DKT-LCIRT: A Deep Knowledge Tracking Model Integrating Learning Capability and Item Response Theory

Guangquan Li 1,2, Junkai Shuai 1, Yuqing Hu 1, Yonghong Zhang 3,* , Yinglong Wang 1, Tonghua Yang 2,* and Naixue Xiong 4

1 School of Computer and Information Engineering, Jiangxi Agriculture University, Nanchang 330045, China
2 School of Vocational Teachers, Jiangxi Agriculture University, Nanchang 330045, China
3 School of Information Engineering, Jiangxi Vocational College of Mechanical & Electrical Technology, Nanchang 330045, China
4 Department of Computer, Mathematical and Physical Sciences, Sul Ross State University, Alpine, TX 79830, USA
* Correspondence: wzfzyh@jxau.edu.cn (Y.Z.); yangth883@jxau.edu.cn (T.Y.)

Abstract: In the realm of intelligent education, knowledge tracking is a critical study topic. Deep learning-based knowledge tracking models have better predictive performance compared to traditional knowledge tracking models, but the models are less interpretable and also often ignore the intrinsic differences among students (e.g., learning capability, guessing capability, etc.), resulting in a lack of personalization of predictive results. To further reflect the personalized differences among students and enhance the interpretability of the model at the same time, a Deep Knowledge Tracking model integrating Learning Capability and Item Response Theory (DKT-LCIRT) is proposed. The model dynamically calculates students’ learning capability by each time interval and allocates each student to groups with similar learning capabilities to increase the predictive performance of the model. Furthermore, the model introduces item response theory to enhance the interpretability of the model. Substantial experiments on four real datasets were carried out, and the experimental results showed that the DKT-LCIRT model improved the AUC by 3% and the ACC by 2% compared to other models. The results confirmed that the DKT-LCIRT model outperformed other classical models in terms of predictive performance, fully reflecting students’ individualization and adding a more meaningful interpretation to the model.

Keywords: knowledge tracking; deep learning; item response theory; learning capability; intelligent education

1. Introduction

As artificial intelligence technology has advanced, online education platforms such as Massive Open Online Courses (MOOC) and Intelligent Tutoring Systems (ITS) have become very popular. No matter what time or place, students can learn high-quality courses and improve their learning efficiency through online education systems, but online education systems are only widely used in large cities and are not fully popular in many small and medium-sized cities. Due to the impact of the new coronavirus pneumonia in 2020, offline education is difficult to carry out. The online education system is being accepted by more and more parents and students under the situation of school closure and non-stop learning. The system also stores a large amount of online education data [1]. While conducting online education, how to provide students with personalized guidance has become another major problem. Personalized guidance aims to provide appropriate learning plans based on students’ knowledge state and improve their learning efficiency, and knowledge tracking is the key to solving this problem [2]. Knowledge tracking is the process of tracking student’s knowledge state depending on the sequence of previous interactions and predicting whether they can answer the upcoming exercise correctly. The
interaction means the response sequence between the student and the online education system. The system receives the result of the student’s answer, changes the student’s knowledge state according to the result, and then recommends an appropriate exercise. The student gives feedback to the system after answering the exercise, and so on and so forth. Students’ mastery of knowledge points is represented by their knowledge state, as well as by keeping track of students’ knowledge state, and the system can better understand students’ knowledge level and provide appropriate learning plans accordingly [3]. The research on knowledge tracking dates from the late 1970s, and a large number of knowledge tracking models [4] have been proposed in these four decades, including models based on item response theory, Bayesian networks, and deep learning.

The probability of correctly answering the exercises depends mainly on their mastery degree of the knowledge points contained in the exercises. Students gradually improve their knowledge through continuous learning [5], and the speed of improvement of knowledge mastery is influenced by learning capability. Each student’s learning capability is different and can change at any time during the learning process. Classifying students into different groups of similar learning capabilities depending on previous performance in order to provide more personalized instruction for each group of students has been the subject of much research in the field of education [6,7]. Although deep knowledge tracking models outperform traditional knowledge tracking models in terms of predictive performance, basic deep knowledge tracking models also have the following shortcomings: (1) The basic deep knowledge tracking models often ignore the intrinsic differences among students, such as learning capability and guessing capability, and all students have the same intrinsic capability, leading to a lack of personalization. (2) The basic deep knowledge tracking models have been poorly interpretable because of the black box characteristics of deep learning.

To overcome these two shortcomings, this paper proposes a deep knowledge tracking model DKT-LCIRT that integrates learning capability and item response theory. The DKT-LCIRT model is an improvement on the Dynamic Key-Value Memory Network model (DKVMN) [8,9] by adding learning capability features to the input layer of the neural network [10,11] and introducing item response theory to the output layer of the neural network [12]. Feature engineering is a crucial part of deep learning models, and features determine the upper limit of deep learning models, and by constructing new features, more information contained in the data can be mined [13,14], making the data further expressive [15–17]. The DKT-LCIRT model extracts potential features from the sequences of students’ historical interactions [18,19], and then uses deep learning neural networks to track the student’s knowledge state, output student capability parameter, and exercise difficulty parameter. Finally, the probability of students answering the exercises correctly is predicted depending on the item response theory.

The primary contributions of this paper are as follows: (1) The DKT-LCIRT model better reflects the individualization of students and enhances the predictive performance by adding students’ learning capability features, dynamically calculating students’ learning capability at each time interval, and assigning students to groups with similar learning capabilities. (2) The DKT-LCIRT model improves the interpretability by introducing item response theory to provide meaningful estimates of students’ capability levels and difficulty levels of the exercises. (3) The study validated the practical effects of adding both learning capability features and item response theory and filled in the gaps of previous research in this area in the literature.

The rest of this paper is organized as follows: The Section 2 reviews relevant research on knowledge tracking. The Section 3 details the specific implementation of the DKT-LCIRT model. The Section 4 describes details of the experimental study. The Section 5 concludes the paper and considers future research prospects.
2. Related Work

First of all, knowledge tracking can be viewed as a supervised sequential learning problem. Suppose there are $|E|$ students and $|F|$ exercises in an online education system, and given the sequence of historical interaction $u = \{(q_1, a_1), (q_2, a_2), \ldots, (q_t, a_t)\}$ between students and the system, where $q_i$ represents the exercises done by the students at the moment $t$ and $a_i$ represents the correctness of the answers to the exercises. In general, $a_i = 1$ represents correct answers and $a_i = 0$ represents incorrect answers. Knowledge tracking can predict the probability of students correctly answering the upcoming exercise $q_{t+1}$ depending on the sequence of historical interactions. Then, three typical types of knowledge tracking models are reviewed here.

2.1. Item Response Theory

Item Response Theory [20] (IRT), which was applied in testing environments in the 1950s, outputs the probability of correctly answering an exercise $j$ on a test depending on the student’s capability level $\theta$ and the difficulty level $\beta_j$ of the exercise. If the student’s capability level is high, the probability of correctly answering the exercise is higher; on the other hand, if the difficulty level of the exercise is high, the probability of correctly answering the exercise is lower. At the beginning, the IRT was designed to be used in testing environments. The IRT presumes that students’ capability levels do not change during the testing process and that students’ knowledge states are static, which is a reasonable assumption for testing environments. However, the student’s knowledge state in knowledge tracking changes all the time. Therefore, it cannot be directly applied to knowledge tracking tasks.

To solve this challenge, researchers have combined IRT with deep learning methods, such as the DIRT model proposed by Cheng et al. [21], and the NeuralCD model proposed by Wang et al. [22,23]. Both models use deep neural networks to extract the complicated information included in the data and track their knowledge state while ensuring the interpretability of IRT.

2.2. Bayesian Knowledge Tracking

Bayesian Knowledge Tracking (BKT) was proposed by Corbett and Anderson in the 1990s [24], and the BKT model is probably the first to relax the assumption of a static knowledge state. The static knowledge state assumption is an unreasonable assumption for a learning environment. The student’s knowledge state is represented by the BKT model as a series of binary variables, each of which indicates whether they have mastered a single knowledge point. Each binary variable is refreshed using a Hidden Markov Model (HMM) when the student answers an exercise. To estimate student knowledge point mastery based on the sequences of their historical interaction, the BKT model requires four parameters: the prior probability $p(L)$, which indicates the probability of having mastered the knowledge point at the beginning; the learning probability $p(T)$, which indicates the probability of transferring the knowledge point from the non-mastery state to the mastery state; the guessing probability $p(G)$, which indicates the probability of correctly answering the exercise even if they did not master the knowledge point; and the slipping probability $p(S)$, which indicates the probability of incorrectly answering the exercise even if they mastered the knowledge point. The BKT model divides the mastery degree of each knowledge point into unmastered and mastered and models them separately, ignoring the intermediate mastery level and not considering the relationship between knowledge points.

Over the years, many variants of BKT have been proposed. Baker et al. [25] enhanced model predictive performance by introducing student slip and guess parameters. Yudelson et al. [26] investigated personalization of student learning rate parameters and found better predictive performance.
2.3. Deep Knowledge Tracking

Similar to the BKT model, Deep Knowledge Tracking (DKT) [27] processes the sequences of student’s historical interaction, but takes advantage of neural networks and breaks the limitations of knowledge point separation and binary state assumptions. In the DKT model, the interaction tuples \((q_t, a_t)\) are first converted into input vectors using one-hot coding. Then, the input vector is passed to the hidden layer. Through Long Short-Term Memory (LSTM) [28] generates the hidden state that theoretically summarizes all past information so that the hidden state can be understood as the potential knowledge state resulting from the student’s previous learning process. Finally, the output layer uses the hidden state to generate the output vector, which represents the probability of correctly answering the exercise.

Basic deep knowledge tracking models suffer from poor interpretability, long-term dependency, and few learning features. To address the interpretability problem, Zhang et al. proposed the DKVMN model [29], which draws on Memory Augmented Neural Network (MANN) [30] to model students’ knowledge mastery process using two memory matrices, key and value. The DKVMN model can capture the relationship between different knowledge points while tracking their mastery states. Liu et al. proposed the EKT model [31] to track students’ knowledge states through a Bi-directional Long Short-Term Memory (Bi-LSTM) network with an attention module. The EKT model uses the textual content of the exercises to encode the exercise embeddings so that the exercise embeddings contain information about the textual features of the exercises. To address the long-term dependency problem, Choi et al. proposed the SAINT model [32] based on a Transformer [33], which utilizes an encoder–decoder structure consisting of superimposed attention layers. The SAINT model inputs the exercise embedding to the encoder, and the encoder output and the interaction embedding to the decoder. To address the problem of missing learning features, Wang et al. proposed the DHKT model [34], which models the hierarchical relationships between exercises using the relationships between exercises and knowledge points. Nakagawa et al. proposed the GKT model [35] that transforms the knowledge tracking problem into a time-series node-level classification task in the Graph Neural Network (GNN) by representing the relationships between knowledge points as one directed graph. Yang et al. proposed the GIKT model [36] that merges problem and skill relevance by embedding propagation utilizing the Graph Convolutional Network (GCN). Shen et al. proposed the CKT model [37] that utilizes the Convolutional Neural Network (CNN) to capture learning rate features from students’ interaction histories.

3. Our Proposed DKT-LCIRT Scheme

Basic deep knowledge tracking models can effectively track students’ knowledge states, but the predictive results lack personalization because they often ignore the intrinsic differences among students (e.g., learning capability, guessing capability, etc.). In addition, the interpretability of the basic deep knowledge tracking models is bad. Therefore, this paper proposes a Deep Knowledge Tracking model integrating Learning Capability and Item Response Theory (DKT-LCIRT). The model tracks students’ knowledge state by adding learning capability features, dynamically calculating students’ learning capability at each time interval, and assigning students to groups with similar learning capabilities [38,39]. This relaxes the assumption that all students have the same learning capability, and learning capability remains constant over time, achieving individualization of students, thus enhancing the predictive performance of the model. The model also introduces item response theory to provide meaningful estimates of students’ capability levels and difficulty levels of the exercises, enhancing the interpretability of the model.

3.1. Learning Capability Features

Students learning capability features are extracted by calculating learning capabilities based on all historical performances of students until the start of the next time interval, and
then dynamically assigning students to groups with similar learning capabilities through the k-means clustering algorithm [40,41].

3.1.1. Time Interval Division

The time interval is the segment in the interaction sequence where the number of responses is fixed [42], and the student’s learning capability is recalculated after each time interval. As shown in Figure 1, the interaction sequence containing 23 answers was divided into 5-time intervals, and students answered five exercises at each time interval.

![Figure 1. Time interval in the student interaction sequence.](image)

3.1.2. Learning Capability Calculation

Students’ learning capability is coded as a vector with the number of elements equal to the number of knowledge points. The difference between the correct and incorrect rates of each knowledge point from students’ previous interactions is converted into elements of the learning capability vector. Students’ learning capability is calculated by Equations (1)–(4):

\[
Correct(s_m)_{1z} = \sum_{i=1}^{z} \left( \frac{s_{mi} = 1}{|N_{mi}|} \right), \tag{1}
\]

\[
Incorrect(s_m)_{1z} = \sum_{i=1}^{z} \left( \frac{s_{mi} = 0}{|N_{mi}|} \right), \tag{2}
\]

\[
R(s_m)_{1z} = Correct(s_m)_{1z} - Incorrect(s_m)_{1z}, \tag{3}
\]

\[
d_{i1z}^{0} = (R(s_1)_{1z}, R(s_2)_{1z}, \ldots, R(s_m)_{1z}), \tag{4}
\]

where Correct(s_m)_{1z} indicates the proportion of knowledge points s_m answered correctly by student i during time intervals 1 to z. Incorrect(s_m)_{1z} indicates the proportion of knowledge points s_m answered incorrectly by student i. R(s_m)_{1z} indicates the difference in students’ performance on knowledge points s_m. d_{i1z}^{0} indicates the student i’s learning capability vector. |N_{mi}| indicates the total number of times student i answered the knowledge point s_m.

3.1.3. K-Means Clustering Grouping

At each time interval, students are assigned to groups c with similar learning capability by performing k-means clustering on the calculated learning capability d_{i1z}^{0} of students. c is based on all historical performance prior to time interval z. In the k-means clustering training phase, the centroid of the k student groups is identified. Once the centroid is determined, the centroid of the k student groups will not change in the following grouping process. The students are assigned to the nearest student group by Equation (5):

\[
Cluster(i, z) = \arg\min_C \sum_{c=1}^{k} \sum_{d_{1z-1} \in C_c} \left\| d_{1z-1}^i - \mu_c \right\|^2, \tag{5}
\]

where Cluster(i, z) represents the student i’s grouping at time interval z. k means that there are k learning capability groupings. d_{1z-1}^i represents the student i’s learning capability at time intervals 1 to z − 1. \mu_c represents the centroid of group c. The specific process of grouping students’ learning capability is shown in Figure 2.
2
( , ) arg min 1: 11 1 : 1
k iCluster i z d zd d dC cz
=  −−= −

where

represents the student i's grouping at time interval z. k means that there are k learning capability groupings. 

represents the student i's learning capability at time intervals 1 to z.

c represents the centroid of group c. The specific process of grouping students' learning capability is shown in Figure 2.

Figure 2. Grouping of students' learning capability at every time interval.

3.2. Framework of the DKT-LCIRT Model

The framework of the DKT-LCIRT model is shown in Figure 3. The DKT-LCIRT model contains two memory matrices, key and value, in the reading and writing process. The key memory matrix, which is immutable, stores the potential knowledge points of the exercises; the value memory matrix, which is variable, stores the mastery of the knowledge points. The DKT-LCIRT model contains three main steps: the acquisition of correlation weights, the prediction of the probability of correct answers to the exercises, and the update of the student’s knowledge state.

3.2.1. Acquisition of Correlation Weights

q_t denotes the exercise done by the student at moment t. Suppose that Q exercises contain N knowledge points. The exercise q_t has its own correlation weight vector \( w_t \in \mathbb{R}^N \), which shows how the exercise q_t and each knowledge point are related. Firstly, the embedding vector \( k_t \in \mathbb{R}^{d_k} \) of the exercise q_t is extracted from the exercise Embedding matrix \( A \in \mathbb{R}^{Q \times d_k} \), where \( d_k \) is the embedding size of a key memory slot. Then, the embedding vector \( k_t \) is cascaded with the learning capability grouping \( c_z \), and finally, the cascaded vector \( [k_t, c_z] \) and the key memory matrix \( M^k \in \mathbb{R}^{N \times d_k} \) are inner-produced to obtain the correlation weight \( w_t \) by the activation function \( \text{softmax} \), and the correlation weight \( w_t \) is calculated as shown in Equation (6):

\[
w_t = \text{Softmax} \left( [k_t, c_z]^T M^k(i) \right),
\]

(6)

where \( M^k(i) \) is the i-th row-vector of \( M^k \).
3.2.1. Acquisition of Correlation Weights

denotes the exercise done by the student at moment \( t \). Suppose that \( Q \) exercises contain \( N \) knowledge points. The exercise \( tq \) has its own correlation weight vector \( \mathbf{w}_t \in \mathbb{R}^N \), which shows how the exercise \( tq \) and each knowledge point are related.

Firstly, the embedding vector \( \mathbf{k}_d \) of the exercise \( tq \) is extracted from the exercise Embedding matrix \( \mathbf{Qd} \in \mathbb{R}^{kQd \times k} \), where \( k_d \) is the embedding size of a key memory slot. Then, the embedding vector \( \mathbf{k}_t \) is cascaded with the learning capability grouping \( \mathbf{zc} \), and finally, the cascaded vector \( [\mathbf{t}, \mathbf{kc}] \) and the key memory matrix \( \mathbf{Nd} \in \mathbb{R}^{kN \times kM} \) are inner-produced to obtain the correlation weight \( \mathbf{w}_t \) by the activation function \( \text{softmax} \), and the correlation weight \( \mathbf{w}_t \) is calculated as shown in Equation (6):

\[
    \mathbf{w}_t = \text{softmax} \left( \mathbf{k}_t \left( \mathbf{M} \right) \right),
\]

where \( \mathbf{k}_M \) is the \( i \)-th row vector of \( \mathbf{M} \).

3.2.2. Prediction of the Probability of Correct Answers to the Exercises

The probability of correctly answering the exercise is predicted by the following process. First, a read vector \( \mathbf{r}_t \) that indicates the student’s mastery of the current exercise \( q_t \) is output using the correlation weights \( \mathbf{w}_t \) reading the knowledge states in the value memory matrix, as shown in Equation (7):

\[
    \mathbf{r}_t = \sum_{i=1}^{N} (\mathbf{w}_t(i) \mathbf{M}_v(i)),
\]

Then, the reading vectors \( \mathbf{r}_t \) are cascaded with previous response \( \mathbf{r}_{t-1} \) and exercise difficulty \( \mathbf{pd}_t \). In this paper, the difficulty \( \mathbf{pd}_t \) of the exercises is measured in 10 levels [43] to measure, the difficulty is related to the exercises and is not related to the knowledge points contained in the exercises [44]. The difficulty of the exercises is calculated by Equations (8) and (9):

\[
    \mathbf{pd}_t(j) = \begin{cases} 
    \delta(j, 10), & \text{if } |N_j| \geq 4 \\
    \mathbf{pd}, & \text{else}
    \end{cases},
\]

\[
    \mathbf{pd}(j) = \begin{cases} 
    \delta(j, 10), & \text{if } |N_j| \geq 4 \\
    \mathbf{pd}, & \text{else}
    \end{cases},
\]

**Figure 3.** Framework diagram of the DKT-LCIRT model.
\[
\delta(j, 10) = \sum_{i=1}^{\lvert N_j \rvert} \frac{\lvert \{a_{ij} == 0\} \rvert}{\lvert N_j \rvert} \cdot 10,
\]

in which \( \lvert N_j \rvert \) represents the group of students who answered the exercises \( j \). \( a_{ij} == 0 \) represents the result of students’ first answer to the exercises \( j \) is wrong. The constant \( p_d \) represents the difficulty level of the exercises that we wish to keep. \( \delta(j, 10) \) is a function that maps the error rate of exercise \( j \) onto (10) difficulty levels. \( p_d(j) \) represents the difficulty of exercise \( j \). For those exercises answered by less than four students, \( p_d = 5 \).

The cascade vectors \([v_{t-1}, r_t, pd_t]\) are input to the hidden layer, thus outputting the potential knowledge state \( h_t \) of the students. As shown in Equation (10):

\[
h_t = \tanh(W_{hx}[v_{t-1}, r_t, pd_t] + W_{hh}h_{t-1} + b_h),
\]

where \( W_{hx} \) indicates the input weight matrix. \( W_{hh} \) indicates the recurrent weight matrix. \( b_h \) indicates the bias vector, and the hidden layer is activated using the hyperbolic tangent function.

Next, the student potential knowledge state \( h_t \) and the exercise embedding vector \( k_t \) are input into two single-layer fully connected neural networks, respectively. The students’ capability levels and difficulty levels of the exercises required by the item response theory are output. According to the role of neural networks, these two neural networks are called student capability network and exercise difficulty network, respectively. Both neural networks use the hyperbolic tangent function as their activation function so that their outputs are scaled to \((-1, 1)\). Student capability \( \theta_{ij} \) and difficulty \( \beta_j \) of the exercises are calculated by Equations (11) and (12):

\[
\theta_{ij} = \tanh(W_\theta v_t + b_\theta),
\]

\[
\beta_j = \tanh(W_\rho k_t + b_\rho),
\]

where \( \theta_{ij} \) represents the capability of students to answer the exercises \( j \) at the moment \( t \). \( \beta_j \) represents the difficulty of the exercises \( j \).

Finally, the student capability \( \theta_{ij} \) and the difficulty \( \beta_j \) of the exercise are input to the item response function to predict the probability of correctly answering the exercise \( j \). As shown in Equation (13):

\[
p_i^c = \sigma(3.0 \times \theta_{ij} - \beta_j),
\]

where the output of the student capability network was multiplied by 3.0 [45] so that the value domain was \((0, 1)\). This is because the predicted probability of correctly answering the exercise is \( \sigma(1 - (-1)) = \sigma(2) = 0.881 \) at the maximum if the student’s capability is not improved.

3.2.3. Update of Students’ Knowledge State

After students answer the exercises, the knowledge state is changed, and the value memory matrix is updated depending on the import tuples \((q_t, a_t)\) and the correlation weights \( w_t \). The embedding vector \( v_t \) is extracted from the exercise response Embedding matrix \( B \), and the embedding vector \( v_t \) represents the knowledge growth after answering the exercise \( q_t \). First, some knowledge is removed using the delete vector \( e_t \), which represents the knowledge forgotten by the student. Then, the increase vector \( a_t \) adds the knowledge growth to the value memory matrix. The student’s knowledge state is updated by Equations (14)–(17):

\[
e_t = \sigma(W_e v_t + b_e),
\]

\[
a_t = \tanh(W_a v_t + b_a),
\]

\[
\tilde{M}^{t+1}_{ij} = M^{t}_{ij} \odot (1 - w_i(a_t)_{T}),
\]

\[
M^{t+1}_{ij} = \tilde{M}^{t+1}_{ij} + w_i(a_t)_{T},
\]

For ease of understanding, we illustrate the DKT-LCIRT model in Algorithm 1.
Algorithm 1: The DKT-LCIRT model

**Input:** interaction sequence \( u_i = \{(q_1, a_1), (q_2, a_2), \ldots, (q_t, a_t)\} \) for student \( i \)

**Output:** the probability \( p_{cz}^t \) of answering the exercise correctly

1. Initialize previous response \( v_t \) and exercise difficulty \( p_d_t \)
2. for \( n = 1, 2, \ldots, n \) do
3. the learning capability grouping \( c_z \) is obtained by Equations (1)–(5)
4. extract the embedding vector \( k_t \) of the exercise and cascade it with the learning capability grouping \( c_z \)
5. the relevant weight vector \( w_t \) of the exercise is obtained by Equation (6)
6. the reading vector \( r_t \) is obtained by Equation (7)
7. cascade the read vector \( r_t \) with \( v_{t-1} \) and \( p_d_t \), and then input to the hidden layer
8. the student’s knowledge state \( h_t \) is obtained by Equation (10)
9. the student’s knowledge state \( h_t \) and the embedding vector \( k_t \) of the exercise are input to the student capability network and the exercise difficulty network
10. output student’s capability \( \theta_{ij} \) and difficulty \( \beta_j \) of the exercises
11. the probability \( p_{cz}^t \) of answering the exercise correctly is obtained by Equation (13)
12. students’ knowledge states are updated by Equations (14)–(17)
13. end for
14. return

3.3. Optimization of the Model

The parameters that need to be trained for the DKT-LCIRT model are the exercise Embedding matrix \( A \), the exercise response Embedding matrix \( B \), the key memory matrix \( M^k_i \), and the weights and biases of neural network. To increase the predictive performance of the model, the model is trained by minimizing the loss function [46]. The loss function is shown in Equation (18):

\[
l = \sum_t (R_t \log p_{cz}^t + (1 - R_t) \log(1 - p_{cz}^t)),
\]

where \( p_{cz}^t \) indicates the predicted value, and \( R_t \) indicates the true value.

4. Performance Analysis

4.1. Datasets

In order to verify the effectiveness of the proposed DKT-LCIRT model, experiments on four publicly available online education datasets were conducted, the specific descriptive information of which is detailed in Table 1. The datasets have a very large data volume and a very diverse data distribution, which can meet the requirements of the experiments well. The experimental results are universally representative and applicable and can be well extended to real-life scenarios. The datasets have good ecological validity.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Number of Exercises</th>
<th>Number of Knowledge Points</th>
<th>Number of Students</th>
<th>Number of Interactions</th>
<th>Correct Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSIST2009</td>
<td>26,684</td>
<td>110</td>
<td>4151</td>
<td>325,637</td>
<td>65.84%</td>
</tr>
<tr>
<td>ASSIST2015</td>
<td>NA</td>
<td>100</td>
<td>19,840</td>
<td>683,801</td>
<td>73.18%</td>
</tr>
<tr>
<td>Synthetic</td>
<td>50</td>
<td>5</td>
<td>2000</td>
<td>100,000</td>
<td>58.83%</td>
</tr>
<tr>
<td>Statics2011</td>
<td>1223</td>
<td>156</td>
<td>333</td>
<td>189,297</td>
<td>76.54%</td>
</tr>
</tbody>
</table>

ASSIST2009 [47]: The dataset, from the ASSISTments intelligent tutoring system, contains 325,637 interactions from 4151 students, covering 26,688 exercises and 110 knowledge points, with a 65.84% correct rate. The correct rate measures the percentage of correct answers to the exercises contained in all interactions in the dataset.

ASSIST2015 [48]: The dataset, also from the ASSISTments intelligent tutoring system, contains 683,801 interactions from 19,840 students, covering 100 knowledge points, with a 73.18% correct rate. In comparison to the ASSIST2009 dataset, although this one has a lot
more interactions, the average number of interactions per student is substantially lower because the number of students is also larger.

Synthetic [27]: The dataset simulates 100,000 interactions of 2000 virtual students, each answering the same 50 exercises, which cover five knowledge points, with a 58.83% correct rate. Statics2011 [49]: The dataset, from a university course on engineering mechanics, contains 189,927 interactions from 333 students, covering 1223 exercises and 156 knowledge points, with a 76.54% correct rate.

4.2. Experimental Setup

The model was implemented on a PC running Windows 10 and equipped with an Intel Core i5-5200U CPU using the Python language and TensorFlow framework. In the experiments, the time interval was set to 20 interactions, and the learning capability was grouped into 8. Predictions were performed using hv-block cross-validation for all datasets. The hv-block cross-validation method is consistent for general smooth observations [50]. The loss function was minimized using a mini-batch stochastic gradient descent algorithm to speed up the training. The model was trained using a batch size of 32 and a learning rate of 0.01, and a dropout was used to prevent model over-fitting.

4.3. Evaluation Index

In this paper, the predictive performance of each model is evaluated using the average ACC and average AUC indexes. ACC is the accuracy rate, which indicates the percentage of correct predictions to all predictions. ACC is calculated by Equation (19):

\[
\text{ACC} = \frac{TN + TP}{FN + TN + FP + TP'}
\]  

(19)

in which \(TN\) represents negative samples correctly predicted. \(TP\) represents positive samples correctly predicted. \(FN\) represents negative samples incorrectly predicted. \(FP\) represents positive samples incorrectly predicted. AUC is the area enclosed by the lower coordinate axis and the ROC curve. The AUC is calculated by Equations (20) and (21):

\[
\text{FPR} = \frac{FP}{FP + TN'},
\]  

(20)

\[
\text{TPR} = \frac{TP}{TP + FN'},
\]  

(21)

where \(FPR\) indicates the horizontal coordinate of the ROC curve, and \(TPR\) indicates the vertical coordinate of the ROC curve. The AUC value equal to 0.5 means that the predictive performance of the model is equivalent to a random guess. The predictive performance of the model is positively correlated with the AUC and ACC values.

4.4. Experimental Results and Analysis

4.4.1. Validity of the DKT-LCIRT Model

To demonstrate the effectiveness of the DKT-LCIRT model, the DKT-LCIRT model was compared with the DKT and DKVMN models. The comparative outcome of the average AUC and ACC values of the three models tested on four publicly available datasets is shown in Table 2.

<table>
<thead>
<tr>
<th>Models</th>
<th>DKT [27]</th>
<th>DKVMN [29]</th>
<th>DKT-LCIRT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUC</td>
<td>ACC</td>
<td>AUC</td>
</tr>
<tr>
<td>ASSIST2009</td>
<td>0.823</td>
<td>0.768</td>
<td>0.825</td>
</tr>
<tr>
<td>ASSIST2015</td>
<td>0.725</td>
<td>0.735</td>
<td>0.730</td>
</tr>
<tr>
<td>Synthetic</td>
<td>0.804</td>
<td>0.752</td>
<td>0.799</td>
</tr>
<tr>
<td>Statics2011</td>
<td>0.794</td>
<td>0.751</td>
<td>0.797</td>
</tr>
</tbody>
</table>
On the ASSIST2009 dataset, the AUC of the DKT and DKVMN models are 0.823 and 0.825, respectively, and the AUC of the DKT-LCIRT model is 0.852, which is an improvement of about 3%. The ACC of the DKT and DKVMN models are 0.768 and 0.771, respectively, and the DKT-LCIRT model has an ACC of 0.785, which is an improvement of about 2%. On the ASSIST2015 dataset, the AUC of the DKT and DKVMN models are 0.725 and 0.730, respectively, and the AUC of the DKT-LCIRT model is 0.764, which is an improvement of about 4%. The ACC of the DKT and DKVMN models are 0.735 and 0.736, respectively, and the ACC of the DKT-LCIRT model is 0.749, which is improved by about 1%. On the Synthetic dataset, the AUC of the DKT and DKVMN models are 0.804 and 0.799, respectively, and the AUC of the DKT-LCIRT model is 0.825, which is about a 2% improvement. The ACC of the DKT and DKVMN models are 0.752 and 0.754, respectively, and the ACC of the DKT-LCIRT model is 0.775, which is about a 2% improvement. On the Statics2011 dataset, the AUC of the DKT and DKVMN models are 0.794 and 0.797, respectively, and the AUC of the DKT-LCIRT model is 0.819, which is improved by about 2%. The ACC of the DKT and DKVMN models is 0.751 and 0.754, respectively, and the ACC of the DKT-LCIRT model is 0.773, which is improved by about 2%. The results confirm that the DKT-LCIRT model outperforms the DKT and DKVMN models on all datasets, while the DKT and DKVMN models have about the same predictive performance.

4.4.2. Validity of Learning Capability Features and Item Response Theory

To demonstrate the effectiveness of adding learning capability features and introducing item response theory, the DKT-LCIRT model was compared with the DKT-LC model, which only adds learning capability features, and the DKT-IRT model, which only introduces item response theory. The comparative outcome of the average AUC and ACC values of the three models tested on four publicly available datasets is shown in Table 3.

<table>
<thead>
<tr>
<th>Models</th>
<th>DKT-LC</th>
<th>DKT-IRT</th>
<th>DKT-LCIRT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUC</td>
<td>ACC</td>
<td>AUC</td>
</tr>
<tr>
<td>ASSIST2009</td>
<td>0.850</td>
<td>0.784</td>
<td>0.826</td>
</tr>
<tr>
<td>ASSIST2015</td>
<td>0.765</td>
<td>0.750</td>
<td>0.732</td>
</tr>
<tr>
<td>Synthetic</td>
<td>0.827</td>
<td>0.776</td>
<td>0.798</td>
</tr>
<tr>
<td>Statics2011</td>
<td>0.816</td>
<td>0.771</td>
<td>0.795</td>
</tr>
</tbody>
</table>

The results showed that the DKT-LCIRT model has similar predictive performance and greater interpretability compared to the DKT-LC model only the learning capability features were added. Because both added learning capability features, the DKT-LCIRT model introduced item response theory to make meaningful estimates of student capability levels and exercise difficulty levels. The DKT-LCIRT model had improved prediction performance and the same interpretability compared to the DKT-IRT model only introduced item response theory. Because the DKT-LCIRT model adds learning capability features to track students’ knowledge states, both introduce item response theory.

4.4.3. Avoid Over-Fitting

If the model initially performs well on the training dataset but not so well on the test dataset, this indicates that the model is over-fitting [51]. To demonstrate that the DKT-LCIRT model can avoid over-fitting better, this paper compares the training and validation AUC of the DKT, DKVMN, and DKT-LCIRT models during training on four publicly available datasets. The comparative outcome is shown in Figure 4.

As can be seen from Figure 4, the DKT model shows over-fitting on all datasets, and its training AUC values and validation AUC values gradually appear to be more different. The DKVMN model does not show over-fitting on the ASSIST2009, ASSIST2015, and Synthetic datasets, but on the Statics2011 dataset, its training AUC values and validation AUC values
gradually appear to be more different after 13 epochs. The DKT-LCIRT model performs better in avoiding over-fitting and maintains similar training and validation AUC values on all datasets.

Figure 4. Training and validation AUC values for the three models on datasets (a) ASSIST2009, (b) ASSIST2015, (c) Synthetic, (d) Statics2011.

5. Conclusions and Future Work

This paper proposes a deep knowledge tracking model DKT-LCIRT that integrates learning capability and item response theory. The model dynamically calculates students’ learning capability by each time interval and allocates each student to groups with similar learning capability to track students’ knowledge state. Finally, the model combines item response theory to estimate students’ probability of correctly answering the exercises. This reflects students’ individualization and improves the predictive performance of the model, and also increases the interpretability of the model. Extensive experiments on four publicly available datasets were carried out. The results confirmed that the predictive performance of the DKT-LCIRT model was raised compared to the classical knowledge tracking models DKT and DKVMN. The model parameters were meaningfully interpreted. The over-fitting problem could also be well avoided. This fully proved the practicality and effectiveness of the DKT-LCIRT model.

The exercises in the dataset used in this paper cover fewer knowledge points, and the DKT-LCIRT model can effectively track the students’ real knowledge state, but the effect of knowledge tracking for multi-knowledge point exercises is unknown. In future research, the model can be further improved for application to knowledge tracking where the exercises cover relatively more knowledge points. The model improves robustness and generalization performance through adversarial training and combines with recommendation algorithms to achieve personalized exercise recommendations. This provides adaptive learning support for students. In addition, how can offline learning be grounded in the real world to help achieve personalized learning path recommendations? This is one of the future research directions.
Author Contributions: Conceptualization, G.L. and J.S.; methodology, G.L., J.S. and Y.H.; validation, Y.Z., Y.W. and T.Y.; writing—original draft preparation, J.S. and G.L.; writing—review and editing, G.L., J.S., Y.H., Y.Z., Y.W. and N.X.; supervision, Y.Z. and T.Y.; funding acquisition, G.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (#62041702); Ministry of Education, Humanities, and Social Sciences Project (#20YJA870010); Jiangxi Provincial Social Science Planning Project (#19TQ05); Key project of Education Science planning in Jiangxi Province (#19ZD024); Jiangxi University Humanities and Social Science Planning Project (#TQ20105); Basic Education Research Project of Jiangxi Province (#SZUNDZH2021-1143); and Jiangxi Province Degree and Postgraduate Education and Teaching Reform Research Project (#JXYJG-2020-075).

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>DKT-LCIRT</td>
<td>Deep Knowledge Tracking model integrating Learning Capability and Item Response Theory</td>
</tr>
<tr>
<td>MOOC</td>
<td>Massive Open Online Courses</td>
</tr>
<tr>
<td>ITS</td>
<td>Intelligent Tutoring Systems</td>
</tr>
<tr>
<td>DKVMN</td>
<td>The Dynamic Key-Value Memory Network model</td>
</tr>
<tr>
<td>IRT</td>
<td>Item Response Theory</td>
</tr>
<tr>
<td>BKT</td>
<td>Bayesian Knowledge Tracking</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>DKT</td>
<td>Deep Knowledge Tracking</td>
</tr>
<tr>
<td>LSTM</td>
<td>Long Short-Term Memory</td>
</tr>
<tr>
<td>MANN</td>
<td>Memory Augmented Neural Network</td>
</tr>
<tr>
<td>Bi-LSTM</td>
<td>Bi-directional Long Short-Term Memory</td>
</tr>
<tr>
<td>GNN</td>
<td>Graph Neural Network</td>
</tr>
<tr>
<td>GCN</td>
<td>Graph Convolutional Network</td>
</tr>
<tr>
<td>CNN</td>
<td>Convolutional Neural Network</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variables</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct (s_m) (_{1:z})</td>
<td>the proportion of knowledge points (s_m) answered correctly by student (i) during time intervals 1 to (z)</td>
</tr>
<tr>
<td>Incorrect (s_m) (_{1:z})</td>
<td>the proportion of knowledge points (s_m) answered incorrectly by student (i)</td>
</tr>
<tr>
<td>(R(s_m)_{1:z})</td>
<td>the difference in students’ performance on knowledge points (s_m)</td>
</tr>
<tr>
<td>(d_i^{1:z})</td>
<td>the student (i)’s learning capability vector at time intervals 1 to (z)</td>
</tr>
<tr>
<td>(</td>
<td>N_i</td>
</tr>
<tr>
<td>Cluster ((i,z))</td>
<td>the student (i)’s grouping at time interval (z)</td>
</tr>
<tr>
<td>(\mu_c)</td>
<td>the centroid of group (c)</td>
</tr>
<tr>
<td>(W_{hx})</td>
<td>the input weight matrix</td>
</tr>
<tr>
<td>(W_{hh})</td>
<td>the recurrent weight matrix</td>
</tr>
<tr>
<td>(b_h)</td>
<td>the bias vector</td>
</tr>
<tr>
<td>(\theta_{ij})</td>
<td>the capability of students to answer the exercises (j) at the moment (t)</td>
</tr>
<tr>
<td>(\beta_{ij})</td>
<td>the difficulty of the exercises (j)</td>
</tr>
<tr>
<td>(p_t^c)</td>
<td>the predicted value</td>
</tr>
<tr>
<td>(R_t)</td>
<td>the true value</td>
</tr>
</tbody>
</table>
References


42. Gao, K.; Han, F.; Dong, P.; Xiong, N.; Du, R. Connected vehicle as a mobile sensor for real time queue length at signalized intersections. Sensors 2019, 19, 2059. [CrossRef]


47. Peng, M.; Heffernan, N.; Koedinger, K. Addressing the assessment challenge with an online system that tutors as it assesses. User Model. User Adapt. Interact. 2009, 19, 243–266. [CrossRef]


