, Weibo Gao, and Guoping Hu. 2019. Exploring multi-objective exercise recommendations in online education systems. In *Proceedings of ACM International Conference on Information and Knowledge Management (CIKM)*. 1261–1270.

[17] Shristi Shakya Khanal, PWC Prasad, Abeer Alsadoon, and Angelika Maag. 2020. A systematic review: machine learning based recommendation systems for e-learning. *Education and Information Technologies* 25, 4 (2020), 2635–2664.

[18] Fei Liu, Xuegang Hu, Chenyang Bu, and Kui Yu. 2022. Fuzzy Bayesian knowledge tracing. *IEEE Transactions on Fuzzy Systems (TFS)* 30, 7 (2022), 2412–2425. https://doi.org/10.1109/TFUZZ.2021.3083177

[19] Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, and Guoping Hu. 2019. EKT: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 33, 1 (2019), 100–115.

[20] Qi Liu, Shuanghong Shen, Zhenya Huang, Enhong Chen, and Yonghe Zheng. 2021. A survey of knowledge tracing. *arXiv preprint arXiv:2105.15106* (2021). doi: 10.48550/arXiv.2105.15106.

[21] Ting Long, Yunfei Liu, Jian Shen, Weinan Zhang, and Yong Yu. 2021. Tracing knowledge state with individual cognition and acquisition estimation. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. 173–182.

[22] Yu Lu, Deliang Wang, Qinggang Meng, and Penghe Chen. 2020. Towards interpretable deep learning models for knowledge tracing. In *Proceedings of the International Conference on Artificial Intelligence in Education (AIED)*. Springer, 185–190.

[23] Haiping Ma, Jingyuan Wang, Hengshu Zhu, Xin Xia, Haifeng Zhang, Xingyi Zhang, and Lei Zhang. 2022. Reconciling cognitive modeling with knowledge forgetting: A continuous time-aware neural network approach. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. 2174–2181.

[24] Haiping Ma, Jinwei Zhu, Shangshang Yang, Qi Liu, Haifeng Zhang, Xingyi Zhang, Yunbo Cao, and Xuemin Zhao. 2022. A prerequisite attention model for knowledge proficiency diagnosis of students. In *Proceedings of the ACM International Conference on Information & Knowledge Management (CIKM)*. ACM, 4304–4308.

[25] Sein Minn, Yi Yu, Michel C Desmarais, Feida Zhu, and Jill-Jênn Vie. 2018. Deep knowledge tracing and dynamic student classification for knowledge tracing. In *Proceddings of IEEE International Conference on Data Mining (ICDM)*. IEEE, 1182–1187.

[26] Chris Piech, Jonathan Spencer, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. In *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*. 505–513.

[27] Saman Shishehchi, Seyed Yashar Banihashem, Nor Azan Mat Zin, and Shahrul Azman Mohd Noah. 2011. Review of personalized recommendation techniques for learners in e-learning systems. In *Proceedings of International Conference on Semantic Technology and Information Retrieval*. IEEE, 277–281.

[28] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris Ding, Si Wei, and Guoping Hu. 2018. Exercise-enhanced sequential modeling for student performance prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, Vol. 32.

[29] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[30] María Cora Urdaneta-Ponte, Amaia Mendez-Zorrilla, and Ibon Oleagordia-Ruiz. 2021. Recommendation systems for education: systematic review. *Electronics* 10, 14 (2021), 1611.

[31] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean field multi-agent reinforcement learning. In *Proceedings of International Conference on Machine Learning (ICML)*. PMLR, 5571–5580.

[32] Jiani Zhang, Xingjian Shi, Irwin King, and Dit Yan Yeung. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of International Conference on World Wide Web (WWW)*. ACM, 765–774.

[33] Yuqiang Zhou, Qi Liu, Jinze Wu, Fei Wang, Zhenya Huang, Wei Tong, Hui Xiong, Enhong Chen, and Jianhui Ma. 2021. Modeling context-aware features for cognitive diagnosis in student learning. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery & Data Mining (SIGKDD)*. 2420–2428.

A PSEUDO-CODES FOR MODULES IN MMER

The pseudo-code of the meta-training stage and exercise recommendation module are presented in Algorithm 2 and Algorithm 3, respectively. The detials of these two modules are presented in Section 4.2 and Section 4.3.

Algorithm 2 Meta-training stage of MMER

Input: Response logs $\{\boldsymbol{L}_1, \boldsymbol{L}_2, ...\boldsymbol{L}_{M-1}\}$ for $M-1$ student groups;
Output: Updated $\boldsymbol{P}$;

```
P ← all the learnable parameters in the exercise recommendation module;
P_1 ← P, G = 0;
for each L_m ∈ {L_1, L_2, ...L_{M-1}} do
    Split L_m into the training and validation sets L_mtr, L_mva;
    for each innerstep do
        if innerstep starts then
            Compute the loss L_mtr and the gradient G_mtr through L_mtr using P_1
            (Algorithm 3);
            Obtain P_2 ← P_1 − lr_2 * G_mtr;
        end if
        Compute the loss L_mtr and the gradient G_mtr through L_mtr using P_2
        (Algorithm 3);
        Obtain P_2 ← P_2 − lr_2 * G_mtr;
        Compute the loss L_mva and the gradient G_mva through L_mva using P_2
        (Algorithm 3);
        if innerstep ends then
            G = G + G_mva;
        end if
    end for
end for
Update P ← P − lr_1 * G.
```

B EXPERIMENTS OF ONLINE EVALUATION

Online exercise recommendation is an important application to intelligent education systems. The systems recommend exercises

Table 9: Statistics of the used datasets.

Dataset	Training data				Fine-tuning data				Testing data			
	# Students	# Logs	Ave. steps	Correct rate	# Students	# Logs	Ave. steps	Correct rate	# Students	# Logs	Ave. steps	Correct rate
Scenario 1	2,000	58,401	29.200	0.508	200	5,145	25.725	0.413	800	21,002	26.253	0.398
Scenario 2	2,000	58,135	29.068	0.520	200	4,422	22.110	0.335	800	20,048	25.060	0.349
Scenario 1-all	12,434	359,055	28.877	0.517	1,204	31,530	26.188	0.399	4,820	127,365	26.424	0.398

Table 10: Comparison experiments for tasks from different continents (Scenario 1-all). MMER demonstrated its best performance when $K = 1$. Moreover, MMER outperforms baselines in NDCG metrics in all cases. In other cases, MMER can also exhibit near-optimal performance. It reflects the effectiveness of MMER.

Models	$K = 1$			$K = 2$			$K = 3$			$K = 4$			$K = 5$		
	NDCG@1	Recall@1	F1@1	NDCG@2	Recall@2	F1@2	NDCG@3	Recall@3	F1@3	NDCG@4	Recall@4	F1@4	NDCG@5	Recall@5	F1@5
BKT [8]	0.597	0.597	0.748	0.737	0.608	0.756	0.777	0.605	0.754	0.792	0.600	0.750	0.800	0.599	0.749
DKT [26]	0.731	0.731	0.845	0.830	0.706	0.828	0.849	0.674	0.805	0.854	0.641	0.781	0.854	0.618	0.764
DKVMN [32]	0.737	0.737	0.849	0.831	0.704	0.826	0.854	0.676	0.807	0.860	0.652	0.789	0.860	0.628	0.771
DKT_DSC [25]	0.744	0.744	0.853	0.837	0.706	0.828	0.856	0.669	0.802	0.858	0.645	0.784	0.857	0.624	0.769
AKT [13]	0.742	0.742	0.852	0.838	**0.713**	**0.832**	0.858	**0.683**	**0.812**	0.862	**0.655**	**0.792**	0.861	**0.631**	**0.773**
DRE [16]	0.745	0.745	0.854	0.837	0.701	0.824	0.855	0.667	0.800	0.857	0.643	0.783	0.856	0.624	0.768
MMER	**0.748**	**0.748**	**0.856**	**0.841**	0.710	0.830	**0.861**	0.681	0.810	**0.865**	0.652	0.789	**0.864**	0.628	0.771

Algorithm 3 Exercise Recommendation Module

Input: $\boldsymbol{P} = \langle \mathcal{S}, \mathcal{A}, MF, \mathcal{P}, \mathcal{R}, Online\_Q, Target\_Q \rangle, D$
Output: $\boldsymbol{e}_m, \mathcal{L}_m, \mathcal{G}_m$
1: Initialize $\boldsymbol{s}, \boldsymbol{a}$;
2: **for** $d = 1$ to $D, d++$ **do**
3: Update $\boldsymbol{a}$ according to $\boldsymbol{Online\_a}$ (Eq. (10));
4: Obtain $\boldsymbol{e}_{m,d}$ (Eq. (12)) and $\boldsymbol{r}$ (Eq. (13)) through $\boldsymbol{a}, \boldsymbol{L}_m$;
5: Update $\boldsymbol{s}'$ and $\overline{\boldsymbol{a}}'$;
6: Obtain $\boldsymbol{target_D}$ and $\boldsymbol{pred_D}$ using $\boldsymbol{r}, \boldsymbol{s}', \overline{\boldsymbol{a}}', Target\_Q$ and $\boldsymbol{s}, \overline{\boldsymbol{a}}, Online\_Q$, respectively;
7: **end for**
8: $\boldsymbol{e}_m = (\boldsymbol{e}_{m,1}, \boldsymbol{e}_{m,2}, ..., \boldsymbol{e}_{m,D})$;
9: $\boldsymbol{target} = (\boldsymbol{target}_1, \boldsymbol{target}_2, ..., \boldsymbol{target_D})$;
10: $\boldsymbol{pred} = (\boldsymbol{pred}_1, \boldsymbol{pred}_2, ..., \boldsymbol{pred_D})$;
11: Compute the loss $\mathcal{L}_m$ and then obtain the gradient $\mathcal{G}_m$ (Eq. (17)).

step by step for students such that we focus on the step-wise evaluation (i.e., hit@1) for the recommendation. To demonstrate the effectiveness of MMER in the online scenario, it was conducted by interacting with a simulator and receiving real-time rewards from students [16]. The simulator needs to be highly accurate in evaluating the students' cognitive states and predicting response performance. AKT [13] performs satisfied performance compared with the state-of-the-art knowledge tracing models. Therefore, we evaluate MMER in the online scenario by interacting with the AKT simulator. It is noticed that the selection of a simulator is not our concern in the experiment.

Specifically, we used 50% of the datasets to training the AKT model and 50% to conduct the online recommendation. We compared the proposed model with random recommendation, shown in Fig. 5. It demonstrated the effectiveness of online recommendation of the proposed model.

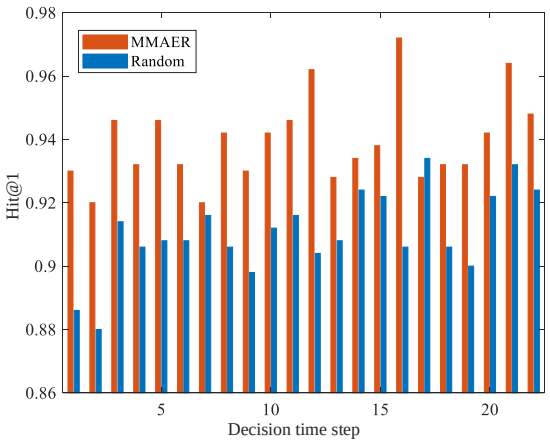

Figure 5: Hit results of online exercise recommendation.

C DISCUSSION

We present the scalability experiments on the number of students, as shown in Table 10. The statistic of the datasets is presented in Table 9. MMER demonstrated its best performance when $K = 1$. Moreover, MMER outperforms baselines in NDCG metrics in all cases. In other cases, MMER can also exhibit near-optimal performance. It reflects the effectiveness of MMER. The consistency between the results of MMER on Scenario 1-all (Table 10) and Scenario 1 (Table 4) demonstrates the scalability of the proposed model.

In addition, an increasing number of KCs can be achieved by expanding the feature dimensions of MMER. There are 74 KCs

in the PISA dataset which is used in our experiments. It is worth mentioning that the numbers of KCs in real-world datasets are not very large. Thus, MMER can meet the requirements for the number of KCs in the field of education data mining.

Received 2 February 2023; accepted 17 May 2023