Deep Knowledge Tracing Incorporating a Hypernetwork With Independent Student and Item Networks

Emiko Tsutsumi, Yiming Guo, Ryo Kinoshita, and Maomi Ueno, *Member, IEEE*

***Abstract*—Knowledge tracing (KT), the task of tracking the knowledge state of a student over time, has been assessed actively by artificial intelligence researchers. Recent reports have described that Deep-IRT, which combines item response theory (IRT) with a deep learning method, provides superior performance. It can express the abilities of each student and the difficulty of each item such as IRT. Nevertheless, its interpretability is inadequate compared to that of IRT because the ability parameter depends on each item. Deep-IRT implicitly assumes that items with the same skills are equivalent, which does not hold when item difficulties for the same skills differ greatly. For identical skills, items that are not equivalent hinder the interpretation of a student's ability estimate. To overcome those difficulties, this study proposes a novel Deep-IRT that models a student response to an item using two independent networks: 1) a student network and 2) an item network. The proposed Deep-IRT method learns student parameters and item parameters independently to avoid impairing the predictive accuracy. Moreover, we propose a novel hypernetwork architecture for the proposed Deep-IRT to balance both the current and the past data in the latent variable storing student's knowledge states. Results of experiments with six benchmark datasets demonstrate that the proposed method improves the prediction accuracy by about 2.0%, on average. In addition, experiments for the simulation dataset demonstrated that the proposed method provides a stronger correlation with true parameters than the earlier Deep-IRT method does at the $p < 0.5$ significance level.**

***Index Terms*—Deep learning, hypernetwork, knowledge tracing (KT), neural network, item response theory (IRT).**

I. Introduction

IMPORTANT tasks for adaptive learning are intended for accurate prediction of a student's performance and for capturing the student's ability change based on the student's prior learning history data. In the field of artificial intelligence, knowledge tracing (KT) has been researched actively to predict a student's performance (correct or incorrect responses to an unknown item) and to discover concepts that the student has not mastered by tracing a student's evolving knowledge state [1], [2], [3], [4], [5]. These tasks are important to help students learn effectively by presenting optimal problems and a teacher's support.

Recently, various KT methods have been developed using major approaches: probabilistic approaches, deep-learning-based approaches, and attention-mechanism-based approaches. Bayesian KT (BKT) is a well-known probabilistic approach that employs a hidden Markov model to trace a student's evolving knowledge state [1].

BKT estimates whether the student has mastered the skill or not according to the student's past response data. It then predicts the student's responses to unknown items. Researchers have proposed several BKT variants to improve interpretability [6], [7], [8], [9]. The BKT models predict a student's knowledge state using only simple discrete values. Therefore, they are inflexible with the student knowledge state changes. Moreover, they assume a single dimension of the ability. They are unable to capture the multidimensional ability sufficiently or predict performance precisely.

Recently, item response theory (IRT) has been used for KT to predict a student's correct answer probability to an unknown item [10], [11], [12]. In fact, IRT has been used in the field of test theory, where it has high parameter interpretability by virtue of its capability of estimating the student's latent ability parameter and item characteristic parameters.

Several studies have extended standard IRT models to ascertain student ability changes for learning processes with the hidden Markov process [11], [12], [13], [14], [15]. These are regarded as generalized models of BKT and IRT because they estimate the ability as a continuous hidden variable following a hidden Markov process. Actually, a learning task is associated with multiple skills. Students must master the knowledge of multiple skills to solve a task. However, BKT and IRT have a restriction: They express only unidimensional ability. Therefore, BKT and IRT are unable to capture the multidimensional ability sufficiently. They are unable to predict the performance precisely.

To overcome this shortcoming, Piech et al. [2] developed deep KT (DKT) as the first method among deep-learning-based approaches.

Manuscript received 19 October 2022; revised 13 November 2023; accepted 18 December 2023. Date of publication 25 December 2023; date of current version 19 January 2024. This work was supported in part by the JSPS KAKENHI under Grant JP19H05663, Grant JP22K19825, and Grant JP22KJ1368, and in part by the JST CREST under Grant JPMJCR21D1. An earlier version of this paper was presented in part at the International Conference on Educational Data Mining (EDM) in 2021 and 2022 [https://educationaldatamining.org/edm2021/proceedings/], [DOI: 10.5281/zenodo.6853107]. *(Corresponding author: Emiko Tsutsumi.)*

Emiko Tsutsumi is with the Graduate School of Information Science and Technology, University of Tokyo Tokyo 113-8656, Japan (e-mail: tsutsumi@mi.u-tokyo.ac.jp).

Yiming Guo, Ryo Kinoshita, and Maomi Ueno are with the Graduate School of Informatics and Engineering, The University of Electro-Communications, Chofu 182-8585, Japan (e-mail: guo_yzmzng@ai.lab.uec.ac.jp; kinoshita@ai.lab.uec.ac.jp; ueno@ai.is.uec.ac.jp).

Digital Object Identifier 10.1109/TLT.2023.3346671

DKT employs long short-term memory (LSTM) to relax the restrictions of skill separation and binary state assumptions [16]. That earlier report describes that DKT can predict a student's performance more precisely than probabilistic models such as BKT can. However, the hidden states include a summary of the past sequence of learning history data in LSTM. Therefore, DKT does not explicitly treat the student's ability of each skill.

To improve DKT performance, a dynamic key-value memory network (DKVMN) was developed to exploit the relation between underlying skills and to trace the respective knowledge states [4]. By employing a memory-augmented neural network, DKVMN can estimate the relations between underlying skills and items addressed by students. In addition, DKVMN has a memory-updating component to allow forgetting and updating of the latent variable memory, which stores the students' knowledge states during the learning process [4]. Furthermore, Deep-IRT has been proposed to improve the explanatory capabilities of the parameters [3]. Deep-IRT can estimate a student's ability and an item's difficulty, just as standard IRT models can by combining DKVMN with an IRT module. However, it has remained insufficient to improve interpretability because the student's ability of Deep-IRT depends on each item characteristic. Although DeepIRT implicitly functions on the assumption that items with the same skills are equivalent, that assumption does not hold true when the item difficulties for the same skills differ greatly. Items of the same skill that are not equivalent interfere with the interpretation of the student's ability estimates.

The self-attentive KT (SAKT) method is the first method to employ an attention mechanism, the transformer method, for KT [18], [19]. To predict student performance, SAKT identifies the relation between skills and an item addressed by a student from past learning data. Most recently, attentive KT (AKT) was developed to improve SAKT performance [5]. To incorporate a forgetting function of past data, AKT employs attention mechanisms. It optimizes parameters to weight past learning data needed to predict student performance. In addition, Ghosh et al. [5] pointed out the error of the assumption in earlier KT methods that items with identical skills are equivalent. To overcome that shortcoming, they employed both items and skills as inputs. In fact, AKT provides state-of-the-art performance for student response prediction. However, the interpretability of the parameters remains inadequate because AKT cannot express a student's ability transition for each skill.

The most challenging aspect of KT is to estimate the interpretable student's ability without decreasing predictive accuracy. This study specifically addresses this point of difficulty. Recent studies of deep learning have clarified that parameter redundancy in training data reduces generalization error, contrary to Occam's razor [20], [21], [22]. Based on those reports, this study proposes a novel Deep-IRT that has two independent redundant networks: 1) a student network and 2) an item network [23]. The proposed method learns the student's ability parameters and the item's characteristic parameters independently. This method provides the high interpretable ability parameters to a greater extent than the earlier Deep-IRT does.

In addition, a student network employs memory network architecture to reflect dynamic changes in student abilities as DKVMN does. The memory updating component in DKVMN is more effective than the forgetting function of AKT because it updates the current latent variable, which stores the students' skills and abilities using only the immediately preceding values.

However, room for improvement remains in the prediction accuracy of the proposed Deep-IRT. In fact, the forgetting parameters that control the degree of forgetting the past latent variable are optimized from only the current input data: the student's latest response to an item. It might degrade the prediction accuracy of the Deep-IRT because the latent variable only insufficiently reflects the past data. As a result, it might interrupt the accurate estimation of the ability transition in a long learning process. It should use not only the current input data but also past latent variables to optimize the forgetting parameters.

A simple solution to this problem is to add new weight parameters that balance the current input data and past latent variables at each time. However, this solution increases the number of weight parameters dynamically when the learning process progresses. It often yields too many weight parameters to support a successful estimate.

To resolve that difficulty, we combine a novel hypernetwork with the proposed method because it optimizes the degree of forgetting of the past latent variables and thereby avoids greatly increasing the number of parameters.

Recent studies in the field of natural language processing (NLP) have proposed several hypernetworks to optimize the latent variables and the weights of the hidden layers for LSTM [24], [25]. Some hypernetworks scale the latent variables and columns of all weight matrices expressing a context-dependent transition [24], [26]. No report of the relevant literature has described a study of the use of hypernetworks for KT methods. Using the proposed method, the proposed hypernetwork balances both current input data and past latent variables that store a student's knowledge state in the learning process. Before the model updates the latent variable, it optimizes not only the weights of the forgetting parameters but also the past latent variables in the hypernetwork.

We conducted experiments to compare the proposed method's performance and those of earlier KT methods. Surprisingly, the results demonstrate that the proposed method improves the prediction accuracy and the interpretability of earlier KT methods, although the parameters of the proposed method are far more numerous than those used for earlier methods.

This study is an extension of our work reported in earlier papers accepted at the International Conference on Educational Data Mining in 2021 and 2022 [23], [27]. The main differences between this article and the earlier papers are the following. Tsutsumi et al. [23] did not propose a new deep-learning technology but combined only existing technologies. Although Tsutsumi et al. [27] proposed a hypernetwork for KT, they described no related details: only the conceptual idea of incorporating a hypernetwork into Deep-IRT [23]. Furthermore, the authors in [23] and [27] improved the parameter interpretability. However, their prediction accuracies did not outperform AKT, which provided the best prediction performance among the earlier methods. In contrast, this study proposes a novel hypernetwork architecture

to optimize the balance between the latest input data and the past latent variables. The proposed method provides the highest prediction accuracy and outperforms AKT with high parameter interpretability to a considerable degree.

The main contributions of the work described in this article are presented as follows.

1) The proposed method can estimate student and item parameters with high interpretability as in IRT by two independent redundant networks. The proposed method provides higher parameter interpretability than other KT methods.
2) The proposed method with hypernetworks improves the prediction accuracy of earlier KT methods. Especially, it functions more effectively for long learning processes because hypernetworks reflect past learning data.

The rest of this article is organized as follows. In Section II, we review IRT and deep learning methods for KT. In Section III, we describe the proposed method to improve parameter interpretability. In Section IV, we describe the proposed method with a hypernetwork to improve the prediction accuracy. Section V shows experiments using benchmark datasets to compare the performances of the proposed methods against existing methods. Section VI explains experiments that were performed to evaluate the interpretability of the ability parameters of the proposed method. Finally, Section VII concludes this article.

Our code is also available on GitHub.[1]

II. Related Work

A. Item Response Theory

Many IRT models exist [10], [28], [29]. This section briefly introduces the two-parameter logistic model (2PLM): an extremely popular IRT model. For 2PLM, u_{ij} represents the response of student i to item j $(1\ldots, J)$ as

$$u_{ij} = \begin{cases} 1, & (\text{student } i \text{ answers correctly to item } j) \\ 0, & (\text{otherwise}). \end{cases}$$

In 2PLM, the probability of a correct answer given to item j by student i with ability parameter $\theta_i \in (-\infty, \infty)$ is assumed as

$$\begin{aligned} P_j(\theta_i) &= P(u_{ij} = 1 \mid \theta_i) \\ &= \frac{1}{1 + exp(-1.7a_j(\theta_i - b_j))} \end{aligned} \tag{1}$$

where $a_j \in (0, \infty)$ represents the jth item's discrimination parameter expressing the discriminatory power for student's abilities, and $b_j \in (-\infty, \infty)$ is the jth item's difficulty parameter representing the degree of difficulty.

Actually, IRT models are known to have high interpretability. However, in standard IRT models, the ability is assumed to be constant throughout the learning process. Therefore, student's ability changes are not reflected in the models. Recently, several studies have extended standard IRT models to capture student's ability changes for the learning processes with the hidden Markov process [11], [12], [13], [14], [30], [31], [32]. These are regarded as generalized models of BKT and IRT because they estimate the ability as a continuous hidden variable following a hidden Markov process.

For example, temporal IRT (TIRT) is a hidden Markov IRT with a parameter to forget past response data [12]. In TIRT, the probability of a correct answer assigned to item j by student i at time t with ability parameter θ_{it} is assumed as

$$P_{ij}(x_{ij} = 1 \mid \theta_{it}) = \frac{1}{1 + \exp\left(-\tilde{a}_{\Delta_t}(\theta_{it} - b_j)\right)} \tag{2}$$

$$\tilde{a}_{\Delta_t} = \frac{a_j}{\sqrt{1 + \epsilon a_j^2 \Delta_t}} \tag{3}$$

where $\Delta_t = t - j$ and $\tilde{a}_{\Delta_t} \in (0, \infty)$ is the discrimination parameter at time t. In addition, $b_j \in (-\infty, \infty)$ is the jth item's difficulty parameter representing the degree of difficulty. Furthermore, $\theta_{it} \in (-\infty, \infty)$ represents the student i ability at time t. The prior of θ_{it} is a normal distribution described as $\theta_{i0} \sim \mathcal{N}(0, 1)\, \theta_{it} \sim \mathcal{N}(\theta_{it-1}, \epsilon)$. Moreover, ϵ is a variance of θ_{it} and a forgetting parameter (tuning parameter), which determines the forgetting degree of the past data. The smoothness of a student's ability transition can be controlled by ϵ. Therefore, as ϵ increases, the fluctuation range of the true ability increases at each time point.

However, these IRT models incorporate the assumption of a single dimension of the ability. In other words, they completely consider independent multiple skills. Apparently, these are unable to accommodate items that require different skills.

B. Deep KT

DKT [2] was proposed as the first deep-learning-based method. It exploits recurrent neural networks and LSTM [16] to simulate transitions of ability. It can capture complex multidimensional features of both items and students and can relax the limitations of traditional methods such as independence between skills. An earlier study demonstrated that DKT outperformed BKT in terms of predictive accuracy [2]. However, DKT summarizes a student's ability of all skills in one hidden state, which makes it difficult to trace the degree to which a student has mastered a certain skill and pinpoint concepts with which a student is proficient or unfamiliar.

C. Dynamic Key-Value Memory Network

To improve the DKT interpretability, researchers have undertaken great efforts to propose novel methods for use with KT [17]. Specifically, a DKVMN exploits a memory-augmented neural network along with attention mechanisms to trace student abilities in different dimensions [4]. Fig. 1 presents a simple illustration.

The salient feature of DKVMN is that it assumes N underlying skills and relations among the input (skills). Underlying skills are stored in key memory $\boldsymbol{M}^k \in \mathbb{R}^{N \times d_k}$. Value memory $\boldsymbol{M}_t^v \in \mathbb{R}^{N \times d_v}$ holds abilities of underlying skills at time t. Here, d_k and d_v are tuning parameters. To express the skill of jth item, the input of DKVMN is an embedding vector $\boldsymbol{s}_j \in \mathbb{R}^{d_k}$ of skill

[1][Online]. Available: https://github.com/UEC-Ueno-lab/Deep-IRT_with_Hypernetwork.git

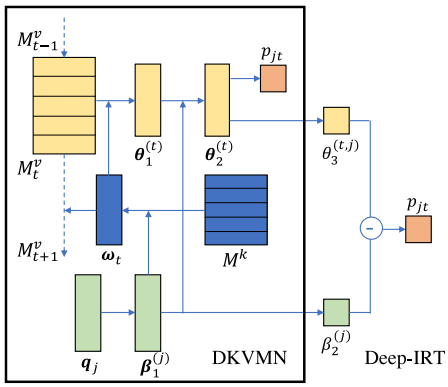


Fig. 1. Network architecture of Deep-IRT with DKVMN. The underside of the structure describes DKVMN. The whole structure describes Deep-IRT. The blue components represent the process of getting the attention weight. The yellow components are associated with the student network and the process of updating the value memory. The green components are associated with the item network. The designation $\ominus$ represents subtraction.

tag of item j. DKVMN predicts the performance of item j at time t as explained ahead.

First, DKVMN calculates the attention, which indicates how strongly an item j is related to each skill as

$$w_{jl} = \text{Softmax}\left(\boldsymbol{M}_l^k \boldsymbol{s}_j\right) \tag{4}$$

where M_l^k represents an lth row vector, and w_{jl} signifies the degree of strength of the relation between the latent skill l and the skill of item j addressed by a student at time t. Next, student vector $\boldsymbol{\theta}_1^{(t)}$ is calculated using the weighted sum of value memory

$$\boldsymbol{\theta}_1^{(t)} = \sum_{l=1}^{N} w_{jl} \left(\boldsymbol{M}_{tl}^v\right)^\top \tag{5}$$

where M_{tl}^v represents an lth row vector. Finally, it concatenates $\boldsymbol{\theta}_1^{(t)}$ with $\boldsymbol{s}_j$ and predicts a correct probability P_{jt} for an item j as

$$\boldsymbol{\theta}_2^{(t)} = \tanh\left(\boldsymbol{W}^{(\theta_2)}\left[\boldsymbol{\theta}_1^{(t)}, \boldsymbol{s}_j\right] + \boldsymbol{\tau}^{(\theta_2)}\right) \tag{6}$$

$$P_{jt} = \sigma\left(\boldsymbol{W}^{(P_{jt})}\boldsymbol{\theta}_2^{(t)} + \boldsymbol{\tau}^{(P_{jt})}\right) \tag{7}$$

where $[\cdot]$ denotes concatenation of vectors, and $\sigma(\cdot)$ represents the sigmoid function. In this report, we express $\boldsymbol{W}^{(\cdot)}$ as the weight matrix and weight vector, and $\boldsymbol{\tau}^{(\cdot)}$ as the bias vector and scalar. Reportedly, DKVMN has the capability of predicting performance accurately. However, unfortunately, it lacks interpretability of the parameters.

D. Deep-IRT

To improve the DKVMN interpretability, Deep-IRT is implemented by combining DKVMN with an IRT module [3]. Deep-IRT exploits both the strong prediction ability of DKVMN and the interpretable parameters of IRT. Fig. 1 presents a simple illustration.

Deep-IRT adds a hidden layer to DKVMN to gain applicable ability and item difficulty. Specifically, when a student attempts item j at time t, an ability $\theta_3^{(t,j)}$ and item difficulty β^j are calculated as described in the following:

$$\theta_3^{(t,j)} = \tanh\left(\boldsymbol{W}^{(\theta_3)}\boldsymbol{\theta}_2^{(t)} + \boldsymbol{\tau}^{(\theta_3)}\right) \tag{8}$$

$$\beta^j = \tanh\left(\boldsymbol{W}^{(\beta)}\boldsymbol{s}_j + \boldsymbol{\tau}^{(\beta)}\right). \tag{9}$$

The prediction is based on the difference between $\theta_3^{(t,j)}$ and β^j such as IRT

$$P_{jt} = \sigma\left(3.0 * \theta_3^{(t,j)} - \beta^j\right). \tag{10}$$

Here, ability $\boldsymbol{\theta}_2^{(t)}$ is calculated using $\boldsymbol{s}_j$ in (6), which depends on the item to solve because it implicitly assumes that items with the same skills are equivalent. In other words, the ability estimate for the same student and time might differ if the student attempts a different item. An important difficulty is that a student's ability, which depends on each item, hinders the interpretability of the parameters.

E. Attentive KT

Ghosh et al. [5] proposed AKT, which combines the attention-based model with the Rasch model, which is also known as the 1PLM IRT model [33]. It is noteworthy that AKT incorporates a forgetting function for past data into attention-based neural networks. Attention weights in AKT express the relation between student's latest data and past data, decaying exponentially during the learning process. Specifically, AKT calculates the attention weight $\boldsymbol{\alpha}$ as

$$\alpha_{t,\lambda} = \frac{\exp\left(f_{t,\lambda}\right)}{\sum_{\lambda'}\exp\left(f_{t,\lambda}\right)} \tag{11}$$

$$\left(f_{t,\lambda}\right) = \frac{\exp\left(-\eta d(t,\lambda)\right)\cdot \boldsymbol{q_t}^\top \boldsymbol{k}_\lambda}{\sqrt{D_k}} \tag{12}$$

where $\eta > 0$ is a decay rate parameter and $d(t,\lambda)$ is a temporal distance measure between time steps t and λ. In addition, $\boldsymbol{q_t} \in \mathbb{R}^{D_k}$ denotes the query corresponding to items to which the student responds at time 1 to t, $\boldsymbol{k_\lambda} \in \mathbb{R}^{D_k}$ denotes the key for the item at time step λ, and D_k denotes the dimensions of the key matrix [5]. The attention weight $\boldsymbol{\alpha}$ decays as the distance between the current input time and the past input time increases. Furthermore, $d(t,\lambda)$ with $\lambda \le t$ is obtained as explained in the following:

$$d(t,\lambda) = |t-\lambda| \sum_{t'=\lambda+1}^{t} \frac{\frac{\boldsymbol{q_t}^\top \boldsymbol{k}_{t'}}{\sqrt{D_k}}}{\sum_{1\le\lambda'\le t'} \frac{\boldsymbol{q_t}^\top \boldsymbol{k}_{\lambda'}}{\sqrt{D_k}}} \quad \forall t' \le t. \tag{13}$$

In fact, $d(t,\lambda)$ adjusts the distance between consecutive time indices according to how the past input is related to the current input [5].

In addition, they pointed out that the earlier KT methods assumed that items with the same skills were equivalent. To resolve the difficulty, AKT employs both items and skill inputs. Results show that, among the earlier KT methods, AKT provides

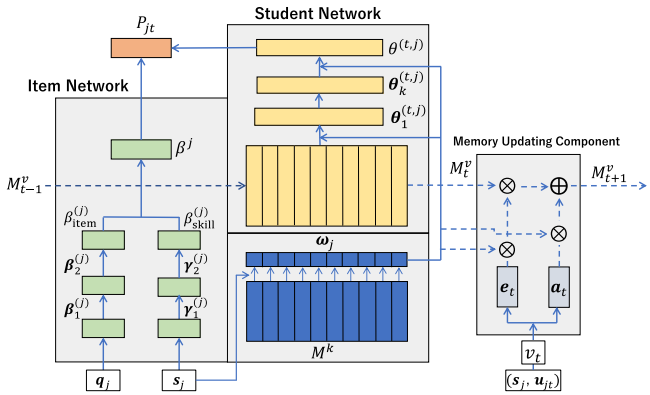


Fig. 2. Network architecture of Deep-IRT with independent student and item networks. The yellow components are associated with the student network. The green components are associated with the item network. In addition, the right side of the figure presents the memory updating component. The designations $\otimes$ and $\oplus$, respectively, represent elementwise multiplication and addition.

the best performance for predicting the students' responses. Nevertheless, the interpretability of its parameters remains inadequate because it cannot express a student's ability transition for each skill.

III. Deep-IRT With Independent Student and Item Networks

The ability parameter of the Deep-IRT [3] depends on each item because it implicitly assumes that items with the same skills are equivalent. That assumption does not hold when the item difficulties for the same skills differ greatly. Therefore, when the items for the same skills are not equivalent, it is difficult to interpret a student's ability estimate.

To resolve the difficulty, this study proposes a novel Deep-IRT method comprising two independent neural networks: 1) the student network and 2) the item deep network [23], as presented in Fig. 2. The student network employs memory network architecture, such as DKVMN to ascertain changes in student ability comprehensively. The item network includes inputs of two kinds: 1) the item attempted by a student and 2) the necessary skills to solve the item. Using the outputs of both networks, the probability of a student answering an item correctly can be calculated.

The proposed method can estimate student parameters and item parameters independently such that the prediction accuracy does not decline because the two independent networks are designed to be more redundant than they are with earlier methods, based on state-of-the-art reports [20], [21], [22]. The proposed method predicts P_{jt}, the probability of a correct answer assigned to the item j at time t, using the item difficulties and the student abilities [23], as shown hereinafter.

A. Item Network

In the item network, two difficulty parameters of item j are estimated: 1) the item characteristic difficulty parameter β^j_{item} and 2) the skill difficulty β^j_{skill} [23]. The item characteristic difficulty parameter represents the unique difficulties of the item, except the required skill difficulty. The proposed method expresses item difficulty as the sum of the two difficulty parameters of β^j_{item} and β^j_{skill}.

In the proposed method, to express the jth item, an input of the item network is an embedding vector $\boldsymbol{q}_j \in \mathbb{R}^{d_k}$ of item j. The item characteristic difficulty parameter of item j is calculated using a feed-forward neural network as

$$\boldsymbol{\beta}^j_1 = \tanh\left(\boldsymbol{W}^{(\beta_1)}\boldsymbol{q}_j + \boldsymbol{\tau}^{(\beta_1)}\right) \tag{14}$$

$$\boldsymbol{\beta}^j_{k'} = \tanh\left(\boldsymbol{W}^{(\beta_{k'})}\boldsymbol{\beta}^j_{k'-1} + \boldsymbol{\tau}^{(\beta_{k'})}\right) \tag{15}$$

$$\beta^j_{\text{item}} = \boldsymbol{W}^{(\beta_{\text{item}})}\boldsymbol{\beta}^j_k + \boldsymbol{\tau}^{(\beta_{\text{item}})}. \tag{16}$$

In this report, we represent $\{k \in \mathbb{N} | 2 \leq k' \leq k\}$ as numerous hidden layers decided depending on the prediction accuracy of actual data. The last layer β^j_{item} represents the jth item characteristic difficulty parameter.

Similarly, to compute the difficulty of skills, the proposed method uses the input of necessary skills $\boldsymbol{s}_j \in \mathbb{R}^{d_k}$. The embedding vector $\boldsymbol{s}_j$ is calculated from the skill tag of item j

$$\boldsymbol{\gamma}^j_1 = \tanh\left(\boldsymbol{W}^{(\gamma_1)}\boldsymbol{s}_j + \boldsymbol{\tau}^{(\gamma_1)}\right) \tag{17}$$

$$\boldsymbol{\gamma}^j_{k'} = \tanh\left(\boldsymbol{W}^{(\gamma_{k'})}\boldsymbol{\gamma}^j_{k'-1} + \boldsymbol{\tau}^{(\gamma_{k'})}\right) \tag{18}$$

$$\beta^j_{\text{skill}} = \boldsymbol{W}^{(\beta_{\text{skill}})}\boldsymbol{\gamma}^j_k + \boldsymbol{\tau}^{(\beta_{\text{skill}})} \tag{19}$$

where $\{k \in \mathbb{N} | 2 \leq k' \leq k\}$. The last layer β^j_{skill} denotes the difficulty parameter of the required skills to solve the jth item.

B. Student Network

In the student network, the proposed method calculates $\boldsymbol{\theta}^{(t,j)}_1$ based on the latent variable $\boldsymbol{M}^v_t$ expressing a student's latent knowledge state at time t [23], as

$$\boldsymbol{\theta}^{(t,j)}_1 = \sum_{l=1}^{N} w_{jl}\left(\boldsymbol{M}^v_{tl}\right)^\top \tag{20}$$

where M^v_{tl} represents an lth row vector and where w_{jl} is the attention weight of underlying skill l. w_{jl} is estimated similarly to DKVMN in (4). Next, an interpretable student's ability vector $\theta^{(t,j)}$ can be estimated as presented in the following:

$$\boldsymbol{\theta}^{(t,j)}_{k'} = \tanh\left(\boldsymbol{W}^{(\theta_{k'})}\boldsymbol{\theta}^{(t,j)}_{k'-1} + \boldsymbol{\tau}^{(\theta_{k'})}\right) \tag{21}$$

$$\theta^{(t,j)} = \sum_{l=1}^{N} w_{jl}\theta^{(t,j)}_{k'l} \tag{22}$$

where $\{k \in \mathbb{N} | 2 \leq k' \leq k\}$ and $\boldsymbol{\theta}^{(t,j)}_k = \{\theta^{(t,j)}_{k1}, \theta^{(t,j)}_{k2}, \ldots, \theta^{(t,j)}_{kN}\}$. Also, $\boldsymbol{\theta}^{(t,j)}_{k'} \in \mathbb{R}^{d_v}$ and $\boldsymbol{\theta}^{(t,j)}_k \in \mathbb{R}^N$. One important difference between the proposed method and the earlier Deep-IRT [3] is that the proposed method does not calculate $\boldsymbol{\theta}^{(t,j)}_k$ using features of items, such as (6) and (8). Therefore, the ability parameter $\theta^{(t,j)}$ is independent of the difficulty parameters of the respective items. In addition, the value of $\boldsymbol{\theta}^{(t,j)}_k$ represents the

abilities of the latent skills. In other words, $\boldsymbol{\theta}_k^{(t,j)}$ can be inferred as a measurement model, such as multidimensional IRT [34].

C. Prediction of Student Response to an Item

The proposed method predicts a student's response probability to an item using the difference between a student's ability $\theta^{(t,j)}$ to solve item j at time t and the sum of two difficulty parameters β^j_{item} and β^j_{skill} [23]

$$P_{jt} = \sigma\left(3.0 * \theta^{(t,j)} - (\beta^j_{\text{item}} + \beta^j_{\text{skill}})\right). \tag{23}$$

After the procedure, the latent value memory $\boldsymbol{M}_t^v$ is updated using the embedding vector of $(s_j, u_{jt}) = s_j + u_{jt} * S$ denoted as $\boldsymbol{v}_t \in \mathbb{R}^{d_v}$ such as DKVMN [4]. Actually, u_{jt} is the student's response to item j at time t: u_{jt} is 1 when the student answers the item correctly; it is 0 otherwise

$$\boldsymbol{e}_t = \sigma(\boldsymbol{W}^e \boldsymbol{v}_t + \boldsymbol{\tau}^e) \tag{24}$$

$$\boldsymbol{a}_t = \tanh(\boldsymbol{W}^a \boldsymbol{v}_t + \boldsymbol{\tau}^a) \tag{25}$$

$$\tilde{\boldsymbol{M}}^v_{t+1,l} = \boldsymbol{M}^v_{t,l} \otimes (1 - w_{jl}\boldsymbol{e}_t)^\top \tag{26}$$

$$\boldsymbol{M}^v_{t+1,l} = \tilde{\boldsymbol{M}}^v_{t+1,l} + w_{jl}\boldsymbol{a}_t^\top. \tag{27}$$

Therein, $\boldsymbol{W}^e \in \mathbb{R}^{d_v \times d_v}, \boldsymbol{W}^a \in \mathbb{R}^{d_v \times d_v}$ are weight matrices and $\boldsymbol{\tau}^e \in \mathbb{R}^{d_v}, \boldsymbol{\tau}^e \in \mathbb{R}^{d_v}$ are bias vectors. l is the underlying skill and $\{l \in \mathbb{N} | 1 \leq l \leq N\}$. $\otimes$ represents the elementwise product. In (24) and (26), $\boldsymbol{e}_t$ controls how much the value memory forgets (remembers) the past ability. In addition, $\boldsymbol{a}_t$ in (25) and (27) controls how strongly current performance is reflected. It is noteworthy that $\boldsymbol{e}_t$ and $\boldsymbol{a}_t$, which control the degree of forgetting the past latent value memory $\boldsymbol{M}_t^v$, are optimized solely from the student's latest response to an item u_{jt}.

In general, deep-learning-based methods learn their parameters using the back-propagation algorithm by minimizing a loss function. The loss function of the proposed method employs cross-entropy, which reflects classification errors. Then, the cross-entropy of the predicted responses P_{jt} and the true responses u_{jt} is calculated as

$$\ell(u_{jt}, P_{jt}) = -\sum_t \left(u_{jt} \log P_{jt} + (1 - u_{jt}) \log(1 - P_{jt})\right). \tag{28}$$

All parameters are learned simultaneously using a well-known optimization algorithm: adaptive moment estimation [35].

IV. Deep-IRT With Hypernetwork

The preceding section described the proposed Deep-IRT method with independent student and item networks [23]. However, room for improvement of the prediction accuracy remains because the parameters that control the degree of forgetting the past latent value memory $\boldsymbol{M}_t^v$ are optimized using only the student's latest response to an item. It might degrade the prediction accuracy of the Deep-IRT because the latent value memory insufficiently reflects past data. As a result, it might prevent difficulty in accurate estimation of the ability transition in a long learning process. It should use not only the current input data but also past data to optimize the forgetting parameters.

One simple solution is to add new weight parameters that balance current input data $\boldsymbol{v}_t$ and past latent values $\boldsymbol{M}_t^v$ at each time. However, the number of weight parameters increases dynamically when the learning process progresses. It often yields too many weight parameters for successful estimation.

Recent reports of studies conducted in the field of NLP have proposed extension components to LSTM [16] in the form of mutual gating of the current input data and earlier hidden variables [24]. These extension components are called hypernetworks. In standard LSTM [16], the hidden variables change with time, but the weights used to update them are fixed values that are not optimized for each time point. To resolve this difficulty, various hypernetworks have been proposed to support the main recurrent neural network by optimizing the nonshared weights for each time point in the hidden layers [24], [26], [36], [37], [38], [39], [40]. Their results demonstrate that LSTM with a hypernetwork works better than the standard LSTM [16]. Furthermore, Melis et al. [26] earlier proposed the "Mogrifier component," which is a kind of hypernetwork for LSTM in the field of NLP. Mogrifier scales the hidden variables using not only the current inputs but also the output of the hidden variable at the earlier time point. They reported that LSTM with the Mogrifier component outperforms other methods for long input data lengths.

Inspired by the results obtained from those studies, we incorporate a novel hypernetwork into the memory updating component (in Fig. 2), which updates the latent variable $\boldsymbol{M}_t^v$ expressing a student's knowledge state, to avoid greatly increasing the number of parameters. Although Tsutsumi et al. [27] proposed a hypernetwork for KT, that report presented no details but just its conceptual idea. This article proposes a novel hypernetwork architecture to optimize the balance between the latest input data and the past latent variables. No report of the relevant literature has described a study of the use of the hypernetworks for KT methods.

Fig. 3 presents the proposed hypernetwork architecture and the memory updating component of the proposed method. The hypernetwork optimizes the degree of forgetting of past data in the proposed Deep-IRT and improves prediction accuracy with parameter interpretability. Specifically, before the method updates the latent variable $\boldsymbol{M}_{t+1}^v$, the proposed hypernetwork balances both the current input data $\boldsymbol{v}_t$ and the latent variable $\boldsymbol{M}_t^v$ using the past latent variables $\{\boldsymbol{M}_t^v, \boldsymbol{M}_{t-1}^v, \ldots, \boldsymbol{M}_{t-\lambda}^v\}$ at time $t - \lambda$ to t. Here, λ represents the degree of the past latent variables to be accessed. For the proposed method, we optimize λ for each learning dataset.

A. Hypernetwork

In the memory updating components of DKVMN and Deep-IRT [3], [4], the forgetting parameters are optimized only from current input data. Therefore, their value memory $\boldsymbol{M}_t^v$ might not adequately forget past data. Therefore, to optimize the forgetting parameters $\boldsymbol{e}_t$, and $\boldsymbol{a}_t$ at time t, the proposed hypernetwork balances the current input data and the past latent value memory

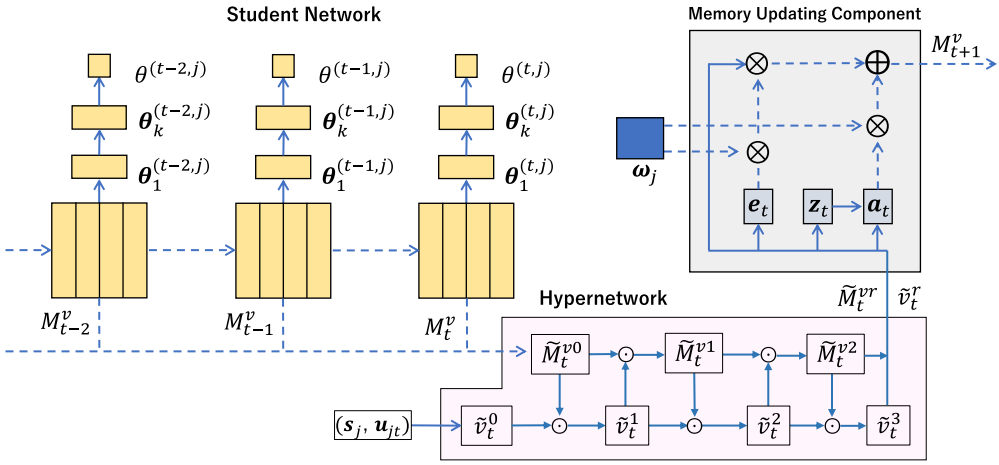


Fig. 3. Memory updating component of the proposed Deep-IRT with hypernetwork. The proposed hypernetwork is located at the beginning of the memory updating component. It estimates the optimal forgetting parameters by balancing both the current input data and the past latent variable before the model updates the latent variable.

to store sufficient information of the learning history data before calculating the latent variables $\boldsymbol{M}_{t+1}^{v}$.

The proposed hypernetwork structure is located at the beginning of the memory updating component (see Fig. 3). The inputs of the hypernetwork are the embedding vector $\boldsymbol{v}_t \in \mathbb{R}^{d_v}$ and the past value memory $\tilde{\boldsymbol{M}}_t^{v}$. The embedding vector $\boldsymbol{v}_t$ is calculated from the current input data (s_j, u_{jt}) when a student responds to item j. In addition, $\tilde{\boldsymbol{M}}_t^{v}$ is calculated as

$$\tilde{\boldsymbol{M}}_t^{v} = \begin{cases} \boldsymbol{M}_t^{v} & (\lambda = 0) \\ \sigma(\boldsymbol{W}[\boldsymbol{M}_t^{v}, \boldsymbol{M}_{t-1}^{v}, \ldots, \boldsymbol{M}_{t-\lambda}^{v}] + \boldsymbol{\tau}) & (\text{otherwise}). \end{cases} \tag{29}$$

Therein, $\boldsymbol{W}$ is the weight vector and $\boldsymbol{\tau}$ is the bias parameter vector. Next, $\boldsymbol{v}_t$ and $\tilde{\boldsymbol{M}}_t^{v}$ are optimized in the hypernetwork as

$$\tilde{\boldsymbol{v}}_t^{r'} = \delta_1 * \sigma(\boldsymbol{W}^{v} \tilde{\boldsymbol{M}}_t^{vr'-1}) \odot \boldsymbol{v}_t^{r'-1} \tag{30}$$

$$\tilde{\boldsymbol{M}}_t^{vr'} = \delta_2 * \sigma(\boldsymbol{W}^{M} \tilde{\boldsymbol{v}}_t^{r'}) \odot \tilde{\boldsymbol{M}}_t^{vr'-1} \tag{31}$$

where $\delta_1 \in \mathbb{R}$, $\delta_2 \in \mathbb{R}$, r is a hyperparameter, and $1 \leq r' \leq r$. r represents the number of rounds in the recurrent architecture. If $r' = 1$, then $\tilde{\boldsymbol{v}}_t^{0} = \boldsymbol{v}_t$ and $\tilde{\boldsymbol{M}}_t^{v0} = \tilde{\boldsymbol{M}}_t^{v}$. Because of the repeated multiplications in (30) and (31), this hypernetwork balances current data $\tilde{\boldsymbol{v}}_t$ and past value memory $\tilde{\boldsymbol{M}}_t^{v}$. For the proposed methods, we optimize the number of rounds r for each learning dataset. Details are presented in the experiment section.

B. Memory Updating Component

Next, we estimate the forgetting parameters $\boldsymbol{e}_t$ and $\boldsymbol{a}_t$ using the optimized $\tilde{\boldsymbol{v}}^{r}$ and $\tilde{\boldsymbol{M}}_t^{vr}$. These forgetting parameters $\boldsymbol{e}_t$ and $\boldsymbol{a}_t$ are important to update the latest value memory $\boldsymbol{M}_{t+1}^{v}$ optimally. The earlier memory updating component of DKVMN and Deep-IRT calculates the forgetting parameters from $\boldsymbol{v}_t$ solely based on current input information in (24) and (25). By contrast, we calculate them using the optimized current input data $\tilde{\boldsymbol{v}}_t^{r}$ and the past latent value $\tilde{\boldsymbol{M}}_t^{vr}$. Furthermore, the unique feature of the proposed method is a new layer $\boldsymbol{z}_t$, which helps to optimize $\boldsymbol{a}_t$. The memory updating component is located next to the hypernetwork on the upper right of Fig. 3. The forgetting parameters $\boldsymbol{e}_t$ and $\boldsymbol{a}_t$ are calculated as

$$\boldsymbol{e}_t^{(l)} = \sigma(\boldsymbol{W}^{e1}\tilde{\boldsymbol{v}}_t^{r} + \boldsymbol{W}^{e2}\tilde{\boldsymbol{M}}_{t,l}^{vr} + \boldsymbol{\tau}^{e}) \tag{32}$$

$$\boldsymbol{z}_t^{(l)} = \sigma(\boldsymbol{W}^{z1}\tilde{\boldsymbol{v}}_t^{r} + \boldsymbol{W}^{z2}\tilde{\boldsymbol{M}}_{t,l}^{vr} + \boldsymbol{\tau}^{z}) \tag{33}$$

$$\boldsymbol{a}_t^{(l)} = \tanh(\boldsymbol{W}^{a1}\boldsymbol{z}_t^{(l)} + \boldsymbol{W}^{a2}\tilde{\boldsymbol{M}}_{t,l}^{vr} + \boldsymbol{\tau}^{a}). \tag{34}$$

Therein, $\boldsymbol{W}^{(\cdot)}$ is the weight vector; $\boldsymbol{\tau}^{(\cdot)}$ is a bias vector. Then, the proposed method updates the latent value $\boldsymbol{M}_{t+1,l}^{v}$ as shown in the following:

$$\boldsymbol{M}_{t+1,l}^{v} = \tilde{\boldsymbol{M}}_{t,l}^{vr} \otimes (1 - w_{jl}\boldsymbol{e}_t^{(l)})^{\top} + w_{jl}\boldsymbol{a}_t^{(l)\top}. \tag{35}$$

By optimizing $\tilde{\boldsymbol{v}}_t$ and $\tilde{\boldsymbol{M}}_t^{v}$ in the hypernetwork, the parameters $\boldsymbol{e}_t$ and $\boldsymbol{a}_t$ are also estimated as optimizing the degree of forgetting of past data and as reflecting the current input data. Furthermore, the proposed method can capture the student knowledge state changes accurately because the latent knowledge state $\boldsymbol{M}_t^{v}$ has sufficient information related to the past learning history data.

V. Predictive Accuracy

A. Datasets

We conduct experiments to compare the performances of the proposed Deep-IRT in Section III (designated as "Proposed-DI") and the proposed Deep-IRT with a hypernetwork in Section IV (designated as "Proposed-HN") against existing solutions. This section presents a comparison of the prediction accuracies for student performance of the proposed methods with those of earlier methods (DKVMN [4], Yeung's Deep-IRT [3] (designated as "Yeung-DI"), AKT [5]) using six benchmark

TABLE I
SUMMARY OF BENCHMARK DATASETS

Dataset	No. students	No. skills	No. Items	Rate Correct	Learning length
ASSISTments2009	4151	111	26684	63.6%	52.1
ASSISTments2015	19840	100	N/A	73.2%	34.2
ASSISTments2017	1709	102	3162	39.0%	551.0
Statics2011	333	1223	N/A	79.8%	180.9
Junyi	48925	705	N/A	82.78%	345
Eedi	80000	1200	27613	64.25%	177

datasets as ASSISTments2009,[2] ASSISTments2015,[3] ASSISTments2017,[4] Statics2011,[5] Junyi,[6] and Eedi.[7] The ASSISTments datasets collected from online tutoring systems have been used as the standard benchmark for KT methods. The Statics2011 dataset was collected from college-level engineering courses on statics. The Junyi dataset was collected by Junyi Academy, a Chinese e-learning website [41]. We use only the students' exercise records in the math curriculum. In addition, we select items that the students attempted for the first time without hints. We also changed the question types into unique skill number tags. The Eedi dataset includes data from the school years of 2018–2020, with student responses to mathematics questions from Eedi, a leading educational platform by which millions of students interact daily around the globe [42]. For Eedi, each item has a list of hierarchical knowledge components. We convert these lists into unique skill number tags.

ASSISTments2009, ASSISTments2017, and Eedi have item and skill tags, although most methods explained in the relevant literature adopt only the skill tag as an input. However, methods with skill inputs rely on the assumption that items with the same skill are equivalent [5]. That assumption does not hold when an item's difficulties in the same skill differ greatly. Therefore, as inputs to AKT and the proposed method, we employ not only skills but also items [5], [23], [27]. Also, for ASSISTments2015, Statics2011, and Junyi with only skill tags, we employ the skill as input data. Table I presents the number of students (No. students), the number of skills (No. skills), the number of items (No. items), the rate of correct responses (Rate correct), and the average length of the items that students addressed (Learning length).

B. Hyperparameter Selection and Evaluation in Deep-IRT

We used standard fivefold cross-validation to evaluate the respective prediction accuracies of the methods. According to Ghosh et al. [5], for each fold, 20% learners are used as the test set, 20% are used as the validation set, and 60% are used as the training set.

For all methods, we chose batch sizes from $\{32, 64, 128, 256\}$ and the hidden layer sizes and memory dimensions of $\{10, 20, 50, 100, 200\}$ using cross-validation according to the earlier studies [3], [4]. Then, we employed Adam optimization with a learning rate of 0.003, as done for the earlier studies [3], [4].

In addition, for the earlier methods, we used the hyperparameters reported from the earlier studies [3], [4], [5]. Additionally, we set 200 items as the upper limit of the input length according to the earlier studies [3], [4], [5]. When the input length of items is greater than 200, we use the first 200 response data for all methods.

To ascertain the number of layers k for the proposed method, we conducted some experiments to gain experience using ASSISTments2009 while changing the layer number. The results are presented in Fig. 4. As the figure shows, the AUC score reaches its highest value when $k = 2$ and $k = 4$. Based on this finding, we employ $k = 2$ for the following experiments because the computation time of the proposal increases exponentially as the number of layers increases.

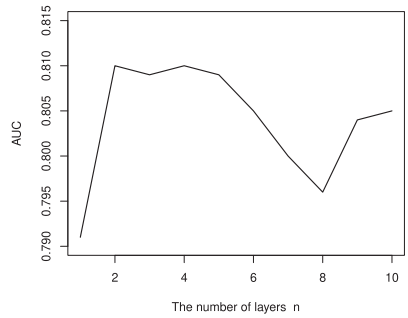


Fig. 4. AUC and the number of layers for ASSISTments2009. The vertical axis shows AUC on the left side. The horizontal axis shows the number of layers.

If the predicted correct answer probability for the next item is 0.5 or more, then the student's response to the next item is predicted as correct. Otherwise, the student's response is predicted as incorrect. For this study, we leverage three metrics for prediction accuracy: 1) accuracy (Acc) score, 2) AUC score, and 3) loss score [43], [44]. The first, Acc, represents the concordance rate between the student predictive responses and the actual responses. The second, AUC, provides a robust metric for binary prediction evaluation. When an AUC score is 0.5, the prediction performance is equal to that of random guessing. Loss represents the cross-entropy in (28).

We used a Tesla T4 GPU to train all methods.

C. Hyperparameter Selection in Hypernetwork

1) Optimal Tuning Parameter δ_1 and δ_2 Estimation: For our experiments, we optimize δ_1 and δ_2 to adjust the hypernetwork

[2][Online]. Available: https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data
[3][Online]. Available: https://sites.google.com/site/assistmentsdata/home/2015-assistments-skill-builder-data
[4][Online]. Available: https://sites.google.com/view/assistmentsdatamining
[5][Online]. Available: https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=507
[6][Online]. Available: http://www.junyiacademy.org/
[7][Online]. Available: https://eedi.com/projects/neurips-education-challenge

TABLE II
PREDICTION ACCURACIES AND HYPERPARAMETERS r

Dataset	Number of rounds r					
	2	3	4	5	6	7
Statics2011 (skill)	**82.25**	82.24	82.20	82.20	82.16	82.11
ASSISTments2009 (skill)	81.19	**81.83**	81.25	81.23	81.2	80.96
ASSISTments2015 (skill)	72.91	**72.95**	72.90	72.89	72.81	72.73
ASSISTments2017 (skill)	**85.06**	82.73	81.64	80.17	73.23	72.64
Junyi (skill)	**79.00**	78.74	78.71	78.67	78.62	78.65
Eedi (skill)	**75.53**	75.48	75.42	75.36	74.97	75.45
ASSISTments2009 (item and skill)	81.30	81.14	81.38	81.49	**82.55**	81.20
ASSISTments2017 (item and skill)	75.94	76.17	76.74	76.70	**76.85**	76.74
Eedi (item and skill)	**79.27**	79.10	79.05	79.05	78.92	79.00

The most max value important experimental values are indicated by bold entities.

for each dataset. To choose optimal parameters δ_1 and δ_2, we conducted some experiments using all training datasets by changing δ_1 and δ_2, respectively. The optimal tuning parameters $\{\delta_1, \delta_2\}$ are estimated as $\{1.5, 1.5\}$ for ASSISTments2009, ASSISTments2015 and ASSISTments2017, $\{1.0, 1.7\}$ for Statics2011, and $\{1.0, 1.0\}$ for Junyi and Eedi. Based on this result, we employ these tuning parameters for the following experiments.

2) Optimal Number of Rounds r Estimation: To ascertain the number of rounds r in the hypernetwork, we conducted some experiments to gain experience using the training datasets by changing the value of r. The results are presented in Table II. As the table shows, the number of rounds r is estimated as $r = 2$ for Statics2011, ASSISTments2017 with skill inputs, Junyi, and Eedi, as $r = 3$ for ASSISTments2009 and ASSISTments2015 with skill inputs and as $r = 6$ for ASSISTments2009 and ASSISTments2017 with item and skill inputs.

We find the number of rounds r using the grid search method. The proposed method estimates the number of each round r by incrementing the value from the initial value $r = 2$ to maximize the prediction accuracy.

3) Optimal Degree of Past Latent Variables to be Assessed: The input of the hypernetwork $\tilde{\boldsymbol{M}}_t^v$ is calculated from the past latent variables $\{\boldsymbol{M}_t^v, \boldsymbol{M}_{t-1}^v, \ldots, \boldsymbol{M}_{t-\lambda}^v\}$ at times $t - \lambda$ to t. We optimize λ by changing the value of $\lambda \in \{0, 1, 2, \ldots, t\}$ using the optimal δ_1, δ_2, and r for each learning dataset. Results show that the optimal λ can be estimated as $\lambda = 1$ for ASSISTments2009 and Junyi with skill inputs, and as ASSISTments2009 and ASSISTments2017 with item and skill inputs. When using the other datasets, optimal λ is estimated as $\lambda = 0$.

D. Results

1) Skill Inputs: The respective values of Acc, AUC, and Loss for all benchmark datasets with only skill inputs are presented in Table III. In addition, this report describes the standard deviations across five test folds. Proposed-DI and Proposed-HN, respectively, represent variants of the proposed method with and without the hypernetwork.

Results show that the averages of AUC, Acc, and Loss obtained using Proposed-DI are better than those using Yeung-DI, which is the earlier Deep-IRT method, although the proposed method separates student and item networks. This result implies that redundant deep student and item networks function effectively for performance prediction. These results are explainable from reports of state-of-the-art methods [20], [21], [22].

Also, Proposed-HN, which optimizes the forgetting parameters in the hypernetwork, provides the best average scores for all metrics. Proposed-HN improves the prediction accuracy of Proposed-DI. In fact, Proposed-HN outperforms AKT, which was reported as having the highest accuracies among earlier methods. For each dataset, results indicate that Proposed-HN provides the best AUC scores for ASSISTments2009, ASSISTments2017, Statics2011, and Junyi. Especially, for ASSISTments2017 with long learning lengths, the performance of the Proposed-HN markedly outperforms that of AKT. By contrast, Proposed-HN tends to have lower prediction accuracies for ASSISTments2015 with a shorter learning length than AKT has. Results suggest that the proposed hypernetwork functions effectively, especially for datasets with long learning lengths.

To investigate the reason for that phenomenon, we analyze the forgetting parameters $\boldsymbol{e_t}$ and $\boldsymbol{a_t}$ in the memory updating component of the proposed method. As described earlier, $\boldsymbol{e}_t$ influences the degree to which the value memory forgets the past ability. In addition, $\boldsymbol{a}_t$ controls how much the value memory reflects the current input data. We calculate the l_2-norm of the forgetting parameters $\boldsymbol{e_t}$ and $\boldsymbol{a_t}$, respectively, for the earlier memory updating component (of Proposed-DI) and the new memory updating component with hypernetwork (of Proposed-HN) using the ASSISTments2017 dataset. Table IV presents the averages of the l_2-norms of $\boldsymbol{e_t}$ and $\boldsymbol{a_t}$ at time $t \in \{1, 2, \ldots, T\}$. Table IV shows that Proposed-DI has the larger l_2-norm value of $\boldsymbol{e_t}$ than $\boldsymbol{a_t}$. The earlier memory updating component drastically forgets the student's past ability information and reflects the current input data when the latent variable memory is updated. The reason is that the forgetting parameters $\boldsymbol{e_t}$ and $\boldsymbol{a_t}$ are calculated using only the current input data. Therefore, their latent value memory $\boldsymbol{M}_t^v$ might not store the student's past ability information. By contrast, Proposed-HN has a larger l_2-norm value of $\boldsymbol{a_t}$ than $\boldsymbol{e_t}$. In the memory updating component of Proposed-HN, $\boldsymbol{a_t}$ and $\boldsymbol{e_t}$ are calculated using both the current input data $\boldsymbol{v_t}$ and the past latent value memory $\boldsymbol{M}_t^v$. Furthermore, these $\boldsymbol{v_t}$ and $\boldsymbol{M}_t^v$ are optimized in the hypernetwork to balance both the current input data and the student's past ability information. The results obtained for the other datasets are almost identical to those obtained for ASSISTments2017, although they are omitted to avoid redundant descriptions. Therefore, results suggest that the Proposed-HN works more effectively for long learning processes because hypernetworks facilitate the reflection of past data.

Findings indicate that AKT provides the best performance for ASSISTments2015. However, the AKT performance results are worse than those of Proposed-HN for ASSISTments2017. Fig. 5 shows the average of attention weights of all students for the 200 items in ASSISTments2017. The vertical axis shows the average of attention weights. The horizontal axis shows the

TABLE III
PREDICTION ACCURACIES OF STUDENT PERFORMANCE WITH SKILL INPUTS

Dataset	metrics	DKVMN	Yeung-DI	AKT	Proposed-DI	Proposed-HN
ASSISTments2009	AUC	81.21 +/- 0.31	81.34 +/- 0.39	80.81 +/- 0.41	81.34 +/- 0.24	**81.83 +/- 0.30**
	Acc	75.11 +/- 0.66	76.55 +/- 0.45	76.57 +/- 0.55	**76.91 +/- 0.24**	76.80 +/- 0.49
	Loss	0.47 +/- 0.05	0.48 +/- 0.10	0.49 +/- 0.08	0.47 +/- 0.10	**0.46 +/- 0.11**
ASSISTments2015	AUC	72.61 +/- 0.16	72.53 +/- 0.23	**72.97 +/- 0.12**	72.34 +/- 0.13	72.95 +/- 0.14
	Acc	75.05 +/- 0.18	74.97 +/- 0.14	**75.25 +/- 0.10**	74.95 +/- 0.39	75.02 +/- 0.15
	Loss	0.51 +/- 0.02	0.52 +/- 0.03	**0.51 +/- 0.01**	0.52 +/- 0.02	**0.51 +/- 0.03**
ASSISTments2017	AUC	72.67+/- 0.37	72.08 +/- 0.32	73.25+/- 0.41	72.32+/- 0.69	**85.06 +/- 1.17**
	Acc	68.46 +/- 0.24	68.36 +/- 0.30	69.17+/- 0.70	68.07 +/- 0.54	**79.11 +/- 1.06**
	Loss	0.58+/- 0.03	0.59 +/- 0.07	0.58+/- 0.09	0.60 +/- 0.08	**0.48 +/- 0.24**
Statics2011	AUC	81.20 +/- 0.42	81.38 +/- 0.42	82.15 +/- 0.35	81.45 +/- 0.45	**82.25 +/- 0.55**
	Acc	79.24 +/- 0.84	80.33 +/- 0.78	80.41 +/- 0.67	79.18 +/- 0.67	**80.63 +/- 0.85**
	Loss	0.42 +/- 0.14	0.42 +/- 0.18	0.42 +/- 0.13	0.42 +/- 0.12	**0.41 +/- 0.20**
Junyi	AUC	78.59 +/- 0.21	78.39 +/- 0.20	78.84 +/- 0.19	78.47 +/- 0.21	**79.00 +/- 0.26**
	Acc	86.61 +/- 0.28	86.57 +/- 0.30	86.54 +/- 0.25	86.58 +/- 0.27	**86.76 +/- 0.24**
	Loss	0.31 +/- 0.07	0.31 +/- 0.07	0.31 +/- 0.04	0.31 +/- 0.06	**0.30+/- 0.05**
Eedi	AUC	75.11 +/- 0.16	75.63 +/- 0.17	**75.81 +/- 0.15**	75.76 +/- 0.17	75.53 +/- 0.15
	Acc	71.23 +/- 0.24	71.34 +/- 0.29	71.38 +/- 0.20	**71.41 +/- 0.25**	71.30 +/- 0.24
	Loss	0.59 +/- 0.06	**0.56 +/- 0.07**	**0.56 +/- 0.03**	**0.56 +/- 0.06**	0.57 +/- 0.06
Average	AUC	76.89	76.83	77.30	76.91	**79.35**
	Acc	74.46	75.05	76.55	76.18	**78.27**
	Loss	0.48	0.48	0.48	0.48	**0.46**

The most AUC, ACC; max value , Loss; minimum value important experimental values are indicated by bold entities.

TABLE IV
FORGETTING PARAMETERS' NORM AVERAGES

norm average	Proposed-DI	Proposed-HN
$\|e_t\|$	5.12	1.99
$\|a_t\|$	3.17	2.58

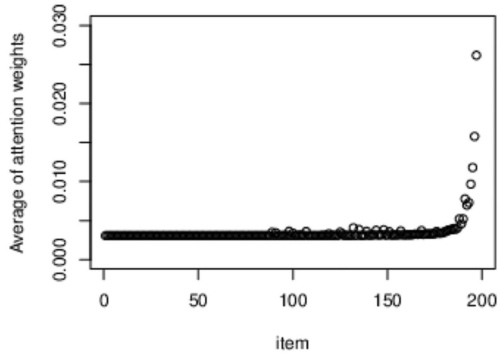


Fig. 5. Average of attention weights in AKT for ASSISTments2017.

number of items the student addressed. Fig. 5 shows that the attention weight α decays as the distance between the current input time and the past input time increases. It is noteworthy that the attention weight α converges to a certain nonzero value. This finding implies that AKT does not completely forget even past data obtained at an extremely long time prior. Consequently, AKT might inadequately forget the past response data from long learning processes. However, Ghosh et al. [5] reported that AKT is more effective for large datasets. Therefore, AKT provides the best performance for AUC of Eedi, which has an extremely large number of students. The performance results obtained using DKVMN are almost identical to those obtained using Yeng-DI because they have similar network structures.

2) Item and Skill Inputs: Furthermore, we compared the performances of the proposed methods with those of AKT for ASSISTments2009, ASSISTments2017, and Eedi with item and skill inputs according to the work in [5]. The respective values of Acc, AUC, and Loss are presented in Table V. Results indicate that the Proposed-HN provides the best performance for all metrics: averages of AUC, Acc, and Loss. For each dataset, the Proposed-HN provides the best scores for ASSISTments2009 and for ASSISTments2017. As described earlier, the Proposed-HN greatly outperforms AKT for ASSISTments2017 with a long learning length because the proposed hypernetwork functions effectively. However, for Eedi, AKT provides the best scores for all the metrics. In fact, AKT with item and skill inputs provides higher performance than those achieved using only skill inputs, as shown in [5]. In contrast, the proposed methods with item and skill inputs do not necessarily outperform those with only skill inputs. The reason might be that input item information cannot be used effectively because the latent value memory $\boldsymbol{M}_t^v$ is optimized using only input skills in the memory updating component. In addition, for Eedi, because of the increased number of parameters, it might not completely tune the hyperparameters in the hypernetwork.

TABLE V
PREDICTION ACCURACIES OF STUDENT PERFORMANCE WITH ITEM AND SKILL INPUTS

Dataset	metrics	AKT	Proposed-DI	Proposed-HN
ASSISTments2009	AUC	82.20 +/- 0.25	80.70 +/- 0.56	**82.55 +/- 0.32**
	Acc	77.30 +/- 0.55	76.13 +/- 0.58	**77.42 +/- 0.49**
	Loss	0.49 +/- 0.10	0.54 +/- 0.10	**0.47 +/- 0.11**
ASSISTments2017	AUC	74.54+/- 0.21	74.15+/- 0.27	**77.69 +/- 0.51**
	Acc	69.83+/- 0.15	68.73+/- 0.11	**72.16 +/- 0.55**
	Loss	0.58+/- 0.06	0.57+/- 0.06	**0.54 +/- 0.13**
Eedi	AUC	**79.42 +/- 0.11**	79.11 +/- 0.14	79.27 +/- 0.15
	Acc	**73.59 +/- 0.16**	73.42 +/- 0.24	73.49 +/- 0.27
	Loss	**0.52 +/- 0.02**	0.53 +/- 0.00	0.53 +/- 0.00
Average	AUC	78.72	78.00	**79.83**
	Acc	73.57	72.76	**74.36**
	Loss	0.53	0.55	**0.51**

The most AUC, ACC; max value , Loss; minimum value important experimental values are indicated by bold entities.

Moreover, we experimented with TIRT [11], [12]. It is a hidden Markov IRT with a parameter to forget past response data, as described earlier in Section II-A. IRT-based methods

TABLE VI
NUMBERS OF TRAINABLE PARAMETERS AND COMPUTATIONAL TIMES FOR MODEL TRAINING

Dataset	DKVMN		Yeung-DI		AKT		Proposed-DI		Proposed-HN	
	params	times	params	times	params	times	params	times	params	times
ASSISTments2009 (skill)	124 951	1294	63 002	770	4 171 033	2935	1 573 102	1348	1 705 766	1433
ASSISTments2009	-	-	-	-	4 272 917	2182	2 420 302	2132	2 563 066	2493
ASSISTments2015	65 051	3045	60 502	3499	4 168 473	4820	1 570 602	3436	1 683 166	3410
ASSISTments2017 (skill)	131 001	2500	61 002	5462	4 168 985	3386	1 571 102	1889	1 786 066	1867
ASSISTments2017	-	-	-	-	4 250 996	3393	2 343 852	1515	2 486 616	1543
Statics2011	345 801	399	341 252	217	4 455 961	468	351 927	480	380 109	403
Junyi	439 951	8591	235 502	3707	4 347 673	5510	1 970 566	3258	2 020 666	3807
Eedi (skill)	619 951	7943	335 502	3353	4 450 073	8851	1 845 602	2510	5 132 758	3053
Eedi	-	-	-	-	5 399 869	8794	3 232 402	2571	6 519 558	2955

Units are in seconds.

rely on an assumption of local independence among the student item responses. They should not be applied to learning processes that allow a student to respond to the same item repeatedly. Therefore, we employ not skills but items as inputs using ASSISTments2009 and ASSISTments2017. In addition, we decompose these datasets into their respective skill groups and estimate the parameters from skill data independently because TIRT assumes a single-dimension skill of the ability. In other words, TIRT predicts performance using only an ability corresponding to one skill for an item. To estimate the student ability and item parameters of TIRT, we use the expected a posterior estimators using the Markov chain Monte Carlo method [45]. The results indicate that AUC is 80.38, Acc is 76.39, and Loss is 0.49 for ASSISTments2009. For ASSISTments2017, results show that AUC is 75.52, Acc is 84.71, and Loss is 0.46. Surprisingly, TIRT outperforms AKT with skill input for ASSISTments2017. That finding suggests that TIRT might estimate the student ability transition accurately. For the Eedi dataset, TIRT cannot complete the calculations within 24 h because of its data size.

E. Computational Costs

This section presents an investigation of the computational costs associated with each method. Concretely, we calculated the number of trainable parameters and training time for each method. We measured the training time for each partition in five-fold cross-validation. Table VI shows the number of trainable parameters and the average training times. According to Table VI, AKT has the largest number of parameters. It requires the longest computation training time. Proposed-DI and Proposed-HN can be trained more quickly than AKT can. Although Proposed-HN has more parameters than Proposed-DI does, the training times are comparable. In addition, DKVMN and Yeung-DI, which have relatively small numbers of parameters, were trained more quickly than the proposed methods and AKT. The computational times of DKVMN are almost identical to those of Yeung-DI because they have similar network structures.

VI. PARAMETER INTERPRETABILITY

A. Estimation Accuracy of Ability Parameters

In the preceding section, we showed that the proposed method has higher prediction accuracy than other methods. As described in this section, to evaluate the interpretability of the ability parameters of the proposed method, we use simulation data to compare the parameter estimates with those of the earlier Deep-IRT [3]. These datasets are generated from TIRT [11], [12]. The prior of θ_{it} is a normal distribution described as $\theta_{i0} \sim \mathcal{N}(0,1)$, $\theta_{it} \sim \mathcal{N}(\theta_{it-1}, \epsilon)$. Therein, ϵ represents the variance of θ_{it}. It controls the smoothness of a student's ability transition. Therefore, as ϵ increases, the range of fluctuation of the true ability increases at each time point. For this experiment, the priors of the jth item parameters are $\log a_j \sim \mathcal{N}(0,1)$, $b_j \sim \mathcal{N}(0,1)$. Each dataset includes 2000 student responses to $\{50, 100, 200, 300\}$ items. Discrimination parameter $\boldsymbol{a}$ and the item's difficulty parameter $\boldsymbol{b}$ are estimated using 1800 students' response data. Given the estimated $\boldsymbol{a}$ and $\boldsymbol{b}$, we estimate the students' ability parameters using the remaining 200 students' response data. In addition, for each dataset, we obtain results while changing $\epsilon = \{0.1, 0.3, 0.5, 1.0\}$.

We evaluate Pearson's correlation coefficients, Spearman's rank correlation coefficients, and Kendall rank correlation coefficients between the true ability parameters of the true model (TIRT) and the estimated ability parameters of the Deep-IRTs (Yeung-DI, Proposed-DI, and Proposed-HN) [46], [47]. Spearman's rank correlation is the nonparametric version of Pearson's correlation. The Kendall rank correlation coefficient is known to provide robust estimates for aberrant values [48]. Generally, the estimation accuracy of the ability parameters is evaluated using the root-mean-square error (RMSE). However, a student's ability of TIRT does not assume a standard normal distribution because the student ability distribution differs at each time. We are unable to evaluate RMSE in this experiment because TIRT, the earlier Deep-IRT method [3], and the proposed methods are unable to not standardize their student abilities.

We calculate a correlation coefficient using student's abilities θ_t at time $t \in \{1, 2, \ldots, T\}$, as estimated using TIRT and the Deep-IRTs. Next, we average these correlation coefficients of all students. Table VII presents the average correlation coefficients of the methods for the respective conditions.

To confirm the significance of the differences between the proposed methods from Yeung-DI, we applied the Tukey–Kramer multiple comparison test [49]. The p-values are presented on the right side of Table VII.

Results show that, for all conditions, Proposed-DI and Proposed-HN provide a stronger correlation with the true ability parameters than Yeung-DI does. The results of Spearman's rank correlation coefficients of the proposed method are greater than those of Pearson's correlation coefficients because the student's ability distribution changes constantly over

TABLE VII
CORRELATION COEFFICIENTS OF THE ESTIMATED ABILITIES

ϵ	No. items / Method	50	100	200	300	50	100	200	300	50	100	200	300	p-value (versus Yeung-DI)
		Pearson				Spearman				Kendall				
0.1	Yeung-DI	0.626	0.667	0.740	0.738	0.626	0.660	0.750	0.745	0.441	0.473	0.550	0.549	-
	Proposed-DI	0.885	0.907	0.924	0.916	0.892	0.915	0.940	0.938	0.710	0.746	0.785	0.782	0.00001
	Proposed-HN	**0.902**	**0.916**	**0.930**	**0.927**	**0.910**	**0.923**	**0.943**	**0.941**	**0.736**	**0.761**	**0.790**	**0.792**	0.00001
0.3	Yeung-DI	0.730	0.799	0.808	0.823	0.751	0.831	0.862	0.873	0.551	0.628	0.659	0.670	-
	Proposed-DI	0.827	0.891	0.883	0.890	0.863	0.926	0.941	0.945	0.671	0.755	0.778	0.785	0.0317
	Proposed-HN	**0.840**	**0.905**	**0.900**	**0.907**	**0.877**	**0.932**	**0.947**	**0.954**	**0.689**	**0.767**	**0.791**	**0.804**	0.0133
0.5	Yeung-DI	0.773	0.800	0.807	0.814	0.812	0.861	0.877	0.890	0.605	0.654	0.676	0.692	-
	Proposed-DI	0.855	0.870	0.860	0.849	0.893	0.928	0.929	0.930	0.705	0.755	0.758	0.761	0.1151
	Proposed-HN	**0.874**	**0.871**	**0.869**	**0.859**	**0.901**	**0.928**	**0.934**	**0.940**	**0.720**	**0.755**	**0.768**	**0.779**	0.0676
1.0	Yeung-DI	0.788	0.809	0.824	0.813	0.834	0.884	0.891	0.888	0.626	0.684	0.695	0.692	-
	Proposed-DI	0.843	0.830	0.844	0.834	0.886	0.911	0.919	0.918	0.696	0.728	0.740	0.740	0.497
	Proposed-HN	**0.854**	**0.840**	**0.854**	**0.836**	**0.894**	**0.920**	**0.930**	**0.919**	**0.708**	**0.744**	**0.762**	**0.743**	0.340

The most max value important experimental values are indicated by bold entities.

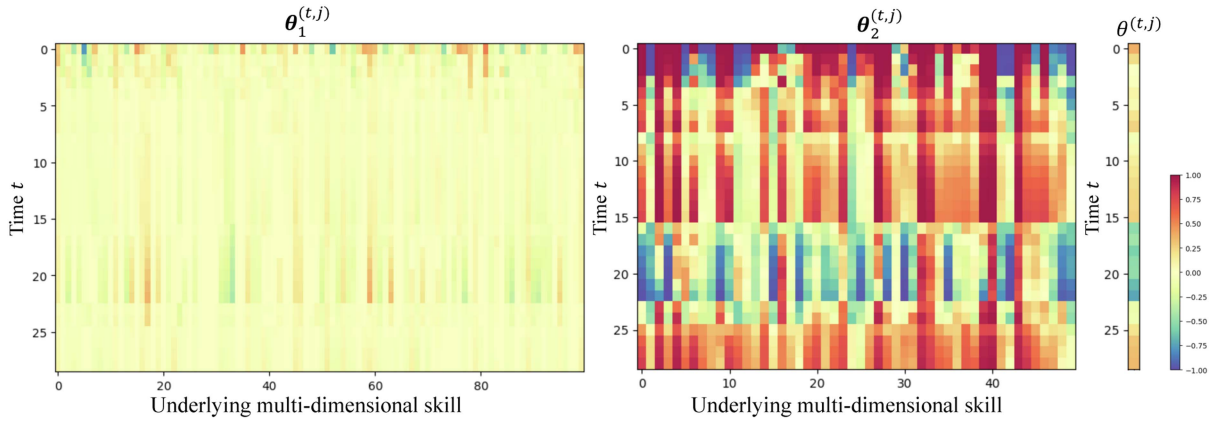


Fig. 6. Examples of student ability $\theta^{(t,j)}$ and latent abilities $\boldsymbol{\theta}_1^{(t,j)}$, $\boldsymbol{\theta}_2^{(t,j)}$ estimated in the input layer and the hidden layer of the student network at times $t = 1$ to $t = 30$.

time in TIRT. Especially, the results obtained for Kendall rank correlation coefficients suggest that Proposed-DI and Proposed-HN estimate the abilities robustly, even for aberrant values. The results demonstrate that the two proposed independent networks function effectively to provide appropriate interpretability of the estimated parameters. Moreover, the students' ability parameters are estimated accurately with sufficient information from past learning history data because the hypernetwork optimized the forgetting parameters using both current input data and past data. Furthermore, the proposed methods tend to produce stronger correlations as the number of items increases. These findings suggest that the proposed methods represent the true student's ability transition accurately in long learning processes.

B. Student Ability Transitions

This section shows student ability transitions using the proposed method.

First, we visualized the student ability parameters for underlying skills in the student network of Proposed-HN. Fig. 6 presents an example of the student ability transition $\theta^{(t,j)}$ and latent abilities $\boldsymbol{\theta}_1^{(t,j)}$, $\boldsymbol{\theta}_2^{(t,j)}$ estimated, respectively, in the hidden layers of the student network at time $t = 1$ to $t = 30$. The vertical axis shows the time stamp at which the student addresses each item. The horizontal axis shows the underlying skills. Fig. 6 depicts that $\boldsymbol{\theta}_2^{(t,j)}$ reflects the features of each underlying skill more strongly than $\boldsymbol{\theta}_1^{(t,j)}$ as the hidden layers of the neural network get deeper. This result suggests that the hidden layer is effective for identifying the underlying skills and for accurately capturing multidimensional ability.

Next, we evaluated the interpretability of the ability parameters of the proposed method by visualizing the ability transition.

Visualizing the ability transition for each skill is helpful for both students and teachers because it can reveal student strengths and weaknesses and can improve the learning method to fill in the learning gaps. Yeung [3] demonstrated a student ability transition for each skill using Yeung-DI. However, their results included some counterintuitive ability estimates. For example, even when the student answered incorrectly, the corresponding student ability estimate increased. Moreover, Yeung-DI cannot identify a relation among multidimensional skills. In some cases, a student's ability for low-level skills decreases even when the student responds correctly to items for high-level skills.

Fig. 7 depicts an example of student ability transitions of each skill estimated using Yeung-DI and Proposed-HN for the ASSISTments2009 according to earlier studies [3], [27]. The

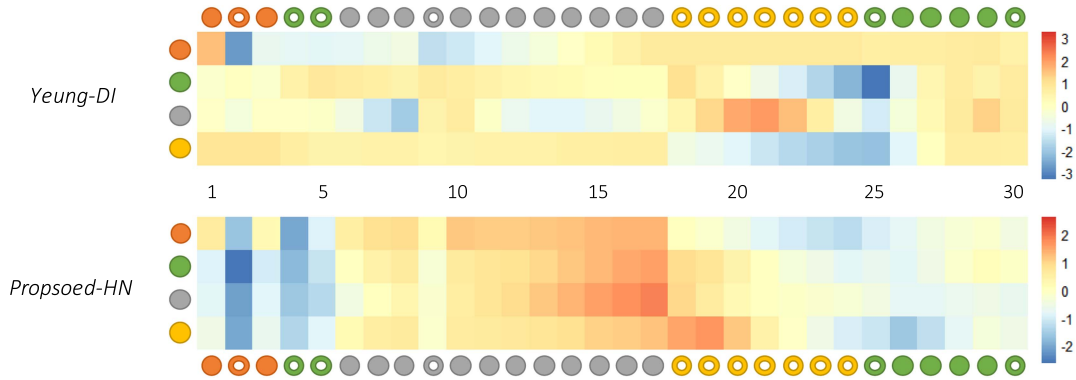


Fig. 7. Example of a student ability transition from the ASSISTments2009 dataset. The skill inputs are classified, respectively, as ordering factions (orange), equation solving more than two steps (gray), equation solving two or fewer steps (green), finding percentages (yellow), and finding percentages (orange). The filled and the hollow circles, respectively, represent correct and incorrect responses.

vertical axis shows the student's ability value on the right side. The horizontal axis shows the item number. The student response is shown by filled circles "•" when the student answers the item correctly; it is shown by hollow circles "○" otherwise. In the first 30 attempts, the student attempted skills of "equation solving more than two steps" (shown in gray), "equation solving two or few steps" (shown in green), "ordering factions" (shown in orange), and "finding percents" (shown in yellow).

For Yeung-DI, as described in earlier reports [3], some of the ability changes might be inconsistent with response data. For instance, the ability of skill "equation solving more than two steps" (gray), which is a higher-level skill, decreases even though the student responds correctly to items 11–17. In another instance, the student responds correctly to items for high-level skills even when a student's ability for low-level skills "equation solving two or few steps" (green) decreases. These unstable behaviors of Yeung-DI might engender severe difficulties, which will consequently confuse students and teachers, as a student model.

In contrast, Fig. 7 shows that the Proposed-HN can provide accurate estimates to reflect the student responses. Additionally, it can estimate relations among the skills. Therefore, when a student responds to an item, not only the corresponding skill ability but also those for other skills change. Especially, because the skills of "equation solving more than two steps" (gray) and "equation solving two or few steps" (green) are similar, the ability changes of each skill also indicate a strong correlation. Consequently, the results demonstrate that the proposed method improves the interpretability of Yeung-DI.

It is noteworthy that the student's responses are not immediately reflected in the estimated ability change when the student provides a different response from the previous several continuous same responses. For example, the ability for "finding percents" (yellow) increases in items 18–19 despite incorrect responses because the Proposed-HN estimates the student's ability with the past responses. Then, the estimated ability values change slightly later when the student provides a different response from the previous several continuous same responses.

VII. Conclusion

This study examined a proposed novel Deep-IRT that models a student's response to an item by two independent redundant networks: 1) a student network and 2) an item network. Because of two independent redundant neural networks, the parameters of the proposed method can be interpreted to a considerable degree while maintaining high prediction accuracy. Furthermore, we improved the prediction accuracy of the proposed method by combining it with a novel hypernetwork. In the earlier memory updating component, the forgetting parameters, which control the degree of forgetting the past latent value memory, are optimized only from the current input data. That restriction might degrade the prediction accuracy of the Deep-IRT because the value memory only insufficiently reflects the past learning information. The proposed hypernetwork can estimate the optimal forgetting parameters by balancing both the current input data and the past latent variables.

Experiments conducted with the benchmark datasets demonstrated that the proposed method improves both the ability parameter interpretability and the prediction accuracies of the earlier KT methods. Especially, results showed that the proposed method with the hypernetwork is effective for tasks with a long-term learning process. Experiments for the simulation dataset demonstrated that the proposed method provides stronger correlations with true parameters of TIRT than the earlier Deep-IRT method. Furthermore, the proposed method estimates the abilities robustly, even with aberrant values.

This study employed slightly redundant deep networks compared to earlier methods. In future work, we intend to use the proposed method to investigate the performances of more-redundant and deeper networks. In addition, we will try to optimize a hypernetwork to maximize the prediction accuracy for large datasets. Most recently, results of some studies have shown that each item's characteristics differ according to their texts, although they require the same skill. To resolve this difficulty, they proposed KT methods to estimate the relation between the item's text content and the student's performance

using the NLP technique or graph neural network [50], [51], [52], [53], [54], [55], [56]. In future work, we expect to incorporate the item's text content into the proposed method to improve the student performance prediction accuracy. Furthermore, deep-learning approaches for KT have been used for computerized adaptive testing (CAT) [57], [58]. The main purpose of CAT is measurement of student ability in personalized tests for online education. Therefore, we infer that the proposed method might be effective for CAT because it can estimate student's capabilities correctly.

References

[1] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User Model. User-Adapt. Interact.*, vol. 4, no. 4, pp. 253–278, Dec. 1995.

[2] C. Piech et al., "Deep knowledge tracing," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 505–513.

[3] C. Yeung, "Deep-IRT: Make deep learning based knowledge tracing explainable using item response theory," in *Proc. 12th Int. Conf. Educ. Data Mining*, 2019, pp. 683–686.

[4] J. Zhang, X. Shi, I. King, and D.-Y. Yeung, "Dynamic key-value memory network for knowledge tracing," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 765–774.

[5] A. Ghosh, N. Heffernan, and A. S. Lan, "Context-aware attentive knowledge tracing," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 2330–2339.

[6] Z. Pardos and N. T. Heffernan, "Modeling individualization in a Bayesian networks implementation of knowledge tracing," in *Proc. 18th Int. Conf. User Model., Adaption, Personalization*, 2010, pp. 255–266.

[7] Z. A. Pardos and N. T. Heffernan, "KT-IDEM: Introducing item difficulty to the knowledge tracing model," in *Proc. 19th Int. Conf. User Model., Adaption, Personalization*, 2011, pp. 243–254.

[8] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon, "Individualized Bayesian knowledge tracing models," in *Proc. Int. Conf. Artif. Intell. Educ.*, 2013, pp. 171–180.

[9] M. Khajah, Y. Huang, J. Gonzalez-Brenes, M. Mozer, and P. Brusilovsky, "Integrating knowledge tracing and item response theory: A tale of two frameworks," *Personalization Approaches Learn. Environ.*, vol. 1181, pp. 5–17, 2014.

[10] F. Baker and S. Kim, *Item Response Theory: Parameter Estimation Techniques (Ser. Statistics: A Series of Textbooks and Monographs)*, 2nd ed. New York, NY, USA: Taylor & Francis, 2004.

[11] C. Ekanadham and Y. Karklin, "T-SKIRT: Online estimation of student proficiency in an adaptive learning system," in *Proc. Mach. Learn. Educat. Workshop*, 2017.

[12] K. H. Wilson, Y. Karklin, B. Han, and C. Ekanadham, "Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation," in *Proc. 9th Int. Conf. Educ. Data Mining*, 2016, pp. 539–544.

[13] F. Bartolucci, F. Pennoni, and G. Vittadini, "Assessment of school performance through a multilevel latent Markov–Rasch model," *J. Educ. Behav. Statist.*, vol. 36, no. 4, pp. 491–522, 2011.

[14] D. Molenaar, D. Oberski, J. Vermunt, and P. D. Boeck, "Hidden Markov item response theory models for responses and response times," *Multivariate Behav. Res.*, vol. 51, pp. 606–626, 2016.

[15] X. Wang, J. Berger, and D. Burdick, "Bayesian analysis of dynamic item response models in educational testing," *Ann. Appl. Statist.*, vol. 7, no. 1, pp. 126–153, 2013.

[16] H. Sepp and S. Jurgen, "Long short-term memory," *Neural Computation*, vol. 14, pp. 1735–1780, 1997.

[17] G. Abdelrahman, Q. Wang, and B. Nunes, "Knowledge tracing: A survey," *ACM Comput. Surv.*, vol. 55, no. 11, pp. 1–37, Feb. 2023, doi: 10.1145/3569576.

[18] S. Pandey and G. Karypis, "A self-attentive model for knowledge tracing," in *Proc. Int. Conf. Educ. Data Mining*, 2019, pp. 384–389.

[19] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[20] H. He, G. Huang, and Y. Yuan, "Asymmetric valleys: Beyond sharp and flat local minima," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 2553–2564. [Online]. Available: http://papers.nips.cc/paper/8524-asymmetric-valleys-beyond-sharp-and-flat-local-minima.pdf

[21] A. Morcos, H. Yu, M. Paganini, and Y. Tian, "One ticket to win them all: Generalizing lottery ticket initializations across datasets and optimizers," in *Adv. Neural Inf. Process. Syst.*, 2019, pp. 4932–4942. [Online]. Available: http://papers.nips.cc/paper/8739-one-ticket-to-win-them-all-generalizing-lottery-ticket-initializations-across-datasets-and-optimizers.pdf

[22] V. Nagarajan and J. Z. Kolter, "Uniform convergence may be unable to explain generalization in deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 11615–11626. [Online]. Available: http://papers.nips.cc/paper/9336-uniform-convergence-may-be-unable-to-explain-generalization-in-deep-learning.pdf

[23] E. Tsutsumi, R. Kinoshita, and M. Ueno, "Deep-IRT with independent student and item networks," in *Proc. 14th Int. Conf. Educ. Data Mining*, 2021, pp. 510–517.

[24] H. David, D. Andrew, and V. L. Quoc, "Hypernetworks," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[25] K. Stanley, D. D'Ambrosio, and J. Gauci, "A hypercube-based encoding for evolving large-scale neural networks," *Artif. Life*, vol. 15, pp. 185–212, 2009.

[26] G. Melis, K. Tomá, and B. Phil, "Mogrifier LSTM," 2020, *arXiv:1909.01792*.

[27] E. Tsutsumi, Y. Guo, and M. Ueno, "DeepIRT with a hypernetwork to optimize the degree of forgetting of past data," in *Proc. 15th Int. Conf. Educ. Data Mining*, 2022, pp. 1–6, doi: 10.5281/zenodo.6853107.

[28] F. Lord and M. Novick, *Statistical Theories of Mental Test Scores*. Reading, MA, USA: Addison-Wesley, 1968.

[29] W. J. van der Linden, *Handbook of Item Response Theory, Volume Two: Statistical Tools* (Ser. Statistics in the Social and Behavioral Sciences). London, U.K.: Chapman & Hall, 2016.

[30] J. Gonzalez-Brenes, Y. Huang, and P. Brusilovsky, "General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge," in *Proc. 7th Int. Conf. Educ. Data Mining*, 2014, pp. 84–91.

[31] J. H. Park, "Modeling preference changes via a hidden Markov item response theory model," in *Handbook of Markov Chain Monte Carlo*. New York, NY, USA: Taylor & Francis, 2011, pp. 479–491.

[32] R. Weng and D. Coad, "Real-time Bayesian parameter estimation for item response models," *Bayesian Anal.*, vol. 13, pp. 115–137, 2016, doi: 10.1214/16-BA1043.

[33] R. Georg, *Probabilistic Models for Some Intelligence and Attainment Tests*. San Diego, CA, USA: MESA Press, 1993.

[34] M. Reckase, *Multidimensional Item Response Theory Models*, Berlin, Germany: Springer, 2009.

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[36] B. Krause, L. Lu, I. Murray, and S. Renals, "Multiplicative LSTM for sequence modelling," 2017 *arXiv:1609.07959*.

[37] Y. Wu, S. Zhang, Y. Zhang, Y. Bengio, and R. Salakhutdinov, "On multiplicative integration with recurrent neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2856–2864.

[38] J. Koutník, F. Gomez, and J. Schmidhuber, "Evolving neural networks in compressed weight space," in *Proc. Genet. Evol. Computation Conf.*, 2010, pp. 619–626.

[39] C. Fernando et al., "Convolution by evolution: Differentiable pattern producing networks," in *Proc. Genet. Evol. Computation Conf.*, 2016, pp. 109–116.

[40] M. Moczulski, M. Denil, J. Appleyard, and N. Freitas, "ACDC: A structured efficient linear layer," 2016, *arXiv:1511.05946*.

[41] H.-S. Chang, H.-J. Hsu, and K.-T. Chen, "Modeling exercise relationships in e-learning: A unified approach," in *Proc. Educ. Data Mining*, 2015, pp. 532–535.

[42] Z. Wang et al., "Results and insights from diagnostic questions: The NeurIPS 2020 education challenge," in *Proc. Mach. Learn. Res.*, 2021, vol. 133, pp. 191–205.

[43] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognit.*, vol. 30, no. 7, pp. 1145–1159, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320396001422

[44] J. Huang and C. Ling, "Using AUC and accuracy in evaluating learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 3, pp. 299–310, Mar. 2005.

[45] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, *Handbook of Markov Chain Monte Carlo*. Boca Raton, FL, USA: CRC Press, 2011.

[46] J. L. Rodgers and W. A. Nicewander, "Thirteen ways to look at the correlation coefficient," *Amer. Statistician*, vol. 42, no. 1, pp. 59–66, 1988, doi: 10.1080/00031305.1988.10475524.

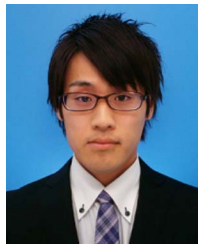
[47] D. Bonett and T. Wright, "Sample size requirements for estimating Pearson, Kendall and Spearman correlations," *Psychometrika*, vol. 65, pp. 23–28, 2000.
[48] E. Tsutsumi, R. Kinoshita, and M. Ueno, "Deep item response theory as a novel test theory based on deep learning," *Electronics*, vol. 10, no. 9, 2021, Art. no. 1020.
[49] J. Ludbrook, "Multiple comparison procedures updated," *Clin. Exp. Pharmacol. Physiol.*, vol. 25, no. 12, pp. 1032–1037, 1998.
[50] Q. Liu et al., "EKT: Exercise-aware knowledge tracing for student performance prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 1, pp. 100–115, Jan. 2021.
[51] S. Pandey and J. Srivastava, "RKT: Relation-aware self-attention for knowledge tracing," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 1205–1214.
[52] Y. Su et al., "Exercise-enhanced sequential modeling for student performance prediction," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 2435–2443.
[53] S. Sonkar, A. E. Waters, A. S. Lan, P. J. Grimaldi, and R. Baraniuk, "qDKT: Question-centric deep knowledge tracing," in *Proc. 13th Int. Conf. Educat. Data Mining*, 2020, pp. 677–681.
[54] Y. Ma, P. Han, H. Qiao, C. Cui, Y. Yin, and D. Yu, "SPAKT: A self-supervised pre-training method for knowledge tracing," *IEEE Access*, vol. 10, pp. 72145–72154, 2022.
[55] Z. Wu, L. Huang, Q. Huang, C. Huang, and Y. Tang, "SGKT: Session graph-based knowledge tracing for student performance prediction," *Expert Syst. Appl.*, vol. 206, 2022, Art. no. 117681.
[56] Y. Luo, B. Xiao, H. Jiang, and J. Ma, "Heterogeneous graph based knowledge tracing," in *Proc. 11th Int. Conf. Educ. Inf. Technol.*, 2022, pp. 226–231.
[57] H. Bi et al., "Quality meets diversity: A model-agnostic framework for computerized adaptive testing," in *Proc. IEEE Int. Conf. Data Mining*, 2020, pp. 42–51.
[58] Y. Zhuang, Q. Liu, Z. Huang, Z. Li, S. Shen, and H. Ma, "Fully adaptive framework: Neural computerized adaptive testing for online education," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 4734–4742. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/20399

Emiko Tsutsumi received the Ph.D. degree in engineering from the University of Electro-Communications, Chofu, Japan, in 2023.

Since 2023, she has been a Project Assistant Professor with the University of Tokyo, Tokyo, Japan. Her research interests include e-learning, e-testing, machine learning, and data mining.

Yiming Guo received the M.E. degree in engineering from the University of Electro-Communications, Chofu, Japan, in 2023.

His main research interests include e-learning, machine learning, and data mining.



Ryo Kinoshita received the M.E. degree in engineering from the University of Electro-Communications, Chofu, Japan, in 2017.

His research interests include e-learning, machine learning, and data mining.

Maomi Ueno (Member, IEEE) received the Ph.D. degree in computer science from the Tokyo Institute of Technology, Tokyo, Japan, in 1994.

Since 2013, he has been a Professor with the Graduate School of Information Systems, University of Electro-Communications, Chofu, Japan. His research interests include e-learning, e-testing, e-portfolio, machine learning, data mining, Bayesian statistics, and Bayesian networks.

Dr. Ueno was the recipient of Best Paper awards from ICTAI2008, ED-MEDIA 2008, e-Learn2004, e-Learn2005, and e-Learn2007.