Privileged Knowledge State Distillation for Reinforcement Learning-based Educational Path Recommendation

Qingyao Li
ly890306@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Wei Xia
xiawei24@huawei.com
Huawei Noah's Ark Lab
Shenzhen, China

Li'ang Yin
yinla@apex.sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Jiarui Jin
jinjiarui97@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Yong Yu*
yyu@apex.sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

ABSTRACT

Educational recommendation seeks to suggest knowledge concepts that match a learner's ability, thus facilitating a personalized learning experience. In recent years, reinforcement learning (RL) methods have achieved considerable results by taking the encoding of the learner's exercise log as the state and employing an RL-based agent to make suitable recommendations. However, these approaches suffer from handling the diverse and dynamic learner's knowledge states. In this paper, we introduce the privileged feature distillation technique and propose the **P**rivileged **K**nowledge **S**tate **D**istillation (**PKSD**) framework, allowing the RL agent to leverage the "actual" knowledge state as privileged information in the state encoding to help tailor recommendations to meet individual needs. Concretely, our PKSD takes the privileged knowledge states together with the representations of the exercise log for the state representations during training. And through distillation, we transfer the ability to adapt to learners to a *knowledge state adapter*. During inference, the *knowledge state adapter* would serve as the estimated privileged knowledge states instead of the real one since it is not accessible. Considering that there are strong connections among the knowledge concepts in education, we further propose to collaborate the graph structure learning for concepts into our PKSD framework. This new approach is termed **GEPKSD** (Graph-Enhanced PKSD). As our method is model-agnostic, we evaluate PKSD and GEPKSD by integrating them with five different RL bases on four public simulators, respectively. Our results verify that PKSD can consistently improve the recommendation performance with various RL methods, and our GEPKSD could further enhance the effectiveness of PKSD in all the simulations.

*Corresponding author.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3671872

CCS CONCEPTS

• **Applied computing** → **E-learning**; • **Information systems** → *Recommender systems.*

KEYWORDS

Educational Path Recommendation; Reinforcement Learning; Online Education; Privileged Feature Distillation

ACM Reference Format:
Qingyao Li, Wei Xia, Li'ang Yin, Jiarui Jin, and Yong Yu. 2024. Privileged Knowledge State Distillation for Reinforcement Learning-based Educational Path Recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3637528.3671872

1 INTRODUCTION

Educational path recommendation acts as an essential aspect of online education [5, 41], which recommends personalized learning resources to learners. The recommendation process can be modeled as a Markov Decision Process (MDP). Based on this formulation, reinforcement learning (RL) techniques [18] have been widely utilized in this domain, allowing for sequential recommendations and the maximization of long-term rewards [13, 17, 22, 27]. The RL models are usually trained based on interactions with the learners. Due to the high cost of interacting with real learners, these methods typically build a simulator to simulate the learner's knowledge state, and the RL agent recommends knowledge concepts based on observations to maximize the reward (the degree of improvement in the learner's knowledge level).

However, RL models are limited in handling the use cases with multiple learners, primarily due to the diverse dynamics among different learners. Specifically, the diverse dynamics in an educational scenario mean that the learners' learning outcomes may vary even if they follow the same educational path because they are at different knowledge levels. This would cause the RL agent to face an environment with unstable dynamics, which could undermine the training effectiveness [2]. Previous approaches addressed the challenge by employing knowledge tracing (KT) models [22, 27]. These methods typically involve a sequence model, like GRU [7] or LSTM [34], to predict the learner's knowledge state based on his/her learning sequence of concepts, often referred to as *exercise log*. The RL agent then recommends an educational path based on

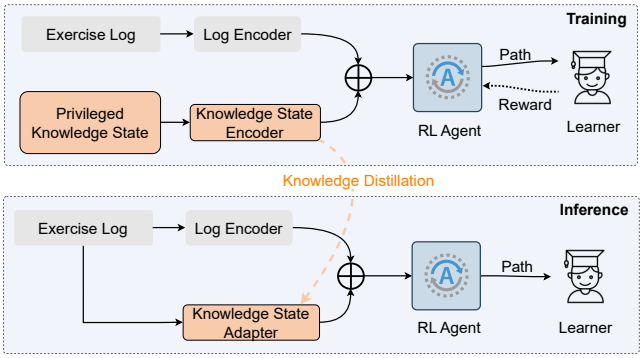


Figure 1: The training-inference paradigm of privileged knowledge state distillation.

the predicted knowledge state. Yet, we contend that KT is unnecessary in this context for two main reasons: 1) While a personalized learning path should reflect the learner's knowledge state, the granular detail KT provides on concept mastery is more than what's needed for making recommendations. 2) Since the predicted knowledge state typically serves as input for another neural network and is converted into a hidden state, which contains partial useful information for recommendation. It might be more efficient to first extract the latent useful information and transfer it to the recommendation model.

This paper presents a unique perspective on the problem. The idea is that we do not need to predict the accurate knowledge state of a learner when doing the recommendation task. By treating the knowledge state as privileged information, we bypass the complexity of its prediction. Our focus shifts to deriving useful insights from the knowledge state and conveying this information to an adapter for times when the privileged knowledge state is not accessible. Essentially, our approach involves three key steps: 1) Creating an oracle model capable of supplying the Privileged Knowledge State (PKS, usually built in the simulator). 2) Learning to harness valuable insights from the PKS for making recommendations when it is available. 3) Distilling the insights gained from the PKS into an adapter, enabling it to gather similar information for recommendations in the absence of the PKS.

In light of this, we propose a privileged feature distillation framework for educational path recommendation called Privileged Knowledge State Distillation (PKSD), where the learner's underlying "actual" knowledge state obtained from the simulator is used as a privileged feature to train RL-based models. Our framework aims at transferring the privileged information extraction ability to the *knowledge state adapter*, enabling the RL agent to adapt to multiple learners effectively. The training-inference paradigm is shown in Figure 1. During training, the privileged knowledge state would be encoded by the *knowledge state encoder* and fed into the RL agent for recommendation. However, it cannot be applied to the real world where the knowledge state cannot be acquired, so we train a *knowledge state adapter*, which aims to extract useful information that's initially derived from the privileged knowledge state from the regular exercise log of the learner. During inference, when the privileged knowledge state is unavailable, the *knowledge state adapter's* output would replace the encoding of the privileged knowledge state to help recommendation. Our approach has advantages in using privileged knowledge state as a more powerful information source, and our model primarily focuses on estimating the output of the *knowledge state encoder* instead of the knowledge state itself, which is easier to learn since the output is the partial information extracted from the privileged knowledge state.

Intuitively, a personalized educational path should be constructed based on two parts of information: 1) The current knowledge state of the learner. 2) The relationships between learning concepts, such as the necessity to understand "addition" before tackling "multiplication". Acknowledging the need to incorporate these intrinsic relationships among knowledge concepts, we further introduce a graph-enhanced version of PKSD, named GEPKSD. This approach combines the knowledge state (considered as a privileged feature) with the knowledge graph. By using a graph neural network to process this combined information, we could gain structure and knowledge state-aware representation of the current state, which helps the RL agent to make recommendations suitable for the learner and adhere to the knowledge structure.

Our main contributions are summarized as follows:

- We propose Privileged Knowledge State Distillation (PKSD), which leverages knowledge states of learners as privileged features and employs a *knowledge state adapter* trained through distillation to help RL agents generate personalized educational paths. To the best of our knowledge, this is the first attempt to incorporate privileged information modeling into educational scenarios.
- To achieve knowledge structure-aware educational path recommendation, we further introduce the Graph-Enhanced PKSD (GEPKSD) framework, which generates a comprehensive representation that encompasses both the learner and knowledge concepts and poses a better way of utilizing the privileged knowledge state.
- We conduct extensive experiments in four simulators. The results demonstrate that our proposed framework can not only prove the performance of representative base RL models but also surpass previous methods, which demonstrates the effectiveness of the framework.

2 RELATED WORK

2.1 Educational Path Recommendation

The goal of educational path recommendation is to customize a sequence of knowledge concepts suggested to individual learners in order to maximize their knowledge advancement [30]. Previous solutions for educational path recommendation can be mainly divided into two categories: non-RL [3, 10, 11, 36] and RL-based methods [13, 17, 22, 24, 27]. Non-RL methods include sequence recommendation [42], graph recommendation [14, 40], and other recommendation methods. These methods usually use the learner's path in the dataset as the label, which does not align with the true objective of promoting the learner's mastery level since the original educational path may not necessarily be the optimal one.

RL-based educational path recommendation models often model the sequence recommendation problem as a Markov Decision Process (MDP) and interact with and train with the simulator's learner

as the environment [25]. The learner's exercise logs are modeled as observed states, and the RL agent takes actions (recommends knowledge concepts) based on states to maximize the reward (the degree of improvement in the learner's knowledge level). One of the most representative works is the CSEAL model proposed in [27], which uses the Deep Knowledge Tracing (DKT) [32] model to predict the knowledge state based on the learner's problem-solving records, and then inputs it as the state into Actor-Critic for recommendation. The other work followed is GEHRL proposed in [24], which is based on hierarchical reinforcement learning that separate the recommendation process into goal planning and goal reaching, improving the efficiency of achieving goals for learners. Moreover, in [22], a model-based RL method is used for recommendation to alleviate the data sparsity problem. Although these RL-based methods have achieved considerable results, using only the knowledge tracing model to extract learner-state information may limit the recommendation effect facing multiple learners. In this paper, we propose using privileged feature distillation to enable the model to recognize learner knowledge states and use them for recommendation.

2.2 Privileged Feature Distillation

Training with privileged features and testing with regular features has been a popular paradigm in recent years [28, 35, 37, 39]. A privileged feature refers to a feature that greatly aids in a task but can only be obtained during the training phase. The idea of privileged feature distillation is to train the model using distillation techniques, enabling it to generate outputs that closely resemble those obtained with privileged features, even when they are unavailable. LUPI proposed in [28] try to minimize the following loss:

$$\min_{W_s}(1-\lambda) * L_c(y, f(X; W_s)) + \lambda * L_d(f(X^*; W_t), f(X; W_s)) \quad (1)$$

where f is the function that map input to the output; L_c is the classification loss; L_d is the distillation loss; X is the regular feature and X^* is the privileged feature. The loss function ensures that the student model, which receives the regular feature as input, produces outputs that closely match the teacher model, which takes the privileged feature as input. After that, [38] proposed a PFD framework for Taobao recommendation, which changes the above loss into the following:

$$\min_{W_s}(1-\lambda) * L_c(y, f(X; W_s)) + \lambda * L_d(f(X^*, X; W_t), f(X; W_s)) \quad (2)$$

The major difference between them is the input of the teacher model. LUPT relies solely on privileged information as input, while PFD combines both standard and privileged information for its input. The paradigm has since been applied across various fields. Wang et al. [35] proposed using a privileged graph to address the cold start issue, and Liu et al. [26] aimed to mitigate recommender system biases with uniform data. Further, integrating reinforcement learning (RL), Kumar et al. [23] introduced the RMA method, enabling legged robots to navigate complex terrains by leveraging environment configurations as privileged information during RL training. Despite achieving impressive results, the exploration and application of privileged information in educational contexts remain largely unexplored. Actually, in the educational path recommendation problem, the knowledge states of learners play a crucial role, particularly for models based on reinforcement learning (RL), which perform optimally in stable environments. Taking the knowledge states of learners can mitigate the challenges posed by dynamic environments, offering a more consistent setting for the RL agent to navigate and improve upon.

3 PROBLEM FORMULATION

This paper addresses the problem of session-based educational path recommendation, where our objective is to recommend a sequential order of knowledge concepts to learners interactively. Normally, at each step of the learning session, the input is the learner's exercise log $\mathcal{H}_t = \{(p_1, score_1), (p_2, score_2), ...(p_t, score_t)\}$, where $score \in \{0, 1\}$ is the learner's feedback of not mastered or mastered. In our work, we additionally take the learner's knowledge state k_t as the privileged information. Here is the definition of it:

Knowledge state. A learner's knowledge state at time step t is a vector k_t with the length equaling the number of knowledge concepts. Each element $k_t[i]$ represents the learner's mastery level of knowledge concept i. It is provided by an oracle model in the simulator. Since all the learner's behavior is simulated based on it, k_t is taken as the learner's underlying "actual" knowledge state.

The model recommends a concept $p_{t+1} \in \mathcal{K}$ to the learner according to $\mathcal{H}_t$ and k_t. The learner would provide the feedback score $score_{t+1}$. Then we update $\mathcal{H}_{t+1}$ by $\mathcal{H}_{t+1} = \mathcal{H}_t \cup (p_{t+1}, score_{t+1})$.

The actual interaction log of a learner in the dataset serves as an indicator of their preferences, making it an appropriate training label for modeling user preferences. However, in educational path recommendation, the path chosen and logged by a learner may not always be the best one. This discrepancy arises because our goal is to find the most effective educational path to improve a learner's proficiency, not just to replicate their preferred choices. To address this issue, we adopt the strategy from Liu et al. [27], using simulators designed to simulate the learner's progress over their education. Below is the explanation of what a simulator is:

Simulator. A simulator functions as a virtual learner that interacts with the recommendation models by performing activities such as completing exercises, taking exams, and advancing their knowledge states. To simulate these actions, a widely used approach involves maintaining a vector that signifies the learner's level of knowledge, as highlighted in [27].

At the beginning of each learning session, the learner undergoes a test and obtains an initial score, denoted as E_{start}. After completing the entire path, the learner takes a final test and obtains a final score, denoted as E_{end}. We aim to maximize $E_{end} - E_{start}$.

4 METHODOLOGY

4.1 Framework Overview

Figure 1 shows the training and inference paradigm. Now we discuss the detailed training process. To ensure that the *knowledge state encoder* accurately estimates valuable information from the privileged knowledge state, we divide the training process into two stages. The first stage is mainly to train the *knowledge state encoder* to learn the latent encoding of the privileged knowledge states while the second stage is mainly for training the *knowledge state adapter* to estimate the latent encoding of the privileged knowledge state when the it is not available (the same situation as the inference time). The overall framework of PKSD is shown in Figure 2.

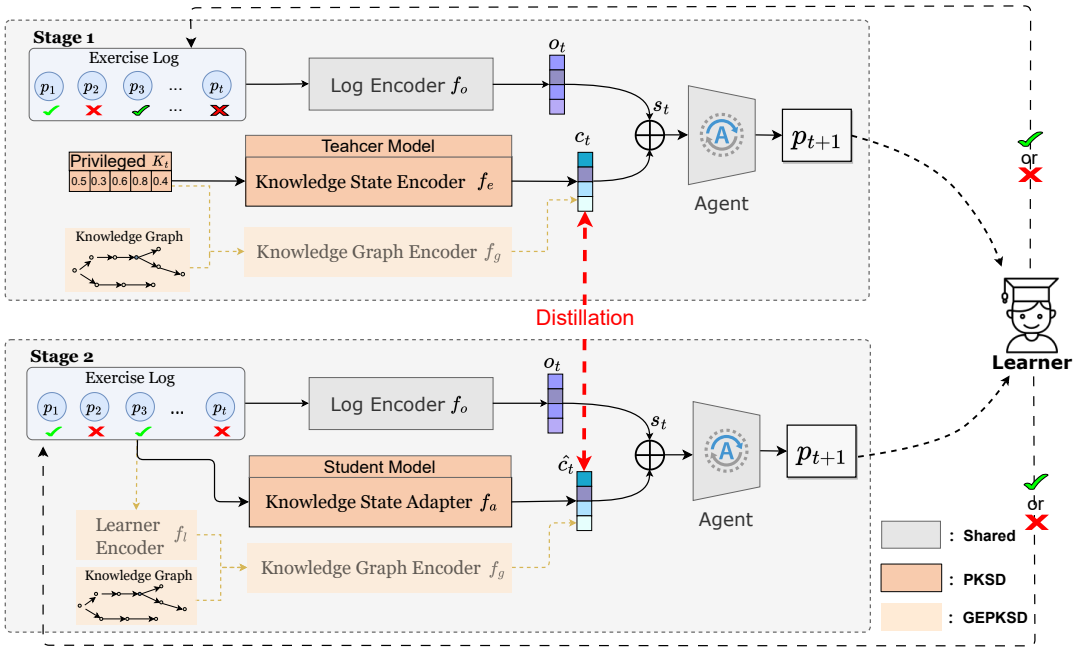


Figure 2: The overall framework of PKSD and GEPKSD. The translucent sections illustrate the GEPKSD framework. In PKSD's stage 1, the privileged knowledge state is encoded using an MLP encoder, while in the graph-enhanced version, it is integrated with the knowledge graph and encoded using a GCN. In stage 2, the difference between PKSD and GEPKSD is also whether the estimation $\hat{c}_t$ is based on the combination of the GRU-encoded exercise log and the knowledge graph or the GRU encoding alone.

In the first stage, the regular and privileged features are combined as input. The regular feature is an RNN-encoded exercise log, which is the same as most RL-based methods. The privileged feature is the encoding of the learner's "actual" knowledge state k_t taken from the simulator. It is encoded into latent vector c_t by an encoding network f_e. The agent decides what knowledge concept to recommend based on the regular feature and privileged feature. In this stage, the agent, trained via RL in simulation, learns to leverage the learner's knowledge state information to generate the educational path and adapt to diverse learners. The *knowledge state encoder* learns to extract useful information from privileged knowledge state to help make recommendations.

In the second stage, we don't take the privileged knowledge state as the direct input. What we need to do is to estimate the latent encoding c_t by $\hat{c}_t$. This estimation task is facilitated by the *knowledge state adapter*, denoted as f_a. During the training phase, we utilize supervised learning to train the *knowledge state adapter* f_a as the privileged knowledge state is available.

4.2 Privileged Knowledge State Distillation

We aim to help the RL agent adapt to the diverse and dynamic knowledge states of learners through privileged knowledge state distillation. To achieve this, we developed a *knowledge state encoder*, which is to extract useful information for recommendation from the privileged knowledge state k_t to a latent vector c_t to help recommendation, and a *knowledge state adapter*, which tries to estimate c_t from the learner's exercise log.

4.2.1 Log encoder f_o. The log encoder is to generate the regular feature. From the learner's exercise log $\mathcal{H}_t$, we try to extract the learner's recent learning interest to help with the recommendation. We use one of the most popular sequence data processing models [8] - Gated recurrent unit (GRU), to model the sequence data.

$$o_t = f_o(\mathcal{H}_t) = GRU(\mathcal{H}_t) \tag{3}$$

$o_t \in \mathbb{R}^{d_o}$ is the learned encoding of exercise log.

4.2.2 Knowledge state encoder f_e. In stage 1, we take the learner's "actual" knowledge state from the simulator as a privileged feature. The *knowledge state encoder* is to extract useful information for the recommendation from the learner's knowledge state vector k_t. Since k_t is a dense vector, we utilize Multilayer Perceptron(MLP) to implement the *knowledge state encoder*.

$$c_t = f_e(k_t) = MLP(k_t) \tag{4}$$

$c_t \in \mathbb{R}^{d_c}$ is the latent encoding of he privileged knowledge state.

The purpose of the log encoder and knowledge state encoder is to extract information useful for the recommendation, so they are input into the agent and trained through RL.

4.2.3 Knowledge State Adapter f_a. In the second stage, when the privileged knowledge state is unavailable, we employ a *knowledge state adapter*, specifically a GRU-based sequence processing model, to estimate the relevant information c_t extracted from the learner's exercise log.

$$\hat{c}_t = f_a(\mathcal{H}_t) = GRU(\mathcal{H}_t) \tag{5}$$

$\hat{c_t} \in \mathbb{R}^{d_c}$ is the estimation of c_t.

4.2.4 Distillation. The objective of the *knowledge state adapter* is to output $\hat{c_t}$ as a substitution for c_t. c_t is the latent encoding extracted by *knowledge state encoder* from the privileged knowledge state and $\hat{c_t}$ is extracted from the regular feature. We take *knowledge state encoder* as the teacher model and the *knowledge state adapter* as the student model and train it through distillation. Specifically, whenever the adapter generates an estimate $\hat{c_t}$, we utilize the actual knowledge state k_t and process it through the *knowledge state encoder* to obtain c_t, which we use as a label for training. Therefore, the distillation loss for the *knowledge state adapter* is the L2 distance between c_t and $\hat{c_t}$:

$$L_d = \|c_t - \hat{c_t}\|_2 \tag{6}$$

It's important to clarify that we do not attempt to predict the learner's entire knowledge state; instead, we focus only on extracting the aspects that are useful for making recommendations. Predicting the full scope of a learner's knowledge state is a more intricate task. Our experiments aim to highlight the distinctions between focusing on useful recommendation information versus predicting the complete knowledge state.

4.3 RL-based Recommendation

An RL agent is developed to decide what action to take under the corresponding state. To conduct an RL-based recommendation model, we need to model the process as a Markov Decision Process(MDP). The key notations are as follows:

- **State s_t.** The state in training stage 1 is defined as the concatenation of the exercise log encoding and privileged knowledge state encoding, formally $s_t = o_t \oplus c_t$. While in training stage 2, the privileged knowledge state encoding is replaced by its estimation $\hat{c_t}$.
- **Action p_t.** The action represents the recommended knowledge concept, $p_t \in \mathcal{K}$.
- **Reward r_t.** The reward is given at the end of the whole path, which represents the learner's knowledge mastery promotion after following the path,

$$r_t = \begin{cases} \dfrac{E_{end} - E_{start}}{E_{sup} - Estart}, & \text{if } t \text{ is the last time step} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

  where E_{sup} is the maximum score of the test; E_{start} and E_{end} are the scores the learner get at the start and the end of the session, respectively.

In general, we take the state s_t at each time step as the input to the RL agent, which outputs the action to be taken at that moment, corresponding to a specific knowledge concept.

$$p_{t+1} \sim \pi(s_t; \theta) \tag{8}$$

where $\pi(s_t; \theta)$ means the policy which we aim to learn of RL agent, and $\sim$ means the p_{t+1} is sampled from the action distribution of the policy. In real-world scenarios, this corresponds to recommending learning resources related to that knowledge concept to the learner. However, for the sake of abstraction, we limit our focus to recommending the knowledge concepts themselves.

4.4 Graph-enhanced Privileged Knowledge State Distillation

Knowing the learner's knowledge state can make the recommended educational paths personalized, while understanding the dependencies between knowledge concepts can improve the alignment of paths with educational principles. Therefore, incorporating the knowledge graph is an intuitive idea for leveraging the privileged knowledge state. Thus, we further propose Graph-Enhanced Privileged Knowledge State Distillation (GEPKSD), which contains a *knowledge graph encoder* f_g to encode both the privileged knowledge state and knowledge graph and a *learner encoder* f_l to replace privileged information during inference.

4.4.1 Knowledge Graph. Knowledge graph in educational scenario represents knowledge concepts' prerequisite relationship [4]. A node in the knowledge graph corresponds to a knowledge concept, and an edge represents the prerequisite relationship between the two connected knowledge concepts.

4.4.2 Knowledge Graph Encoder f_g. We propose to incorporate the knowledge graph by adding the privileged knowledge state as the initial feature of the graph nodes. By leveraging Graph Convolutional Network (GCN), we propagate information throughout the graph to generate a comprehensive representation encompassing both the learner's knowledge state and the relationships between concepts. Any graph embedding models could be used here. We choose GCN for its simplicity and effectiveness in the graph encoding area.

$$h_i^0 = x_i \oplus k_t[i] \tag{9}$$

$$h_i^l = \sigma\left(\frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i \cup \{i\}} w^l h_j^{l-1} + b^l\right) \tag{10}$$

$$c_t = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} h_i^l \tag{11}$$

where $k_t[i]$ is the i-th element of the knowledge state; h_i^0 is the initial feature of node i; $\mathcal{N}_i$ is the neighbor set of node i; $\mathcal{N}$ is the node set; w^l and b^l are the weight and bias of the l-th layer of GCN. The GCN is trained by optimizing the final recommendation objective in an end-to-end manner.

4.4.3 Learner Encoder f_l. Since the knowledge state is not available in stage 2, we need to replace k_t with the encoding of the exercise log. We employ a GRU-based learner encoder to complete the task. The knowledge graph is not privileged information which is then used in both stage 1 and stage 2.

4.5 Optimization

The whole model is updated depending on the RL algorithm used. Generally, there would be a policy net $\pi(s_t; \theta)$ predicting the action and a value net $V(s_t; \phi)$ estimating the future reward of the state, where θ and ϕ are the parameters of the corresponding neural network.

The value net is trained based on the classical mean squared loss (MSE):

$$L_v = \mathbb{E}(||\sum_{i=0}^{T-t}\gamma^i r_{t+i} - V(s_t;\phi)||)^2 \quad (12)$$

And the policy net is trained based on the value net's estimation:

$$L_p = \mathbb{E}[\sum_{t=0}^{T}(r_t + \gamma V(s_{t+1};\phi) - V(s_t;\phi)) \cdot log(\pi(p_{t+1}|s_t;\theta))] \quad (13)$$

After all, in stage 1, all the modules are updated based on L_v and L_p In stage 2, the agent is updated based on L_v and L_p while the log encoder and *knowledge state adapter* are updated based on L_d, L_v and L_p.

5 EXPERIMENT

In this section, we conduct extensive experiments on 4 simulators to evaluate the effectiveness of our model. Our experiments aim at answering the following questions:

- **RQ1.** Could PKSD/GEPKSD framework improve the educational path recommendation performance across different base RL agents?
- **RQ2.** How does PKSD/GEPKSD perform compared with previous educational path methods?
- **RQ3.** Is distillation on the latent encoding of the privileged knowledge state a better way for recommendation than tracing the exact knowledge state?
- **RQ4.** In GEPKSD, is the privileged knowledge state important or is it working only for using the knowledge graph?
- **RQ5.** Could our approach handle multiple learners with diverse knowledge states better?

5.1 Experiment Setup

5.1.1 Datasets. Our experiments are based on three public educational datasets: *Junyi*[1] *ASSISTments2009*[2] and *ASSITments2015*[3]. *Junyi* dataset provides a prerequisite graph of the knowledge concepts while others don't, so we build a transition graph [31] as an estimation of the prerequisite graph. The statistics of the datasets are presented in Table 1.

Table 1: Dataset Statistics

Dataset	Junyi	ASSIST15	ASSIST09
#Concepts	835	100	167
#Learners	525,061	69,675	4,217
#Records	21,460,249	2,420,200	346,860
Positive label rate	54.38%	73.17%	63.81%

5.1.2 Simulators and Oracle Models. We construct two types of simulators proposed in previous work [27]: the Knowledge Structure-based Simulator (KSS) and the Knowledge Evolution-based Simulator (KES). The KSS is based on Item Response Theory and a rule-based formula to determine the learner's knowledge state changes. KES is a data-driven simulator that employs a trained Deep Knowledge Tracing (DKT) model on a specific dataset to simulate the learner's knowledge growth and learning feedback. Specifically, each simulated learner is initialized based on the first 60% of the exercise log in the dataset(not available for the recommendation model). Subsequently, as the recommendations are made, new exercise logs are incorporated, and the DKT is used to update the learner's knowledge state. We constructed 4 simulators based on 3 datasets: KSS (rule-based), KES-junyi, KES-ASSIST15, and KES-ASSIST09. The KES simulators are trained using three different datasets to serve as the foundations for their respective DKT-based simulators.

In the simulator, the algorithm that determines a learner's knowledge state serves as the oracle providing privileged knowledge states. For KSS, this is represented by the formulas deciding a learner's knowledge, and in KES, it's the environment's DKT model. The DKT model is trained on the entire dataset, making it a reliable oracle. However, DKT models outside the simulator can only use the data interacting with the simulator to train, making the environment's DKT-determined knowledge states the decisive and privileged ones.

5.1.3 Methods for comparison. Since our framework is model-agnostic, we first combine our method and several representative RL methods to show the effectiveness in prompting RL methods' performance:

- DQN [29]: One of the basic RL algorithms, which mainly estimates the value of each state and action pair by a neural network.
- AC [21]: Vanilla Actor-Critic algorithm that contains a policy net to output the action distribution and a value net to determine the value of the state.
- PPO [33]: An advanced RL algorithm that constrains the update magnitude of the policy network so that the training is more stable.
- TD3 [12]: Twin Delayed DDPG is a model-free off-policy RL algorithm that uses twin critics to estimate the Q-values and reduces overestimation bias in Q-learning.
- SAC [15]: SAC (Soft Actor-Critic) is an off-policy RL algorithm that incorporates an entropy regularization term to balance exploration and exploitation, allowing for more stable and efficient learning.

Then, we further compare with two classes of previous educational path recommendation methods: Non-RL methods and RL-based methods:

(1) Non-RL methods:

- GRU4Rec [16]: An RNN-based model for the session-based recommendation, where the session data is first one-hot encoded and then embedded. This processed data is fed into a GRU network, which outputs the probability of each item being recommended.
- SASRec [19]: A self-attention-based sequential recommendation model that effectively captures long-term dependencies but focuses on a few important items in sequential data for accurate item recommendation.

[1] https://www.kaggle.com/datasets/junyiacademy/learning-activity-public-dataset-by-junyi-academy
[2] https://www.kaggle.com/datasets/junyiacademy/learning-activity-public-dataset-by-junyi-academy
[3] https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data

Table 2: Comparison between PKSD/GEPKSD and backbone RL methods. "*" denotes that the improvement are significant at level of $p < 0.05$ with paired t-test (PKSD compares with base and GEPKSD compares with PKSD).

Base Models	KSS			Junyi			ASSISTments2015			ASSISTments2009		
	base	PKSD	GEPKSD	base	PKSD	GEPKSD	base	PKSD	GEPKSD	base	PKSD	GEPKSD
SAC	0.3738	0.4199*	**0.5558***	-0.2224	0.0505*	**0.2216***	0.2280	0.3918*	**0.6947***	0.6545	0.6614*	**0.6638***
TD3	0.3732	0.4527*	**0.4842***	-0.1405	-0.1046*	**-0.0598***	0.2972	0.5160*	**0.5721***	0.2468	0.4270*	**0.6671***
DQN	0.3855	0.4191*	**0.4476***	0.2386	0.2982*	**0.3215***	0.3675	0.6004*	**0.7020***	0.0503	0.4511*	**0.5857***
AC	0.5051	0.5454*	**0.5539***	0.3093	0.7600*	**0.7746***	0.7282	0.8181*	**0.8457***	0.6619	0.6773*	**0.6782***
PPO	0.7337	0.7546*	**0.7723***	0.2399	0.3260*	**0.3309***	0.8856	0.9176*	**0.9213***	0.6602	0.6673*	**0.6676**

Table 3: Comparison between PKSD/GEPKSD and previous methods. "*" denotes that the improvement are significant at level of $p < 0.05$ with paired t-test comparing with previous methods' best performance.

Simulators	Non-RL Methods			RL-based Methods				Ours			
	GRU4Rec	SASRec	DKTRec	CB	RLtutor	CSEAL	GEHRL	PKSD(PPO)	GEPKSD(PPO)	PKSD(GEHRL)	GEPKSD(GEHRL)
KSS	0.1905	0.7183	0.3069	0.4779	0.6227	0.6833	0.8331	0.7546	0.7723	0.8480	**0.8536***
Junyi	0.0009	-0.1518	-0.1471	0.1730	-0.1306	0.3942	0.5830	0.3260	0.3309	0.6398	**0.6568***
ASSISTments2015	0.1025	0.1975	0.5280	0.6413	0.8713	0.8015	0.8121	0.9176	**0.9213***	0.8265	0.8825
ASSISTments2009	-0.2347	-0.5222	-0.0814	0.3586	0.6396	0.6347	0.5758	0.6673	**0.6676***	0.6004	0.6009

- DKTRec: Use a DKT to predict the learner's knowledge state on the knowledge concepts and recommend the one closest to 0.5

(2) RL-based methods:

- CB [17]: An RL-based method that takes the educational path recommendation problem as a contextual bandits problem and applies Q-learning to solve it.
- CSEAL [27]: Actor-Critic based educational path recommendation model, which contains a cognitive navigation module so that the path is aligned with educational principles.
- RLTutor [22]: Utilizing model-based RL, which builds an inner model DAS3H [6] to simulate the learner and apply PPO algorithm to build the RL agent.
- GEHRL [24]: Utilizing hierarchical reinforcement learning as the base with graph-based candidate selector to coordinate the learning path with learning structure.

We evaluate these methods with the average promotion of the mastery of the learning goals after path recommendation, which is given by the converge reward from the simulators.

5.1.4 Implementation Details. We use Adam [20] as the optimizer, and the learning rate is set to be 5×10^{-5}. For each model, we run for three random seeds: 1, 5, 10, and calculate the average mastery promotion on 2000 learner samples across three seeds. The path length is set to 20. For the models that employ DKT, we perform pre-training by utilizing the data interacting with the simulator. In the case of KSS, we pre-train the DKT model and other non-RL methods using the experimental data obtained from the interaction of the trained PPO with the simulator. The source code of our proposed approach is available[4].

[4]Source code for model implementations: https://github.com/mindspore-lab/models/tree/master/research/huawei-noah/PKSD

5.2 Overall Performance

5.2.1 Improvement over Backbone RL Models (RQ1). We pick five representative RL methods as the backbone to assess the effects of PKSD/GEPKSD, with the results displayed in Table 2. Key observations include: 1) The significant performance enhancement with PKSD framework. For example, DQN's performance on ASSIST09 improved markedly from 0.0503 to 0.4511 with PKSD, highlighting the effectiveness of introducing privileged knowledge states. 2) GEPKSD, incorporating a knowledge graph, further boosts PKSD's impact across all methods, demonstrating the advantage of combining learner-specific information (privileged knowledge state) and knowledge domain insights (knowledge graph) for more effective recommendation paths based on learner states and knowledge structures.

5.2.2 Improvement over Other Recommendation Methods (RQ2). Next, we compare PKSD/GEPKSD with other recommendation baselines. We also integrate our framework with one of the baselines GEHRL, proposed by Li et al. [24], to show the effectiveness of our framework with the advancing RL-based educational path recommendation method. The outcomes are presented in Table 3, revealing that: 1) The best results were achieved by GEPKSD, with PKSD also surpassing previous baselines, including non-RL and RL-based methods, demonstrating the advancement of our approach comparing with previous methods. 2) Generally, RL-based methods outperformed non-RL methods, indicating the limitations of merely mimicking existing learners' paths. In contrast, RL methods, through exploration and interaction, can discover more effective strategies. 3) The combination of PKSD and GEPKSD with strong backbone models like PPO and GEHRL led to superior performance, which shows that our framework possesses strong model-agnostic characteristics; it not only improves the effectiveness of base RL methods but also synergizes with advanced RL-based methods to achieve superior results.

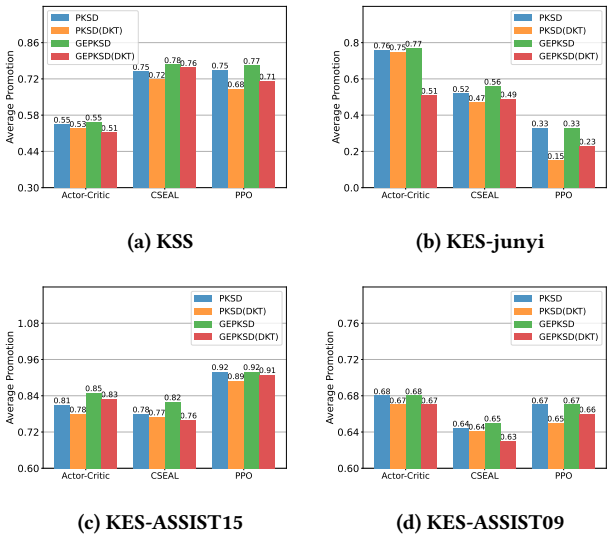
Table 4: The DBI of different representation learned in three datasets.

Embeddings	KSS	Junyi	ASSIST15	ASSIST09
Knowledge States	1.2220	1.1813	1.4018	1.2803
Extracted Latent Vectors	0.6155	0.5985	0.4910	0.4472

5.3 Compare with knowledge tracing-based methods (RQ3)

We argued that for the task of recommending learning paths, it is unnecessary to predict the learner's exact knowledge state. To support this claim, we perform experiments to investigate the feature of the knowledge states and the extracted latent encodings.

First of all, we explore the inherent properties of the privileged knowledge states and the latent encodings. Specifically, we calculate the Davies-Bouldin Index (DBI) [9] of 1000 learners' knowledge states and extracted latent vectors c_t. DBI assesses the data separation state between clusters. A lower DBI indicates better-separated classes. We use K-Means to cluster the embeddings and calculate DBIs, as shown in Table 4. We can see that the extracted latent vectors have a much lower DBI than the original knowledge states, which indicates that a better clustering pattern that is easier to catch.



(a) KSS (b) KES-junyi

(c) KES-ASSIST15 (d) KES-ASSIST09

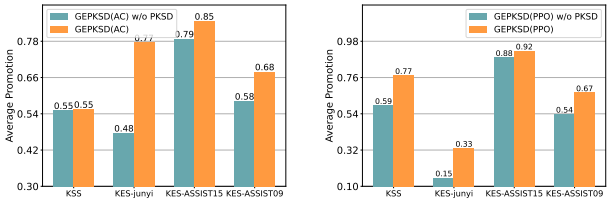
Figure 3: Compare with DKT-based PKSD

Furthermore, we conduct experiments that trace the knowledge state directly instead of c_t as a knowledge tracing way to leverage the privileged knowledge state. For the graph-enhanced version GEPKSD, we use DKT to estimate the learner's knowledge state in stage 2 to combine with the knowledge graph.

The results are shown in Figure 3. It can be observed that using the DKT model to estimate the learner's exact knowledge state results in a decline in performance. This decline is particularly evident on the KES-junyi dataset, which consists of 835 knowledge concepts. Due to the larger number of knowledge concepts, the prediction difficulty for the DKT model is greater, leading to a more significant decrease in performance.

5.4 PKSD for GEPKSD (RQ4)

GEPKSD enhances the recommendation capabilities of RL agents by incorporating a graph that considers not only the learner's knowledge state but also the dependency relationships between concepts on the knowledge graph. To ascertain whether the improvements are due to the proposed PKSD or the introduction of the graph neural network, experiments comparing GEPKSD with and without PKSD are conducted. The results are depicted in Figure 4.



(a) Actor-Critic (b) PPO

Figure 4: Compare the performance of with/without PKSD on GEPKSD.

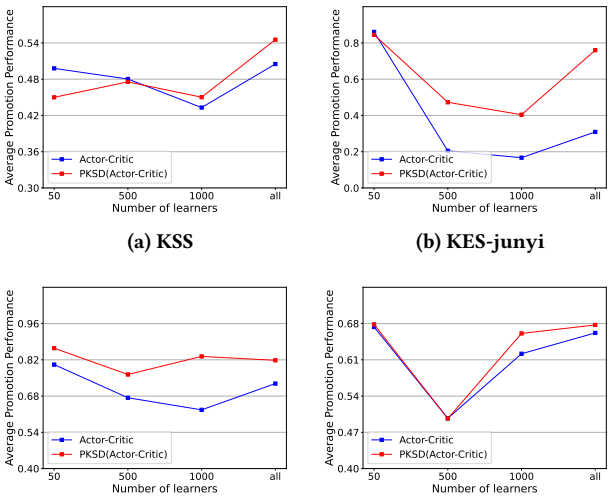
Removing PKSD significantly diminishes GEPKSD's performance across four simulators. This suggests that merely incorporating a graph neural network without a meaningful training task is ineffective. The distillation process provides a clear goal for the knowledge state adapter, which is to estimate information related to the learner's knowledge level that is crucial for recommendations. This is vital since the recommended learning path should be primarily based on the learner's knowledge state and the structure of the knowledge itself.

5.5 Performance across various number of learners (RQ5)

To illustrate that our proposed PKSD framework could adapt to multiple learners better, we conduct experiments on different numbers of learners. The result is shown in 5. The experimental results reveal that the advantages of our method become increasingly evident as the number of learners grows. This is not only due to the privileged knowledge state providing stronger signals for training compared to the learner's exercise log but also because the *knowledge state adapter* effectively helps the RL agent identify differences between learners to enhance generalization performance.

As the number of learners increases, the effectiveness of PKSD initially decreases and then rises. This pattern occurs because when the learner population starts to grow, the dynamic changes in learners' knowledge states make it challenging for the RL agent to adapt, leading to a dip in performance. However, as the number of learners reaches a certain level, the volume of data becomes sufficient for the *knowledge state adapter* to be fully trained, resulting in a recovery in effectiveness. While vanilla RL methods without PKSD exhibit a similar trend, there is a noticeable difference in performance when dealing with a larger learner count. These methods struggle to accommodate learners with dynamic knowledge states without the *knowledge state adapter* and the distillation process.

The experiments generally demonstrate that our model could adapt to multiple learners better.



(a) KSS (b) KES-junyi

(c) KES-ASSIST15 (d) KES-ASSIST09

Figure 5: Performances with various number of learners.

6 CONCLUSION

We propose a novel framework called Privileged Knowledge State Distillation (PKSD) for educational path recommendation. We transfer the privileged information knowledge through distillation to the knowledge state adapter and combine it with the knowledge graph to generate a comprehensive representation, enabling the RL agent to adapt more effectively to learners with diverse knowledge states. This marks the first attempt to incorporate privileged knowledge distillation into educational path recommendations. Extensive experiments demonstrate the framework's effectiveness in efficiently recommending educational paths to enhance learners' knowledge levels.

ACKNOWLEDGMENTS

The SJTU team is partially supported by Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102) and National Natural Science Foundation of China (62177033). The work is also sponsored by Huawei Innovation Research Program. We gratefully acknowledge the support of MindSpore [1], CANN (Compute Architecture for Neural Networks), and Ascend AI Processor used for this research.

REFERENCES

[1] 2020. MindSpore. https://www.mindspore.cn/
[2] Mohammadamin Barekatain, Ryo Yonetani, and Masashi Hamaya. 2019. Multipolar: Multi-source policy aggregation for transfer reinforcement learning between diverse environmental dynamics. *arXiv preprint arXiv:1909.13111* (2019).
[3] Cun-Ling Bian, De-Liang Wang, Shi-Yu Liu, Wei-Gang Lu, and Jun-Yu Dong. 2019. Adaptive learning path recommendation based on graph theory and an improved immune algorithm. *KSII Transactions on Internet and Information Systems (TIIS)* 13, 5 (2019), 2277–2298.
[4] Haw-Shiuan Chang, Hwai-Jung Hsu, and Kuan-Ta Chen. 2015. Modeling Exercise Relationships in E-Learning: A Unified Approach.. In *EDM*. 532–535.
[5] Chih-Ming Chen. 2009. Ontology-based concept map for planning a personalised learning path. *British Journal of Educational Technology* 40, 6 (2009), 1028–1058.
[6] Benoit Choffin, Fabrice Popineau, Yolaine Bourda, and Jill-Jenn Vie. 2019. DAS3H: modeling student learning and forgetting for optimally scheduling distributed practice of skills. *arXiv preprint arXiv:1905.06873* (2019).
[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).
[8] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.
[9] David L Davies and Donald W Bouldin. 1979. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence* 2 (1979), 224–227.
[10] Pragya Dwivedi, Vibhor Kant, and Kamal K Bharadwaj. 2018. Learning path recommendation based on modified variable length genetic algorithm. *Education and information technologies* 23, 2 (2018), 819–836.
[11] Lumbardh Elshani and Krenare Pireva Nuçi. 2021. Constructing a personalized learning path using genetic algorithms approach. *arXiv preprint arXiv:2104.11276* (2021).
[12] Scott Fujimoto, Herke Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*. PMLR, 1587–1596.
[13] Jibing Gong, Yao Wan, Ye Liu, Xuewen Li, Yi Zhao, Cheng Wang, Yuting Lin, Xiaohan Fang, Wenzheng Feng, Jingyi Zhang, et al. 2022. Reinforced moocs concept recommendation in heterogeneous information networks. *ACM Transactions on the Web* (2022).
[14] Jibing Gong, Shen Wang, Jinlong Wang, Wenzheng Feng, Hao Peng, Jie Tang, and Philip S Yu. 2020. Attentional graph convolutional networks for knowledge concept recommendation in moocs in a heterogeneous view. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*. 79–88.
[15] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*. PMLR, 1861–1870.
[16] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).
[17] Wacharawan Intayoad, Chayapol Kamyod, and Punnarumol Temdee. 2020. Reinforcement learning based on contextual bandits for personalized online learning recommendation systems. *Wireless Personal Communications* 115, 4 (2020), 2917–2932.
[18] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research* 4 (1996), 237–285.
[19] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.
[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[21] Vijay Konda and John Tsitsiklis. 1999. Actor-critic algorithms. *Advances in neural information processing systems* 12 (1999).
[22] Yoshiki Kubotani, Yoshihiro Fukuhara, and Shigeo Morishima. 2021. RLTutor: Reinforcement Learning Based Adaptive Tutoring System by Modeling Virtual Student with Fewer Interactions. *arXiv preprint arXiv:2108.00268* (2021).
[23] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. 2021. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034* (2021).
[24] Qingyao Li, Wei Xia, Li'ang Yin, Jian Shen, Renting Rui, Weinan Zhang, Xianyu Chen, Ruiming Tang, and Yong Yu. 2023. Graph Enhanced Hierarchical Reinforcement Learning for Goal-oriented Learning Path Recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 1318–1327.
[25] Yuanguo Lin, Yong Liu, Fan Lin, Lixin Zou, Pengcheng Wu, Wenhua Zeng, Huanhuan Chen, and Chunyan Miao. 2021. A survey on reinforcement learning for recommender systems. *arXiv preprint arXiv:2109.10665* (2021).
[26] Dugang Liu, Pengxiang Cheng, Zinan Lin, Jinwei Luo, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. 2022. KDCRec: Knowledge Distillation for Counterfactual Recommendation Via Uniform Data. *IEEE Transactions on Knowledge and Data Engineering* (2022).
[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] David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. 2015. Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643* (2015).
[29] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[30] Amir Hossein Nabizadeh, José Paulo Leal, Hamed N Rafsanjani, and Rajiv Ratn Shah. 2020. Learning path personalization and recommendation methods: A survey of the state-of-the-art. *Expert Systems with Applications* 159 (2020), 113596.

[31] Hiromi Nakagawa, Yusuke Iwasawa, and Yutaka Matsuo. 2019. Graph-based knowledge tracing: modeling student proficiency using graph neural network. In *IEEE/WIC/ACM International Conference on Web Intelligence*. 156–163.

[32] 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).

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

[34] Ralf C Staudemeyer and Eric Rothstein Morris. 2019. Understanding LSTM–a tutorial into long short-term memory recurrent neural networks. *arXiv preprint arXiv:1909.09586* (2019).

[35] Shuai Wang, Kun Zhang, Le Wu, Haiping Ma, Richang Hong, and Meng Wang. 2021. Privileged graph distillation for cold start recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1187–1196.

[36] Zhengyang Wu, Ming Li, Yong Tang, and Qingyu Liang. 2020. Exercise recommendation based on knowledge concept prediction. *Knowledge-Based Systems* 210 (2020), 106481.

[37] Ruobing Xie, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. 2022. A peep into the future: Adversarial future encoding in recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1177–1185.

[38] Chen Xu, Quan Li, Junfeng Ge, Jinyang Gao, Xiaoyong Yang, Changhua Pei, Fei Sun, Jian Wu, Hanxiao Sun, and Wenwu Ou. 2020. Privileged features distillation at taobao recommendations. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2590–2598.

[39] Guan Yang, Minghuan Liu, Weijun Hong, Weinan Zhang, Fei Fang, Guangjun Zeng, and Yue Lin. 2022. Perfectdou: Dominating doudizhu with perfect information distillation. *Advances in Neural Information Processing Systems* 35 (2022), 34954–34965.

[40] Hang Yin, Zhiyu Sun, Yanchun Sun, and Gang Huang. 2021. Automatic Learning Path Recommendation for Open Source Projects Using Deep Learning on Knowledge Graphs. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 824–833.

[41] Qian Zhang, Jie Lu, and Guangquan Zhang. 2021. Recommender Systems in E-learning. *Journal of Smart Environments and Green Computing* 1, 2 (2021), 76–89.

[42] Yuwen Zhou, Changqin Huang, Qintai Hu, Jia Zhu, and Yong Tang. 2018. Personalized learning full-path recommendation model based on LSTM neural networks. *Information sciences* 444 (2018), 135–152.