Set-to-Sequence Ranking-Based Concept-Aware Learning Path Recommendation

Xianyu Chen[1], Jian Shen[1], Wei Xia[2], Jiarui Jin[1], Yakun Song[1], Weinan Zhang[1*], Weiwen Liu[2],
Menghui Zhu[1], Ruiming Tang[2], Kai Dong[3], Dingyin Xia[3], Yong Yu[1*]

[1] Shanghai Jiao Tong University
[2] Huawei Noah's Ark Lab
[3] Huawei Technologies Co Ltd
{xianyujun,r_ocky,jinjiarui97,ereboas,wnzhang,zerozmi7}@sjtu.edu.cn, yuyong@apex.sjtu.edu.cn
{xiawei24,liuweiwen8,tangruiming,dongkai4,xiadingyin}@huawei.com

Abstract

With the development of the online education system, personalized education recommendation has played an essential role. In this paper, we focus on developing path recommendation systems that aim to generate and recommend an entire learning path to the given user in each session. Noticing that existing approaches fail to consider the correlations of concepts in the path, we propose a novel framework named **S**et-to-Sequence **R**anking-Based **C**oncept-Aware Learning Path Recommendation (**SRC**), which formulates the recommendation task under a set-to-sequence paradigm. Specifically, we first design a concept-aware encoder module that can capture the correlations among the input learning concepts. The outputs are then fed into a decoder module that sequentially generates a path through an attention mechanism that handles correlations between the learning and target concepts. Our recommendation policy is optimized by policy gradient. In addition, we also introduce an auxiliary module based on knowledge tracing to enhance the model's stability by evaluating students' learning effects on learning concepts. We conduct extensive experiments on two real-world public datasets and one industrial dataset, and the experimental results demonstrate the superiority and effectiveness of SRC. Code now is available at https://gitee.com/mindspore/models/tree/master/research/recommend/SRC.

Introduction

Different from providing the same learning content for all students in each classroom session in traditional learning, adaptive learning aims to tailor different learning objectives to meet the individual needs of different learners (Carbonell 1970). Existing recommendation methods of learning content can be summarized into two categories: (i) Step by step, the following learning item is recommended for students in real-time, and the interaction of each step (i.e., students' answers) will be integrated into the recommendation for the next step (Liu et al. 2019; Cai, Zhang, and Dai 2019; Huang et al. 2019). (ii) Plan a certain length of the learning path for students at one time. The latter is because users sometimes want to know the entire learning path at the beginning (for example, universities need to organize courses for students) (Joseph, Abraham, and Mani 2022; Chen et al. 2022;

Shao, Guo, and Pardos 2021; Bian et al. 2019; Dwivedi, Kant, and Bharadwaj 2018). As the latter direction is more restricted and complex (e.g., larger search space, less available feedback), it is more challenging and is also the main focus of this paper. Previous studies (Piaget and Duckworth
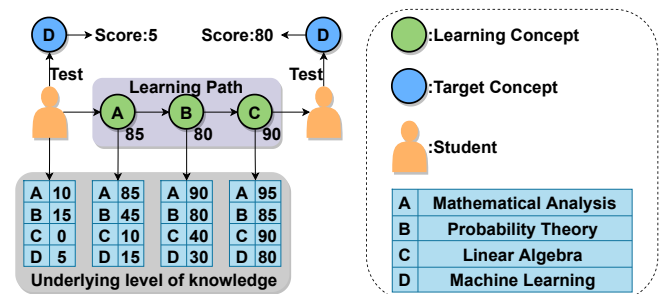

Figure 1: Illustration of the student learning process to improve a student's mastery of concept D by learning the path composed of three concepts A, B, and C. The student is given a test before and after the path to knowing his mastery of D. The bottom four tables show the student's potential mastery of all concepts at each step, which is not accessible during the training process. We can only know the mastery of the current learning concept. For example, after learning B, we know his mastery of B is 80.

1970; Inhelder et al. 1976; Pinar et al. 1995) reveal that cognitive structure greatly influences adaptive learning, which includes both the relationship between items (e.g., premise relationship and synergy relationship) and the characteristics of students' dynamic development with learning. Most existing methods to solve learning path planning are either based on a knowledge graph (or some relationship between concepts) to constrain path generation (Liu et al. 2019; Shi et al. 2020; Wang et al. 2022), or based on collaborative filtering of features to search for paths (Joseph, Abraham, and Mani 2022; Chen et al. 2022; Nabizadeh, Jorge, and Leal 2019). However, these models can not penetrate into the important features of the cognitive structure perfectly, and the model is relatively simple, resulting in the path generated either with a low degree of individuation or with a poor learning effect. From this perspective, we argue that *how to effectively mine the correlations among concepts and the important charac-*

teristics of students in learning path planning is still challenging, and summarize the specific challenges as follows:

- **(C1)** How to effectively explore the correlations among concepts? There may be complex and diverse correlations between concepts, such as prerequisite relationship and synergy relationship, which will affect students' learning of concepts (Tong et al. 2020; Liu et al. 2019). As shown in Figure 1, mastery, of course, A (Mathematical Analysis) is of greater help to mastery of course B (Probability Theory), and of less help to mastery, of course, C (Linear Algebra). Therefore, it should be taken into account when planning the learning path.

- **(C2)** How to evaluate and optimize the generation algorithm by effectively using the students' learning effect on the target concepts? As shown in Figure1, we expect students to achieve the best improvement in the target concept D (Machine Learning). However, the existing path recommendation algorithms either do not use this feedback but use indirect factors such as similarity degree and occurrence probability (Joseph, Abraham, and Mani 2022; Shao, Guo, and Pardos 2021), or lack of excellent generation algorithms (Zhou et al. 2018; Nabizadeh, Jorge, and Leal 2019). As a result, it is difficult for them to provide an efficient learning path. This is because it is still challenging to optimize a path using only feedback that is available at the end of the path. In contrast, in the stepwise recommendation scenario, immediate feedback can be obtained at the end of each step, which allows some more advanced reinforcement learning (RL) algorithms (Sun et al. 2021; Li et al. 2021) to be applied.

- **(C3)** How can student feedback on learning concepts be incorporated into the model? As shown in Figure 1, students have different learning feedback for concepts A, B, and C on the path after learning. In the field of knowledge tracing (KT), this information plays a great role in modeling students' knowledge levels. Many models (Piech et al. 2015; Yang et al. 2020; Zhang et al. 2017) take students' past answers as features to predict the current answer. In (Liu et al. 2019), its DKT (Piech et al. 2015) module used this information to trace students' knowledge levels in real-time to adjust recommendations for the next step. However, in path recommendation, this feedback can only be obtained after the path ends, so the above approach is difficult to implement here.

To address these challenges, we propose a novel framework Set-to-Sequence Ranking-based Concept-aware Learning Path Recommendation (**SRC**). We formulate the learning path recommendation task as a set-to-sequence paradigm. In particular, firstly, in order to mine the correlation between concepts (**C1**), we design a concept-aware encoder module. This module can globally calculate the correlation between each learning concept and other learning concepts in the set so as to get a richer representation of the concept. At the same time, in the decoder module, on the one hand, we use a recurrent neural network to update the state of students; on the other hand, we use the attention mechanism to calculate the correlation between the remaining learning concepts in the set and the target concepts, so as

to select the most suitable concept in the current position of the path. Secondly, we need to effectively utilize feedback on the target concepts (**C2**). Since the feedback is generally continuous and considering the large path space, the policy gradient algorithm is more suitable in this case. Thus the correlation between the learning concept and the target concepts calculated by the previous decoder can be expressed in the form of selection probability. So we get a parameterized policy, and we can update the model parameters in a way that maximizes the reward. Finally, we designed an auxiliary module to utilize feedback on learning concepts (**C3**). Similar to the KT task, the student state updated by the previous decoder at each step is fed into an MLP to predict the student's answer at that step. In this way, students' feedback on the learning concepts can participate in the updating of model parameters to enhance the stability of the algorithm.

Related Works

Learning Path Recommendation. A class of branches (Joseph, Abraham, and Mani 2022; Chen et al. 2022; Zhou et al. 2018; Nabizadeh, Jorge, and Leal 2019; Liu and Li 2020; Shao, Guo, and Pardos 2021) in existing methods models the task as a general sequence recommendation task, which is dedicated to reconstructing the user's behavior sequence. For example, in Zhou et al. (2018), KNN (Cover and Hart 1967) is used to complete the collaborative filtering and then RNN is used to estimate the learning effect; in Shao, Guo, and Pardos (2021), the BERT (Devlin et al. 2018) paradigm is directly used to solve this problem. Another branch (Liu et al. 2019; Shi et al. 2020; Wang et al. 2022; Zhu et al. 2018) focuses on mining the role of knowledge structure. For example, Zhu et al. (2018) formulates some rules based on the knowledge structure to constrain the generation of paths. In general, most of the above methods fail to take full advantage of student feedback on the target concepts. One of the better ones is Liu et al. (2019), which uses this feedback through reinforcement learning methods to optimize the generative model. However, on the one hand, it can obtain real-time feedback on learning concepts, and its application scenario is actually a step-by-step recommendation; on the other hand, it uses the concept relationship graph as a rule to constrain path generation without mining deeply into their correlations, which makes it challenging to apply in the general case. In our method, we use an attention mechanism to mine inter-concept correlations and make full use of various feedback from students to optimize the modeling of correlations, which makes our method more general.

Learning Item Recommendation. In step-by-step learning item recommendations, immediate feedback is available. This allows them (Cai, Zhang, and Dai 2019; Huang et al. 2019; Sun et al. 2021; Li et al. 2021) to use more complex RL algorithms. Such as Sun et al. (2021) use the DQN(Mnih et al. 2013), and Cai, Zhang, and Dai (2019) use the Advantage Actor-Critic. Our method also uses policy gradient in RL for optimization, but since we have no immediate feedback, only delayed feedback after the path ends, training may be more difficult. Therefore we introduce the KT auxiliary task to enhance the model stability.

Set-to-Sequence Formulation. The set-to-sequence task

aims to permute and organize a set of unordered candidate items into a sequence whose solutions can be roughly divided into three fields: point-wise, pair-wise, and list-wise. Among them, the point-wise method is the most widely used, which is designed to score each item individually, and then rank the items in descending order of their scores (Friedman 2001). The pair-wise methods (Burges et al. 2005; Joachims 2006) do not care about the specific score of each item. Instead, formulate the problem pair-wise, focusing on predicting the relative orders among each pair of items. The list-wise algorithms (Burges 2010; Cao et al. 2007; Xia et al. 2008) treat the entire sequence as a whole, which allows the model to mine the deep correlations among the items carefully. Noticing that students' feedback on a concept is likely to be significantly affected by the other concepts on the same path, we here design our model in a list-wise manner.

The main difficulty with list-wise is that the sorting process is not completely differentiable because there are no gradients available for sorting operations (Xia et al. 2008). One solution randomly optimizing the ranking network by continuous relations (Grover et al. 2019; Swezey et al. 2020). Another class of branches, named Plackett-Luce (PL) ordering model (Burges 2010; Luce 2012; Plackett 1975), represents ordering as a series of decision problems, where each decision is made by softmax operation. Its probabilistic nature leads to more robust performance (Bruch et al. 2020), but computing the gradient of the PL model requires iterating over every possible permutation. A solution proposed in the recent literature (Oosterhuis 2021) is the policy gradient algorithm (Williams 1992).

Problem Formulation

Consider a student u, whose historical sequence of concepts learning is $H = \{h_1, h_2, \cdots, h_k\}$. The record $h_t = \{c_t, y_t\}$ of each time t includes the learned concept c_t and the degree of mastery y_t of the concept. Now given a set $S = \{s_1, s_2, \cdots, s_m\}$ consisting of m candidate concepts, the student u is to learn n non-repetitive concepts from S in some order (hence $m \geq n$). Through the study of such a learning path $\pi = \{\pi_1, \pi_2 \cdots, \pi_n\}$, he can improve his mastery of some target concepts $T = \{t_1, t_2, \cdots\}$. Following (Liu et al. 2019), we quantify the learning effect as

$$E_T = \frac{E_e - E_b}{E_{sup} - E_b} \qquad (1)$$

where E_e and E_b represent the student's mastery of the target concepts before and after the path π (which can be obtained through exams), and E_{sup} represents the upper bound of mastery. At the same time, we can also observe the students' mastery $Y_\pi = \{y_{\pi_1}, y_{\pi_2}, \cdots, y_{\pi_n}\}$ of learning concepts after the end of the path. Then, we can formulate our problem as follows:

Definition 1. *Learning Path Recommendation. Given a student's historical learning sequence H, target concepts T, and candidate concepts set S, it is required to select n concepts from S without repetition and rank them to generate a path π to recommend to the student. The ultimate goal is to maximize the learning effect E_T at the end of the path.*

SRC Framework

Figure 2 shows the overview framework of our SRC model. As shown, first we design a concept-aware encoder to model the correlations among candidates' learned concepts to obtain their global representations. Then in the decoder module, we use the recurrent neural network to model the knowledge state of the students along the path and calculate the correlation between the learning concepts and the target concepts through the attention mechanism to determine the most suitable concept for the position. In addition to this, based on the knowledge state obtained in the decoder, we further predict the student's answer to the learning concepts. At the end of the learning path, we pass the obtained feedback E_T and Y_π to the model to optimize the parameters.

Encoder

First, for each concept s_i in the candidate concept set S, we access the embedding layer to obtain its continuous representation x_{s_i}. However, as we discussed in the introduction, there are complex and diverse correlations between concepts, and these correlations can seriously affect the final learning effect of the path. However, the embedding representation we obtained can only reflect the characteristics of the concept itself in isolation, and cannot reflect the correlation between concepts. Therefore, we need a function f^e to capture these correlations within the set and fuse them into the concept representation to get the global representation E_s:

$$E_s = f^e(X_s), X_s = [x_{s_1}, x_{s_2}, \cdots, x_{s_m}]^T. \qquad (2)$$

For the implementation of f^e, a simple approach is to add these concept representations to each concept after a pooling operation (e.g., average pooling operation), unfortunately, it is not capable to model complex correlations. Recent literature (Pang et al. 2020; Lee et al. 2019) uses a more complex Transformer to extract information, but it mainly focuses on correlation and thus pays less attention to the unique characteristics of the concept itself. And note that our training is based on the paradigm of a policy gradient with only one reward per path. With label sparsity, complex models like Transformer are extremely difficult to train due to potential over-smoothing issues (Liu et al. 2021). We empirically verify it in our follow-up experiments, which motivates us to combine the above two approaches.

First, we apply the self-attention mechanism to X_S:

$$E_s^a = softmax(\frac{QK^T}{\sqrt{d}})V, \qquad (3)$$

where

$$Q = X_s W^Q, K = X_s W^K, V = X_s W^V, \qquad (4)$$

W^Q, W^K, W^V are all trainable weights, d is the dimension of the embedding. At the same time, we pass a simple multi-layer perceptron(MLP) to the embedding and add the average pooling part:

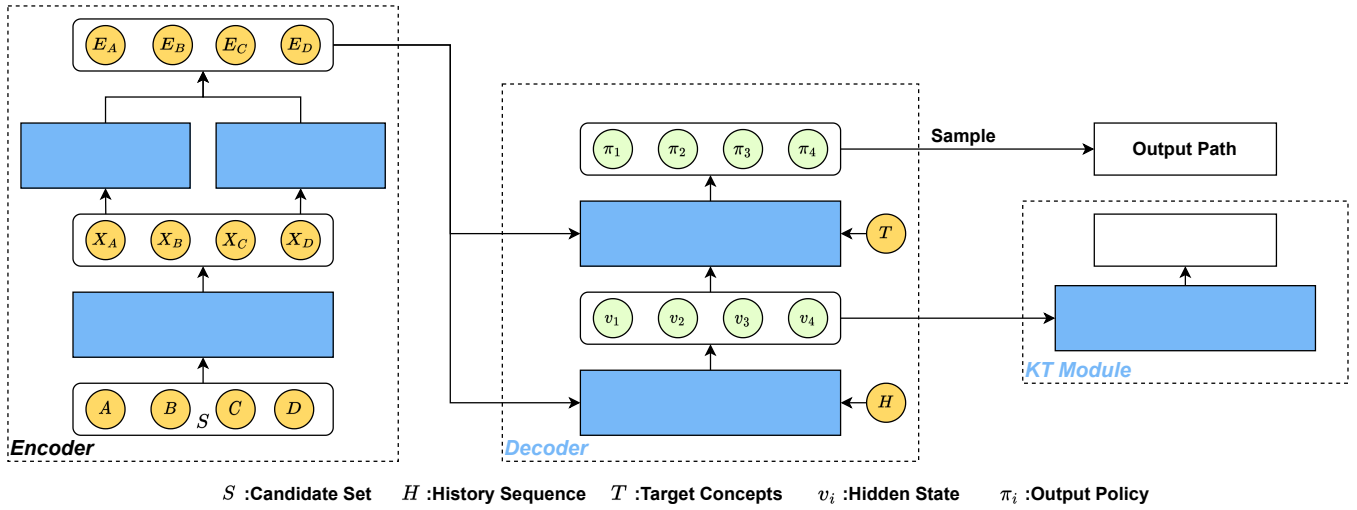$$E_s^l = E^l + \frac{\sum_i^m e_i^l}{m}, E^l = f^l(X_S), \qquad (5)$$

Figure 2: The overview of our framework. SRC is composed of the encoder, decoder, and KT auxiliary module. The encoder captures the correlation between concepts in the candidate set S to obtain the representation of concepts E_S. The decoder generates a ranking of S based on the information of E_S, T, and H, and outputs the policy π. KT auxiliary module is responsible for predicting the correct probability of each step on the path.

where f^l is the MLP, e_i^l is the feature of the i-th concept in E^l. Then the final representation of learning concepts in S is

$$E_s = [E_s^a; E_s^l], \qquad (6)$$

where $[.;.]$ is the concat operation. In this way, we obtain the representation E_s that are being aware of the other concepts in the set and retain their own characteristics.

Decoder

After obtaining a representation of each learned concept, we will generate their permutation and the probability in the decoder module:

$$\pi, P = f^d(E_s, H, T). \qquad (7)$$

The implementation of f^d refers to Pointer Network (Vinyals, Fortunato, and Jaitly 2015). First, we design an LSTM (denoted as g) (Hochreiter and Schmidhuber 1997) to trace student states. The initial state v_0 of the student in g before the start of the path should be related to the student's past learning sequence H. Considering that each step i in H contains both the learning concept c_i and the mastery degree y_i, v_0 is calculated as:

$$v_o = g([x_{c_1}; y_1]W_h, [x_{c_2}; y_2]W_h, \cdots, [x_{c_k}; y_k]W_h), \quad (8)$$

where x_{c_i} is the embedding of concept c_i, W_h is a trainable matrix that transforms $[x_{c_1}; y_1]$ into the same input dimension as E_s.

Now we assume that the state after the $(i-1)$-th concept π_{i-1} in the learning path is v_{i-1}. Let $\pi_{<i}$ denote the set of learned concepts, namely $\pi_{<i} = \{\pi_1, \pi_2, \cdots, \pi_{i-1}\}$. Then we calculate the probability distribution of step i as follows:

$$d_i^j = w_1^T(W_1 v_{i-1} + W_2 e_s^j + W_3 x_T + b), \qquad (9)$$

$$P(\hat{\pi}_i = s_j) = \begin{cases} e^{d_i^j} / \sum_{s_k \in S/\pi_{<i}} e^{d_i^k}, & \text{if } s_j \notin \pi_{<i} \\ 0, & \text{otherwise.} \end{cases}, \quad (10)$$

where $j = 1, 2 \cdots, m$; s_j represents the j-th concept in S and e_s^j represents its representation obtained by encoder; x_T represents the fusion of embeddings of concepts in T; w_1, W_1, W_2 and b are learnable weights or matrices. As shown in Eq. (9), we comprehensively consider student knowledge states, learning concepts, and target concepts to calculate the score of each learning concept under the current step. Then in Eq. (10), we further use softmax to calculate the probability among the remaining concepts, and the selected concepts are set to 0. Then according to the obtained probability distribution $P(\hat{\pi}_i)$, we can sample it to get the concept π_i of position i. And then we can update the state v_i accordingly:

$$v_i = g(e_s^{\pi_i}, v_{i-1}). \qquad (11)$$

According to the above method, we generate the final path $\pi = \{\pi_1, \cdots, \pi_n\}$ step by step, and the probability $P = \{p_{\pi_1}, \cdots, p_{\pi_n}\}$ corresponding to each step. This path will be recommended to students later.

Knowledge Tracing Auxiliary Module

In the decoder, we use LSTM to trace the student state to evaluate the current best-fit learning concept. However, unlike the general practice (Piech et al. 2015; Liu et al. 2019), we have no way to get instant access to the mastery of the student's previous step. Lack of utilization of this feedback may affect the performance of the decoder. To this end, we developed this module to predict mastery in a student's process, which acts as an auxiliary task to enhance the reliability and stability of other modules. Specifically, for the i-th

concept π_i on the path, we predict that the probability $p_{\pi_i}^k$ of successful mastery by students is:

$$p_{\pi_i}^y = Sigmoid(f^y(v_i)), \qquad (12)$$

where f^y denotes a MLP, $Sigmoid()$ denotes the $sigmoid$ function.

Optimization Objective

We build a student simulation to obtain E_e, E_b, and E_{sup}, which will be further specified in the experiment section. Then, we are able to compute E_T according to Eq. (1). We treat it as the reward in RL and formulate the loss of policy gradient for the path as:

$$L_\theta = -E_T \log \prod_i^n p_{\pi_i} = -E_T \sum_i^n \log p_{\pi_i}. \qquad (13)$$

Besides L_θ, we also introduce a cross-entropy between the predicted probability $p_{\pi_i}^y$ in the knowledge tracing auxiliary module and the actual feedback y_{π_i} of learning concept:

$$L_y = -\sum_i^n (y_{\pi_i} \log p_{\pi_i}^y + (1 - y_{\pi_i}) \log(1 - p_{\pi_i}^y)). \qquad (14)$$

By combining the above two losses, we can obtain the final loss of the full path:

$$L = L_\theta + \beta L_y, \qquad (15)$$

where β can be 0 or 1, which is used to control whether the KT task is used to assist the training. Algorithm 1 shows the training procedure of our method.

Algorithm 1: SRC

1: Randomly initialize the learning parameters.
2: **while** not converged **do**
3:     Randomly sample T, H, S.
4:     Calculate representation E_s of concepts in S (Eq. (2)).
5:     Generate path π and probability P (Eq. (7)).
6:     Predict feedback P_π^y on the learning path (Eq. (12)).
7:     Get feedback E_T, Y_π from student after learning π.
8:     Compute the gradient and update the parameters w.r.t the loss L (Eq. (15)).
9: **end while**

Experiment

In this section, we detail our experimental setup and results. We also do some discussions and extended investigations to illustrate the effectiveness of our model.

Dataset

Our experiments are performed on two real-world public datasets: ASSIST09[1] (Feng, Heffernan, and Koedinger 2009) and Junyi[2] (Chang, Hsu, and Chen 2015). Some statistics of the two datasets are shown in Table 1.

---

[1] https://sites.google.com/site/assistmentsdata/home/2009-2010-assistment-data

[2] https://www.kaggle.com/datasets/junyiacademy/learning-activity-public-dataset-by-junyi-academy

Dataset	#students	#concepts	#average length
ASSIST09	3841	167	49.54
Junyi	5002	712	54.19

Table 1: Dataset

However, there are some problems with using these two datasets directly to evaluate the model. Specifically, their data are all static, i.e. only answers to concepts that have been answered by students beforehand are known. Our model and other baseline models need to generate new paths and learn student feedback on them for training and evaluation. Therefore, the static original dataset cannot meet our requirements.

To this end, we refer to some of the practices of (Liu et al. 2019; Hu et al. 2018), and design a simulator that can dynamically assess students' knowledge level and return feedback. Specifically, this simulator is data-driven. We train the KT model on static data. The input of the model is the student's past learning sequence, and the output is the current concept answer probability. After the simulator training is completed, we can use it to simulate the learning situation of students on the paths recommended by various models and obtain the corresponding E_T to complete the effect evaluation. To enhance the reliability of the experiments, we use two KT models: DKT (Piech et al. 2015) and CoKT (Long et al. 2022) to build different simulators.

Meanwhile, to be able to compare the performance of various models more comprehensively, we formulate 4 different sources of candidate concept set S. Specifically, if the recommended path length is n, then the source of S will be: **1.** A fixed number of n concepts; **2.** Group all concepts, each with a size of n, and randomly select one group at a time; **3.** Randomly select n concepts each time; **4.** All concepts. Of course, the division in the first two sources is consistent for all models. In the experimental results reported later, $p = 0, 1, 2, 3$ represents these four sources in turn.

Baselines

To demonstrate the effectiveness and robustness of our framework, we compare it with the following methods:

- **Random**: Randomly select concepts from S and then randomly arrange the generated paths.

- **Rule-based**: Let the simulator return the learning effect of the target concepts after learning each concept separately in S. Then, according to this effect, the concepts are sorted from smallest to largest to generate paths.

- **MPC**: Using the Model Predictive Control (Deisenroth and Rasmussen 2011) in RL combined with KT, each step predicts the effect of several random search paths and makes the current action.

- **GRU4Rec**: A classic sequential recommendation model (Hidasi et al. 2015). After inputting the past sequence, predict the probability distribution of the next concept. Note that it is learned based on the original dataset.

- **DQN**: Classic reinforcement learning model (Mnih et al.

	Model	ASSIST09				Junyi			
		p=0	p=1	p=2	p=3	p=0	p=1	p=2	p=3
DKT	Random	0.0707	0.0995	0.1159	0.1290	-0.0854	-0.0054	-0.0091	-0.0075
	Rule-based	<u>0.1675</u>	<u>0.2070</u>	<u>0.1950</u>	<u>0.4233</u>	0.0525	<u>0.0969</u>	<u>0.1115</u>	<u>0.3481</u>
	MPC	0.0834	0.1163	0.1293	0.1399	-0.0584	0.0176	0.0121	0.0209
	GRU4Rec	0.0862	0.1505	0.1682	0.0755	-0.0390	0.0112	0.0152	-0.1394
	DQN	0.1215	0.1767	0.0723	0.2949	<u>0.0713</u>	0.0234	-0.0118	0.2023
	SRC	**0.3135***	**0.2971***	**0.2345***	**0.5567***	**0.2555***	**0.1761***	**0.1508***	**0.5809***
CoKT	Random	0.0858	0.0932	0.0917	0.0968	-0.1022	-0.0700	-0.0664	-0.0773
	Rule-based	0.0928	0.1010	0.0960	0.0990	-0.0988	-0.0580	<u>-0.0503</u>	-0.0522
	MPC	0.1145	0.1056	0.0918	0.1035	-0.0568	-0.0699	-0.0823	-0.0576
	GRU4Rec	0.1334	0.1242	<u>0.1240</u>	0.0589	-0.0333	-0.0675	-0.0659	-0.1385
	DQN	<u>0.1403</u>	<u>0.1281</u>	0.0710	<u>0.1253</u>	<u>-0.0145</u>	<u>-0.0524</u>	-0.0691	<u>0.0781</u>
	SRC	**0.1885***	**0.1559***	**0.1574***	**0.2340***	**0.0569***	**-0.0238***	**-0.0360***	**0.1709***

Table 2: Performance comparison of different models under four scenarios for two simulators built on each dataset. * indicates p<0.001 in significance tests compared to the best baseline.

2013). Here we pre-train a DKT model based on the original data to generate the required state in DQN.

Experiment Setting

The learning rate is decreased from the 1×10^{-3} to 1×10^{-5} during the training process. The batch size is set as 128. The weight for L2 regularization term is 4×10^{-5}. The dropout rate is set as 0.5. The dimension of embedding vectors is set as 64. All the models are trained under the same hardware settings with 16-Core AMD Ryzen 9 5950X (2.194GHZ), 62.78GB RAM, and NVIDIA GeForce RTX 3080 cards.

Experiment Result

Table 2 shows the performance comparison between SRC and other baseline models in their respective cases, where the evaluation index is the learning effect E_T, and the path length is 20. From these results, we have the following findings.

- Our model SRC outperforms all baselines in any case. This may be because our model adequately models interconcept correlations and effectively utilizes feedback.

- In DKT, the rule-based method generally achieves the best performance compared to other baselines. However, in CoKT, the performance of this method is poor, and most cases are close to the random method. Moreover, under CoKT, in many cases even SRC its E_T is negative and the maximum value of E_T learned by various models is also lower than the value under DKT. This may reflect the difference in the properties of the two simulators, such as the consideration of the relationship between concepts, the speed of forgetting knowledge, etc. Overall, learning under CoKT is more difficult.

- GRU4Rec performs well in some cases. This shows that the original learning sequences of students have some value and can reflect the relationship between concepts to some extent. Except that the performance is even worse

than random in the case of $p = 3$, which may be because this kind of scenario of choosing the optimal path from all concepts requires a wider range of collocations, and the limited raw data cannot extract this paradigm.

- DQN performs well in most cases (3rd in DKT, 2nd in CoKT). This reflects the superiority of interaction-based reinforcement learning methods in this setting. Its poor performance at $p = 2$ may be because the situation is more complex and the exploration of DQN is insufficient.

Ablation Study

Impact of Encoder and KT Auxiliary Module. Table 3 shows the performance comparison of various variants of SRC, where SRC_A and SRC_M represent the case where the encoder of SRC only uses self-attention or MLP, respectively. SRC^- means $\beta = 0$ during training, i.e. no KT auxiliary module is used. First of all, it can be seen that the performance of the model is degraded after replacing the original encoder with or without the KT module. This shows that the encoder in SRC combining self-attention and MLP indeed retains the advantages of both, which not only mines the correlation between concepts but also retains its own features.

Then, note that in SRC_A^-, after removing the KT module, a very large drop in model performance occurs compared to SRC_A, far exceeding that in the other two cases. In addition to this, SRC_A^- sometimes does not converge in experiments that other models have never experienced. This validates our previous concern that such complex networks would be significantly more difficult to train under this sparse reward reinforcement learning paradigm. Therefore, after using the KT module, the student feedback in Y_π not only brings more information, but also reduces the training difficulty of the encoder, and the performance is greatly improved. In SRC and SRC_M, there is no encoder training problem, so the performance improvement is not so obvious.

Effects of different path lengths. Figure 3 shows the per-

Model	p=0	p=1	p=2	p=3
SRC_M^-	0.2881	0.2878	0.2276	0.5391
SRC_M	0.3098	0.2954	0.2311	0.5405
SRC_A^-	0.2234	0.2018	0.1587	0.4539
SRC_A	0.2943	0.2891	0.2301	0.5378
SRC^-	0.3025	0.2910	0.2291	0.5290
SRC	**0.3135***	**0.2971***	**0.2345***	**0.5567***

Table 3: Performance comparison of different variants of SRC under DKT built on ASSIST09.

formance of paths of various lengths generated by different models under two complex scenarios of $p = 2, 3$. First, the performance rankings of various models are basically unchanged at various lengths, further illustrating the effectiveness of our model. Then, the learning effect E_T all grows with the length of the path, which is also in line with the intuition in education. Also note that in the $p = 3$ scenario, the performance growth of all models becomes very slow after length> 20. This is probably because the concepts that make up the path in this scenario are selectable by the model. While the number of concepts that are helpful for learning the target concept is limited, they are already selected when the path is short. Concepts added on longer paths have little value and are offset by factors such as forgetting.
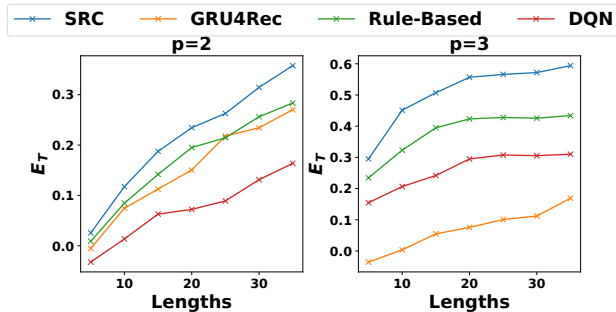

Figure 3: Impact of Length

Result for Industrial Dataset

To verify the effectiveness of SRC in practical applications, we deploy our model in the online education department of Huawei. Table 4 shows some experimental results on the company's internal industrial dataset. The dataset includes 159 students and 614 concepts and the average length of trajectories is 108.99.

It can be seen that SRC shows the best performance. The performance of other methods is also similar to those on public datasets. The main difference is that the negative rewards appear more frequently here. Under DKT, the random method even has negative rewards close to -1; correspondingly, under DKT, the SRC method can learn rewards close to the upper bound of 1. Under CoKT, although the reward of the learned optimal path is still negative in some cases, the

fluctuation range is greatly reduced. These results reflect the different properties of difficulty curves and forgetting curves under different simulators built on different datasets. And our model can show the best performance in a variety of situations, indicating the effectiveness and generalization of our method.

Further online experimental results are being deployed and collected in the coming weeks.

	Model	p=0	p=1	p=2	p=3
DKT	Rule-based	0.0319	0.2092	0.1622	0.9507
	Random	-0.8202	-0.7088	-0.8098	-0.8885
	DQN	0.4495	-0.2504	-0.6021	0.8800
	SRC	**0.9319***	**0.4701***	**0.2842***	**0.9861***
CoKT	Rule-based	-0.0445	-0.0631	-0.0819	-0.0595
	Random	-0.0637	-0.0848	-0.0832	-0.0548
	DQN	0.0101	-0.0092	-0.0215	-0.0504
	SRC	**0.1288***	**-0.0042***	**-0.0159***	**0.2577***

Table 4: Result on the industrial dataset.

Conclusion

In this paper, we formulate the path recommendation of the online education system as a set-to-sequence task and provide a new set-to-sequence ranking-based concept-aware framework, named SRC. Specifically, we first design a concept-aware encoder module that captures correlations between input learning concepts. The output is then fed to a decoder module that sequentially generates paths through an attention mechanism that handles correlations between learning and target concepts. The recommendation policy will be optimized through policy gradients. In addition, we introduce an auxiliary module based on knowledge tracking to enhance the stability of the model by evaluating students' learning effects on the learned concepts. We conduct extensive experiments on two real-world public datasets and one industrial proprietary dataset, where SRC demonstrates its performance superiority over other baselines.

In future work, it might be an interesting direction to further explore the relationships between concepts, such as using graph neural networks. In addition, we plan to further deploy our model in the real-world online education system.

Acknowledgements

References

Bian, C.-L.; Wang, D.-L.; Liu, S.-Y.; Lu, W.-G.; and Dong, J.-Y. 2019. Adaptive learning path recommendation based on graph theory and an improved immune algorithm. *KSII Transactions on Internet and Information Systems (TIIS)*, 13(5): 2277–2298.

Bruch, S.; Han, S.; Bendersky, M.; and Najork, M. 2020. A stochastic treatment of learning to rank scoring functions. In *Proceedings of the 13th international conference on web search and data mining*, 61–69.

Burges, C.; Shaked, T.; Renshaw, E.; Lazier, A.; Deeds, M.; Hamilton, N.; and Hullender, G. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, 89–96.

Burges, C. J. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581): 81.

Cai, D.; Zhang, Y.; and Dai, B. 2019. Learning path recommendation based on knowledge tracing model and reinforcement learning. In *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, 1881–1885. IEEE.

Cao, Z.; Qin, T.; Liu, T.-Y.; Tsai, M.-F.; and Li, H. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, 129–136.

Carbonell, J. R. 1970. AI in CAI: An artificial-intelligence approach to computer-assisted instruction. *IEEE transactions on man-machine systems*, 11(4): 190–202.

Chang, H.-S.; Hsu, H.-J.; and Chen, K.-T. 2015. Modeling Exercise Relationships in E-Learning: A Unified Approach. In *EDM*, 532–535.

Chen, Y.-H.; Huang, N.-F.; Tzeng, J.-W.; Lee, C.-a.; Huang, Y.-X.; and Huang, H.-H. 2022. A Personalized Learning Path Recommender System with Line Bot in MOOCs Based on LSTM. In *2022 11th International Conference on Educational and Information Technology (ICEIT)*, 40–45. IEEE.

Cover, T.; and Hart, P. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1): 21–27.

Deisenroth, M.; and Rasmussen, C. E. 2011. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 465–472. Citeseer.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dwivedi, P.; Kant, V.; and Bharadwaj, K. K. 2018. Learning path recommendation based on modified variable length genetic algorithm. *Education and information technologies*, 23(2): 819–836.

Feng, M.; Heffernan, N.; and Koedinger, K. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User modeling and user-adapted interaction*, 19(3): 243–266.

Friedman, J. H. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Grover, A.; Wang, E.; Zweig, A.; and Ermon, S. 2019. Stochastic optimization of sorting networks via continuous relaxations. *arXiv preprint arXiv:1903.08850*.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Hu, Y.; Da, Q.; Zeng, A.; Yu, Y.; and Xu, Y. 2018. Reinforcement learning to rank in e-commerce search engine: Formalization, analysis, and application. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 368–377.

Huang, Z.; Liu, Q.; Zhai, C.; Yin, Y.; Chen, E.; Gao, W.; and Hu, G. 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.

Inhelder, B.; Chipman, H. H.; Zwingmann, C.; et al. 1976. *Piaget and his school: a reader in developmental psychology*. Springer.

Joachims, T. 2006. Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 217–226.

Joseph, L.; Abraham, S.; and Mani, B. P. 2022. Exploring the Effectiveness of Learning Path Recommendation based on Felder-Silverman Learning Style Model: A Learning Analytics Intervention Approach. *Journal of Educational Computing Research*, 07356331211057816.

Lee, J.; Lee, Y.; Kim, J.; Kosiorek, A.; Choi, S.; and Teh, Y. W. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, 3744–3753. PMLR.

Li, X.; Xu, H.; Zhang, J.; and Chang, H.-h. 2021. Optimal hierarchical learning path design with reinforcement learning. *Applied psychological measurement*, 45(1): 54–70.

Liu, D.; Wang, S.; Ren, J.; Wang, K.; Yin, S.; and Zhang, Q. 2021. Trap of Feature Diversity in the Learning of MLPs. *arXiv preprint arXiv:2112.00980*.

Liu, H.; and Li, X. 2020. Learning path combination recommendation based on the learning networks. *Soft Computing*, 24(6): 4427–4439.

Liu, Q.; Tong, S.; Liu, C.; Zhao, H.; Chen, E.; Ma, H.; and Wang, S. 2019. Exploiting cognitive structure for adaptive learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 627–635.

Long, T.; Qin, J.; Shen, J.; Zhang, W.; Xia, W.; Tang, R.; He, X.; and Yu, Y. 2022. Improving Knowledge Tracing with Collaborative Information. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 599–607.

Luce, R. D. 2012. *Individual choice behavior: A theoretical analysis*. Courier Corporation.

MindSpore. 2022. MindSpore. https://www.mindspore.cn/.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Nabizadeh, A. H.; Jorge, A. M.; and Leal, J. P. 2019. Estimating time and score uncertainty in generating successful learning paths under time constraints. *Expert Systems*, 36(2): e12351.

Oosterhuis, H. 2021. Computationally efficient optimization of plackett-luce ranking models for relevance and fairness. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1023–1032.

Pang, L.; Xu, J.; Ai, Q.; Lan, Y.; Cheng, X.; and Wen, J. 2020. Setrank: Learning a permutation-invariant ranking model for information retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 499–508.

Piaget, J.; and Duckworth, E. 1970. Genetic epistemology. *American Behavioral Scientist*, 13(3): 459–480.

Piech, C.; Bassen, J.; Huang, J.; Ganguli, S.; Sahami, M.; Guibas, L. J.; and Sohl-Dickstein, J. 2015. Deep knowledge tracing. *Advances in neural information processing systems*, 28.

Pinar, W. F.; Reynolds, W. M.; Slattery, P.; Taubman, P. M.; et al. 1995. *Understanding curriculum: An introduction to the study of historical and contemporary curriculum discourses*, volume 17. Peter lang.

Plackett, R. L. 1975. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2): 193–202.

Shao, E.; Guo, S.; and Pardos, Z. A. 2021. Degree planning with plan-bert: Multi-semester recommendation using future courses of interest. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 14920–14929.

Shi, D.; Wang, T.; Xing, H.; and Xu, H. 2020. A learning path recommendation model based on a multidimensional knowledge graph framework for e-learning. *Knowledge-Based Systems*, 195: 105618.

Sun, Y.; Zhuang, F.; Zhu, H.; He, Q.; and Xiong, H. 2021. Cost-effective and interpretable job skill recommendation with deep reinforcement learning. In *Proceedings of the Web Conference 2021*, 3827–3838.

Swezey, R.; Grover, A.; Charron, B.; and Ermon, S. 2020. Pirank: Learning to rank via differentiable sorting. *arXiv preprint arXiv:2012.06731*.

Tong, S.; Liu, Q.; Huang, W.; Hunag, Z.; Chen, E.; Liu, C.; Ma, H.; and Wang, S. 2020. Structure-based knowledge tracing: an influence propagation view. In *2020 IEEE International Conference on Data Mining (ICDM)*, 541–550. IEEE.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. *Advances in neural information processing systems*, 28.

Wang, X.; Liu, K.; Wang, D.; Wu, L.; Fu, Y.; and Xie, X. 2022. Multi-level recommendation reasoning over knowledge graphs with reinforcement learning. In *Proceedings of the ACM Web Conference 2022*, 2098–2108.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3): 229–256.

Xia, F.; Liu, T.-Y.; Wang, J.; Zhang, W.; and Li, H. 2008. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*, 1192–1199.

Yang, Y.; Shen, J.; Qu, Y.; Liu, Y.; Wang, K.; Zhu, Y.; Zhang, W.; and Yu, Y. 2020. GIKT: a graph-based interaction model for knowledge tracing. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 299–315. Springer.

Zhang, J.; Shi, X.; King, I.; and Yeung, D.-Y. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*, 765–774.

Zhou, Y.; Huang, C.; Hu, Q.; Zhu, J.; and Tang, Y. 2018. Personalized learning full-path recommendation model based on LSTM neural networks. *Information Sciences*, 444: 135–152.

Zhu, H.; Tian, F.; Wu, K.; Shah, N.; Chen, Y.; Ni, Y.; Zhang, X.; Chao, K.-M.; and Zheng, Q. 2018. A multi-constraint learning path recommendation algorithm based on knowledge map. *Knowledge-Based Systems*, 143: 102–114.