Deep-IRT: Make Deep Learning Based Knowledge Tracing Explainable Using Item Response Theory

Chun-Kit Yeung
Find Solution Ai Limited
Hong Kong
chunkit@findsolutiongroup.com

ABSTRACT

Deep learning based knowledge tracing model has been shown to outperform traditional knowledge tracing model without the need for human-engineered features, yet its parameters and representations have long been criticized for not being explainable. In this paper, we propose *Deep-IRT* which is a synthesis of the item response theory (IRT) model and a knowledge tracing model that is based on the deep neural network architecture called dynamic key-value memory network (DKVMN) to make deep learning based knowledge tracing explainable. Specifically, we use the DKVMN model to process the student's learning trajectory and estimate the item difficulty level and the student ability over time. Then, we use the IRT model to estimate the probability that a student will answer an item correctly using the estimated student ability and the item difficulty. Experiments show that the Deep-IRT model retains the performance of the DKVMN model, while it provides a direct psychological interpretation of both students and items.

Keywords

Knowledge tracing, item response theory, deep learning.

1. INTRODUCTION

With the advancement of digital technologies, online platforms for intelligent tutoring systems (ITSs) and massive open online courses (MOOCs) are becoming prevalent. These platforms produce massive datasets of student learning trajectories about the *knowledge components* (KCs), where KC is a generic term for concept, skill, exercise, item, etc. The availability of online activity logs of students has accelerated the development of learning analytics and educational data mining tools for predicting the performance and advising the learning of students. Among many topics, knowledge tracing is considered to be important for enhancing personalized learning. Knowledge tracing is the task of modeling student's *knowledge state*, which is a general representation of the mastery level of KCs, e.g., a scalar value representing a student ability level, or a vector representation similar to word embedding [10]. With the estimated students' knowledge state, teachers or tutors can gain a better understanding of the attainment levels of their students and can tailor the learning materials accordingly. Moreover, students may also take advantage of the learning analytics tools to come up with better learning plans to deal with their weaknesses and maximize their learning efficacy.

学生能力水平

查漏补缺，最大化学习效率

Generally, the knowledge tracing task can be formalized as follows: given a sequence of student's historical interactions $\mathbf{X}_t = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_t)$ up to time t on a particular learning task, it predicts some aspects of his next interaction $\mathbf{x}_{t+1}$. Question-and-answer interactions are the most common type in knowledge tracing, and thus $\mathbf{x}_t$ is usually represented as an ordered pair (q_t, a_t) which constitutes a tag for the question q_t being answered at time t and an answer label a_t indicating whether the question has been answered correctly. In many cases, knowledge tracing seeks to predict the probability that a student will answer a question q_{t+1} correctly given the sequence $\mathbf{X}_t$, i.e., $P(a_{t+1} = 1|q_{t+1}, \mathbf{X}_t)$.

Xt=(qt, at)是一个有序问题tag-答案tag对

Many mathematical and computational models have been developed to solve the knowledge tracing task. These models can be grouped into two categories [9]:

1. a highly structured model whose parameters have a direct meaningful interpretation, e.g., Bayesian knowledge tracing (BKT) [4] and performance factors analysis (PFA) [13];
2. a highly complex but general-purpose model whose parameters are difficult to interpret, e.g., deep knowledge tracing (DKT) [14] and dynamic key-value memory network (DKVMN) for knowledge tracing [22].

DKVMN：动态键值记忆网络

The former category typically provides more insight besides the prediction result, while the latter usually performs better without requiring substantial feature engineering by humans. To the best of our knowledge, there has not yet been a model that is highly complex and general-purpose, yet simultaneously explainable. Therefore, it is appealing to devise a model that inherits the merits of these two categories.

In this paper, we propose *deep item response theory* (Deep-IRT) to make the deep learning based knowledge tracing model explainable. The Deep-IRT model is inspired by the Bayesian deep learning [17] and is a synthesis of a deep learning model and a psychometric model. Specifically, the Deep-IRT model utilizes the DKVMN model to process input data and return psychologically meaningful parameters of the IRT model. The DKVMN model performs feature engineering job to extract latent features from student's historical question-and-answer interactions. Then, the extracted latent features are used to infer the difficulty level of and the student ability on each KC over time. Based on the estimated student ability and the KC difficulty level, the IRT model predicts the probability that the student will answer a KC correctly. By formulating the knowledge tracing task with both the DKVMN model and the IRT model, we are getting the merits from these two models. The Deep-IRT model benefits from the advance of deep learning techniques, e.g., capturing features that are hard to be human-engineered. On the other hand, we empower the explainability by introducing a well-known psychometric model which can be easily understood by many people.

Our experiments show that the proposed Deep-IRT model retains the performance of the DKVMN model. We also conduct analyses on the difficulty level and the student ability learned by the Deep-IRT model. Analyses reveal that the difficulty level estimated by the Deep-IRT model aligns with other traditional methods, e.g., the IRT model and the item analysis [1]. However, the Deep-IRT model still suffers from the reconstruction issue which is discovered in the DKT model [20].

Our main contributions are summarized as follows:

1. The proposed Deep-IRT knowledge tracing model is capable of inferring meaningful estimation of student ability and KCs' difficulty level while simultaneously retains the predictive power of the deep learning based knowledge tracing model.
2. The Deep-IRT model potentially provides an alternative way for estimating KC's difficulty level by utilizing the entire learning trajectory, rather than the traditional educational testing environment.
3. We propose to use a deep learning model to output parameters of a psychometric model so as to leverage the deep learning capability and provide explainable psychometric parameters. This idea can be applied elsewhere apart from the knowledge tracing task.

2. LITERATURE REVIEW

2.1 Item Response Theory

Item response theory [15] (IRT) has been used in the educational testing environment since the 1950s. It outputs the probability $P(a)$ that a student will answer an item (i.e., a question) j correctly during a test, based on the student's ability level θ and the item's difficulty level β_j (in the simplest one-parameter IRT[1].) This probability is defined by an item response function which has the following characteristics: If a student has a higher ability level, the student has a higher probability to answer an item correctly; on the other hand, if an item is more difficult, a student has a lower probability to answer the item correctly. Most commonly, the logistic regression model is used in the IRT model as the item response function:

学生能力

试题难度

$$P(a) = \sigma(\theta - \beta_j) = \frac{1}{1 + \exp(-(\theta - \beta_j))} \tag{1}$$

where $\sigma(\cdot)$ is the sigmoid function.

In addition to estimating the probability $P(a)$, the IRT model has also been widely used to estimate the student ability θ and the item difficulty level β_j. Yet, as the IRT model is originally devised for the educational testing environment, the model assumes that the student's ability is not changing during the test. Thus, it cannot be applied directly to the task of knowledge tracing, where student's knowledge state is changing over time.

IRT模型假设学生能力在测试中不会改变

2.2 Knowledge Tracing

To tackle the knowledge tracing task, researchers have been investigating mathematical and computational models since the 1990s. Various approaches, ranging from probabilistic models to deep neural networks, have been developed over the past three decades.

方法：从概率模型，到深度神经网络

2.2.1 *Bayesian Based Knowledge Tracing*

The Bayesian knowledge tracing (BKT) model was proposed by Corbett and Anderson [4] during the 1990s. It is used to model a skill acquisition process by a hidden Markov model (HMM). However, many simplifying assumptions adopted by the BKT model are unrealistic. One of the assumptions is that all students and questions of a same skill are treated equally in the BKT model. Thus, researchers have investigated in various individualizations on the BKT model. Some researchers have empowered the individualization of the BKT model on both skill-specific parameters [11, 12] and student-specific parameters [21]. Some other researchers has also investigated the synthesis of the BKT model and the IRT model [8, 18] to empower the individualization over questions and students. However, it should be noted that such extensions often require considerable feature engineering efforts and may incur a significant increase in the computational requirements.

BKT通过HMM建模技能获得过程。然而其中多个假设不太现实。

2.2.2 *Factors Analysis Based Knowledge Tracing*

In the 2000s, learning factors analysis (LFA) [3] and performance factors analysis (PFA) [13] are proposed to tackle the knowledge tracing task using the logistic regression model. Both models are similar to the IRT model, yet they estimate the probability that a student will answer a question correctly by learning skill-level parameters. LFA is formulated as follows:

PFA效果因素分析

$$P(a) = \sigma(\theta + \sum_{j \in skills} (\gamma_j N_j - \beta_j)), \tag{2}$$

where $\sigma(\cdot)$ is the sigmoid function, θ, γ_j and β_j are the model parameters, and N_j is the input to the model. Similar to the IRT model, θ and β_j can be conceived as the ability of a student and the difficulty level of a skill j, respectively. N_j means the number of attempts that the student has on the skill j, so γ_j can be interpreted as the learning rate for the skill j.

[1] The one-parameter IRT has the property of specific objectivity. It means that the rank of the item difficulty is the same for all students independent of ability, and the rank of the person ability is the same for all items independent of difficulty.

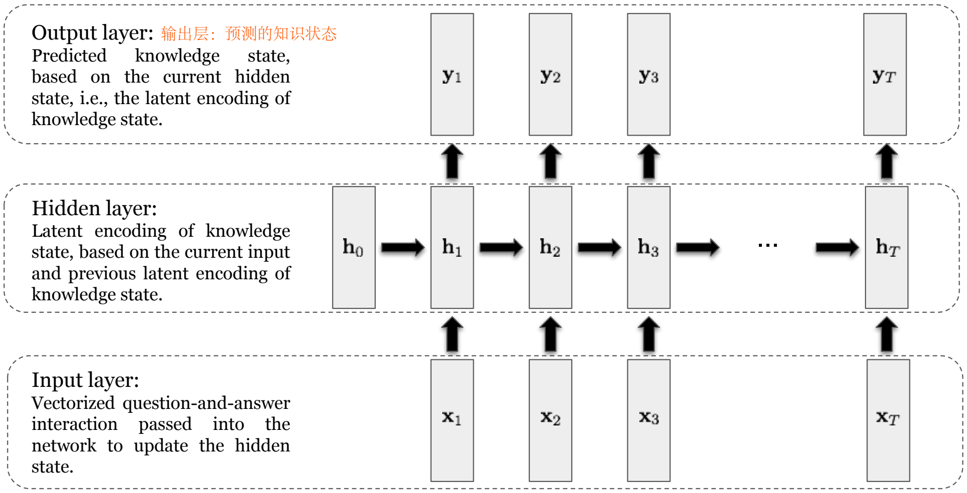
Output layer: 输出层：预测的知识状态
Predicted knowledge state, based on the current hidden state, i.e., the latent encoding of knowledge state.
$\mathbf{y}_1$
$\mathbf{y}_2$
$\mathbf{y}_3$
$\mathbf{y}_T$
Hidden layer:
Latent encoding of knowledge state, based on the current input and previous latent encoding of knowledge state.
$\mathbf{h}_0$
$\mathbf{h}_1$
$\mathbf{h}_2$
$\mathbf{h}_3$
···
$\mathbf{h}_T$
Input layer:
Vectorized question-and-answer interaction passed into the network to update the hidden state.
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{x}_T$

Figure 1: Unfolded version of the RNN architecture for DKT. The hidden state is processed differently in the vanilla RNN or LSTM-RNN. $\mathbf{h}_0$ is the initial hidden state in the RNN, and it is usually initialized randomly or to a zero vector.

After the emergence of the LFA model, Pavlik et al., who believed that student performance has a higher influence than the student ability in tackling the KT task, proposed the PFA model which offers higher sensitivity to student performance rather than student ability [13]. Concretely, it discards the parameter θ in the LFA model and splits the input N_j into S_j and F_j, which represent the number of successful and failed attempts, respectively, the student has on the skill j. The PFA model is formulated as follows:

$$P(a) = \sigma(\sum_{j \in skills} (\alpha_j S_j + \rho_j F_j - \beta_j)), \tag{3}$$

where α_j and ρ_j are the new model parameters. Similarly, both α_j and ρ_j can be considered as the learning rate for the skill j when it is applied successfully and unsuccessfully, respectively. In analogue with the IRT model, we can deem that the PFA model treats $\alpha_j S_j + \rho_j F_j$ to be the student ability θ on the skill j, such that a student can have different ability levels on different skills. It turns out that the PFA model performs better than the LFA model [5].

学生在不同技能上有不同的能力水平

LFA、PFA可以处理一个学习任务与多个技能相关。

As we can see from the model formulation, both LFA and PFA models can handle a learning task that is associated with multiple skills. However, it should be noted that they require manual labeling about the skills involved in solving the learning task, and they cannot deal with the inherent dependency among skills. Furthermore, the LFA assumes that $P(a)$ increases monotonically with the total number of attempts, so it is impossible to transit a student's knowledge state from learned to unlearned. Although the PFA model relieves this assumption by introducing the count of failed attempts, it is still difficult for the PFA model to decrease $P(a)$ once a student has numerous successful attempts. Even this issue is alleviated in the recent-PFA model [5] which uses an exponentially weighted average approach to obtain the recent correct rate of a student, we consider that the features used in the LFA and PFA models are still relatively simple and human-engineered, so it cannot adequately represent the students' knowledge state.

缺点：1.需要针对特定学习任务手工标注学习技能；2.无法处理不同技能之间的依赖关系

指数加权平均

2.2.3 *Deep Learning Based Knowledge Tracing*

Recently, with a surge of interest in deep learning models, deep knowledge tracing (DKT) [14], which models student's knowledge state based on a recurrent neural network (RNN), has been shown to outperform the traditional models, e.g., BKT and PFA, without the need for human-engineered features such as recency effect, contextualized trial sequence, inter-skill relationship, and students' ability variation [9].

In DKT, an interaction (q_t, a_t) is first transformed into a fixed-length input vector $\mathbf{x}_t$ using one-hot encoding. After the transformation, DKT passes the $\mathbf{x}_t$ to the hidden layer and computes the hidden state $\mathbf{h}_t$ using long short-term memory (LSTM) cells [7]. The hidden state of the RNN, in theory, summarizes all of the information from the past, so the hidden state in the DKT model can therefore be conceived as the latent knowledge state of a student that is resulted from his past learning trajectory. This latent knowledge state is then disseminated to the output layer to compute the output vector $\mathbf{y}_t$, which represents the probabilities of answering each KC correctly. If the student has a sequence of question-and-answer interactions of length T, the DKT model maps the inputs $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$ to the outputs $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T)$ accordingly. The unfolded RNN architecture for DKT is visualized in Figure 1, with a high-level interpretation.

ht隐藏层状态，即为学生在过去学习轨迹的潜在知识状态

yt输出向量，表征正确回答每个KC的概率

However, since all of the information captured by the RNN lives in a same vector space in the hidden layer, the DKT model is consequently difficult to provide consistent prediction across time and therefore failing to pinpoint accurately which KCs a student is good at or unfamiliar with [20]. An-

无法精确定位学生擅长哪个KC，或者不熟悉哪个KC

other novel neural network architecture, dynamic key-value memory network (DKVMN), is therefore proposed to alleviate this problem [22].

DKVMN使用单头注意力机制的记忆增强网络(两类记忆，静态Key memory，动态value memory)去建模技能获得过程，

DKVMN exploits the (single-head) attention mechanism in the memory augmented neural network [6] to model the skill acquisition process with two types of memory – the static *key memory* and the dynamic *value memory*. The static *key memory* and the dynamic *value memory* are an analogue of the Python dictionary. The *key memory* is immutable and stores the embedding vectors of different *query keys* for the attention mechanism; while the *value memory* is mutable and stores the *numerical values* of the corresponding *key*. To adapt the DKVMN model in the knowledge tracing setting, consider that there is a set of latent concepts (*key memory*) which is associated with a set of corresponding latent knowledge states (*value memory*). These latent concepts underlie all of the available KCs in a dataset, so each KC is a combination of these latent concepts. To construct a knowledge state of a KC, the same combination is applied on those corresponding latent knowledge states. This combination process generates vector representations of the KC and the knowledge state. The DKVMN model then estimates $P(a)$ based on the extracted KC and knowledge state vectors. In order to govern the KC acquisition process, the latent knowledge states in the *value memory* are updated when a student has done a learning task. Zhang et al. shows that the DKVMN model not only outperforms the DKT model without suffering from overfitting, but also discovers underlying concepts for input KCs precisely.

假设：存在一系列潜在概念(key memory) 与对应的潜在知识状态(value memory)相关

这些潜在概念位于数据集中所有可用的KC之下，所以每个KC都是这些潜在概念的组合。

Although DKVMN has provided an elegant methodology to model the KC acquisition process by distributing the vector representations of a KC and its knowledge state into different memory locations, these vector representations are still lack of interpretability, which is a common problem in most of the deep learning models. Therefore, there have been a tension in the learning science community between highly structured models, e.g., IRT, BKT and PFA, whose parameters have a direct psychological interpretation, versus, a highly complex but general-purpose models, e.g., DKT and DKVMN, whose parameters and representations are difficult to interpret. In the later section, we would like to relieve this tension by merging the IRT model and the DKVMN model so as to make the DKVMN model explainable.

3. DEEP ITEM RESPONSE THEORY

学生能力网络
KC难度网络

In this section, we explicate the Deep-IRT model which is a synthesis of the DKVMN model and the IRT model. Firstly, we explain the working mechanism of the DKVMN model mathematically. Then, we propose to augment the DKVMN model with the *student ability network* and the *difficulty network* so as to output the student ability level and the KC difficulty level that are required in the IRT model. Eventually, we use the one-parameter IRT model to predict the probability that a student will answer a KC correctly based on the estimated student ability and the KC difficulty level.

3.1 Working Mechanism of DKVMN

In summary, the DKVMN model works as follows: at time t, it first receives a KC q_t, then predicts the probability of answering q_t correctly, and eventually updates the memory using the question-and-answer interaction (q_t, a_t).

Assume that there are N latent concepts underlying Q distinct KCs. These latent concepts are stored in the key memory $\mathbf{M}^k \in \mathbb{R}^{N\times d_k}$, where d_k is the embedding size of a key memory slot. The student's knowledge states on these latent concepts are stored in the value memory $\mathbf{M}^v \in \mathbb{R}^{N\times d_v}$, where d_v is the embedding size of a value memory slot. To trace the student's knowledge state, the DKVMN model involves three major steps – getting an attention weight, making a prediction and updating the value memory.

3.1.1 Getting Attention Weight

The DKVMN model can make a prediction when it receives an input q_t at any time t. It first extracts the embedding vector of q_t from a KC embedding matrix $\mathbf{A} \in \mathbb{R}^{Q\times d_k}$. Then, it uses this embedding vector, denoted as $\mathbf{k}_t \in \mathbb{R}^{d_k}$ later on, to query the key memory matrix $\mathbf{M}^k$ in the DKVMN model. The query result is the weighting of how much attention should be paid on each value memory slot. This attention weight $\mathbf{w}_t \in \mathbb{R}^N$ is computed by the softmax activation of the inner product between $\mathbf{k}_t$ and each key memory slot $\mathbf{M}^k_i$:

$$w_{ti} = \text{Softmax}(\mathbf{M}^k_i \mathbf{k}_t), \tag{4}$$

where $\sum_{i=1}^N w_{ti} = 1$ and w_{ti} is the i-th element in the weight vector $\mathbf{w}_t$, and $\mathbf{M}^k_i$ is the i-th row-vector of $\mathbf{M}^k$.

3.1.2 Making Prediction

With the attention weight $\mathbf{w}_t$, the DKVMN model can predict the probability that a student will answer q_t correctly by the following process. First, the DKVMN model reads the latent knowledge state in the value memory $\mathbf{M}^v_t$ at time t to form a read vector

$$\mathbf{r}_t = \sum_{i=1}^N w_{ti} (\mathbf{M}^v_{ti})^T, \tag{5}$$

where $\mathbf{M}^v_{ti}$ is the i-th row-vector of $\mathbf{M}^v_t$. Then, the read vector $\mathbf{r}_t$ and the KC embedding vector $\mathbf{k}_t$ are concatenated vertically and disseminated to a fully connected layer with the hyperbolic tangent activation so as to generate a feature vector $\mathbf{f}_t$. The feature vector $\mathbf{f}_t$ is then used to calculate the probability p_t that the student will answer the KC q_t correctly. These steps can be expressed mathematically as follows:

$$\mathbf{f}_t = \tanh(\mathbf{W}_f[\mathbf{r}_t, \mathbf{k}_t] + \mathbf{b}_f), \tag{6}$$

$$p_t = P(a_t) = \sigma(\mathbf{W}_p \mathbf{f}_t + \mathbf{b}_p), \tag{7}$$

where $[\cdot]$ denotes concatenation, and both the sigmoid function $\sigma(\cdot)$ and the hyperbolic tangent $\tanh(\cdot)$ are applied in an element-wise manner. These are parameterized by a weight matrix $\mathbf{W}$ and a bias vector $\mathbf{b}$ with appropriate dimensions. Here we adopt p_t to represent $P(a_t)$ for the sake of avoiding confusion when we state the loss function later on.

3.1.3 Updating Value Memory

The DKVMN model updates the value memory $\mathbf{M}^v_t$ based on the input tuple (q_t, a_t) and the attention weight $\mathbf{w}_t$. The DKVMN model first retrieves an embedding vector of (q_t, a_t) from a KC-response embedding matrix $\mathbf{B} \in \mathbb{R}^{2Q\times d_v}$. This embedding vector, denoted as $\mathbf{v}_t \in \mathbb{R}^{d_v}$ later on, represents the knowledge growth after working on the KC q_t with the correct label a_t. When updating the memory, some of the memory is first erased with an erase vector $\mathbf{e}_t \in \mathbb{R}^{d_v}$ before

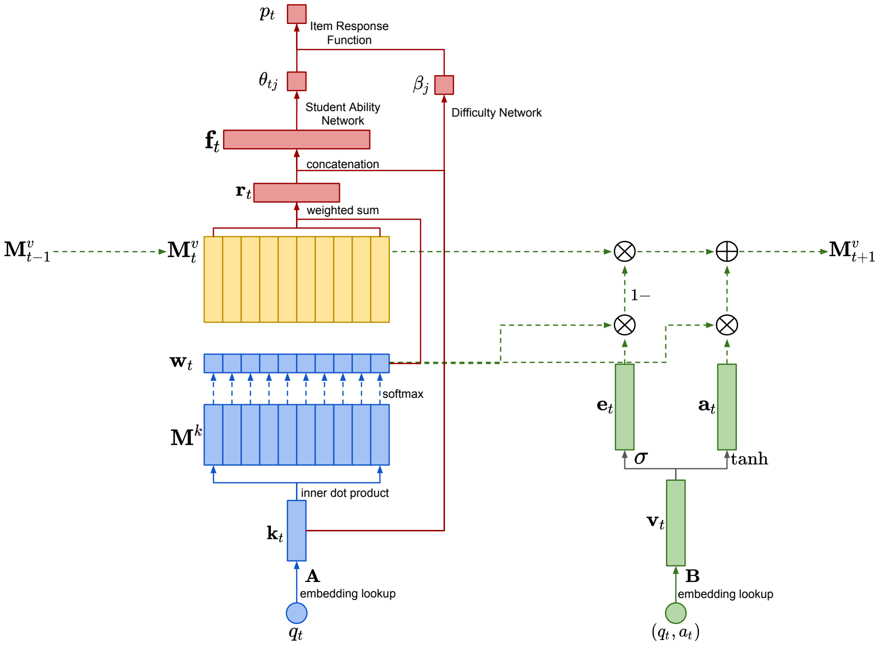

Figure 2: Network architecture for the Deep-IRT model. The model is drawn at time t only. The blue components describe the process of getting the attention weight, the green components describe the process of updating the value memory and the red components describe the process of making a prediction. The $\otimes$ and $\oplus$ represent element-wise multiplication and addition, respectively.

new information is added to the memory with the add vector $\mathbf{a}_t \in \mathbb{R}^{d_v}$. Erasing the memory offers the ability of forgetting similar to the LSTM cell. All in all, each value memory slot is updated as follows:

$$\mathbf{e}_t = \sigma(\mathbf{W}_e \mathbf{v}_t + \mathbf{b}_e), \tag{8}$$

$$\mathbf{a}_t = \tanh(\mathbf{W}_a \mathbf{v}_t + \mathbf{b}_a), \tag{9}$$

$$\tilde{\mathbf{M}}^v_{t+1,i} = \mathbf{M}^v_{ti} \otimes (\mathbf{1} - w_{ti}\mathbf{e}_t)^T, \tag{10}$$

$$\mathbf{M}^v_{t+1,i} = \tilde{\mathbf{M}}^v_{t+1,i} + w_{ti}\mathbf{a}_t^T, \tag{11}$$

where $\mathbf{1}$ is a vector of all ones, and $\otimes$ represents element-wise multiplication.

3.2 Student Ability and Difficulty Networks

As the architecture of DKVMN is elegant, it can be easily augmented to provide other meaningful information during the model influence. Firstly, the knowledge state of each latent concept can be exploited to calculate the student ability. Specifically, when the DKVMN model receives a KC q_t, it forms the feature vector $\mathbf{f}_t$ during the influence. As $\mathbf{f}_t$ is the concatenation of the read vector $\mathbf{r}_t$ and the KC embedding vector $\mathbf{k}_t$, it contains information of both the student's knowledge state on q_t and the embedded information of q_t. We believe that the $\mathbf{f}_t$ can be used to infer the student ability on q_t by further processing $\mathbf{f}_t$ via a neural network. Similarly, the difficulty level of q_t can be elicited by passing the KC embedding vector $\mathbf{k}_t$ to a neural network.

According to the purpose of the neural network, we call these two networks to be student ability network and difficulty network. Using a single fully-connected layer, we express these two networks as follows:

$$\theta_{tj} = \tanh(\mathbf{W}_\theta \mathbf{f}_t + \mathbf{b}_\theta), \tag{12}$$

$$\beta_j = \tanh(\mathbf{W}_\beta \mathbf{q}_t + \mathbf{b}_\beta), \tag{13}$$

where θ_{tj} and β_j can interpreted as the student ability on the KC j at time t and the difficulty level of the KC j, respectively. We use the hyperbolic tangent to be the activation function for both networks such that both outputs are scaled into the range $(-1, 1)$. Then, these two values are passed to the item response function to calculate the probability that a student will answer the KC j correctly:

$$p_t = \sigma(3.0 * \theta_{tj} - \beta_j). \tag{14}$$

The output of the student ability network are multiplied by a factor of 3.0 for a practical reason [19]. For example, if we do not scale up the student ability, the maximum value that can be obtained is $\sigma(1 - (-1)) = \sigma(2) = 0.881$.

The network architecture at time t is plotted in Figure 2. It should be noted that the student ability network and the KC difficulty network can be applied to any types of neural network. For example, these two networks can be inserted to the DKT model, i.e., the RNN, surrounded by the hidden layer and the output layer. By formulating the knowledge tracing task with both the DKVMN model and the IRT model, we are getting the best from two worlds. The model benefits from the advance of deep learning techniques such that it captures features that are hard to be human-engineered. On the other hand, we empower the explainability by introducing a well-known psychometric model which can be easily understood by many people. This idea of merging deep learning models and psychometric models can be

Table 1: The summary of datasets.

Dataset	No. students	No. skills	No. questions	No. interactions	Sequence length	Correct rate
ASSIST2009	4,151	110	26,684	325,637	78.45 ± 155.86	65.84%
ASSIST2015	19,840	100	N/A	683,801	34.47 ± 41.39	73.18%
Statics2011	333	156	1,223	189,297	568.46 ± 370.30	76.54%
Synthetic	2000	5	50	100,000	50.00 ± 0.00	58.83%
XXXX-F1toF3	310	99	2266	51,283	165.43 ± 163.65	46.69%

applied elsewhere apart from the knowledge tracing task, e.g., forgetting curve [2].

4. EXPERIMENTS

4.1 Datasets

We employ four public datasets and one proprietary dataset in our experiment. For the public datasets, we used the processed data that provided by Zhang et al. [22]. The information of these datasets is summarized in Table 1.

ASSIST2009 This dataset is provided by the ASSISTments online tutoring platform and has been used in several papers for the evaluation of knowledge tracing models. The dataset contains $4,151$ students with $325,637$ question-and-answering interactions from $26,688$ questions of 110 skills. The average sequence length of a student is 78.45 with a standard deviation of 155.86. The correct rate of this dataset is 65.84%. As most of the literature adopts the skill tag when they conducted the experiment, we also adopt the skill tag as the input to a model in our experiment.

ASSIST2015 This dataset contains $19,840$ student responses for 100 skills with a total of $683,801$ question-and-answering interactions. Although it contains more interactions than ASSIST2009, the average number of records per skill and student is actually smaller due to a larger number of students. The average sequence length of a student is 34.47 with a standard deviation of 41.39. The correct rate of this dataset is 73.18%. As this dataset only provides the skill tag, we adopt the skill tag as the input to a model.

Statics2011 This dataset is obtained from an engineering statics course with $189,927$ interactions from 333 students and $1,223$ question tags of 156 skills. The average sequence length of a student is 568.46 with a standard deviation of 370.30. The correct rate of this dataset is 76.54%. Moreover, the question tag is used as the input to a model.

Synthetic Piech et al. [14] also simulated 2000 virtual students' question-and-answer trajectories. Each student answers the same sequence of 50 questions each of which belong to a single concept $j \in \{1, \ldots, 5\}$ and has a difficulty level β, with an assigned ability θ of solving the task related to the concept j. The probability of a student answering a question correctly is defined based on the this IRT model $p(a) = c + \frac{1-c}{1+\exp(-(\theta-\beta))}$, where c denotes the probability of a student guessing it correctly and it is set to 0.25. The question tag is used as the input to a model.

FSAI-F1toF3 This dataset is provided by the Find Solution Ai Limited and is collected via an adaptive learning tablet application called 4LittleTrees[2]. We extracted the student interactions that are related to mathematics curriculum from F.1 to F.3 in Hong Kong[3]. It consists of $51,283$ question-and-answer interactions from 310 students on $2,266$ questions of 99 skills. The average sequence length of a student is 165.43 with a standard deviation of 163.65. The correct rate of this dataset is 46.69%. For this dataset, we use the question tag as the input to a model.

4.2 Implementation

4.2.1 *Experiments Setting*

We feed the input q_t and (q_t, a_t) to the network using their ID tag, where $ID(q_t) = q_t \in \{1, 2, \ldots, Q\}$ and $ID(q_t, a_t) = q_t + a_t * Q \in \{1, 2, \ldots, 2Q\}$ if there are Q distinct KCs. The IDs of q_t and (q_t, a_t) are used to lookup the embedding vectors in the the KC embedding matrix $\mathbf{A}$ and the KC-response embedding matrix $\mathbf{B}$, respectively.

Prior to training, the KC embedding matrix $\mathbf{A}$ and the KC-response embedding matrix $\mathbf{B}$, as well as the key memory matrix $\mathbf{M}^k$, the value memory matrix $\mathbf{M}^v$ and other model parameters $\mathbf{W}$ and $\mathbf{b}$, are initialized randomly from a Gaussian distribution with zero mean and a small standard deviation. All of these model parameters are learned during the training process by minimizing the cross-entropy loss:

$$\mathcal{L} = -\sum_t \left(a_t \log p_t + (1 - a_t) \log(1 - p_t) \right). \tag{15}$$

We learn the model using the Adam optimization with a learning rate of 0.003 and a batch size of 32. We also consistently set the norm clipping threshold to 10.0 in order to avoid gradient exploding. In addition, since all of the input sequences are of different lengths, all sequences, except the Synthetic dataset, are set to be a length of 200. For sequences that have less than 200 time-steps, they are padded with zeros to fill up the remaining time steps. Accordingly, masking is also applied when computing the loss.

We implement the models using Tensorflow on a computer with a single NVIDIA RTX 2080 GPU.

4.2.2 *Hyperparameter Selection and Evaluation*

Of all the datasets, 30% of the sequences are held out as a test set and the remaining 70% are used as a training set. Furthermore, five-fold cross-validation is applied on the training set to perform hyperparameter selection. The

[2] More information about the 4LittleTrees can be found on https://www.4littletrees.com/.

[3] F.1 (Form 1) means the first grade in the secondary school, which is equivalent to the 7th grade elsewhere. Thus, F.1 to F.3 is equivalent to 7th to 9th grade.

Table 2: The average test results of the evaluation measures, as well as their standard deviations, from 5 trials are reported. As the PFA model can be learned by a closed-form solution, the learned parameters and thus its performance are the same in every trail. Therefore, the standard deviation is not reported for the PFA model.

Dataset	PFA			DKT			DKVMN			Deep-IRT		
	AUC	Acc	Loss	AUC	Acc	Loss	AUC	Acc	Loss	AUC	Acc	Loss
ASSIST2009	59.68	69.24	7.08	81.56 ± 0.18	**77.17 ± 0.04**	**5.26 ± 0.01**	81.61 ± 0.06	77.01 ± 0.04	5.29 ± 0.01	**81.65 ± 0.02**	77.00 ± 0.06	5.30 ± 0.01
ASSIST2015	52.85	73.37	6.13	72.85 ± 0.05	**75.29 ± 0.02**	**5.69 ± 0.01**	**72.94 ± 0.06**	75.18 ± 0.03	5.71 ± 0.01	72.88 ± 0.07	75.14 ± 0.02	5.72 ± 0.01
Statics2011	64.99	79.85	4.64	82.71 ± 0.18	81.37 ± 0.08	4.29 ± 0.02	**83.17 ± 0.11**	**81.57 ± 0.05**	**4.24 ± 0.01**	83.09 ± 0.12	81.56 ± 0.04	4.24 ± 0.01
Synthetic	61.68	65.20	8.01	81.65 ± 0.06	74.84 ± 0.09	5.79 ± 0.02	82.97 ± 0.06	75.58 ± 0.07	5.62 ± 0.02	**82.98 ± 0.07**	**75.61 ± 0.04**	**5.61 ± 0.01**
FSAI-F1toF3	54.52	54.57	10.46	**69.42 ± 0.34**	**64.11 ± 0.29**	**8.26 ± 0.07**	68.40 ± 0.89	63.40 ± 0.15	8.42 ± 0.03	68.69 ± 0.28	63.43 ± 0.24	8.42 ± 0.06

Table 3: The corresponding hyperparameter settings of the models reported in Table 2.

Dataset	DKT		DKVMN			Deep-IRT		
	s. dim	# params	N	s. dim	# params	N	s. dim	# params
ASSIST2009	100	140, 011	10	50	27, 801	20	100	67, 602
ASSIST2015	200	341, 901	100	50	35, 301	20	50	27, 352
Statics2011	100	1, 142, 824	5	200	836, 701	5	100	398, 502
Synthetic	200	251, 851	50	100	55, 501	20	50	19, 852
FSAI-F1toF3	100	2, 082, 567	10	100	712, 301	5	50	350, 752

network architectures of the DKT, DKVMN and Deep-IRT models are varied with different numbers of state dimension and memory size. A grid search is applied over the combinations of state dimension and memory size.

As for the DKT model, the hidden layer size is chosen from {10, 50, 100, 200}. For the DKVMN and Deep-IRT models, the dimensions of key memory matrix (d_k) and the value memory matrix (d_v) are chosen from {10, 50, 100, 200} as well. We also choose the memory dimension N for the DKVMN and Deep-IRT models from {5, 10, 20, 50, 100}. As the search space is large if we perform a grid search over all possible combinations of N, d_k and d_v, we therefore set $d_k = d_v$ for simplicity. Lastly, we use the combination that results in the smallest cross-entropy loss to retrain the model with the entire training set and evaluate the model performance on the test set.

To report the model performance, we run the training and evaluation process for 5 times. The average and standard deviation of area under the ROC curve (AUC), the accuracy and the cross-entropy loss are reported. AUC provides a robust metric for binary prediction evaluation. When interpreting the value of AUC, the larger the AUC score is, the better the prediction performance is of the model. An AUC score of 0.5 indicates that the model performance is merely as good as random guess.

4.3 Results

The model performance of the experiment is shown in Table 2 and the corresponding hyperparameters are stated in Table 3. In addition, we also include the performance of the PFA model in Table 2 as a baseline model for reference. It is noted that the model performance of the DKT model in our experiment is better than the one reported in [22], even thought we adopt the same processed data provided by Zhang et al. [22]. This might be attributed to different random seeds and deep learning libraries that are used to train the model.

For the ASSIST2009 and ASSIST2015 datasets, the DKT model achieves the highest accuracy and the smallest loss, while the Deep-IRT model and the DKVMN model result in the highest AUC score in the ASSIST2009 and the ASSIST2015, respectively. As for the Statics2011 dataset, the DKVMN model obtains the highest AUC score, the highest accuracy and the smallest loss. Regarding to the Synthetic dataset, the Deep-IRT results in the best performance in AUC, accuracy and loss. On the FSAI-F1toF3 dataset, the DKT model performs the best. All in all, the DKT, DKVMN and Deep-IRT models have a similar performance.

In addition, we conduct two-tailed independent t-tests on each dataset and each evaluation measure between the Deep-IRT model and the DKVMN model. We found that the difference between their performance is not significant for majority of the datasets. The p-values of the t-tests are reported as follows in Table 4:

Table 4: The p-values of the independent t-tests between the DKVMN model and the Deep-IRT model.

Dataset	p-value		
	AUC	Accuracy	Loss
ASSIST2009	0.2581	0.7061	0.7061
ASSIST2015	0.1695	0.0472	0.0472
Statics2011	0.3239	0.7836	0.7836
Synthetic	0.7565	0.3907	0.3907
FSAI-F1toF3	0.5119	0.8101	0.8101

Although we cannot claim that their performance is, more or less, the same based on the large p-value, this result, however, might imply that the Deep-IRT model potentially retains the performance of the DKVMN model.

5. DISCUSSION

5.1 Going Deeper in Difficulty Level

To evaluate the KC difficulty estimated from the Deep-IRT model, we compare the difficulty level learned for the FSAI-F1toF3 dataset with four other sources. The reason why we

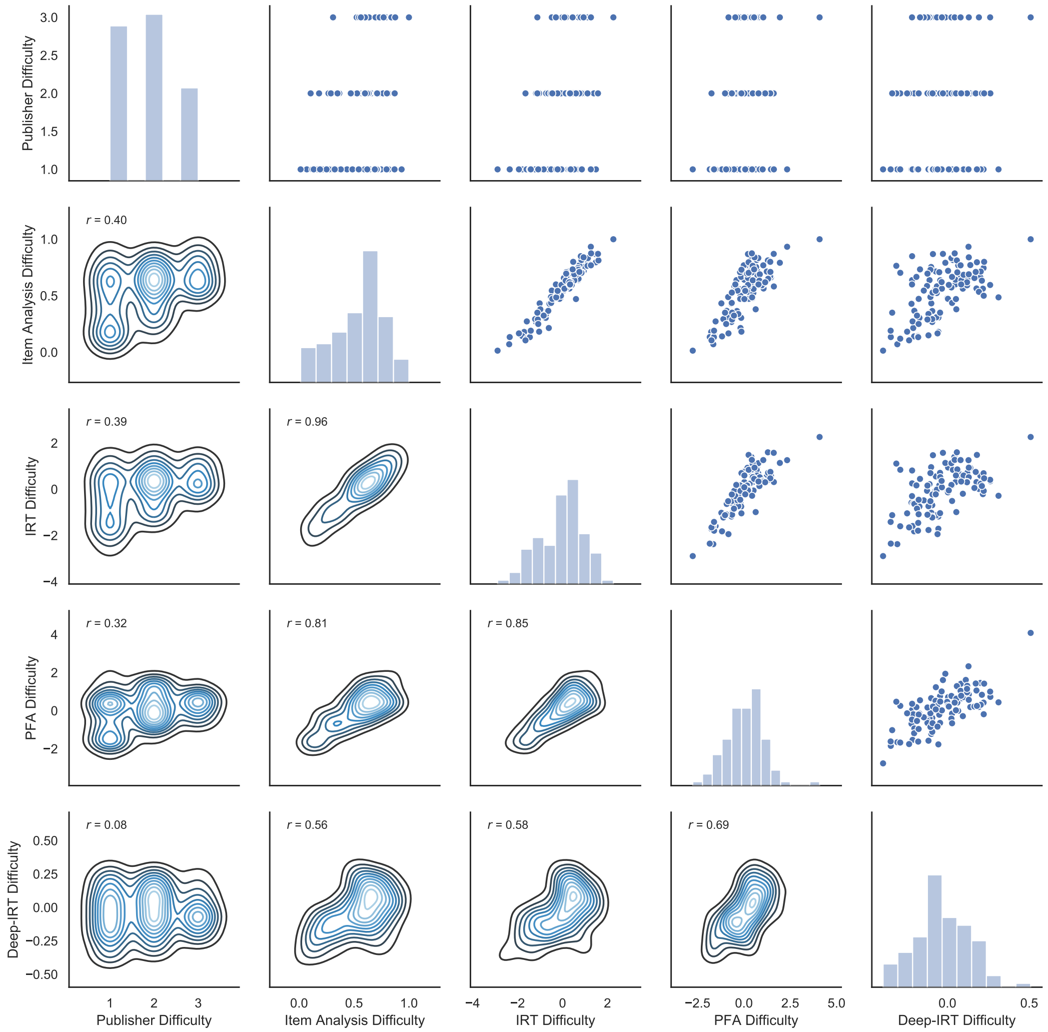

Figure 3: Pairwise comparison of the difficulty level obtained from different sources. The positions in the pairs plot are ordered according to the evaluation setting and the complexity of getting the difficulty level.

use the proprietary dataset is that we have the individual questions' difficulty level provided by the publisher. Each question is associated with a difficulty level in $\{1, 2, 3\}$ which represents easy, medium and hard, respectively.

The second source of difficulty level is calculated according to the item analysis [1]. The difficulty level of the item analysis is the percentage of students who answer a question correctly in a test environment. Yet, to be consistent with interpretation with other models, we adopt the percentage of students who answer a question *incorrectly*. Furthermore, as our dataset is not collected from a test environment, a student can answer a same question multiple times. Thus, we only adopt the student's first attempt when calculating the difficulty level. Moreover, we only consider the question on which at least 10 distinct students has answered.

The third source of difficulty level is learned by the one-parameter IRT model. We also use the student's first attempt, only, on a question to learn the IRT model for the sake of avoiding multiple attempts on the same question. Moreover, we adopt $\sigma(\theta_i - \beta_j)$ to be the item response function when learning the IRT model.

Lastly, we trained a PFA model to extract the difficulty level in a knowledge tracing setting, rather than a test environment setting. In other words, we use the entire student's learning trajectory to learn the question's difficulty level.

Since there are more than two thousands questions in this dataset, we only evaluate a set of questions that belong to a subset of skills. This subset contains five skills that constitute around a fifth of the interactions in the FSAI-F1toF3 dataset, and has in total 131 questions. These skills are "Significant Figures", "Approximation and Errors in Measurement", "Index Notation", "Laws of Indices" and "Polynomials". We visualize the difficulty level obtained from different sources in a pairs plot in Figure 3 with the Pearson correlation r stated in the lower triangular part of the pairs plot. The positions in the pairs plot are ordered according to the evaluation setting and the complexity of getting the difficulty level.

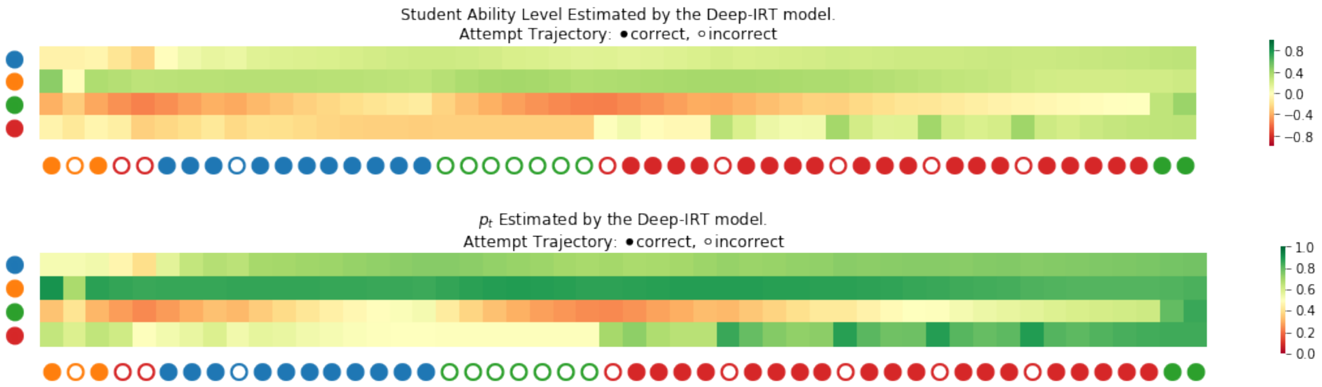

Figure 4: An example of a student's learning trajectory from the ASSIST2009 dataset. The student ability (top) and the probability prediction (bottom) are shown. The labels in the vertical dimension correspond to different skill tags. They are, from top to bottom, "equation solving two or fewer steps" (blue), "ordering factions" (orange), "finding percents" (green) and "equation solving more than two steps" (red), respectively. The learning trajectory is shown in the horizontal dimension by filled or hollow circles with corresponding colors. The filled and the hollow circles represent a correct response and an incorrect response, respectively.

The pairs plot reveals that the difficulty level learned from the Deep-IRT model aligns with most of the other sources with a strong correlation, except for the difficulty level provided by the publisher. Moreover, it is observed that the more similar the models' evaluation setting and complexity are, the higher the Pearson correlation is between the models. For example, the Pearson correlation between the difficulty level from the item analysis model and the IRT model is 0.96, while the Pearson correlation between the item analysis model and the Deep-IRT model is 0.56. Furthermore, it is observed that the difficulty level provided by the publisher is moderately correlated to the one obtained from the item analysis model (0.40) and the IRT model (0.39), but weakly correlated to the one obtained from the Deep-IRT model (0.08). Thus, it would be interesting to examine whether the difficulty level inferred from the Deep-IRT model would be more accurate than other traditional models.

5.2 Going Deeper in Student Ability

As stated in [20], there are two problems exist in the DKT model. The first one is that the DKT model fails to reconstruct the observed input. This means that the estimated performance of a student decreases even if the student makes a successful attempt, and vice versa. The second problem is that the estimated performance of different KCs is not consistent over time. This means that the student mastery level alternates between mastered and not-yet-mastered during the model influence. These two behaviors are undesirable and therefore we would like to examine if these problems exist in the Deep-IRT model.

We randomly select a student in the ASSIST2009 dataset and then evaluate the transition of the student's ability level and the prediction probability of getting the next KC correctly in the next time-step during the learning trajectory by visualizing the first 50 attempts of that student (shown in Figure 4). In the first 50 attempts, the student has attempted the skills of "equation solving two or fewer steps" (shown in blue), "ordering factions" (orange), "finding percents" (green) and "equation solving more than two steps" (red).

As observed from the figure, the prediction transition is smooth while there are some counter intuitive prediction. For example, when the student answers "equation solving more than two steps" (red) incorrectly, the corresponding estimated student ability level and the prediction probability increase, instead. Moreover, it seems that the model cannot identify the prerequisite relationship among the skills. For instance, when the student is answering the skill "equation solving two or fewer steps" (blue) correctly, the estimated student ability level and the prediction probability of the skill "equation solving more than two steps" (red) decrease. However, it is expected if the student is doing well in the skill "equation solving two or fewer steps" (blue), it should, at least, does not have an adverse effect on the skill "equation solving more than two steps" (red). Thus, it would be interesting to investigate methods to make deep learning based knowledge tracing model reliable, e.g., the prediction-consistent regularization proposed in [20].

6. CONCLUSION

In this paper, we propose the Deep-IRT model which empowers the deep learning based knowledge tracing model with explaniability. Experiments show that the Deep-IRT model retains the performance of the deep learning based knowledge tracing model while simultaneously being able to estimate the KC difficulty level and the student ability level over time. Moreover, the difficulty level estimated by the Deep-IRT model aligns with the difficulty level obtained by other traditional methods, e.g., the IRT model and the item analysis. Thus, it potentially provides an alternative way to estimate KC's difficulty level by utilizing the entire learning trajectory, rather than the traditional educational testing environment. However, there are still rooms for improvement on learning the inherent KC dependency. One of the potential ways is to learn a better KC vector representation by incorporating the content of questions [16].

References

[1] Understanding Item Analyses. Office of Educational Assessment, University of Washington. Retrieved February 11, 2019 from http://www.washington.edu/assessment/scanning-scoring/scoring/reports/item-analysis/.

[2] Lee Averell and Andrew Heathcote. The form of the forgetting curve and the fate of memories. *Journal of Mathematical Psychology*, 55(1):25–35, 2011.

[3] Hao Cen, Kenneth R. Koedinger, and Brian Junker. Learning factors analysis – a general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference in Intelligent Tutoring Systems*, pages 164–175, Berlin, Heidelberg, 2006. Springer.

[4] Albert T. Corbett and John R. Anderson. Knowledge tracing: modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4 (4):253–278, March 1995.

[5] April Galyardt and Ilya Goldin. Move your lamp post: Recent data reflects learner knowledge better than older data. *Journal of Educational Data Mining*, 7(2):83–108, 2015.

[6] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio G. Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016.

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

[8] Mohammad M. Khajah, Yun Huang, José P. González-Brenes, Michael C. Mozer, and Peter Brusilovsky. Integrating knowledge tracing and item response theory: A tale of two frameworks. In *Proceedings of the 4th International Workshop on Personalization Approaches in Learning Environment*, pages 7–15, 2014.

[9] Mohammad M. Khajah, Robert V. Lindsey, and Michael C. Mozer. How deep is knowledge tracing. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 94–101, 2016.

[10] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[11] Zachary A. Pardos and Neil T. Heffernan. Modeling individualization in a Bayesian networks implementation of knowledge tracing. In *Proceedings of the 18th International Conference on User Modeling, Adaptation, and Personalization*, volume 6075, pages 255–266, Berlin, Heidelberg, 2010. Springer.

[12] Zachary A. Pardos and Neil T. Heffernan. KT-IDEM: Introducing item difficulty to the knowledge tracing model. In *Proceedings of the 19th International Conference on User Modeling, Adaption and Personalization*, volume 6787, pages 243–254. Springer, 2011.

[13] Philip I. Pavlik, Hao Cen, and Kenneth R. Koedinger. Performance factors analysis – a new alternative to knowledge tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, pages 531–538, Amsterdam, Netherlands, 2009. IOS Press.

[14] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J. Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pages 505–513, 2015.

[15] Georg Rash. Probabilistic models for some intelligence and attainment tests. *Copenhagen: Danish Institute for Educational Research*, 1960.

[16] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris Ding, Si Wei, and Guoping Hu. Exercise-enhanced sequential modeling for student performance prediction. In *Proceedings of 32nd AAAI Conference on Artificial Intelligence*, pages 2435–2443, 2018.

[17] Hao Wang and Dit-Yan Yeung. Towards Bayesian deep learning: A framework and some existing methods. *IEEE Transactions on Knowledge and Data Engineering*, 28(12):3395–3408, 2016.

[18] Kevin H. Wilson, Yan Karklin, Bojian Han, and Chaitanya Ekanadham. Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 539–544, 2016.

[19] Frances M. Yang and Solon T. Kao. Item response theory for measurement validity. *Shanghai Archives of Psychiatry*, 26(3):171–177, 2014.

[20] Chun-Kit Yeung and Dit-Yan Yeung. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the 5th ACM Conference on Learning @ Scale*, pages 5:1–5:10. ACM, 2018.

[21] Michael V. Yudelson, Kenneth R. Koedinger, and Geoffrey J. Gordon. Individualized Bayesian knowledge tracing models. In *Proceedings of the 16th International Conference on Artificial Intelligence in Education*, volume 7926, pages 171–180, Berlin, Heidelberg, 2013. Springer.

[22] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*, pages 765–774. International World Wide Web Conferences Steering Committee, 2017.