Simulating Student Interactions with Two-stage Imitation Learning for Intelligent Educational Systems

Guanhao Zhao
School of Data Science, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
ghzhao0223@mail.ustc.edu.cn

Zhenya Huang*
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
huangzhy@ustc.edu.cn

Yan Zhuang
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
zykb@mail.ustc.edu.cn

Jiayu Liu
School of Data Science, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
jy251198@mail.ustc.edu.cn

Qi Liu
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
qiliuql@ustc.edu.cn

Zhiding Liu
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
doge@mail.ustc.edu.cn

Jinze Wu
iFLYTEK AI Research (Central China), iFLYTEK Co., Ltd
Hefei, Anhui, China
hxwjz@mail.ustc.edu.cn

Enhong Chen
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
cheneh@ustc.edu.cn

ABSTRACT

The fundamental task of intelligent educational systems is to offer adaptive learning services to students, such as exercise recommendations and computerized adaptive testing. However, optimizing required models in these systems would always encounter the collection difficulty of high-quality interaction data in practice. Therefore, establishing a student simulator is of great value since it can generate valid interactions to help optimize models. Existing advances have achieved success but generally suffer from exposure bias and overlook long-term intentions. To tackle these problems, we propose a novel *D*irect-*A*dversarial *I*mitation Student *Sim*ulator (*DAISim*) by formulating it as a Markov Decision Process (MDP), which unifies the workflow of the simulator in training and generating to alleviate the exposure bias and single-step optimization problems. To construct the intentions underlying the complex student interactions, we first propose a direct imitation strategy to mimic the interactions with a simple reward function. Then, we propose an adversarial imitation strategy to learn a rational distribution with the reward given by a parameterized discriminator. Furthermore, we optimize the discriminator in adversarial imitation in a pairwise manner, and the theoretical analysis shows that the pairwise discriminator would improve the generation quality. We conduct extensive experiments on real-world datasets, where the results demonstrate that our *DAISim* can simulate high-quality student interactions whose distribution is close to real distribution and can promote several downstream services.

*Corresponding Author.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3615060

CCS CONCEPTS

• **Computing methodologies → Adversarial learning**; • **Applied computing → E-learning**.

KEYWORDS

Adaptive Learning, Student Simulator, Imitation Learning

ACM Reference Format:
Guanhao Zhao, Zhenya Huang, Yan Zhuang, Jiayu Liu, Qi Liu, Zhiding Liu, Jinze Wu, and Enhong Chen. 2023. Simulating Student Interactions with Two-stage Imitation Learning for Intelligent Educational Systems. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham,*

United Kingdom. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3583780.3615060

1 INTRODUCTION

As the landscape of education is evolving significantly, intelligent educational systems (e.g., MOOC and Coursera [16, 41, 45]) have attracted many students. These systems provide several adaptive learning services, including personalized recommendations, knowledge tracing (KT), and computerized adaptive testing (CAT) [17, 26, 29, 47]. Figure 1 (left) shows a typical learning interaction procedure between a student and a system. Generally, the system poses a learning item (i.e., exercise) at each step. After receiving feedback from the student (e.g., right or wrong), the system provides a new item according to the student's personalized needs.

The systems predominantly depend on machine learning algorithms that utilize extensive interaction data, either offline or online [17, 25], to optimize their models and provide the aforementioned services. Despite their effectiveness, there exist main drawbacks of such algorithms: On the one hand, offline-trained systems would be hindered by the bias in the collected data [11]. On the other hand, online systems may offer students inappropriate learning items during the trial-and-error process, resulting time wasting, unfairness [23] and high cognitive load [9], etc. To address these problems, a viable approach is to employ a simulator imitating real students and interacting with the system, as shown in the left part of Figure 1. The primary goal of the simulator is to generate high-quality interactions where the notion of high-quality generated data refers to the data whose distribution is equivalent to the real distribution, as shown in the right part of Figure 1. By utilizing the simulator, the system can undergo initial optimization with sufficient training before real students become involved, thereby offering students better adaptive learning services [36].

In the literature, researchers have dedicated their efforts to designing ideal student simulators ranging from the earlier manually crafted rule-based simulators [27, 33] to recent deep-learning-based advances [17, 27]. Though they have achieved some successes, there are still some limitations. First, existing simulators generally encounter exposure bias [2] because of the inconsistency between training and testing. Second, most simulators straightforwardly treat the learning interaction generation task as a single-step prediction task (i.e., KT task [17, 27, 31]), which generally overlooks the long-term intention determining how students interact with the systems. Overall, both problems would hinder generation quality.

In this paper, we tackle the above problems in a principled way by proposing a novel student simulator framework, namely *D*irect-*A*dversarial *I*mitation Student *Sim*ulator (*DAISim*). Specifically, we reformulate the interaction process as a sequential decision-making task and formalize it as a Markov Decision Process (MDP) to alleviate exposure bias and optimize the long-term intention of students within a reinforcement learning (RL) paradigm.

However, students' interactions are complex, and the underlying intention of interactions is implicit and still needs to be apprehended, making it infeasible to manually define the reward function of MDP. To this end, we reconstruct two reward functions in two-stage imitation, including direct and adversarial imitation with inspiration from imitation learning [18], to model intentions and

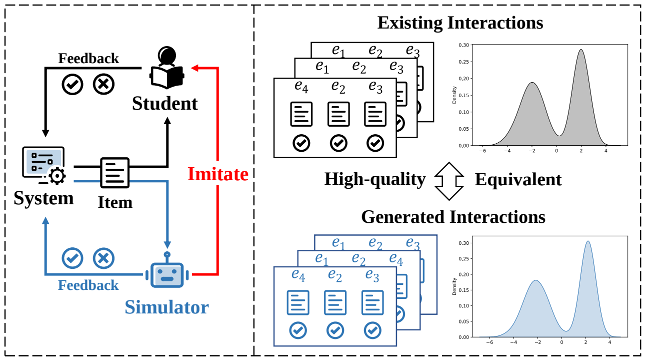


Figure 1: Left: Interaction process of student/simulator. Right: High-quality data refers to generated interactions equivalent to real interactions in distribution.

guide generating high-quality interactions. In the direct imitation stage, we propose direct imitation with a simple reward inspired by imitation learning by reinforcement learning [10, 32, 38]. Expressly, we force the generated interactions to recover the real distribution. Nevertheless, real interactions are sometime sparse [11], causing insufficiently rational and diverse interaction generation in direct imitation. Thus, in the second stage, we propose an adversarial imitation strategy to overcome the data sparsity problem. Specifically, our simulator first generates interactions beyond real interactions to explore diverse patterns further. Then, the simulator updates with the reward from a parameterized discriminator. However, the traditional classification discriminator would provide inappropriate rewards and hinder the generated distribution of the simulator from improving. Because its discriminator only tries to distinguish between the generated and the real interactions rather than explicitly considering the extent to which the generated interactions differ from the real ones [19]. We address this by introducing a pairwise discriminator, which focuses on the degree of the difference between the generated and real interactions to improve generation quality. Our theoretical analysis demonstrates that the objective of the simulator in adversarial imitation is minimizing a divergence which would be more sensitive to the difference between distributions than the conventional discriminator.

In summary, we highlight the main contributions as follows:

- To the best of our knowledge, *DAISim* is the first framework to formulate the task of building the student simulator as MDP, further addressing the exposure bias and short-term objective problems in previous advances.
- We tackle technical challenges such as implicit intentions underlying interactions, data sparsity, and inappropriate rewards with direct and pairwise adversarial imitation. Through theoretical analysis, we demonstrate that the simulator with the pairwise discriminator in adversarial imitation minimizes a more sensitive divergence.
- We conduct extensive experiments to evaluate the proposed framework on two real-world datasets. Quantification distance measuring between distributions and downstream adaptive learning services improvement demonstrate that *DAISim* enhances these services using its generated data.

While our proposed framework is initially motivated by the need for a student simulator in the context of adaptive learning, it is essential to note that the fundamental idea is generalizable and applicable to other domains involving user simulation [1, 46].

2 RELATED WORK

Student Simulation Student simulators, constructed to mitigate the problem of lacking high-quality interaction data in intelligent education systems, have been applied in many previous works. Memory-based [33] and KSS simulators [27] rely on manually crafted rules to predict student answering or memory behavior. EERNN [17] and KES [27] use an RNN-based model to predict the exercise performance of students. Nevertheless, memory-based and KSS simulators are too simple to simulate complex student interactions. Moreover, KES and EERNN, optimized in a maximum likelihood estimation (MLE) manner, may hardly ensure the quality of generated interactions because of exposure bias and the single-step optimization [2, 11]. In this paper, we formulate the task of building a student simulator as an MDP, which overcomes the limitations of existing student simulators, to ensure that generated interactions resemble real distribution closely.

Imitation Learning Imitation Learning aims to imitate human behavior in a specific task by learning the map between states and actions from the trajectories of expert demonstrations [12, 18]. Imitation learning can be divided into the following categories: First, Behavior Clone (BC) directly learns the map between states and actions through supervised learning but suffers from compounding error, which means mistakes in testing would cause the agent to deviate from the learned state distribution and lead to more mistakes [38]. Traditional KES and EERNN can be viewed as BC methods. Second, Inverse Reinforcement Learning (IRL) and Generative Adversarial Imitation Learning (GAIL) learn the implicit reward of the optimal decision from given expert demonstrations and alleviate compounding error within a minimax manner [15]. Third, imitation learning by reinforcement learning (ILbyRL), which is proposed to address the implementation and usage difficulty of IRL/GAIL [10, 24], generally defines the reward to be 1 for state-action pairs from the expert trajectory and 0 for other state-action pairs [10, 32, 38]. In this paper, we adopt the inspiration from imitation learning to reconstruct the implicit intentions underlying real interactions (i.e., expert demonstrations). Specifically, we propose two types of rewards in direct and adversarial imitation stages to model intentions and generate high-quality interactions.

Adaptive Learning Services Intelligent educational systems offer students adaptive learning services, including Computerized Adaptive Testing (CAT) and Knowledge Tracing (KT), to help students enhance their skills. CAT is an advanced educational measurement method that evaluates the knowledge level of examinees in minor exercises [3, 14, 47], while KT is a fundamental education research that assesses student knowledge proficiency based on their performance on previous exercises [6, 13, 26, 30, 31, 35, 42, 43]. Both of them are of great importance in intelligent educational systems. However, they have a high demand for high-quality learning interactions, which is generally hard to collect. Thus, in this paper, we utilize these two services as downstream tasks to evaluate the quality of the interactions generated by the simulator.

3 DAISIM: DIRECT-ADVERSARIAL IMITATION STUDENT SIMULATOR

3.1 Problem Statement

Suppose there are $|U|$ students and $|E|$ exercises in an intelligent educational system. Specifically, real student interaction data(i.e., expert demonstrations) is recorded as $U = \{u_1, u_2, ..., u_{|U|}\}$. For a certain student $u \in U$, his/her interactions are denoted as $u = \{(e_1, a_1), (e_2, a_2), ...\}$, where $e_t \in E$ represents the exercise that student u practice at step t, and a_t is the response to e_t[1]. If he/she answers e_t correctly, a_t is 1 otherwise a_t is 0. For a certain exercise $e \in E$, we denote it as $e = \{i, k\}$, i is the id of exercise, and $k \in K$ is the corresponding knowledge concepts (e.g., Function, Addition). In this context, the goal of the simulator is to generate interactions $\hat{U} = \{\hat{u}_1, \hat{u}_2, ...\}$ that are equivalent to U in distribution. We denote real and generated distribution as $\mathbb{U}$ and $\hat{\mathbb{U}}$, respectively.

3.2 Framework Overview

To address the problems of exposure bias and single-step optimization, we perceive the interaction process as a sequential decision-making process and formulate it as a Markov Decision Process (MDP). The corresponding elements of the MDP are as follows:

- **State** $\mathcal{S}$: $\mathcal{S}$ is the state space that models the state before the simulator responds. At the step t, the state $s_t \in S$ is defined as the combination of prior ability p_0 the historical interactions of a simulator before t and the current exercise e_t, that is $s_t = \{p_0, (e_0, \hat{a}_0), (e_1, \hat{a}_1), ..., e_t\}$. Note, p_0 is a prior knowledge vector $[p_{0,1}, p_{0,2}, ..., p_{0,|K|}]$ over a student, representing the student's ability on each concept. Intuitively, a student with higher $p_{0,k}$ would answer exercises correctly related to concept k with a higher probability.
- **Action** $\mathcal{A}$: At step t, $\hat{a}_t$ is the imitated student response to an exercise e_t. Whether the simulated student could answer an exercise correctly is determined by $\pi_\theta(\hat{a}_t|s_t)$.
- **Transition** $\mathcal{T}$: The next state s_{t+1} is determined by the current state s_t, action $\hat{a}_t$ and exercises e_{t+1}. Thus, we set $s_{t+1} = \{s_t, \hat{a}_t, e_{t+1}\} = \{p_0, (e_0, \hat{a}_0), (e_1, \hat{a}_1), ..., (e_t, \hat{a}_t), e_{t+1}\}$.
- **Reward** $\mathcal{R}$: After the simulator taking an action $\hat{a}_t$ based on state s_t, a reward $r(s_t, \hat{a}_t)$ is given to the simulator according to state s_t and action $\hat{a}_t$.

Based on the above MDP formulation, we can address the problems of exposure bias and single-step optimization. Specifically, we first make the training and testing stages consistent with the definition of s to alleviate exposure bias. Then, we optimize a γ-discounted long-term reward to fit students' intentions and train the simulator in the RL manner.

3.3 Imitation Flow

Generally, the student interactions are complex, and the intention underlying the complex interactions is unknown. Therefore, manually defining the reward function $r(s_t, a_t)$ is infeasible. Thus, the remaining issue now is how to construct the reward. We will further construct the reward functions and introduce how to optimize *DAISim* in a two-stage imitation learning framework.

[1]There are various types of student interactions, we only consider the exercise-answering interaction here since it is the most common one.

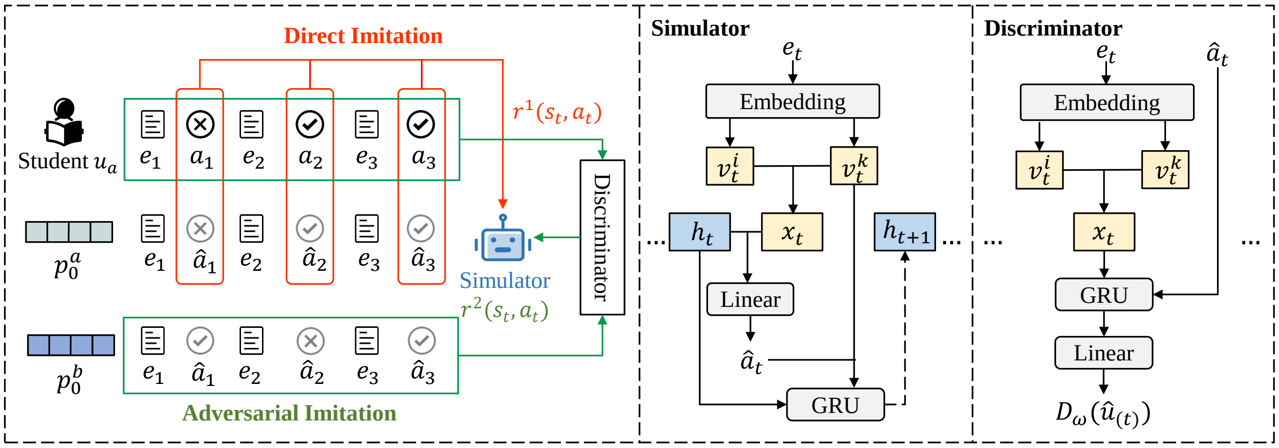


Figure 2: The overall framework. Left: In the direct imitation stage, the simulator mimics u_a to recover the interactions of existing u_a sequentially and receive a pre-defined r^1. In the adversarial imitation stage, the simulator first generates interactions sequentially beyond the dataset by mimicking another u_b to answer exercises of u_a. Then the discriminator accesses the simulator with r^2 based on the difference between real interactions and generated interactions. Right: Model implementation of the simulator and the discriminator.

Direct Imitation. To overcome the difficulty of manually designing the intention which drives student interactions, inspired by ILbyRL [10, 32, 38], we propose a simple supervised reward function (red line in Figure 2) to imitate students. To be specific, for a certain student $u = \{(e_0, a_0), (e_1, a_1), ...\} \in U$, we utilize the real feedback as supervised signals. When the action $\hat{a}$ generated by the $\pi(\hat{a}|s)$ is consistent with the real student action a, the reward is 1; otherwise, it is 0:

$$r_t^1 = \begin{cases} 1 & if\ \hat{a}_t == a_t, \\ 0 & \text{otherwise}, \end{cases} \tag{1}$$

where r_t^1 is the reward at the direct imitation stage. In this way, π_θ will be more inclined to imitate the real response behavior of students directly.

At this stage, we sample N real students $\tau = \{u_1, u_2, ..., u_N\}$ from given interactions U. In Figure 2, for an individual student u_a, we first extract initial state p_0^a from historical interactions, then we use exercise list $(e_0, e_1, e_2, ...)$ of u_a, and $\pi_\theta(\hat{a}|s)$ to sequentially generate $\hat{u} = \{(e_0, \hat{a}_0), (e_1, \hat{a}_1), ...\}$. We utilize Proximal Policy Optimization (PPO) [34] with the reward according to Equation (1):

$$\min_\theta \mathbb{E}[-r^1 \log \pi_\theta(\hat{a}|s)] - H(\pi_\theta), \tag{2}$$

where the $H(\pi_\theta)$ is the entropy term serving as a regularizer [15].

Adversarial Imitation. Nevertheless, direct imitation would encounter the sparsity problem with limited real interactions because each student would only interact with fractional items. This problem would cause insufficient rationality and diversity in generated interaction data of direct imitation [11]. To address this issue, we propose adversarial imitation by introducing interactions not presented in the dataset. To enrich the real interactions and ensure consistency between generated interactions and the real distribution, we make an assumption, which takes the real learning scenario into account, that students with similar abilities are often exposed to similar exercises. According to the assumption, the simulator generates new interactions beyond the dataset, as shown in Figure 2: (1) Sampling N real sequences $\tau = \{u_1, u_2, ..., u_N\}$ from expert demonstrations U; (2) For student $u_a \in \tau$ with proficiency p_0^a, finding another student u_b with the closest proficiency p_0^b according to cosine similarity; (3) Simulating the response student b to the exercises answered by student a (i.e., $\hat{u}_a$) based on $\pi_\theta(\hat{a}|s)$, which introduce a sequence beyond real interactions. Thus, $\hat{u}_a = \{(e_0, \hat{a}_0), (e_1, \hat{a}_1), ...\}$ is generated, and u_a and $\hat{u}_a$ are regarded as real and fake samples respectively to train the discriminator.

Generally, some previous GAIL-based works [11, 36] use cross-entropy loss to optimize the discriminator D_ω:

$$\max_\omega \mathbb{E}_{(s,a)\in U}[\log(\sigma(D_\omega(s,a)))] + \mathbb{E}_{(s,a)\in \hat{U}}[\log(1-\sigma(D_\omega(s,a)))]. \tag{3}$$

Then π_θ is updated with the reward given by D_ω in a reinforcement learning manner. In this setting, the objective of π_θ is:

$$\min_\theta \max_\omega \mathbb{E}_{(s,a)\in U}[\log(\sigma(D_\omega(s,a)))] + \mathbb{E}_{(s,a)\in \hat{U}}[\log(1-\sigma(D_\omega(s,a)))], \tag{4}$$

This objective is isovalent to minimize the JS divergence between $\mathbb{U}$ and $\hat{\mathbb{U}}$:

$$\min_\theta Div_{JS}(\mathbb{U}, \hat{\mathbb{U}}) - H(\pi_\theta), \tag{5}$$

Here, $\mathbb{U}$ and $\hat{\mathbb{U}}$ represent the distributions of real interactions and generated interactions, respectively [11, 15].

Pairwise discriminator training strategy. However, the traditional discriminator would provide improper rewards to hinder generation quality and stability for these reasons: First, student interaction is generally complex due to many confounding factors like guessing and slipping [37]. The discriminator would converge much faster than the simulator since the recovering distribution task of the simulator is indeed much more challenging than the task of the discriminator [11]. Then, the simulator would only receive negative rewards before it resembles the real distribution and fails to train. Second, the traditional discriminator would lack sensitivity to the difference between distributions. Because when the discriminator reaches optimality, it only tries to distinguish between the generated and real interactions and stops considering why real samples are more "real" than fake samples and vice versa [19]. Therefore, it is not feasible to utilize the traditional discriminator.

To address the challenges and achieve more stable training of the student simulator with appropriate rewards, we employ a pairwise strategy to update the discriminator D_ω. The optimization of D_ω is carried out by:

$$\max_{\omega} \mathbb{E}_u[f(D_\omega(u)-\mathbb{E}[D_\omega(\hat{u})])]+\mathbb{E}_{\hat{u}}[f(\mathbb{E}[D_\omega(u)]-D_\omega(\hat{u}))], \quad (6)$$

where $f(\Delta)$ is the transfer function, we use a Log-Sigmoid function (i.e., $f(x) = \log(\sigma(x)) + \log(2)$). Specifically, in Equation (6), for a real interaction u, its score should be higher than any randomly sampled generated interaction $\hat{u}$ as shown in the first expectation, and vice versa. Through this, the pairwise strategy compels the discriminator to explicitly measure the discrepancy between real and generated samples, expanding its focus beyond mere discrimination and enabling a deeper understanding of the authenticity of real student interaction data. So the reward function for the simulator in adversarial imitation (green line in Figure 2) is defined as:

$$r^2 = \sigma(D_\omega(\hat{u}) - \mathbb{E}[D_\omega(u)]), \quad (7)$$

Then the simulator is optimized by PPO [34] according to:

$$\min_{\theta} \mathbb{E}[-r^2 \log \pi_\theta(\hat{a}|s)] - H(\pi_\theta). \quad (8)$$

Through the above objectives of the simulator and the discriminator, according to [20], we find:

Lemma 3.1. *Given the objective of the pairwise discriminator in Equation (6), the maximum of this objective is a divergence:*

$$\begin{aligned} Div_P(\mathbb{U}, \hat{\mathbb{U}}) = \quad & \max_{\omega} \mathbb{E}_{x\sim\mathbb{U}}[f(D_\omega(x) - \mathbb{E}[D_\omega(\hat{x})])] + \\ & \mathbb{E}_{x\sim\hat{\mathbb{U}}}[f(\mathbb{E}[D_\omega(x)] - D_\omega(\hat{x}))], \end{aligned} \quad (9)$$

where $\mathbb{U}$ *and* $\hat{\mathbb{U}}$ *represent the distributions of real interactions and generated interactions, respectively.* $Div_P(\mathbb{U}, \hat{\mathbb{U}})$ *denotes the divergence between* $\mathbb{U}$ *and* $\hat{\mathbb{U}}$.

Thus, through Lemma 3.1, we can conclude that the objective of the simulator in adversarial imitation with the pairwise discriminator is minimizing a divergence Div_P, the objective can be written similar to Equation (5):

$$\min_{\theta} Div_P(\mathbb{U}, \hat{\mathbb{U}}) - H(\pi_\theta). \quad (10)$$

Compared to the conventional discriminator, our pairwise discriminator surpasses in performance for the following reasons: First, the pairwise discriminator explicitly considers the degree of differences between real and generated interactions rather than solely focusing on distinguishing them [19]. As a result, even if the discriminator becomes well-trained earlier than the simulator, the rewards provided to the simulator remain appropriate. These rewards indicate the extent to which the generated interactions deviate from the "real" interactions instead of simply categorizing generated interactions as "not real". Second, the pairwise discriminator would be more sensitive to the difference between distributions since it consistently incorporates real interactions and generated interactions into its gradient [20].

Based on the above intuitive advantages, we further theoretically analyze why the pairwise discriminator is better than the conventional one. Recall that the objective of the simulator in GAIL is to minimize Equation (3), which is equivalent to minimizing JS divergence (i.e., Equation (5)). Simplifying without affecting the result, we denote the Equation (3) as Div_{JS} by ignoring some constant term [19]. We have:

Theorem 3.2. *Given Log-Sigmoid function* f*, Let* $\mathbb{U}$ *and* $\hat{\mathbb{U}}$ *be probability distribution with support* X*. The following inequality holds true.*

$$Div_{JS}(\mathbb{U}, \hat{\mathbb{U}}) \le Div_P(\mathbb{U}, \hat{\mathbb{U}}) \quad (11)$$

Proof. Let:

$$\omega_0^* = \underset{\omega}{\operatorname{argmax}}\, \mathbb{E}_{x\sim\mathbb{U}}[\log(\sigma(D_\omega(x)))] + \mathbb{E}_{y\sim\hat{\mathbb{U}}}[\log(1 - \sigma(D_\omega(y)))] \quad (12)$$

Then:

$$\begin{aligned} & \max_{\omega} \mathbb{E}_{x\sim\mathbb{U}}[\log(\sigma(D_\omega(x)))] + \mathbb{E}_{y\sim\hat{\mathbb{U}}}[\log(1 - \sigma(D_\omega(y)))] \\ = \quad & \max_{\omega} \mathbb{E}_{x\sim\mathbb{U}}[f(D_\omega(x))] + \mathbb{E}_{y\sim\hat{\mathbb{U}}}[f(-D_\omega(y))] \\ = \quad & 2\, \mathbb{E}_{x\sim\mathbb{U}}[\tfrac{1}{2} f(D_{\omega_0^*}(x))] + \mathbb{E}_{y\sim\hat{\mathbb{U}}}[\tfrac{1}{2} f(-D_{\omega_0^*}(y))] \\ = \quad & 2\, \mathbb{E}_{x\sim\mathbb{U}, y\sim\hat{\mathbb{U}}}[\tfrac{1}{2} f(D_{\omega_0^*}(x)) + \tfrac{1}{2} f(-D_{\omega_0^*}(y))] \\ \le \quad & 2\, \mathbb{E}_{x\sim\mathbb{U}, y\sim\hat{\mathbb{U}}}[f(\tfrac{1}{2} D_{\omega_0^*}(x) - \tfrac{1}{2} D_{\omega_0^*}(y))] \\ \le \quad & \max_{\omega} 2\, \mathbb{E}_{x\sim\mathbb{U}, y\sim\hat{\mathbb{U}}}[f(D_\omega(x) - D_\omega(y))] \end{aligned} \quad (13)$$

Let:

$$\omega_1^* = \max_{\omega} \mathbb{E}_{x\sim\mathbb{U}, y\sim\hat{\mathbb{U}}}[f(D_\omega(x) - D_\omega(y))] \quad (14)$$

Then:

$$\begin{aligned} & \max_{\omega} 2\, \mathbb{E}_{x\sim\mathbb{U}, y\sim\hat{\mathbb{U}}}[f(D_\omega(x) - D_\omega(y))] \\ = \quad & 2\, \mathbb{E}_{x\sim\mathbb{U}, y\sim\hat{\mathbb{U}}}[f(D_{\omega_1^*}(x) - D_{\omega_1^*}(y))] \\ = \quad & \mathbb{E}_{x\sim\mathbb{U}}[\mathbb{E}_{y\sim\hat{\mathbb{U}}}[f(D_{\omega_1^*}(x) - D_{\omega_1^*}(y))|x]] + \\ & \mathbb{E}_{y\sim\hat{\mathbb{U}}}[\mathbb{E}_{x\sim\mathbb{U}}[f(D_{\omega_1^*}(x) - D_{\omega_1^*}(y))|y]] \\ \le \quad & \mathbb{E}_{x\sim\mathbb{U}}[f(\mathbb{E}_{y\sim\hat{\mathbb{U}}}[D_{\omega_1^*}(x) - D_{\omega_1^*}(y)|x])] + \\ & \mathbb{E}_{y\sim\hat{\mathbb{U}}}[f(\mathbb{E}_{x\sim\mathbb{U}}[D_{\omega_1^*}(x) - D_{\omega_1^*}(y)|y])] \\ = \quad & \mathbb{E}_{x\sim\mathbb{U}}[f(D_{\omega_1^*}(x) - \mathbb{E}_{y\sim\hat{\mathbb{U}}}[D_{\omega_1^*}(y)])] + \\ & \mathbb{E}_{y\sim\hat{\mathbb{U}}}[f(\mathbb{E}_{x\sim\mathbb{U}}[D_{\omega_1^*}(x)] - D_{\omega_1^*}(y))] \\ \le \quad & \max_{\omega} \mathbb{E}_{x\sim\mathbb{U}}[f(D_\omega(x) - \mathbb{E}_{y\sim\hat{\mathbb{U}}}[D_\omega(y)])] + \\ & \mathbb{E}_{y\sim\hat{\mathbb{U}}}[f(\mathbb{E}_{x\sim\mathbb{U}}[D_\omega(x)] - D_\omega(y))] \end{aligned} \quad (15)$$

Done. □

According to Theorem 3.2, we can conclude that the pairwise discriminator has an objective with a more prominent upper bound. Consequently, the simulator guided by the pairwise discriminator would minimize a divergence Div_P (see Equation (10)) that exhibits a heightened sensitivity to the differences between distributions compared to the traditional one. The experimental results in Section 4.3 also support this conclusion. Thus, the simulator can be improved to resemble the real distribution more closely with the pairwise discriminator.

The detail of the two-stage imitation is presented above. We summarize the two-stage imitation workflow in Algorithm 1.

3.4 Model Implementation

This section will discuss the implementation details of the simulator and discriminator models. To showcase the effectiveness of the two imitation stages, we adopt a simple yet effective RNN-based architecture similar to KES [31] to build the student simulator. And an

Algorithm 1 *DAISim*

```
Input: Real student sequences $D$
Output: Simulator $\pi_\theta$
Parameter: $\theta$, $\omega$
1:  Initialize parameter $\theta$ and $\omega$
2:  while NOT CONVERGENCE do
3:      Sample a batch of Expert Demonstrations $\tau$ from $D$;
4:      (1) Direct Imitation
5:      for $i = 0, 1, 2, ...$ do
6:          $p_0 \sim u_i, e_1 \sim u_i$;
7:          for $j = 0, 1, 2, ...$ do
8:              Predict $\hat{a}_j \sim \pi_\theta(\cdot|s_j)$,
9:              Transition $s_{j+1}$,
10:             Compute $r_j^1$ by Equation (1);
11:         end for
12:     end for
13:     Update $\theta$ by PPO according to Equation (2);
14:     (2) Adversarial Imitation
15:     for i = 0, 1, 2, ... do
16:         Find $u_b$ based on $p_0 \sim u_i$;
17:         $p_0^b \sim u_b, e_1 \sim u_i$;
18:         for j = 0, 1, 2, ... do
19:             Predict $\hat{a}_j \sim \pi_\theta(\cdot|s_j^b)$,
20:             Transition $s_{j+1}^b$,
21:             Compute $r_j^2$ by Equation (7);
22:         end for
23:     end for
24:     Update $\omega$ by Equation (6);
25:     Update $\theta$ by PPO by Equation (8);
26: end while
```

RNN structure serves as the discriminator. Although Transformer-based backbones are effective sequential models, they are not involved here for these reasons: Firstly, due to the limited availability of training data for the simulator, using a transformer-based model may result in poor generalization performance. Secondly, the state transition in our model depends on each step's action. However, the transformer architecture, which heavily relies on parallelization for efficient computation, is not well-suited for this dependency on sequential actions here.

First, for each exercise $e_t = \{i, k\}$ at step t, we map the id of exercise i and concepts k to low-dimensional vectors $v_t^i, v_t^k \in \mathbb{R}^d$ via embedding layers and linear layers. The encoding x_t of exercise e_t can be concatenated with its id and knowledge representation, i.e., $x_t = v_t^i \oplus v_t^k \in \mathbb{R}^{2d}$, where $\oplus$ denotes the concat operator.

3.4.1 Simulator Architecture. The simulator should precisely generate student interaction based on $\pi_\theta(\hat{a}_t|s_t)$ and update state based on s_t and $\hat{a}_t$. To implement it, we adopt a structure similar to the classical and simple deep knowledge tracing model [31, 40] as illustrated in the middle part of Figure 2.

- Before generation, as mentioned in section 3.2, the simulator needs to extract p_0 from historical interactions to set up. Here we compute the correct rate of each knowledge concept in historical interactions to set p_0.

Table 1: Dataset statistics.

Dataset	Assistment0910	EdNet-KT1
Students	4,163	784,309
Exercises	17,746	13,169
Concepts	123	189
Interactions	324,572	95,293,926
Avg.length	68.00	121.50
Sparsity	0.439%	0.923%

- At step t, we further utilize GRU to capture information in historical interactions, here h_t encodes the last hidden state h_{t-1} and interactions $(e_{t-1}, \hat{a}_{t-1})$:

$$h_t = \begin{cases} p_0 & \text{if } t = 0, \\ GRU(h_{t-1}, m(e_{t-1}, \hat{a}_{t-1})) & otherwise, \end{cases} \tag{16}$$

Since the right (i.e., 1) and wrong responses (i.e., 0) to the exercise have different influences, we extend e_t on all concepts (i.e., v_t^k) to a feature vector $\mathbf{0}^x \in \mathbb{R}^d$ with the same dimensions of v_t^k. We note this operation as m_t:

$$m_t(e_t, \hat{a}_t) = \begin{cases} v_t^k \oplus \mathbf{0}^k & \text{if } \hat{a}_t = 1, \\ \mathbf{0}^k \oplus v_t^k & \text{if } \hat{a}_t = 0, \end{cases} \tag{17}$$

- At step t, the simulator makes a decision based on the current state s_t:

$$\pi_\theta(\hat{a}_t|s_t) = \sigma(Linear(h_t \oplus x_t)). \tag{18}$$

3.4.2 Discriminator Architecture. In the adversarial imitation procedure, the discriminator D_ω is designed to distinguish the interaction sequences, which gives a high reward to the real sequences. Therefore, it forces the simulator to generate reasonable interaction sequences similar to real ones. As illustrated in the right part of Figure 2, we also use RNN[7] to build the discriminator. Specifically, for a generated interaction sequence $\hat{u} = \{(e_1, \hat{a}_1), (e_2, \hat{a}_2), ...\}$, the discriminator assesses it as:

$$D_\omega(\hat{u}_{(t)}) = Linear(GRU(z_t)), \tag{19}$$

where $D_\omega(\hat{u}_{(t)}) \in \mathbb{R}$ is the score and z_t is concatenated vector from x_t and $\mathbf{0}^x \in \mathbb{R}^{2d}$ as follows:

$$z_t = \begin{cases} x_t \oplus \mathbf{0}^x & \text{if } \hat{a}_t = 1, \\ \mathbf{0}^x \oplus x_t & \text{if } \hat{a}_t = 0. \end{cases} \tag{20}$$

4 EXPERIMENTS

4.1 Experiment Setup

4.1.1 Datasets. We conduct experiments on two educational benchmark datasets, namely Assistment0910 and EdNet-KT1. Table 1 shows the basic statistics of the datasets. **ASSISTment0910**[2] is an open dataset that provides interaction log with responses and knowledge concept. **EdNet-KT1**[3] [8] is a large-scale dataset in Santa with more than 780K users.

As Section 3.2 mentions, the simulator utilizes the initial prior and exercises to generate interaction sequences. Thus, we separate the interaction sequence of each student into early data and late data, where the early data is to extract the initial prior p_0 of students, and the late data provide exercises for the generation. The exact

[2]https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data
[3]https://github.com/riiid/ednet

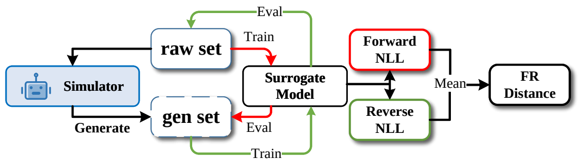


Figure 3: Workflow: computing Forward/Reverse NLL and FR Distance with *raw set* and *gen set*.

separation method of early and late data depends on the statistic of each dataset.

The data preprocessing is as follows: First, the average length of interaction sequences in ASSISTment0910 is 68, and the lengths of over 75% of interaction sequences are less than 60. Thus, for the reliability of experiments, we only apply this dataset in a short-sequence scenario. Specifically, we filter the students whose interaction sequences are shorter than 30. Then we select the last 20 interactions as late data and other interactions as early data. Second, to further evaluate the simulator's performance on long sequences, we apply EdNet in our experiments. Since EdNet is too large [21, 39], we random sample 2 subsets of 5,000 students. The late data's length in each subset is respectively set as 20/50 while the length of early data is limited the same as the ASSISTment0910. After data preprocessing, we can get three datasets: Assist, EdNet-20, and EdNet-50.

4.1.2 Baselines. To demonstrate the effectiveness of our *DAISim*, we first compare it with KES [31] and EERNN [26], which were once used as student simulators in previous work [17, 27]. Second, we compare it with AICM [11], which is a representative GAIL-based user simulator in information retrieval. Finally, there are only a few works investigating student simulation. Thus, for a convincing comparison, we compare *DAISim* with other classical or state-of-the-art knowledge tracing models because KT models are similar to the simulator in the form of input and output, including DKVMN [42], SAKT [30], AKT [13].

4.1.3 Implement Details. Our model is implemented by PyTorch. The dimension of embeddings and the hidden vector is 128. The parameters of the simulator and discriminator are initialized by Xavier initialization. The dropout rate is 0.5. We use the Adam [22] optimizer to optimize our model. The learning rate of the discriminator is 0.0001. The learning rate of the simulator in the direct imitation stage is $l_d = 0.0001$ while in the adversarial imitation stage is $l_a = \lambda l_d$. The simulator will be updated four times in PPO, while the discriminator will be updated only once in one step. All experiments are run on a Linux server with four 2.30GHz Intel Xeon Gold 5218 CPUs and a Tesla V100 GPU. Our code is available at https://github.com/bigdata-ustc/DAISim.

4.2 RQ1: Quantification Distance Measuring

The objective of the simulator is that its generated distribution is close to the real distribution, as mentioned in Section 1.

To set up, we divide students in each dataset with a proportion of 60%/40% into *trainset* and *rawset* for training and evaluation. We utilize the *trainset* dataset to train our *DAISim* model as well as other baseline models with a validation dataset derived from the *trainset* (20%). Then we use well-trained simulators to generate sequences based on the exercise list in the *rawset*. After generation, we can get a new *genset* with the same size as *rawset*. Whether the distribution of *genset* is close to that of *rawset* indicates whether the simulator is effective in generating high-quality data. However, commonly used metrics in continuous distribution like JS-divergence are unsuitable for quantitatively measuring the distance between distributions because our generated interactions are discrete exercise-answer pairs. Thus, we apply two quantitative measurements to evaluate the distance or similarity between the generated and real distribution.

First, we follow the idea of Dai et al. [11] and Zhao et al. [44] to build a surrogate model to measure the distance between generated and real data quantitatively. Specifically, we use *rawset* to train the surrogate model and evaluate in *genset* to compute Forward NLL and compute Reverse NLL conversely as shown in Figure 3. Forward and Reverse NLL reflect the asymmetric discrepancy between distributions to some extent. Thus the distance between the generated and real distribution can be approximately defined as the average of the above two values. We name it **FR Distance**, where a lower value indicates a better performance of the simulator. Here we utilize a simple RNN as a surrogate model because the performance of different surrogate models is not our concern. Second, we introduce **ROUGE**, widely used in NLP and sensitive to the generations beyond reality, to compute the similarity between generated sequences and the given sequences because our data is discrete exercise-answer pair sequences similar to texts. Here we compute ROUGE-2 and ROUGE-L by viewing *rawset* as the reference and *genset* as the generation, where the higher value, the better performance.

Table 2 reports the performance of our proposed model and baseline models on the two datasets. From the experiment results, we have the following observations: First, *DAISim* outperforms all baseline methods on each dataset, demonstrating that the distribution generated by *DAISim* covers the real distribution best. The reason may be as follows: (1) reconstructing the long-term intentions and alleviating exposure bias in *DAISim* could help cover complex patterns; (2) taking reverse KL divergence, which constrains the simulator to generate patterns that are not in real distribution, into account in both imitation stages further to ensure data quality [4, 19]. Second, KES, which has a similar model architecture with *DAISim* but is trained in the MLE manner with exposure bias and single-step optimization, performs worse than *DAISim*. This observation strengthens that the MDP formulation and two-stage imitation learning help the model recover the real distribution better. Third, AICM, which has a similar MDP formulation and GAIL-based training procedure with *DAISim*, performs worse than *DAISim*. Such a result may be caused by: (1) its traditional classification training manner in its discriminator may impact negatively due to unstable training; (2) its model is designed based on preference rather than a student simulator based on knowledge proficiency. In other words, it is not suitable for modeling student interactions. Finally, Although transformer-based KT models outperform RNN-based models in the KT task, they perform worse than RNN-based models in ROUGE-2, ROUGE-L, and FR Distance. This observation demonstrates that the KT task and the simulation task prefer different models, and RNN-structure is more suitable for the simulation task. In summary, we can conclude that *DAISim*

Table 2: Experimental results on quantification distance measuring. The second-best results are marked by an underline, while the best results are bold. * indicates significant difference (p-value < 0.01 in the t-test).

	Data Distribution								
	Assist			EdNet-20			EdNet-50		
	ROUGE-2 ↑	ROUGE-L ↑	FR Distance ↓	ROUGE-2 ↑	ROUGE-L ↑	FR Distance ↓	ROUGE-2 ↑	ROUGE-L ↑	FR Distance ↓
KES	0.4342	0.6052	0.7295	0.4014	0.6276	0.7260	0.4359	0.6520	0.7412
EERNN	0.3931	0.5778	0.6536	0.4105	0.6320	0.6761	0.3606	0.6037	0.6506
DKVMN	0.3044	0.4943	0.6344	0.3096	0.5453	0.6558	0.2507	0.4919	0.6762
SAKT	0.3176	0.5190	0.7356	0.3227	0.5522	0.6398	0.3117	0.5415	0.6799
AKT	0.3786	0.5624	0.7437	0.3949	0.6193	0.6363	0.3525	0.5848	0.6560
AICM	0.2634	0.4981	0.6354	0.2763	0.5227	0.6352	0.3069	0.5583	0.6940
DAISIM	**0.4392***	**0.6331***	**0.6292***	**0.4243***	**0.6492***	**0.6263***	**0.4596***	**0.6788***	**0.6314***
DAISIM-DI	0.4220	0.6301	0.6346	0.4008	0.6312	0.6315	0.4354	0.6602	0.6337
DAISIM-GAIL	0.3793	0.5724	0.7540	0.2728	0.4994	0.9826	0.2831	0.5505	0.6882
DAISIM-PAIR	0.3463	0.5690	0.6891	0.3299	0.5523	0.7422	0.3643	0.5968	0.6307
DAISIM-DI+GAIL	0.4136	0.6183	0.6348	0.4049	0.6343	0.6311	0.4365	0.6618	0.6318

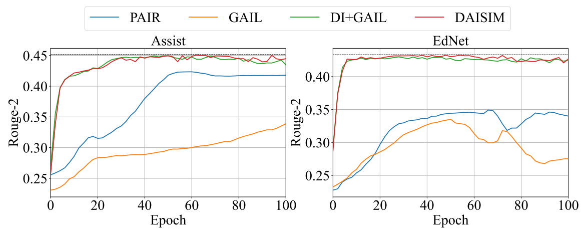


Figure 4: Rouge-2 performance on test set during training.

is effective by taking full advantage of the long-term objective and two stages of imitation to generate a distribution better recovering real distribution.

4.3 RQ2: Ablation Study

To further investigate the contributions of the imitation flow and pairwise strategy in *DAISim*, we conduct some ablation studies on both datasets with the same setting as Section 4.2:

- *DAISim-DI*, only contains the direct imitation stage.
- *DAISim-GAIL*, only contains the adversarial imitation stage with a classification discriminator and minimizes JS divergence as Equation (5).
- *DAISim-PAIR*, only contains the adversarial imitation stage with a pairwise discriminator and minimizes divergence defined in Equation (9).
- *DAISim-DI+GAIL* contains the direct and adversarial imitation stages, but its discriminator is optimized in the classification manner.

The experimental results are listed in Table 2 and Figure 4, from which we can get some interesting observations: First, ROUGE-2, ROUGE-L, and FR distance of *DAISim-DI* are typically better than almost all baselines, indicating that the long-term objective and alleviating exposure bias are helpful. Also, all DI-based methods converge fast, as shown in Figure 4, demonstrating the essentiality of the direct imitation stage. Second, *DAISim* and *DAISim-DI+GAIL* outperform *DAISim-DI*, indicating that interactions beyond the given dataset can help the simulator cover real distribution. It is reasonable because these interactions may introduce various patterns, and the discriminator helps choose the patterns consistent with real distribution to improve the simulator. Thus, this observation confirms that adversarial imitation is crucial in student interaction generation. Third, as shown in Figure 4, *DAISim-PAIR* and *DAISim* are respectively better and more stable than *DAISim-GAIL* and *DAISim-DI+GAIL*. This observation and the theoretical analysis indicate that the pairwise discriminator can better differentiate between real and generated interactions and provide appropriate rewards to help the simulator resemble the real distribution.

Table 3: The improvement of KT models.

	Data Argumentation-KT					
Model	Train AUC	Arg AUC	Impr	Train ACC	Arg ACC	Impr
DKT	0.5983	**0.6013**	0.5%	0.6861	**0.7189**	4.8%
EERNN	0.5078	**0.5113**	0.7%	0.5287	**0.6446**	21.9%
SAKT	0.5773	**0.6059**	5.0%	0.6176	**0.7206**	16.7%
AKT	0.6565	0.6530	-0.5%	0.7251	**0.7346**	1.3%

Table 4: The improvement of CAT strategies.

	Data Augmentation-CAT					
	@5			@10		
Strategy	Train ACC	Arg ACC	Impr	Train ACC	Arg ACC	Impr
FSI	0.5410	**0.6567**	21.38%	0.5373	**0.6567**	22.22%
KLI	0.5597	**0.6082**	8.67%	0.5597	**0.6045**	8.00%
MAAT	0.5933	**0.6940**	16.98%	0.6045	**0.6978**	15.43%

4.4 RQ3: Adaptive Learning Improvement

The primary goal of student simulators is to improve adaptive learning services with generated data. To evaluate our proposed *DAISim*, we choose the standard KT and CAT as our evaluation tasks since they are representative, of great value in education, and sensitive to high-quality data [27, 31, 47], where KT is often used to track student knowledge states, and CAT estimates students' ability with minor exercises. In a word, if the generated data can improve the performance of KT and CAT, it demonstrates that our proposed model is adequate.

To set up, we first divide students in Assist the same as Section 4.2 and train our simulator with *trainset*. Then, to simulate the situation that offline training data of the adaptive learning services is biased in the actual scenario, we divide students in *rawset* with 60%/40% into *dtrainset* and *dtestset* according to the initial state of

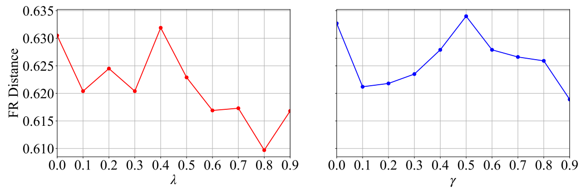


Figure 5: FR Distance in different λ and γ values.

each student to make the *dtrainset* and *dtestset* apart in distribution. Next, we use *DAISim* to augment *dtrainset*. Specifically, we randomly generate the initial states. Then we use the exercise lists in *dtrainset* and the generated initial states to generate new student interaction sequences and merge these sequences into *dtrainset* as a new dataset, named *argset*. In the KT task, we use *dtrainset* and *argset* to train some representative KT models [13, 26, 30, 31], and evaluate these models by *dtestset*. To verify the tracking performance of models, we utilize commonly used AUC and ACC in KT task. In the CAT task, we use *dtrainset* and *argset* to train the CDM (IRT) and then evaluate by different question selection strategies, including FSI [28], KLI [5], and MAAT [3] at different interaction steps (@5, @10) in *dtestset*. Then, we utilize the commonly used metric ACC to evaluate its estimation performance.

The results are listed in Table 3 and Table 4. The results show that almost all models and strategies can get promotion with the help of *DAISim*. They further indicate that *DAISim* can easily generate high-quality interactions with randomly initialized students and enrich the given dataset because it can resemble real distribution.

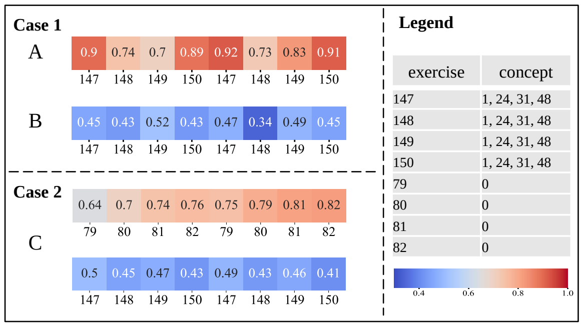
4.5 RQ4: Hyper-parameter Influence

The learning rate proportion λ and discount factor γ in the PPO algorithm [34] are crucial factors in our framework. Precisely, the λ in Section 4.1.3 controls the impact of two stages in the training process. Moreover, the discount factor γ determines the weight of the long-term reward when making current actions.

The result is obtained from the experiment with the same setting as Section 4.2 in Assist and shown in Figure 5. We can have such observations: First, the simulator performs best when λ is 0.8 and performs worst when λ is 0, which indicates that the contribution of different imitation stages to the parameter update also influences the quality of distribution recovery. Second, the simulator performs worst when γ is 0, while it performs best when γ is 0.9, showing that the long-term objective is helpful when generating student interaction sequences.

4.6 Case Study

To demonstrate the quality of interaction sequences from *DAISim*, we conduct case studies illustrated in Figure 6. In Case 1, two students A and B are compared. p_0^A for each knowledge concept is 1, while p_0^B is all zeros. After assigning exercises and recording correct answer probabilities, it's evident that A outperforms B, aligning with the expectation that higher prior ability leads to better performance. Moving to Case 2, student C, who only masters concept 0, is considered. Two sets of exercises are given to C: one related to concept 0, the other unrelated. Student C's higher correct answer probability for concept 0-related exercises (e.g., exercises 79, 80, ...)



exercise	concept
147	1, 24, 31, 48
148	1, 24, 31, 48
149	1, 24, 31, 48
150	1, 24, 31, 48
79	0
80	0
81	0
82	0

Figure 6: The initial state p_0 of student A is represented as $[1, 1, ..., 1]$, student B as $[0, 0, ..., 0]$, and student C as $[1, 0, ..., 0]$. The heat bar indicates the probability of answering the exercises correctly. For instance, student A has a 0.9 probability of correctly answering exercise 147. The legend table provides the correspondence between exercise ID i and the associated concept k. For example, exercise 79 is related to concept 0.

aligns logically with the anticipated outcome. Notably, both cases reveal that practicing mastered exercises (e.g., A and the upward sequence of C) enhances performance. For instance, A's probability of correctly answering exercise 147 improves from 0.9 to 0.92 upon repeated practice. Conversely, practicing exercises not mastered (e.g., B and the lower sequence of C) yields no improvement. This phenomenon is intuitive because one can hardly naturally master an exercise by practicing again and again without help from the teacher, but once one has mastered an exercise, keeping practicing strengthens proficiency and memory. In summary, these observations underscore the congruence of generated interactions with common sense, affirming their high quality.

5 CONCLUSION

In this paper, we proposed a novel framework named *DAISim*, alleviated exposure bias and single-step optimization in existing advances by formulating the task of building the student simulator as an MDP. One step further, the two imitation stages, together with the pairwise discriminator, tackled the technical challenges to realize *DAISim*, including implicit intentions, data sparsity, and inappropriate rewards. Experimental results and our theoretical analysis demonstrated the effectiveness of *DAISim* in distribution matching and enhancing the downstream adaptive learning tasks.

There are still some potential future directions: First, in addition to answering, students also have clicking, hesitating, buying, quitting, and other behaviors when interacting with the agent. They are of great value and can be added in the future. Second, we have theoretically analyzed the optimization objective of the simulator in adversarial imitation, and it is also valuable to further analyze the error bound or convergence of *DAISim*.

ACKNOWLEDGMENTS

This research was partial supported by grants from the National Key Research and Development Program of China (Grant No. 2021YFF 0901003) and the National Natural Science Foundation of China (Grants No. 62106244, No. U20A20229), the University Synergy Innovation Program of Anhui Province (No. GXXT-2022-042).

REFERENCES

[1] Xueying Bai, Jian Guan, and Hongning Wang. 2019. A model-based reinforcement learning with adversarial training for online recommendation. *Advances in Neural Information Processing Systems* 32 (2019).

[2] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems* 28 (2015).

[3] Haoyang Bi, Haiping Ma, Zhenya Huang, Yu Yin, Qi Liu, Enhong Chen, Yu Su, and Shijin Wang. 2020. Quality meets diversity: A model-agnostic framework for computerized adaptive testing. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 42–51.

[4] Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*. Vol. 4. Springer.

[5] Hua-Hua Chang and Zhiliang Ying. 1996. A Global Information Approach to Computerized Adaptive Testing. *Applied Psychological Measurement* 20 (1996), 213 – 229.

[6] Mingzhi Chen, Quanlong Guan, Yizhou He, Zhenyu He, Liangda Fang, and Weiqi Luo. 2022. Knowledge Tracing Model with Learning and Forgetting Behavior. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (2022).

[7] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).

[8] Youngduck Choi, Youngnam Lee, Dongmin Shin, Junghyun Cho, Seoyon Park, Seewoo Lee, Jineon Baek, Chan Bae, Byungsoo Kim, and Jaewe Heo. 2020. Ednet: A large-scale hierarchical dataset in education. In *International Conference on Artificial Intelligence in Education*. Springer, 69–73.

[9] Hui-Chun Chu. 2014. Potential negative effects of mobile learning on students' learning achievement and cognitive load—A format assessment perspective. *Journal of Educational Technology & Society* 17, 1 (2014), 332–344.

[10] Kamil Ciosek. 2022. Imitation Learning by Reinforcement Learning. In *International Conference on Learning Representations*. https://openreview.net/forum?id=1zwleytEpYx

[11] Xinyi Dai, Jianghao Lin, Weinan Zhang, Shuai Li, Weiwen Liu, Ruiming Tang, Xiuqiang He, Jianye Hao, Jun Wang, and Yong Yu. 2021. An Adversarial Imitation Click Model for Information Retrieval. In *Proceedings of the Web Conference 2021*. 1809–1820.

[12] Chao Feng, Defu Lian, Xiting Wang, Zheng Liu, Xing Xie, and Enhong Chen. 2023. Reinforcement Routing on Proximity Graph for Efficient Recommendation. *ACM Transactions on Information Systems* 41, 1 (2023), 1–27.

[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*.

[14] Aritra Ghosh and Andrew Lan. 2021. BOBCAT: Bilevel Optimization-Based Computerized Adaptive Testing. In *International Joint Conference on Artificial Intelligence*.

[15] Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning. *Advances in neural information processing systems* 29 (2016).

[16] 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 (2020), 1–33.

[17] 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 the 28th ACM International Conference on Information and Knowledge Management*. 1261–1270.

[18] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. 2017. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)* 50, 2 (2017), 1–35.

[19] Alexia Jolicoeur-Martineau. 2019. The relativistic discriminator: a key element missing from standard GAN. In *International Conference on Learning Representations*. https://openreview.net/forum?id=S1erHoR5t7

[20] Alexia Jolicoeur-Martineau. 2020. On relativistic f-divergences. In *International Conference on Machine Learning*. PMLR, 4931–4939.

[21] Minsam Kim, Yugeun Shim, Seewoo Lee, Hyunbin Loh, and Juneyoung Park. 2021. Behavioral Testing of Deep Neural Network Knowledge Tracing Models. *International Educational Data Mining Society* (2021).

[22] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[23] René F Kizilcec and Hansol Lee. 2020. Algorithmic fairness in education. *arXiv preprint arXiv:2007.05443* (2020).

[24] Karol Kurach, Mario Lucic, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. 2019. The GAN Landscape: Losses, Architectures, Regularization, and Normalization. https://openreview.net/forum?id=rkGG6s0qKQ

[25] Yu Lei, Hongbin Pei, Hanqi Yan, and Wenjie Li. 2020. Reinforcement learning based recommendation with graph convolutional q-network. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*. 1757–1760.

[26] 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* 33, 1 (2019), 100–115.

[27] Qi Liu, Shiwei Tong, Chuanren Liu, Hongke Zhao, Enhong Chen, Haiping Ma, and Shijin Wang. 2019. Exploiting cognitive structure for adaptive learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 627–635.

[28] Frederic M. Lord. 1980. Applications of Item Response Theory To Practical Testing Problems.

[29] Yu Lu, Yang Pian, Penghe Chen, Qinggang Meng, and Yunbo Cao. 2021. RadarMath: An Intelligent Tutoring System for Math Education. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 16087–16090.

[30] Shalini Pandey and George Karypis. 2019. A self-attentive model for knowledge tracing. *arXiv preprint arXiv:1907.06837* (2019).

[31] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. *Advances in neural information processing systems* 28 (2015).

[32] Siddharth Reddy, Anca D. Dragan, and Sergey Levine. 2020. SQIL: Imitation Learning via Reinforcement Learning with Sparse Rewards. In *International Conference on Learning Representations*. https://openreview.net/forum?id=S1xKd24twB

[33] Siddharth Reddy, Sergey Levine, and Anca Dragan. 2017. Accelerating human learning with deep reinforcement learning. In *NIPS'17 Workshop: Teaching Machines, Robots, and Humans*. 5–9.

[34] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[35] Shuanghong Shen, Zhenya Huang, Qi Liu, Yu Su, Shijin Wang, and Enhong Chen. 2022. Assessing Student's Dynamic Knowledge State by Exploring the Question Difficulty Effect. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2022).

[36] Jing-Cheng Shi, Yang Yu, Qing Da, Shi-Yong Chen, and An-Xiang Zeng. 2019. Virtual-taobao: Virtualizing real-world online retail environment for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 4902–4909.

[37] Jill-Jênn Vie, Fabrice Popineau, Éric Bruillard, and Yolaine Bourda. 2017. A review of recent advances in adaptive assessment. *Learning analytics: fundaments, applications, and trends* (2017), 113–142.

[38] Ruohan Wang, Carlo Ciliberto, Pierluigi Vito Amadori, and Yiannis Demiris. 2019. Random expert distillation: Imitation learning via expert policy support estimation. In *International Conference on Machine Learning*. PMLR, 6536–6544.

[39] Qize Xie, Liping Wang, Peidong Song, and Xuemin Lin. 2021. SQKT: A Student Attention-Based and Question-Aware Model for Knowledge Tracing. In *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*. Springer, 221–236.

[40] Chun-Kit Yeung and Dit-Yan Yeung. 2018. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. 1–10.

[41] Jifan Yu, Yuquan Wang, Qingyang Zhong, Gan Luo, Yiming Mao, Kai Sun, Wenzheng Feng, Wei Xu, Shulin Cao, Kaisheng Zeng, et al. 2021. MOOCCubeX: a large knowledge-centered repository for adaptive learning in MOOCs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4643–4652.

[42] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*. 765–774.

[43] Moyu Zhang, Xinning Zhu, Chunhong Zhang, Yang Ji, Feng Pan, and Changchuan Yin. 2021. Multi-factors aware dual-attentional knowledge tracing. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2588–2597.

[44] Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In *International conference on machine learning*. PMLR, 5902–5911.

[45] Wayne Xin Zhao, Wenhui Zhang, Yulan He, Xing Xie, and Ji-Rong Wen. 2018. Automatically learning topics and difficulty levels of problems in online judge systems. *ACM Transactions on Information Systems (TOIS)* 36, 3 (2018), 1–33.

[46] Xiangyu Zhao, Long Xia, Lixin Zou, Hui Liu, Dawei Yin, and Jiliang Tang. 2021. Usersim: User simulation via supervised generativeadversarial network. In *Proceedings of the Web Conference 2021*. 3582–3589.

[47] Yan Zhuang, Qi Liu, Zhenya Huang, Zhi Li, Shuanghong Shen, and Haiping Ma. 2022. Fully Adaptive Framework: Neural Computerized Adaptive Testing for Online Education. In *AAAI*, Vol. 36. 4734–4742.