Tracing Knowledge State with Individual Cognition and Acquisition Estimation

Ting Long, Yunfei Liu, Jian Shen, Weinan Zhang, Yong Yu*
Shanghai Jiao Tong University
{longting,liuyunfei,r_ocky,wnzhang}@sjtu.edu.cn,yyu@apex.sjtu.edu.cn

ABSTRACT

Knowledge tracing, which dynamically estimates students' learning states by predicting their performance on answering questions, is an essential task in online education. One typical solution for knowledge tracing is based on Recurrent Neural Networks (RNNs), which represent students' knowledge states with the hidden states of RNNs. Such type of methods normally assumes that students have the same cognition level and knowledge acquisition sensitivity on the same question. Thus, they (i) predict students' responses by referring to their knowledge states and question representations, and (ii) update the knowledge states according to the question representations and students' responses. No explicit cognition level or knowledge acquisition sensitivity is considered in the above two processes. However, in real-world scenarios, students have different understandings on a question and have various knowledge acquisition after they finish the same question. In this paper, we propose a novel model called Individual Estimation Knowledge Tracing (IEKT), which estimates the students' cognition on the question before response prediction and assesses their knowledge acquisition sensitivity on the questions before updating the knowledge state. In the experiments, we compare IEKT with 11 knowledge tracing baselines on four benchmark datasets, and the results show IEKT achieves the state-of-the-art performance.

CCS CONCEPTS

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

KEYWORDS

knowledge tracing; cognition; knowledge acquisition sensitivity; reinforcement learning

ACM Reference Format:
Ting Long, Yunfei Liu, Jian Shen, Weinan Zhang, Yong Yu. 2021. Tracing Knowledge State with Individual Cognition and Acquisition Estimation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3404835.3462827

*Corresponding author.

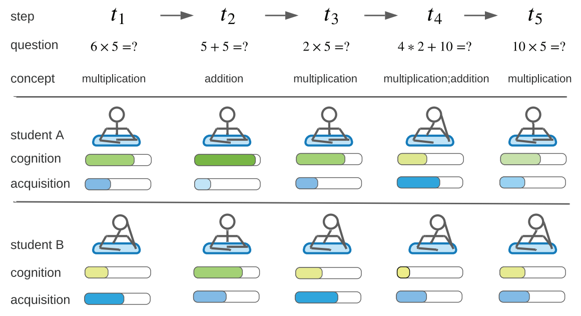


Figure 1: An example of the question-answering process. Although students A and B answer the same questions, their cognition and acquisition levels are different. As for cognition, most of the questions are easy for student A, while these questions are difficult for student B. As for acquisition, since student A has mastered the *addition* and *multiplication*, she (he) acquires less improvement after finishing these questions. In contrast, as student B has only mastered the *addition*, answering these questions can help her master the *multiplication* and make more progress accordingly.

1 INTRODUCTION

Many online learning sites, like Yuantiku[1], offer students a platform to enhance the knowledge they have acquired. Students can improve their skills by answering questions on such platforms. However, since the questions are prepared in advance, students may answer the questions whose related concepts they have already mastered, which has little help for improving their overall mastery of the subject. Instead, students should focus on the questions they have difficulty answering correctly, which means the students have not mastered the concepts behind the questions. Therefore, dynamically estimating students' learning states, identifying the questions that are difficult for students to answer, and pushing them into students' to-answer lists is a crucial issue for online learning sites. This consideration motivates the study of *knowledge tracing*. Knowledge tracing can automatically estimate students' learning states, which uses the question-answering history of students to predict if they can correctly answer a new question. Due to this attribute, knowledge tracing task is also beneficial to the related tasks in online education, such as arranging the individual learning path [12, 20] and recommending the learning materials [10, 35].

In the literature, lots of efforts have been devoted to designing deep models for knowledge tracing due to the power of deep learning [1, 5, 22, 26, 27, 29, 32, 36, 39]. One of representative among

[1]https://www.yuantiku.com/

them is based on auto-regressive architectures, such as recurrent neural networks (RNNs) [6, 16], where student's knowledge states (the mastery level of the concepts) are represented by the hidden states of recurrent units. These frameworks can usually be formulated into a two-stage process, namely the *read* and *write* stages, as shown in Figure 2(a). To be more specific, in the *read* stage, the hidden states of recurrent units and the question representations are loaded and then fed into a multi-layer perceptron to predict the probability of the students correctly answering the questions. On the other hand, the *write* stage models the dynamic changes of students' knowledge states, by updating the corresponding recurrent cell after receiving the feedback of one question.

Despite the success of the previous methods, some important limitations still exist. First, these methods assume that students share the same cognition level for a specific question, and thus directly use the question representations and the knowledge states to make predictions without explicitly representing students' cognition levels on specific questions. However, students' knowledge backgrounds are distributed [2, 11, 28]. As a result, they have different cognition levels for the same question, as Figure 1 shows. Second, the previous methods assume that different students have the same knowledge increment after they answer the same question and give the same response. However, the distribution of students' intelligence is diverse according to the theory of multiple intelligence [14], and therefore students may have different sensitivity in knowledge acquisition though they answer the same question and give the same response, which as Figure 1 illustrates. Thus, the lack of explicit modeling of cognition level and acquisition sensitivity limits the performance of the previous works.

Based on these considerations, in this paper, we propose a novel knowledge tracing model called Individual Estimation Knowledge Tracing (IEKT). To be specific, IEKT introduces a cognition estimation (CE) module in the *read* stage and a knowledge acquisition sensitivity estimation (KASE) module in the *write* stage to tackle the above challenges. The CE module estimates students' cognition levels on questions according to their knowledge states and question representations. The KASE module estimates students' knowledge acquisition sensitivity according to students' knowledge states, responses and question representations. To search the optimal parameters of IEKT, we further apply a reinforcement learning method to train IEKT. We evaluate IEKT on four benchmark datasets by comparing it with 11 previous methods. The experiment results demonstrate that IEKT achieves the new state-of-the-art performance.

The main contributions of our paper are summarized as:

- We introduce the cognition level estimation module and knowledge acquisition sensitivity estimation module to knowledge tracing. To our knowledge, IEKT is the first one that explicitly estimates individual cognition and knowledge acquisition sensitivity on questions.
- We validate the performance of IEKT on four public datasets, where IEKT outperforms all the compared baselines.
- We find that CE and KASE modules are also applicable to other knowledge tracing methods. The experiments reveal that introducing CE and KASE modules to the other models also enhances their performance.

2 RELATED WORKS

The knowledge tracing methods can be grouped into traditional methods and deep learning-based methods. The traditional methods have comparable performance with the deep learning-based methods in some cases, but the deep learning-based methods are more powerful in general [38].

Most traditional methods are factor-based. They predict students' responses according to the factors related to learning. One of the classical methods is Bayesian Knowledge Tracing (BKT) [8, 40]. It uses binary variables to represent students' knowledge states: know the knowledge concepts (1) and not know the knowledge concepts (0). BKT uses Hidden Markov Model to model the knowledge states. It considers four factors affecting students' responses: initial knowledge states, learning rate, slip probability, and guess probability. Another typical type is Factor Analysis methods [34, 37]. The simplest model is the Item Response Theory (IRT) [9]. It measures students' ability and the difficulty of questions to evaluate students' capability and the probability of they correctly answer the questions. Recent works elaborate the factors related to learning. For instance, Vie and Kashima [34] introduced the factors like school ID, teacher ID, and they find that the performance becomes better as the number of factors increases.

Most deep learning methods are state-based, which maintain vectors to represent students' knowledge states. One of the representative methods is Deep Knowledge Tracing (DKT), which is proposed by Piech et al. [27]. DKT represents students' knowledge states with the hidden states of LSTM [16], and predicts students' responses by feeding the knowledge states to a binary classifier. As students' cognition improves when they interact with the questions, DKT updates their knowledge states by using question representation and students' responses. Many works obtain a better performance by extending DKT: Nagatani et al. [24] considered the forgetting behavior; Chen et al. [5] labeled the prerequisite relations among concepts; Su et al. [30] encoded question embeddings with text description and Liu et al. [21] pre-trained the embeddings of questions. Unlike the DKT-based methods which represent the students' knowledge states by a single vector, there are some methods using multiple vectors to represent the knowledge states on different concepts. One of the approaches is Dynamic Key-Value Memory Networks (DKVMN) [41]. DKVMN stores the concept representation in the key matrix and the knowledge states in the value matrix. Also, many works follow DKVMN. For instance, Abdelrahman and Wang [1] improved DKVMN by introducing a hop-LSTM layer, and Huang et al. [17] used the attention mechanism to aggregate history states. Further, Nakagawa et al. [25] used the graph to represent the concept relation, and updated the knowledge state based on the graph. In addition to the state-based methods, Shen et al. [29] applied convolutional neural networks (CNN) [19] on students' question-answering history, and Pandey and Karypis [26] utilized attention mechanism [33] to replace the RNNs to assess students' knowledge states.

Although the previous methods have achieved sound results, they assume that an arbitrary question means the same for all the students. However, the students with different knowledge backgrounds have different cognition and knowledge acquisition sensitivity on the questions. Thus, in this paper, we propose a method called Individual Estimation Knowledge Tracing (IEKT) to solve this problem. It evaluates students' cognition before response prediction and assesses their knowledge acquisition sensitivity before updating the knowledge state.

3 PROBLEM DEFINITION

In this section, we briefly introduce the knowledge tracing task with the notations used throughout the paper. We summarize them in Table 1.

Suppose the question-answering history of a student is $\mathcal{X}_{t-1} = \{(q_1, r_1), (q_2, r_2), \ldots, (q_{t-1}, r_{t-1})\}$. Here, q_t denotes the question the student answers at step t. And the binary variable r_t denotes the student's response to the question q_t:

$$r_t = \begin{cases} 1, & \text{if the student's answer is right;} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

We then denote the set of all questions as $Q = \{q_i\}_{i=1}^{|Q|}$, and the embedding of q_i is represented by $\mathbf{e}_i^q \in \mathbf{R}^{d_q}$. In knowledge tracing, each question is related to multiple concepts as it is illustrated in Figure 1. We further denote the set of concepts as $C = \{c_j\}_{j=1}^{|C|}$, and the corresponding embedding of concept c_j is represented by $\mathbf{e}_j^c \in \mathbf{R}^{d_c}$.

The task of knowledge tracing is formulated as predicting the probability that the student will correctly answer a new question q_t at step t, *i.e.*, $P(r_t = 1|q_t, \mathcal{X}_{t-1})$. We approach that by learning a function f_Θ parameterized by Θ to estimate the probability:

$$\hat{r}_t = f_\Theta(\cdot) \tag{2}$$

Here, $\hat{r}_t = P(r_t = 1|q_t, \mathcal{X}_{t-1})$, and $(\cdot)$ denotes the features we use to predict the student's response.

To obtain the probability that the student correctly answer q_t, we define the following terminologies:

Definition 3.1. (**Knowledge State**). The student's knowledge state is her (his) mastery level of all the concepts. We denote a student's knowledge state at step t as $\mathbf{h}_t \in \mathbf{R}^{d_h}$.

Definition 3.2. (**Cognition Level, Cognition Vector and Cognition Matrix**). Given a question q_t, the student's cognition level on q_t represents the level she (he) comprehends the question q_t. We use a cognition vector $\mathbf{m}_t \in \mathbf{R}^{d_m}$ to represent the cognition level. We represent the different cognition levels by a cognition matrix $\mathbf{M} \in \mathbf{R}^{k \times d_m}$, where k denotes the number of cognition levels. In particular, $\mathbf{m}_t$ is a row vector of $\mathbf{M}$.

Definition 3.3. (**Knowledge Acquisition Sensitivity, Acquisition Vector and Acquisition Matrix**). Given a question q_t, the student's knowledge acquisition sensitivity represents the level of knowledge increment after the student answers the question q_t and receives the feedback. We use an acquisition vector $\mathbf{s}_t \in \mathbf{R}^{d_s}$ to represent the knowledge acquisition sensitivity. We represent the different knowledge acquisition sensitivity levels by acquisition matrix $\mathbf{S} \in \mathbf{R}^{b \times d_s}$, where b denotes the number of sensitivity levels. In particular, $\mathbf{s}_t$ is a row vector of $\mathbf{S}$.

Table 1: Notations and descriptions.

Notations	Descriptions
$\mathcal{X}_{t-1}$	Student's question-answering history.
q_i, c_j	The question and the concept.
Q, C	The set of questions and the set of concepts.
$\mathbf{e}_i^q, \mathbf{e}_j^c$	The embedding of question and concept.
$\hat{r}_t, r_t$	The predicted probability and the true label.
$\mathbf{M}, \mathbf{m}_t$	The cognition matrix and the cognition vector.
$\mathbf{S}, \mathbf{s}_t$	The acquisition matrix and the accquistion vector.
$\mathbf{h}_t$	The knowledge state.

4 METHOD

Previous state-based methods for the knowledge tracing task can usually be formulated into *read* and *write* stages. However, these methods do not take the cognition level and knowledge acquisition sensitivity into account, which leads to inadequate performance. To bridge this gap, we propose a model called Individual Estimation Knowledge Tracing (IEKT), which introduces a Cognition Estimation (CE) module into the *read* stage and the Knowledge Acquisition Sensitivity Estimation (KASE) module into the *write* stage as Figure 2(b) shows. Specifically, the CE module is designed to estimate students' cognition levels on questions, which enhances response prediction. And the KASE module estimates students' knowledge acquisition sensitivity on the questions, which assists the update of knowledge states. The details of IEKT are introduced as follows.

4.1 The read stage

Like the previous methods, IEKT also relies on the knowledge states and question representations to predict students' responses. However, different from the previous methods, IEKT introduces a cognition estimation (CE) module, which estimates students' cognition on the questions before making final predictions.

Cognition estimation (CE). As we assume that students have different cognition on the questions, we use students' knowledge states and the question representations to estimate their cognition levels on the questions. Like the previous methods, we represent students' knowledge states with the hidden states of RNN. We represent the question q_t with the following form:

$$\mathbf{v}_t = \mathbf{e}_t^q \oplus \overline{\mathbf{e}}_t^c, \tag{3}$$

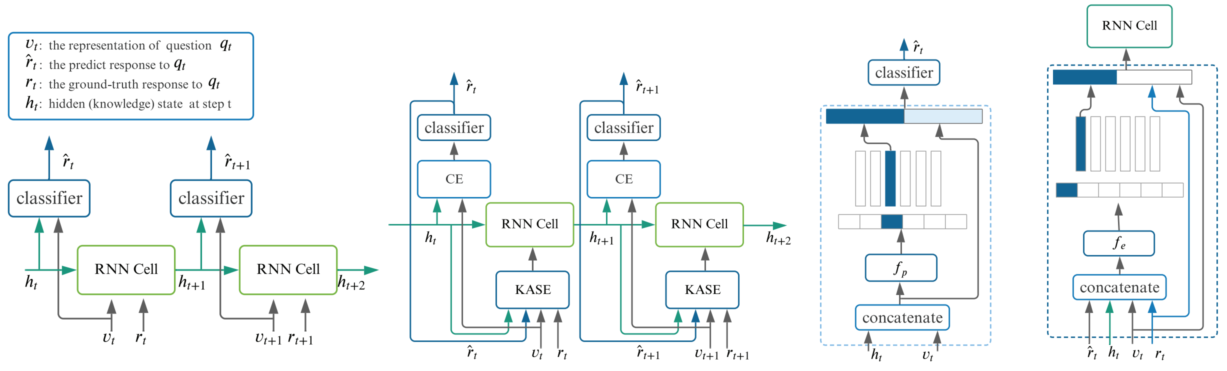
where we have $\mathbf{v}_t \in \mathbf{R}^{(d_q+d_c)}$ and $\overline{\mathbf{e}}_t^c \in \mathbf{R}^{d_c}$ denotes the average embedding of the concepts which are related to question q_t. $\oplus$ denotes the concatenation.

Then, we concatenate the question representation $\mathbf{v}_t$ with the knowledge state $\mathbf{h}_t$ of the student,

$$\mathbf{h}_v = \mathbf{h}_t \oplus \mathbf{v}_t, \tag{4}$$

where we have $\mathbf{h}_v \in \mathbf{R}^{d_v}$ and $d_v = d_h + d_q + d_c$. Then we feed the concatenation to a mapping function f_p to obtain the index probability distribution of the cognition vectors. We sample an index from the distribution:

$$i_{m,t} \sim f_p(\mathbf{h}_v), \tag{5}$$



v_t: the representation of question q_t
$\hat{r}_t$: the predict response to q_t
r_t: the ground-truth response to q_t
h_t: hidden (knowledge) state at step t
classifier
RNN Cell
CE
KASE
concatenate
f_p
f_e
h_t
h_{t+1}
h_{t+2}
v_t
r_t
$\hat{r}_t$
$\hat{r}_{t+1}$
v_{t+1}
r_{t+1}

(a) The previous RNN-based works. (b) The framework of IEKT. (c) The CE modules. (d) The KASE module.

Figure 2: The frameworks and details of IEKT

and then pick the cognition vector which represents the student's cognition levels from the cognition matrix $\mathbf{M}$:

$$\mathbf{m}_t = \mathbf{M}[i_{m,t}], \tag{6}$$

Response prediction. We introduce the cognition vector to the response prediction by concatenating it with the knowledge states and question representation. Then, we use the concatenation to obtain the probability that the student correctly answer question q_t:

$$\begin{aligned} \mathbf{y}_t &= ReLU(\mathbf{W}_1 \cdot [\mathbf{m}_t \oplus \mathbf{h}_v] + \mathbf{b}_1), \\ \hat{r}_t &= \delta(\mathbf{W}_2 \cdot \mathbf{y}_t + \mathbf{b}_2), \end{aligned} \tag{7}$$

where $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{b}_1$, $\mathbf{b}_2$ are trainable parameters. $\mathbf{W}_1 \in \mathbf{R}^{d_r \times d_r}$, $d_r = d_v + d_m$, $\mathbf{W}_2 \in \mathbf{R}^{d_r \times 1}$, $\mathbf{b}_1 \in \mathbf{R}^{d_r}$, $\mathbf{b}_2 \in \mathbf{R}$. δ denotes the *sigmoid* function:

$$\delta(\mathbf{x}) = \frac{1}{1 + e^{-\mathbf{x}}} \tag{8}$$

4.2 The write stage

Students acquire knowledge as they interact with the questions, and their knowledge states change accordingly. We assume that students have different knowledge acquisition sensitivity on the same question. Thus, we use the KASE module to explicitly estimate students' knowledge acquisition sensitivity.

Knowledge acquisition sensitivity estimation (KASE). Our estimation considers the knowledge state, the question representation, the response, and the predicted response.

First, we concatenate $\mathbf{h}_v$ with the predicted response:

$$\mathbf{v}_p = \begin{cases} \mathbf{h}_v \oplus \mathbf{0}, & \hat{r}_t \geq 0.5, \\ \mathbf{0} \oplus \mathbf{h}_v, & \hat{r}_t < 0.5, \end{cases} \tag{9}$$

and we concatenate $\mathbf{h}_v$ with the ground-truth response as

$$\mathbf{v}_g = \begin{cases} \mathbf{h}_v \oplus \mathbf{0}, r_t = 1, \\ \mathbf{0} \oplus \mathbf{h}_v, r_t = 0, \end{cases} \tag{10}$$

where $\mathbf{0} \in \mathbf{R}^{d_v}$, $\mathbf{v}_p \in \mathbf{R}^{2d_v}$ and $\mathbf{v}_g \in \mathbf{R}^{2d_v}$.

Then, we concatenate $\mathbf{v}_p$ and $\mathbf{v}_g$ to obtain the vector that considers both the predicted response and the ground-truth response under the knowledge state and question representation,

$$\mathbf{v}_m = \mathbf{v}_p \oplus \mathbf{v}_g. \tag{11}$$

Here, we utilize both the predicted response and the ground-truth response. The motivation is that students may guess or slip when they interact with questions. We consider both of them to decrease the noise caused by these factors.

We use $\mathbf{v}_m$ and a mapping function f_e to select the vector from $\mathbf{S}$, which represents the level of the student's knowledge acquisition sensitivity,

$$\begin{aligned} \mathbf{i}_{s,t} &\sim f_e(\mathbf{v}_m), \\ \mathbf{s}_t &= \mathbf{S}[i_{s,t}]. \end{aligned} \tag{12}$$

Knowledge state update. We concatenate the acquisition vector $\mathbf{s}_t$ with the question representation according to the response r_t,

$$\mathbf{v}_i = \begin{cases} \mathbf{v}_t \oplus \mathbf{s}_t, r_t = 1, \\ \mathbf{s}_t \oplus \mathbf{v}_t, r_t = 0, \end{cases} \tag{13}$$

where $\mathbf{v}_i \in \mathbf{R}^{d_i}$, $d_i = d_q + d_c + d_s$.

Then, we feed $\mathbf{v}_i$ and the current knowledge state $\mathbf{h}_t$ to the RNN cell to update the student's knowledge state. In our case, we adopt the GRU [6],

$$\begin{aligned} \mathbf{u}_r &= \delta(\mathbf{W}_r \cdot (\mathbf{v}_i \oplus \mathbf{h}_t) + \mathbf{b}_r), \\ \mathbf{u}_z &= \delta(\mathbf{W}_z \cdot (\mathbf{v}_i \oplus \mathbf{h}_t) + \mathbf{b}_z), \\ \mathbf{u}_h &= tanh(\mathbf{W}_h \cdot (\mathbf{v}_i \oplus (\mathbf{u}_r * \mathbf{h}_t)) + \mathbf{b}_h), \\ \mathbf{h}_{t+1} &= (\mathbf{1} - \mathbf{u}_z) * \mathbf{u}_h + u_z * \mathbf{h}_t. \end{aligned} \tag{14}$$

Here, $\mathbf{W}_r$, $\mathbf{b}_r$, $\mathbf{W}_z$, $\mathbf{b}_z$, $\mathbf{W}_h$, $\mathbf{b}_h$ are trainable parameters, where $\mathbf{W}_r, \mathbf{W}_z, \mathbf{W}_h \in \mathbf{R}^{d_h \times (d_i + d_h)}$ and $\mathbf{b}_r, \mathbf{b}_z, \mathbf{b}_h \in \mathbf{R}^{d_h}$.

4.3 Model Learning

As we directly sample the index of cognition vector and acquisition vector in Eq. 5 and Eq. 12, some gradients in the network are cut off and cannot back-propagate. We apply the Policy Gradient [31], which is a classical reinforcement learning algorithm, to optimize the selection of cognition vectors and acquisition vectors.

For the cognition vector selection (Eq. 5), we use the $\mathbf{h}_v$ (Eq. 4) as the state in reinforcement learning. The action set is $A_m =$

$\{0,\ 1,\ 2, ..., k\}$. Each element in A_m represents the index of the cognition vector. The stochastic policy is denoted by a mapping function f_p (Eq. 5). The reward $\hat{u}_t$ depends on the prediction:

$$\hat{u}_t = \begin{cases} \frac{1}{T}\ , \text{if the prediction is correct,} \\ 0\ , \text{otherwise} \end{cases} \tag{15}$$

Here, T denotes the length of the question-answering sequence. The loss for cognition vector selection is:

$$\mathcal{L}_m = -\sum_t^T \log f_p(i_{m,t}|\mathbf{h}_v)u_t, \tag{16}$$

$f_p(i_{m,t}|\mathbf{h}_v)$ represents the probability of the policy f_p taking the action $i_{m,t}$ at the state $\mathbf{h}_v$ (Eq. 5). u_t denotes the cumulative reward at step t, which is computed from $\hat{u}_t$:

$$u_t = \hat{u}_t + \gamma u_{t+1} \tag{17}$$

where γ is the discount factor.

For the acquisition vector selection (Eq. 12), we use the $\mathbf{v}_m$ (Eq. 11) as the state. The action set is $A_s = \{0,\ 1,\ 2, ..., b\}$, in which each element represents the index of the acquisition vector. The policy is the mapping function f_e (Eq. 12). The reward is similar to the cognition vector selection. The loss for acquisition vector selection is:

$$\mathcal{L}_s = -\sum_t^T \log f_e(i_{s,t}|\mathbf{v}_m)u_t, \tag{18}$$

where $f_e(i_{s,t}|\mathbf{v}_m)$ represent the probability of the policy f_e taking the action $i_{s,t}$ at state $\mathbf{v}_m$ (Eq. 12).

Thus, for the cognition vector selection and acquisition vector selection, the loss is:

$$\mathcal{L}_{ms} = \mathcal{L}_m + \mathcal{L}_s \tag{19}$$

Our objective for selecting vectors is to minimize $\mathcal{L}_{ms}$.

For the knowledge tracing task, the objective is response prediction. It predicts the probability that the student can correctly answer questions. The objective function for this goal is to minimize the negative log-likelihood of predicted probability $\hat{r}_t$ and the true label r_t. That is:

$$\mathcal{L}_c = -\sum_{i=1}^T (r_i\ \log \hat{r}_i + (1 - r_i)\ \log(1 - \hat{r}_i)). \tag{20}$$

The learning parameters of our method are the embedding of concepts and questions, the cognition matrix $\mathbf{M}$, and the knowledge acquisition sensitivity matrix $\mathbf{S}$, the weights in GRU, the weights in mapping function f_p, f_e and the weights in response prediction $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{b}_1$, $\mathbf{b}_2$. The parameters are jointly learned by minimizing the loss for both knowledge tracing and vectors selection:

$$\mathcal{L} = \mathcal{L}_c + \lambda\mathcal{L}_{ms}. \tag{21}$$

The overall training procedure is described as in Algorithm 1.

5 EXPERIMENT

In this section, we present our experiment settings and the corresponding results in detail. We also make some discussions with an extended investigation to illustrate the effectiveness of our model. Moreover, we have also released our code[2] of IEKT.

Algorithm 1: Training procedure of IEKT

```
Randomly initialize the learning parameters;
while not converged do
    for batch in data do
        for (t = 0; t < T; t = t + 1) do
            estimate the students' cognition levels (Eq. 6);
            predict the responses (Eq.7);
            estimate the students' acquisition levels (Eq.12);
            update the knowledge states (Eq. 14);
            obtain the current rewards (Eq. 15);
        end
        compute the cumulative rewards (Eq. 17);
        compute the vector selection loss (Eq. 19);
        compute the knowledge tracing loss (Eq. 20);
        compute the gradient and update the parameters
          w.r.t the loss $\mathcal{L}(Eq. 21)$;
    end
end
```

5.1 Dataset

We evaluate our method on four public datasets: ASSIST09, ASSIST12, EdNet, and Junyi. All of the four datasets are sampled from the logs of students' question-answering history. We take the IDs of questions, the IDs of the concepts which are related to the corresponding questions, and students' responses from records. The statistics of the four datasets are shown in Table 2. The maximum length of students' question-answering history is set to 200. We split 80% data for training and validation, and 20% for testing.

ASSIST09. This dataset is gathered from the ASSISTments online tutoring platform [13]. We filter out the records without concept tags and the students whose question-answering history length is less than ten. Each question in this dataset is related to one to four knowledge concepts.

ASSIST12. This dataset is also gathered from the ASSISTments online tutoring platform [13], and we do the same preprocessing as ASSIST2009. Each question in this dataset is related to one knowledge concept.

EdNet. This dataset is contributed by Choi et al. [7]. It is collected from Santa, an AI tutoring service platform. EdNet is composed of 131,441,538 records, which involve 784,309 students. As the full dataset requires many computation resources, we randomly sample 4,702 students' learning records from this dataset. In the samples, the question-answering history of all the students is more than ten. Each question is related to one to seven knowledge concepts.

Junyi. This dataset was collected from Junyi Academy, an e-learning website [4]. This is also a large scale dataset like EdNet.

[2]Source code and datasets will be available at https://github.com/githubg0/iekt

Table 2: Dataset Statistics.

Dataset	ASSIST09	ASSIST12	EdNet	Junyi
Students	2,968	22,422	4,702	7,000
Records	185,110	1,839,429	326,267	622,781
Questions	15,003	45,543	11,060	1,978
Concepts	121	99	189	39
Questions Per Concept	150.76	460.03	128.73	50.72
Concepts Per Question	1.22	1.0	2.21	1.0
Attempts Per Question	12.34	40.39	29.50	314.85
Attempts Per Concept	1914.21	18,580.10	4026.93	15,968.74
Positive Label Rate	63.80%	69.60%	59.69 %	67.30%

To make the computation resource affordable for us, we randomly sample 7,000 students whose learning history length is more than ten from the dataset. Each question in this dataset is related to one knowledge concept.

5.2 Baselines

To evaluate the effectiveness of our model, we compare our method with four groups of 11 knowledge tracing models. They are:

Traditional methods. These methods predict the students' responses according to the factors that affect students' learning.

- BKT [8] uses binary variables to represent students' knowledge states, which indicate whether a student masters the related concepts. It applies the Bayesian network to learn the factors which affect students' responses, such as learning rate, slip and guess probability.
- KTM [34] considers the side information like questions, concepts, and uses factorization machine to predict students' responses. In this paper, we only take questions, and concepts into consideration to keep consistent with other models.

Single-state methods. These methods maintain one vector to represents the knowledge state of a student.

- DKT [27] represents students' knowledge states with the hidden states of RNN. It updates students' knowledge states by feeding the question representation into RNN, and uses the knowledge states to predict students' responses.
- DFKT [24] is similar to DKT in knowledge states representation and update. However, DFKT considers the impact of forgetting behavior on students' responses. Thus, it introduces the repeated time gap, sequence time gap, and past trial counts into the input of RNN.
- DHKT [36] also uses the hidden states of RNN to represent the knowledge states like DKT. However, different from DKT, it considers both the knowledge state and question in students' response prediction.
- EERNNA [17] uses the hidden states of RNN to represent students' knowledge states. Except for the current knowledge state and question, it considers the impact of the previous states on the current response prediction.

Multi-state methods. These methods maintain a vector for each concept to represent students' knowledge states.

- DKVMN [41] uses the Key-value memory network to store students' knowledge states, and it updates the knowledge states of all the concepts when students interact with questions.
- GKT [25] introduces a graph to represent the relation of concepts. It updates the knowledge states by referring to the graph.

State-free methods. These methods maintain no vector to represent students' knowledge states.

- CKT [29] considers the historical relevant performance and concept-wised percent correct of students. It applies CNN on the concatenation of these factors to capture the individual learning rate and predict students' responses.
- SAKT [26] introduces the self-attention [33] to capture the correlation of questions and the responses of the students.
- AKT-NR [15] uses the Transformer [33] to model the question-answering sequence of students. However, it adopts the exponential decay in multi-head attention score.

5.3 Implementation Details

The dimension of the knowledge state (hidden state of RNN) is 64. The dimension of question embedding and concept embedding are also 64. We use multiple layer perceptron to implement the mapping function f_p, f_e. The number of hidden layer is one for both mapping functions. For all the datasets, we set $\mathbf{M} \in \mathbf{R}^{10\times128}$, $\mathbf{S} \in \mathbf{R}^{10\times128}$. We use a one-layer GRU to update the knowledge states of students. The γ in Eq. 17 is 0.93. The λ in Eq. 21 is 40. The optimizer is Adam [18], and the learning rate is 0.001, weight decay is 0.0001.

For the baselines which use RNN to update the knowledge states, we also apply GRU for fairness. The hyperparameters of each model are carefully tuned to the best performances to make the results comparable. For all of the deep methods,we represent the question in a same manner.

5.4 Experiment Result

We measure the ACC, AUC, and the statistical significance to evaluate the performance of compared models. Specifically, we deploy a MannWhitney U test [23] under AUC metric, and a t-test [3] under ACC metric. A higher AUC (ACC) indicates a better performance in predicting students' responses. Table 3 shows the converged ACC, AUC, and the statistical significance of our model against the baseline models.

According to Table 3, we observe that our method outperforms all of the 11 baselines and achieves the best performance. On the ASSIST09, our model outperforms the best baseline (GKT) by 0.84% in ACC and 1.77% in AUC. On the ASSIST12, our model outperforms the best baseline by 1.07% (DHKT) in ACC and 2.43% (AKT-NR) in AUC. On the EdNet, our model is better than the best baseline (AKT-NR) by 1.42% in ACC and 2.17% in AUC. On the Junyi, our model is better than the best baseline (DHKT) by 0.58% in ACC and 0.60% in AUC. It deserves to mention that although our method is similar to DHKT in the basic framework, our model is significantly better than DHKT. That means introducing the CE and KASE modules into knowledge tracing benefits the performance.

Table 3: The performance on four public datasets. * indicates p-value < 0.05 in the significance test.

Group	Model	ASSIST09		ASSIST12		EdNet		Junyi	
		ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
Traditional	BKT	0.6142	0.6418	0.6028	0.5987	0.5334	0.5516	0.7043	0.7165
	KTM	0.6764	0.6552	0.7166	0.6934	0.6634	0.7004	0.7211	0.7207
State-free	CKT	0.6845	0.7025	0.6992	0.6465	0.6208	0.6363	0.8216	0.8805
	SAKT	0.6730	0.6804	0.7150	0.6914	0.6660	0.6984	0.7609	0.8004
	AKT-NR	0.6825	0.6949	0.7221	0.7098	0.6703	0.7087	0.8096	0.8595
Multi-state	DKVMN	0.7076	0.7066	0.7048	0.6252	0.6660	0.6979	0.8342	0.8884
	GKT	0.7258	0.7544	0.7179	0.6872	0.6639	0.6936	0.8072	0.8586
Single-state	DKT	0.7102	0.7220	0.7041	0.6398	0.6454	0.6580	0.7969	0.8462
	DFKT	0.7073	0.7215	0.7045	0.6447	0.6428	0.6526	0.8009	0.8567
	DHKT	0.7203	0.7423	0.7239	0.7088	0.6594	0.6877	0.8398	0.8939
	EERNNA	0.7115	0.7359	0.7227	0.7012	0.6693	0.7032	0.8091	0.8665
	IEKT	**0.7342***	**0.7720***	**0.7346***	**0.7341***	**0.6845***	**0.7305***	**0.8456***	**0.8999***

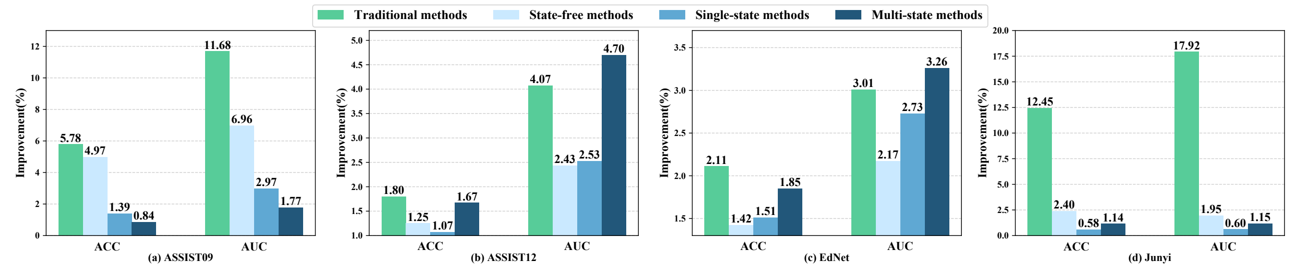


Figure 3: The improvement on different types of methods.

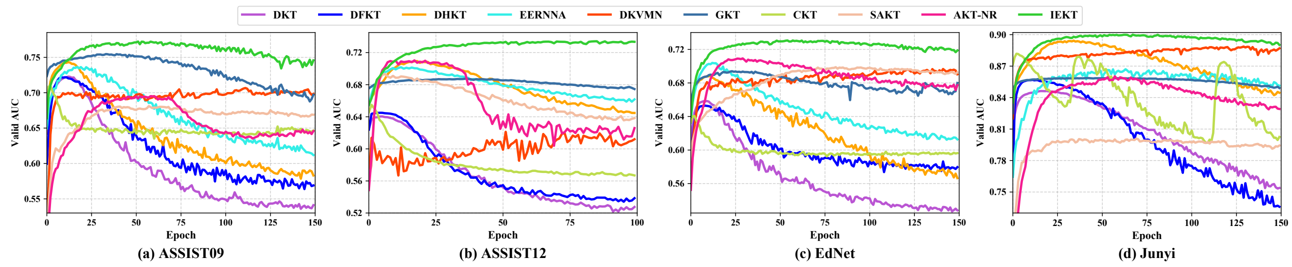


Figure 4: The convergence curves.

Table 4: The performance of introducing CE or KASE to single-state methods. $*$ indicates p-value < 0.05 in the significance test.

Model	ASSIST09		ASSIST12		EdNet		Junyi	
	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
DKT	0.7102	0.7220	0.7041	0.6398	0.6454	0.6580	0.7969	0.8462
DKT + CE	0.7097	0.7224	0.7046*	0.6422	0.6452	0.6561	0.7993*	0.8516
DKT + KASE	0.7136*	0.7280	0.7046*	0.6418	0.6459*	0.6591	0.7971*	0.8474
DFKT	0.7073	0.7215	0.7045	0.6447	0.6428	0.6526	0.8009	0.8567
DFKT + CE	0.7119*	0.7316*	0.7045	0.6450*	0.6459*	0.6604*	0.8024*	0.8592*
DFKT + KASE	0.7139*	0.7350*	0.7055*	0.6484*	0.6483*	0.6659*	0.8037*	0.8604*
EERNNA	0.7115	0.7359	0.7227	0.7012	0.6693	0.7032	0.8091	0.8665
EERNNA + CE	0.7139*	0.7438*	0.7263*	0.7126*	0.6738*	0.7105*	0.8108*	0.8671*
EERNNA + KASE	0.7163*	0.7467*	0.7257*	0.7093*	0.6778*	0.7177*	0.8128*	0.8690*

We can observe that deep learning-based methods are superior to the traditional methods in most cases. However, traditional methods have comparable even better performance in some cases. For instance, KTM is better than DKT, DFKT, DKVMN, CKT and SAKT in ASSIST12, and better than DKT, DFKT, DHKT and CKT in EdNet. State-free methods have a deficiency in ASSIST09. However, they have comparable performance with the state-based methods

Table 5: The different variants of comparative settings.

	CE module	$\mathcal{L}_m$ in Eq. 19	KASE module	$\mathcal{L}_s$ in Eq. 19	GRU layers
MovCE	×	×	✓	✓	1
MovKASE	✓	✓	×	×	1
MovRL	✓	×	✓	×	1
MovCK	×	×	×	×	1
MovCKI	×	×	×	×	3
IEKT	✓	✓	✓	✓	1

in ASSIST12, EdNet and Junyi. In general, there is no obvious differences in performance among state-free methods, multi-state methods and single-state methods.

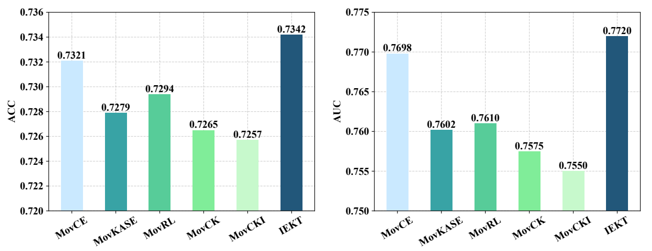
We also contrast the performance of our method with the best methods in different groups, which as Figure 3 shows. Compared with multi-state methods, our method only uses one vector to represents students' knowledge states on all concepts. However, our method is superior to all the multi-state methods. It outperforms these methods by 0.84% - 1.85% in ACC and 1.15% - 4.70% in AUC. Compared with the single-state methods, our method also demonstrates superior performance, especially in ASSIST09, ASSIST12, and EdNet. It improves the ACC by 1.07% - 1.51%, and promotes the AUC by 2.53% - 2.97%. However, it has the least improvement on Junyi, which outperforms these methods by 0.58% in ACC and 0.60% in AUC. We assume the reason is that the task is easier in Junyi than other datasets. Compared with the state-free methods, although we applied no attention mechanism, our method has 1.25% - 4.97% improvement in ACC and 1.95% - 6.96% improvement in AUC.

We also compare the learning curves of deep learning methods. As shown in Figure 4, our method converges quickly and demonstrates good stability.

5.5 Ablation Study

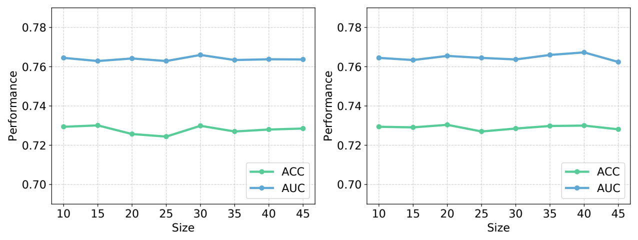
To further investigate the contributions of the modules in IEKT, we conduct some ablation studies on ASSIST09. We have five comparative settings:

- MovCE removes CE module from IEKT. That means, we ignore students' cognition levels ($\mathbf{m}_t$ in Eq. 7) when predicting their responses. It deserves to mention that we also remove the $\mathcal{L}_m$ in Eq. 19, which is related to the cognition vector selection.
- MovKASE removes KASE module from IEKT. We ignore students' knowledge acquisition sensitivity and directly concatenate $\mathbf{v}_t$ with the students' responses like the previous works. Correspondingly, we also remove $\mathcal{L}_s$ in Eq. 19.
- MovRL removes the Policy Gradient from the model learning. That means our training strategy is same to the previous work.
- MovCK removes both CE and KASE modules in IEKT. That means we do not consider the individual cognition and acquisition sensitivity on the questions.
- MovCKI removes both CE and KASE modules in IEKT and increases the layer of GRU to three layers (as we reuse the knowledge states in CE module and KASE module. We aim to investigate whether the reusing of knowledge state causes the improvement in performance).



(a) The ACC of comparative settings. (b) The AUC of comparative settings.

Figure 5: The contribution of modules in IEKT.



(a) Cognition Space Size. (b) Acquisition Space Size.

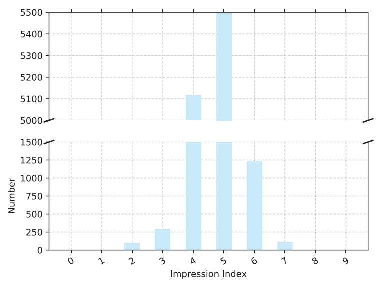
Figure 6: The impact of hyperparameters.

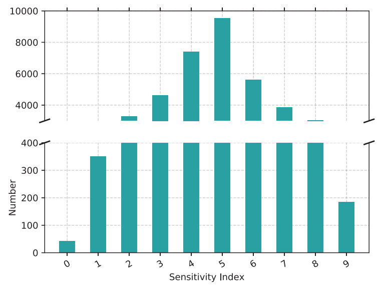
We contrast the differences of the comparative settings in Table 5, and we visualize the performance of the comparative settings in Figure 5. From Figure 5, we can observe that the performance of IEKT will decrease no matter which module of the model is removed. That means all of these modules have a contribution to the performance. We can also observe that removing the KASE from IEKT (MovKASE) has more significant influence on the performance than removing CE (MovCE). We assume that because the KASE affects all the predictions in the future, but the CE only affects the current response prediction. Thus, removing CE module has less influence on performance. We can also find that removing the optimization of reinforcement learning from training (MovRL) also affects IEKT's performance. We have stated the reason previously, the vectors selection cut off some gradients in the network, which affects the learning of parameters. The IEKT has a better performance than MovCKI, which indicates that the performance of IEKT is not the result of rough increasing the layer of networks. That means IEKT is reasonable.

To validate whether CE and KASE are applicable to the other single-state methods, we introduce the CE and KASE to DKT, DFKT, and EERNNA. Here, we did not apply CE and KASE to DHKT, because our basic framework is similar to DHKT. We transfer the same CE and KASE modules from IEKT to these models without refining. We test the performance of these models on all four datasets. The experiment results are shown in Table 4. From the table, we can observe that introducing CE and KASE will improve the performance in most cases.

5.6 Sensitivity Analysis

To investigate the sensitivity of our model, we evaluate the impact of different hyperparameters on the performance of our method.


(a) Cognition vectors distribution.


(b) Acquisition vectors distribution.

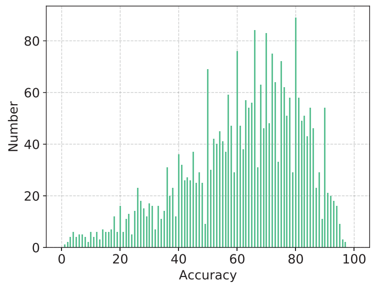
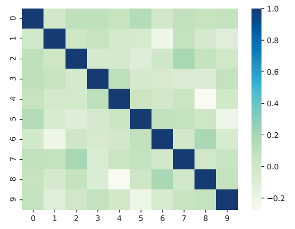

(c) Student accuracy distribution.

Figure 7: The cognition distribution and acquisition sensitivity distribution.


(a) The similarity of cognition vectors.

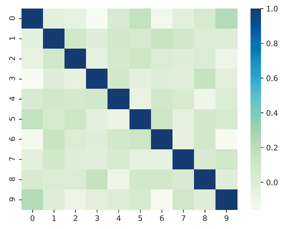

(b) The similarity of acquisition vectors.

Figure 8: The similarity of vectors.

We test the performance of our method under different action space for cognition vectors, different action space for acquisition vector. All experiments are conducted on ASSIST09. We set the action space from 10 to 45 with five intervals, and the other hyperparameters remain the same in the experiments. As it is illustrated in Figure 6, for both cognition and acquisition sensitivity tests, the performance has no obvious volatility in ACC and AUC when we change the value of the hyperparameters from 10 to 45. That means our method has good stability. Thus, the optimal hyperparameters of our method are easily obtained.

5.7 Vector Distributions

We compute the similarity of the cognition vectors and the acquisition vectors. The results are shown in Figure 8. We can find that these vectors are different from each other. That means each vector in the cognition matrix represents the different cognition levels. Each vector in the acquisition matrix also represents the different knowledge acquisition sensitivity levels.

As students' cognition levels and their knowledge acquisition sensitivity represent their learning capability, we contrast the distribution of the selected vector indexes with question-answering accuracy as shown in Figure 7. Like many question-answering and examination results in the off-line education scenario [2, 11, 28], the students' accuracy in this dataset is also close to the Z-score distribution as Figure 7(c) shows. There are a large number of students whose accuracy ranges from 50% - 85%. However, the number of students whose accuracy less than 50% or more than 85% is small. We can also observe the same pattern in cognition vector and acquisition vector selection as Figure 7(a) and (b) shows. Some cognition vectors and sensitivity vectors are selected many times, while some are selected limited times. It demonstrates that our model learns that some of the cognition vectors and acquisition vectors are shared by many students. However, some are only applicable to a minority of students. That means most students have a similar understanding on the same question. In contrast, minority students have different understanding on the question. Furthermore, Most of the students have ordinary knowledge acquisition sensitivity. However, a small number of students have strong sensitivity on questions. They can master a knowledge concept within a few interactions with the related questions, but some students need to pay more efforts to their studies until they master the knowledge concepts. This phenomenon is consistent with the capability reflected by question-answering accuracy in Figure 7(c).

6 CONCLUSION

In this paper, we propose a knowledge tracing model called IEKT, which is a single-state RNN-based method. Compared with the previous RNN-based methods, it introduces the CE and KASE modules to model the student characters. The CE module estimates students' cognition on the questions according to their knowledge states and the question representation. The KASE estimates students' knowledge acquisition sensitivity according to their knowledge states, their responses, and the question representation. In model training, we apply the reinforcement learning to assist the optimal searching. We validate the performance of IEKT on four public datasets, and compare it with 11 knowledge tracing methods. As IEKT considers students' cognition levels and knowledge acquisition sensitivity on questions, it has superior performance compared with other knowledge tracing models. We also introduce our CE and KASE to the other single-state models. The modules improve their performance in most cases. That means our CE and KASE also applicable to these models.

IEKT improves the performance of knowledge tracing. However, we find that the selected cognition vectors are concentrated on a small part of the vectors in the cognition matrix. That means there is still a gap between the learning result and students' capability distribution in the real world. The expectation might be better than the result we obtained. Thus, future works can further explore the solutions to compensate for this deficiency.

ACKNOWLEDGEMENT

The corresponding author Yong Yu thanks the support of NSFC (61772333).

REFERENCES

[1] Ghodai Abdelrahman and Qing Wang. 2019. Knowledge Tracing with Sequential Key-Value Memory Networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 175–184.

[2] Herman Aguinis and Steven A Branstetter. 2007. Teaching the concept of the sampling distribution of the mean. *Journal of Management Education* 31, 4 (2007), 467–483.

[3] Bhaskar Bhattacharya and Desale Habtzghi. 2002. Median of the p value under the alternative hypothesis. *The American Statistician* 56, 3 (2002), 202–206.

[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] Penghe Chen, Yu Lu, Vincent W Zheng, and Yang Pian. 2018. Prerequisite-driven deep knowledge tracing. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 39–48.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

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

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

[9] Paul De Boeck and Mark Wilson. 2004. *Explanatory item response models: A generalized linear and nonlinear approach*. Springer Science & Business Media.

[10] Phung Do, Kha Nguyen, Thanh Nguyen Vu, Tran Nam Dung, and Tuan Dinh Le. 2017. Integrating knowledge-based reasoning algorithms and collaborative filtering into e-learning material recommendation system. In *International Conference on Future Data and Security Engineering*. Springer, 419–432.

[11] Neil J Dorans. 2002. The recentering of SAT® scales and its effects on score distributions and score interpretations. *ETS Research Report Series* 2002, 1 (2002), i–21.

[12] 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.

[13] Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction* 19, 3 (2009), 243–266.

[14] Howard Gardner et al. 1992. *Multiple intelligences*. Vol. 5. Minnesota Center for Arts Education.

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

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[17] Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, Guoping Hu, et al. 2019. Ekt: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering* (2019).

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

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[20] 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.

[21] Yunfei Liu, Yang Yang, Chen Xianyu, Shen Jian, Zhang Haifeng, and Yu Yong. 2020. Improving Knowledge Tracing via Pre-training Question Embeddings. In *Twenty-Ninth International Joint Conference on Artificial Intelligence*.

[22] Yu Lu, Deliang Wang, Qinggang Meng, and Penghe Chen. 2020. Towards Interpretable Deep Learning Models for Knowledge Tracing. *arXiv preprint arXiv:2005.06139* (2020).

[23] Simon J Mason and Nicholas E Graham. 2002. Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* 128, 584 (2002), 2145–2166.

[24] Koki Nagatani, Qian Zhang, Masahiro Sato, Yan-Ying Chen, Francine Chen, and Tomoko Ohkuma. 2019. Augmenting knowledge tracing by considering forgetting behavior. In *The World Wide Web Conference*. 3101–3107.

[25] Hiromi Nakagawa, Yusuke Iwasawa, and Yutaka Matsuo. 2019. Graph-based Knowledge Tracing: Modeling Student Proficiency Using Graph Neural Network. In *2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. IEEE, 156–163.

[26] Shalini Pandey and George Karypis. 2019. A Self-Attentive model for Knowledge Tracing. *arXiv preprint arXiv:1907.06837* (2019).

[27] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. In *Advances in neural information processing systems*. 505–513.

[28] LJ Salomon, JP Bernard, and Y Ville. 2005. Analysis of Z-score distribution for the quality control of fetal ultrasound measurements at 20–24 weeks. *Ultrasound in Obstetrics and Gynecology: The Official Journal of the International Society of Ultrasound in Obstetrics and Gynecology* 26, 7 (2005), 750–754.

[29] Shuanghong Shen, Qi Liu, Enhong Chen, Han Wu, Zhenya Huang, Weihao Zhao, Yu Su, Haiping Ma, and Shijin Wang. 2020. Convolutional Knowledge Tracing: Modeling Individualization in Student Learning Process. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1857–1860.

[30] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris Ding, Si Wei, and Guoping Hu. 2018. Exercise-enhanced sequential modeling for student performance prediction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[31] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. 1999. Policy gradient methods for reinforcement learning with function approximation.. In *NIPs*, Vol. 99. Citeseer, 1057–1063.

[32] Hanshuang Tong, Yun Zhou, and Zhen Wang. 2020. HGKT: Introducing Problem Schema with Hierarchical Exercise Graph for Knowledge Tracing. *arXiv preprint arXiv:2006.16915* (2020).

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[34] Jill-Jênn Vie and Hisashi Kashima. 2019. Knowledge tracing machines: Factorization machines for knowledge tracing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 750–757.

[35] Shuhan Wang, Hao Wu, Ji Hun Kim, and Erik Andersen. 2019. Adaptive learning material recommendation in online language education. In *International Conference on Artificial Intelligence in Education*. Springer, 298–302.

[36] Tianqi Wang, Fenglong Ma, and Jing Gao. 2019. Deep hierarchical knowledge tracing. In *Proceedings of the 12th International Conference on Educational Data Mining*.

[37] Kevin H Wilson, Yan Karklin, Bojian Han, and Chaitanya Ekanadham. 2016. Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. *arXiv preprint arXiv:1604.02336* (2016).

[38] Kevin H Wilson, Xiaolu Xiong, Mohammad Khajah, Robert V Lindsey, Siyuan Zhao, Yan Karklin, Eric G Van Inwegen, Bojian Han, Chaitanya Ekanadham, Joseph E Beck, et al. 2016. Estimating student proficiency: Deep learning is not the panacea. In *In Neural Information Processing Systems, Workshop on Machine Learning for Education*. 3.

[39] Shanghui Yang, Mengxia Zhu, Jingyang Hou, and Xuesong Lu. 2020. Deep Knowledge Tracing with Convolutions. *arXiv preprint arXiv:2008.01169* (2020).

[40] Michael V Yudelson, Kenneth R Koedinger, and Geoffrey J Gordon. 2013. Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education*. Springer, 171–180.

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