Addressing Two Problems in Deep Knowledge Tracing via Prediction-Consistent Regularization

Chun-Kit Yeung
Hong Kong University of Science and Technology
ckyeungac@cse.ust.hk

Dit-Yan Yeung
Hong Kong University of Science and Technology
dyyeung@cse.ust.hk

ABSTRACT

Knowledge tracing is one of the key research areas for empowering personalized education. It is a task to model students' mastery level of a knowledge component (KC) based on their historical learning trajectories. In recent years, a recurrent neural network model called deep knowledge tracing (DKT) has been proposed to handle the knowledge tracing task and literature has shown that DKT generally outperforms traditional methods. However, through our extensive experimentation, we have noticed two major problems in the DKT model. The first problem is that the model fails to reconstruct the observed input. As a result, even when a student performs well on a KC, the prediction of that KC's mastery level decreases instead, and vice versa. Second, the predicted performance for KCs across time-steps is not consistent. This is undesirable and unreasonable because student's performance is expected to transit gradually over time. To address these problems, we introduce regularization terms that correspond to *reconstruction* and *waviness* to the loss function of the original DKT model to enhance the consistency in prediction. Experiments show that the regularized loss function effectively alleviates the two problems without degrading the original task of DKT.[1]

ACM Classification Keywords

I.2.6 Artificial Intelligence: Learning; K.3.m Computer and Education: Miscellaneous

Author Keywords

Knowledge tracing; Deep learning; Regularization; Educational data mining; Personalized learning; Sequence modeling

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 skill, concept, exercise, 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 (KT) is considered to be important for enhancing personalized learning. KT is the task of modeling student's *knowledge state*, which represents the mastery level of KCs, based on historical data. 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.

Generally, the KT task can be formalized as follows: given a 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 KT, 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, KT usually seeks to predict the probability that the student will answer the question correctly during the next time-step, i.e., $p(a_{t+1} = 1 | q_{t+1}, \mathbf{X}_t)$. Many approaches have been developed to solve the KT problem, such as using the hidden Markov model (HMM) in Bayesian knowledge tracing (BKT) [2] and the logistic regression model in performance factors analysis (PFA) [11]. More recently, a recurrent neural network (RNN) model has been applied in a method called deep knowledge tracing (DKT) [12]. Experiments show that DKT outperforms traditional methods without requiring substantial feature engineering by humans.

Although DKT achieves impressive performance for the KT task, we have noticed two major problems in the prediction results of DKT when trying to replicate the experiments in [12] (where the authors adopted the skill-level tag as the question tag.) These two problems are illustrated using a heatmap in Figure 1, which visualizes the predicted knowledge state at each time-step of a student (namely id-1) from the ASSISTment 2009 skill builder dataset.

The first problem of DKT is that the DKT model fails to reconstruct the input information in prediction. When a student

[1]The implementation of this work is available on https://github.com/ckyeungac/deep-knowledge-tracing-plus.

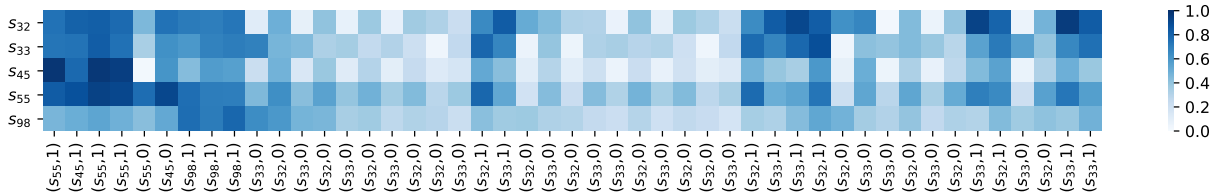


Figure 1: A student, namely id-1, from the ASSISTment 2009 dataset is used to plot the heatmap illustrating the two problems in DKT. The label s_i in the vertical dimension corresponds to the skill tag, and only those which have been answered by the student are shown. The label in the horizontal dimension refers to the input fed into the DKT at each time-step. The color of the heatmap indicates the predicted probability that the student will answer s_i correctly in the next time-step, i.e., $p(a_{t+1} = 1 \mid q_{t+1} = s_i)$. The darker the color, the higher the probability.

performs well in a learning task related to a skill s_i, the model's predicted performance for the skill s_i may drop instead, and vice versa. For example, at the 6^{th} time-step in Figure 1, the probability of correctly answering the exercise related to s_{45} increases compared to the previous time-step even though the student answered s_{45} incorrectly.

Second, it is observed that the transition in prediction outputs, i.e., the student's knowledge states, across time-steps is not consistent. As depicted in Figure 1, there are sudden surges and plunges in the predicted performance of some skills across time-steps. For example, the probabilities of correctly answering s_{32}, s_{33}, s_{45}, and s_{55} fluctuate when the student answers s_{32} and s_{33} in the middle of the learning sequence. This is intuitively undesirable and unreasonable as students' knowledge state is expected to transit gradually over time, but not to alternate between mastered and not-yet-mastered. Such wavy transitions are therefore not favorable as it would mislead the interpretation of the student's knowledge state.

To address the problems described above, we propose to augment the original loss function of the DKT model by introducing additional quality measures other than the original one which solely considers the prediction accuracy of the next interaction. Specifically, we define the *reconstruction* error and *waviness* measures and use them to augment the original loss function as a regularized loss function. Experiments show that the regularized DKT is more accurate in reconstructing the answer label of the observed input and is more consistent in its prediction across time-steps, yet without sacrificing the prediction accuracy for the next interaction.

Our main contributions are summarized as follows:

- Two problems in DKT that have not been revealed in the literature are raised: failure in current observation reconstruction and wavy prediction transition;
- Three regularization terms for enhancing the consistency of prediction in DKT are proposed: r to address the reconstruction problem, and w_1 and w_2 to address the wavy prediction transition problem;
- Five other performance measures are proposed to evaluate three aspects of goodness in KT: AUC(C) for the prediction performance of the current interaction, w_1 and w_2 for the waviness in KT's prediction overall, and m_1 and m_2 for the consistency between the current observation and the corresponding change in prediction.

BACKGROUND

Researchers have been investigating mathematical and computational models to tackle the KT task since the 1990s. Various approaches, ranging from probabilistic models to deep neural networks, have been developed over the past two decades.

The Bayesian knowledge tracing (BKT) model was proposed in [2] during the 1990s. It models the knowledge states of KCs using one HMM for each KC. Specifically, the hidden state in the HMM represents the student's knowledge state which indicates whether or not the KC is mastered. However, many simplifying assumptions adopted by BKT are unrealistic. For example, BKT assumes that forgetting does not occur and the KCs are mutually independent. To address these shortcomings, some variants of BKT such as those with forgetting power [4] and skill dependency [5] have been proposed. Extensive works have also been conducted to empower the individualization of BKT on both skill-specific parameters [9, 10] and student-specific parameters [20]. Some other attempts have been made to extend the capabilities of BKT on the partial score [17], sub-skills and temporal features [3], and more features from cognitive science such as the recency effect and contextualized trial sequences [6]. However, it should be noted that such extensions often require considerable feature engineering efforts and may incur a significant increase in the computational requirements.

In the 2000s, learning factors analysis (LFA) [1] was proposed to model student knowledge states using the logistic regression model to deal with the multi-KCs issue and to incorporate student ability into the model. There is a reconfiguration of the LFA, called performance factors analysis (PFA) [11], which offers higher sensitivity to student performance rather than student ability. Both LFA and PFA exploit the number of successes or failures of applying a KC to predict whether a student has acquired the knowledge about the KC. Although both LFA and PFA can handle a learning task that is associated with multiple KCs, they cannot deal with the inherent dependency among KCs, e.g., "addition" is a prerequisite of "multiplication". Moreover, the features used in LFA and PFA are relatively simple and they cannot provide a deep insight into students' latent knowledge state.

Recently, with a surge of interest in deep learning models, DKT [12], which models student's knowledge state based on an RNN, has been shown to outperform the traditional models, such as BKT and PFA, without the need for human-engineered features such as recency effect, contextualized trial sequence, inter-skill relationship, and students' ability variation [6]. Since the DKT was proposed, a few comprehensive studies have been reported to compare DKT with other KT models [18, 19] or to apply the ideas of DKT to other applications [14, 16, 15]. Nevertheless, to the best of our knowledge, all such attempts in the literature evaluate the DKT model mainly with respect to the prediction of the next interaction based on the *area under the ROC curve* (AUC) measure, without considering other quality aspects in the prediction result.

Review of Deep Knowledge Tracing

Recurrent Neural Networks

DKT employs the RNN as its backbone model (see Figure 2). A (vanilla) RNN [7] aims to map an input sequence $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$ to an output sequence $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T)$. To map the input to the output, the input vector undergoes a series of transformations via a hidden layer, which captures useful information that is hard to human-engineer, and forms a sequence of hidden states $(\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_T)$. More concretely, at time-step t, the hidden state $\mathbf{h}_t$ is the encoding of the past information obtained up to time-step $t-1$, i.e., $\mathbf{h}_{t-1}$, and the current input $\mathbf{x}_t$. The input-to-hidden transformation and hidden-to-output transformation can be stated mathematically as follows:

$$\mathbf{h}_t = \tanh(\mathbf{W}_{hx}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h) \quad (1)$$

$$\mathbf{y}_t = \sigma(\mathbf{W}_{hy}\mathbf{h}_t + \mathbf{b}_y) \quad (2)$$

where both the hyperbolic tangent $\tanh(\cdot)$ and the sigmoid function $\sigma(\cdot)$ are applied in an element-wise manner. The model is parameterized by a weight matrix $\mathbf{W}$ and a bias vector $\mathbf{b}$ with appropriate dimensions.

Piech et al. [12] adopts an RNN variant with long short-term memory (LSTM) cells. An LSTM cell incorporates three gates to imitate the human memory system [8] so as to calculate the hidden state $\mathbf{h}_t$. The three gates are forget gate $\mathbf{f}_t$, input gate $\mathbf{i}_t$ and output gate $\mathbf{o}_t$, which control a memory cell state $\mathbf{c}_t$. Mathematically, they are simply three vectors calculated based on the current input $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$:

$$\begin{aligned}
\mathbf{f}_t &= \sigma(\mathbf{W}_f[\mathbf{x}_t, \mathbf{h}_{t-1}] + \mathbf{b}_f), \\
\mathbf{i}_t &= \sigma(\mathbf{W}_i[\mathbf{x}_t, \mathbf{h}_{t-1}] + \mathbf{b}_i), \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o[\mathbf{x}_t, \mathbf{h}_{t-1}] + \mathbf{b}_o),
\end{aligned}$$

where $[\cdot]$ denotes concatenation. Different gates play different roles to control what information should be stored in $\mathbf{c}_t$. The forget gate $\mathbf{f}_t$ decides what information to forget from the previous memory cell state $\mathbf{c}_{t-1}$, while the input gate $\mathbf{i}_t$ decides what new information $\tilde{\mathbf{c}}_t$ should be added to the recent cell state $\mathbf{c}_t$. Thus, the recent cell state $\mathbf{c}_t$ depends on the previous cell state after forgetting and the new information added from the input gate. Eventually, the output gate $\mathbf{o}_t$ determines what information should be extracted from $\mathbf{c}_t$ to form the hidden state $\mathbf{h}_t$. These can be expressed mathematically as follows:

$$\begin{aligned}
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_c[\mathbf{x}_t, \mathbf{h}_{t-1}] + \mathbf{b}_c), \\
\mathbf{c}_t &= \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \tilde{\mathbf{c}}_t, \\
\mathbf{h}_t &= \mathbf{o}_t \otimes \tanh(\mathbf{c}_t),
\end{aligned} \quad (3)$$

where $\otimes$ denotes element-wise multiplication. This formulation empowers RNN to store information occurred in a distance history, and thus has a more powerful capability than the vanilla RNN. The unfolded RNN architecture is visualized in Figure 2, with a high-level interpretation of DKT.

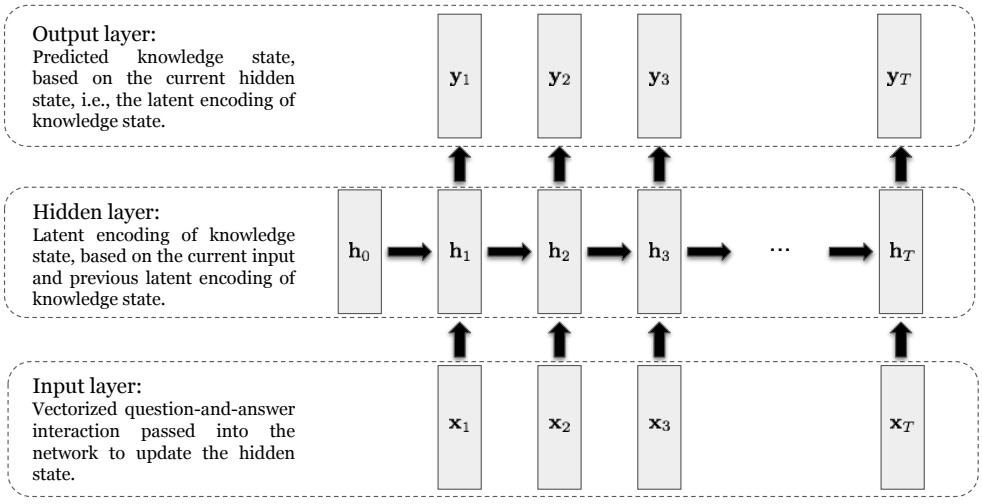


Figure 2: Unfolded version of the RNN architecture for DKT. The hidden state is processed differently in the vanilla RNN (eq. (1)) or LSTM-RNN (eq. (3)). $\mathbf{h}_0$ is the initial hidden state in the RNN, and it is usually initialized randomly or to a zero vector.

DKT Problem Formulation

To train a DKT model, an interaction (q_t, a_t) needs to be transformed into a fixed-length input vector $\mathbf{x}_t$. As a question can be identified by a unique ID, it can be represented using one-hot encoding as a vector $\boldsymbol{\delta}(q_t)$. The corresponding answer label can also be represented as the one-hot vector $\boldsymbol{\delta}(q_t)$ of the corresponding question if a student answers it correctly, or a zero vector $\mathbf{0}$ otherwise. Therefore, if there are M unique questions, then $\mathbf{x}_t \in \{0, 1\}^{2M}$.

After the transformation, DKT passes the $\mathbf{x}_t$ to the hidden layer and computes the hidden state $\mathbf{h}_t$ using the vanilla RNN or LSTM-RNN. As the hidden state summarizes the information from the past, the hidden state in the DKT can therefore be conceived as the latent knowledge state of student 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 question correctly. For student i, if she has a sequence of question-and-answer interactions of length T_i, the DKT model maps the inputs $(\mathbf{x}_1^i, \mathbf{x}_2^i, \ldots, \mathbf{x}_{T_i}^i)$ to the outputs $(\mathbf{y}_1^i, \mathbf{y}_2^i, \ldots, \mathbf{y}_{T_i}^i)$ accordingly.

The objective of the DKT is to predict the next interaction performance, so the target prediction will be extracted by performing a dot product of the output vector $\mathbf{y}_t^i$ and the one-hot encoded vector of the next question $\boldsymbol{\delta}(q_{t+1}^i)$. Based on the predicted output and the target output a_{t+1}^i, the loss function $\mathcal{L}$ can be expressed as follows:

$$\mathcal{L} = \frac{1}{\sum_{i=1}^{n}(T_i - 1)} \left(\sum_{i=1}^{n} \sum_{t=1}^{T_i - 1} l\left(\mathbf{y}_t^i \cdot \boldsymbol{\delta}(q_{t+1}^i), a_{t+1}^i\right) \right) \quad (4)$$

where n is the number of students and $l(\cdot)$ is the cross-entropy loss.

SOME PROBLEMS OF DKT AND THEIR REMEDIES

While we were replicating the experiments on the original DKT proposed in [12] using the skill builder dataset provided by ASSISTment in 2009 (denoted ASSIST2009)[2], we noticed two major problems in the prediction results of DKT. First, it sometimes fails to reconstruct the input observation because the prediction result is counterintuitive. When a student answers a question of skill s_i correctly/incorrectly, the predicted probability of that student answering s_i correctly sometimes decreases/increases instead. Second, the predicted knowledge state is wavy and inconsistent over time. This is not desirable because student's knowledge state is expected to transit only gradually and steadily over time. Therefore, we propose three regularization terms to rectify the consistency problem in the prediction of DKT: reconstruction error r to resolve the reconstruction problem, and waviness measures w_1 and w_2 to smoothen the predicted knowledge state transition.

Reconstruction Problem

As we saw from Figure 1, when the student answers s_{32} incorrectly, the probability of correctly answering s_{32} grows significantly compared to the previous time-step. This problem can be attributed to the loss function defined in the DKT model (eq. (4)). Specifically, the loss function takes only the predicted performance of the next interaction into account, but not the predicted performance of the current one. Accordingly, when the input order $((s_{32}, 0), (s_{33}, 0))$ occurs frequently enough, the DKT model will tend to learn that if a student answers s_{32} incorrectly, he/she will likely answer s_{33} incorrectly, but not s_{32}. Consequently, the prediction result is counterintuitive for the current observed input.

However, one might argue that such transition in prediction reveals that s_{32} is a prerequisite of s_{33}.[3] This is because the predicted performance for s_{32} is lower only when the DKT model receives $(s_{33}, 0)$, but it is higher when the DKT model receives $(s_{32}, 0)$. To gainsay the above argument, we are going to impeach by contradiction. We hypothesize that if s_{32} is indeed a prerequisite of s_{33}, then when a student answers s_{32} incorrectly in the current time-step, it is more probable that he/she will answer s_{33} incorrectly in the next time-step, but not vice versa. To verify this hypothesis, Tables 1 and 2 are made to tabulate the frequency counts when s_{32} and s_{33} appear consecutively in different orders. According to the above hypothesis, it is expected that the lower right cell will have a larger value than the lower left cell in Table 1, but not Table 2.

		Next = s_{33}		
		Correct	Incorrect	Total
Current = s_{32}	Correct	1543	159	1702
	Incorrect	81	367	448
	Total	1624	526	2510

Table 1: Correctness matrix when the current skill is s_{32} and the next skill is s_{33}.

		Next = s_{32}		
		Correct	Incorrect	Total
Current = s_{33}	Correct	1362	72	1434
	Incorrect	90	361	451
	Total	1452	433	1885

Table 2: Correctness matrix when the current skill is s_{33} and the next skill is s_{32}.

From Table 1, we can see that if a student answers s_{32} incorrectly in the current time-step, it is more probable that he/she will answer s_{33} incorrectly in the next time-step. However, Table 2 shows that if a student answers s_{33} incorrectly, it is also more probable that he/she will answer s_{32} incorrectly in the next time-step. This means that an inverse dependency also exists and contradicts the above hypothesis, and therefore the statement that s_{32} is a prerequisite of s_{33} becomes questionable. Moreover, the distributions of these two matrices would advocate that s_{32} and s_{33} are likely to be interdependent and acquired simultaneously.

If s_{32} is not a prerequisite of s_{33} and they should be acquired at the same time, then there should be room for improvement to deal with cases like s_{32} and s_{33} in DKT. As mentioned above,

[2]More information about the dataset can be found in https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data.

[3]We note that s_{32} is "Ordering Positive Decimals" and s_{33} is "Ordering Fractions".

the loss function considers only the predicted performance of the next interaction but ignores the current one. An immediate remedy to alleviate the problem is to regularize the DKT model by taking the loss between the prediction and the current interaction into consideration. By doing so, the model will adjust the prediction with respect to the current input. Thus, a regularization term for the reconstruction problem is defined as follows:

$$r = \frac{1}{\sum_{i=1}^{n}(T_i - 1)} \left(\sum_{i=1}^{n} \sum_{t=1}^{T_i - 1} l\left(\mathbf{y}_t^i \cdot \boldsymbol{\delta}(q_t^i), a_t^i\right) \right). \tag{5}$$

Wavy Transition in Prediction

The second problem is the wavy transition in the student's predicted knowledge state. This problem may be attributed to the hidden state representation in the RNN. The hidden state $\mathbf{h}_t$ is determined by the previous hidden state $\mathbf{h}_{t-1}$ and the current input $\mathbf{x}_t$. It summarizes the student's latent knowledge state of all the exercises in one single hidden layer. Although it is difficult to explicate how the elements in the hidden layer influence the predicted performance of the KCs, it is plausible to confine the hidden state representation to be more invariant via regularization over the output layer.

We define two waviness measures w_1 and w_2 as regularization terms to smoothen the transition in prediction:

$$w_1 = \frac{\sum_{i=1}^{n} \sum_{t=1}^{T_i - 1} \|\mathbf{y}_{t+1}^i - \mathbf{y}_t^i\|_1}{M \sum_{i=1}^{n}(T_i - 1)}, \tag{6}$$

$$w_2^2 = \frac{\sum_{i=1}^{n} \sum_{t=1}^{T_i - 1} \|\mathbf{y}_{t+1}^i - \mathbf{y}_t^i\|_2^2}{M \sum_{i=1}^{n}(T_i - 1)}. \tag{7}$$

To quantify how disparate the two prediction vectors are, both $L1$-norm and $L2$-norm are used to measure the difference between the prediction results $\mathbf{y}_t^i$ and $\mathbf{y}_{t+1}^i$. This is similar to the elastic net regularization [22]. The two measures are averaged over the total number of input time-steps and the number of KCs M. Thus, the magnitude of w_1 would be seen as the average value change of each component in the output vector between $\mathbf{y}_t$ and $\mathbf{y}_{t+1}$. The larger the values of w_1 and w_2, the more inconsistent the transitions in the model.

In summary, the original loss function is augmented by incorporating three regularization terms to give the following regularized loss function:

$$\mathcal{L}' = \mathcal{L} + \lambda_r r + \lambda_{w_1} w_1 + \lambda_{w_2} w_2^2 \tag{8}$$

where λ_r, λ_{w_1} and λ_{w_2} are regularization parameters. By training with this new loss function, the DKT model is expected to address the reconstruction and wavy transition problems.

EXPERIMENTS

Implementation

Experiment settings

In the following experiments, 20% of the data is used as a test set and the other 80% is used as a training set. Furthermore, 5-fold cross-validation is applied on the training set to select the hyperparameter setting. The test set is used to evaluate the model, and also to perform early stopping [13]. The weights of the RNN are initialized randomly from a Gaussian distribution with zero mean and small variance. For fair comparison, we follow the hyperparameter setting in [12] even though it may not be optimal. A single-layer RNN-LSTM with a state size of 200 is used as the basis of the DKT model. The learning rate and the dropout rate are set to 0.01 and 0.5, respectively. In addition, we also consistently set the norm clipping threshold to 3.0. Moreover, our preliminary experiment using ASSIST2009 shows that using the exercise tag as the question tag induces data sparsity and results in poor performance[4], so we adopt the skill tag to be the question tag in the following experiment.

Hyperparameter search

We perform hyperparameter search for the regularization parameters λ_r, λ_{w_1} and λ_{w_2}. At first, each parameter is examined separately to identify a range of values giving good results according to some evaluation measures to be explained later. The initial search ranges for λ_r, λ_{w_1} and λ_{w_2} are {0, 0.25, 0.5, 1.0}, {0, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1.0, 3.0, 10.0}, and {0, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1.0, 3.0, 10.0, 30.0, 100.0}, respectively. After narrowing down the range of each parameter, a grid search over combinations of λ_r, λ_{w_1} and λ_{w_2} is conducted. The final search ranges for λ_r, λ_{w_1} and λ_{w_2} are {0, 0.05, 0.10, 0.15, 0.20, 0.25}, {0, 0.01, 0.03, 0.1, 0.3, 1.0}, and {0, 0.3, 1.0, 3.0, 10.0, 30.0, 100.0}, respectively.

Evaluation Measures

The performance of the DKT is customarily evaluated by AUC, which provides a robust metric for binary prediction evaluation. An AUC score of 0.5 indicates that the model performance is merely as good as random guess. In this paper, we report not only the AUC for the next performance prediction (named AUC(N) in this paper for clarity) which is tantamount to the evaluation in [12], but also five other quantities with respect to the reconstruction accuracy and consistency of the input observation as well as the waviness of the prediction result.

For the reconstruction accuracy, the AUC for the current performance prediction (called AUC(C)) is used. With regard to the consistency in prediction of the input observation, two more quantities m_1 and m_2 are defined to measure the consistency between the observed input and the change in the corresponding prediction. For a single student i at time t, we define

$$m_{1,t}^i = (-1)^{1-a_t^i} \operatorname{sign}\left((\mathbf{y}_t^i - \mathbf{y}_{t-1}^i) \cdot \boldsymbol{\delta}(q_t^i)\right), \tag{9}$$

$$m_{2,t}^i = (-1)^{1-a_t^i} \left((\mathbf{y}_t^i - \mathbf{y}_{t-1}^i) \cdot \boldsymbol{\delta}(q_t^i)\right), \tag{10}$$

and

$$m_1 = \frac{\sum_{i=1}^{n} \sum_{t=2}^{T_i} m_{1,t}^i}{\sum_{i=1}^{n}(T_i - 1)}, \tag{11}$$

$$m_2 = \frac{\sum_{i=1}^{n} \sum_{t=2}^{T_i} m_{2,t}^i}{\sum_{i=1}^{n}(T_i - 1)}. \tag{12}$$

Accordingly, when the model gives a correct change in prediction with respect to the input, we will obtain positive values for $m_{1,t}^i$ and $m_{2,t}^i$. Otherwise negative values will be obtained.

[4] An AUC of 0.73 if 26,668 exercise IDs are used; an AUC of 0.82 if 124 unique skill IDs are used.

Dataset	λ_r	λ_{w_1}	λ_{w_2}	AUC(N)	AUC(C)	w_1	w_2	m_1	m_2
ASSIST2009	0.0	0.0	0.0	0.8212±0.00023	0.9044±0.00151	0.0830±0.00381	0.1279±0.00535	0.3002±0.01832	0.0156±0.00205
	0.1	0.003	3.0	**0.8227±0.00041**	**0.9625±0.00365**	**0.0229±0.00022**	**0.0491±0.00033**	**0.4486±0.00427**	**0.0573±0.00132**
ASSIST2015	0.0	0.0	0.0	0.7365±0.00045	0.8846±0.00185	0.0282±0.00116	0.0414±0.00162	0.6208±0.00799	0.0476±0.00044
	0.05	0.03	3.0	**0.7371±0.00017**	**0.9233±0.00180**	**0.0124±0.00017**	**0.0210±0.00018**	**0.8122±0.00915**	**0.0591±0.00029**
ASSISTChall	0.0	0.0	0.0	**0.7343±0.00021**	0.7109±0.00579	0.0690±0.00130	0.1045±0.00181	0.1151±0.00920	-0.0055±0.00199
	0.1	0.3	3.0	0.7285±0.00024	**0.8570±0.00175**	**0.0147±0.00053**	**0.0301±0.00091**	**0.3052±0.00729**	**0.0441±0.00039**
Statics2011	0.0	0.0	0.0	0.8159±0.00037	0.7404±0.00556	0.1358±0.00970	0.1849±0.01308	-0.2590±0.01124	-0.0658±0.00502
	0.20	1.0	30.0	**0.8349±0.00029**	**0.9038±0.00431**	**0.0074±0.00023**	**0.0130±0.00016**	**0.4761±0.03587**	**0.0315±0.00303**
Simulated-5	0.0	0.0	0.0	0.8255±0.00034	0.8642±0.00265	0.0426±0.00136	0.0588±0.00199	-0.1512±0.02501	-0.0134±0.00569
	0.20	0.001	10.0	**0.8264±0.00061**	**0.9987±0.00081**	**0.0196±0.00013**	**0.0426±0.00164**	**0.9064±0.01948**	**0.1659±0.00830**

Table 3: The average test results of the evaluation measures, as well as their standard deviations from 3 trials are reported for both the DKT (with $\lambda_r = \lambda_{w_1} = \lambda_{w_2} = 0.0$) and DKT+ models. The hyperparameter setting reported for DKT+ is selected according to the following procedure using 5-fold cross-validation: (1) select the DKT+ models with a lower value of w_1 than DKT, and (2) among them, select the DKT+ with the highest value of AUC(N) + AUC(C) + m_1 + m_2.

A positive value of m_1 indicates that more than half of the predictions comply with the input observations; a zero value implies that the model makes half of the predictions change in the right direction while another half change in a wrong direction; a negative value means that the model makes more than half of the predictions change in a wrong direction. A similar interpretation also holds for m_2 though it takes changes in both the direction and magnitude into account. Accordingly, the higher the values of m_1 and m_2 are, the better the model is from the perspective of consistency in prediction for the current observation.

Besides, the waviness measures w_1 and w_2 are also used as performance measures to quantify the consistency in prediction of the other KCs in the model. We deem that a good DKT model should achieve a high AUC score while maintaining a low waviness value.

Datasets

ASSISTment 2009 (ASSIST2009)

This dataset is provided by the ASSISTments online tutoring platform and has been used in several papers for the evaluation of DKT models. Owing to the existence of duplicated records in the original dataset [19], we have removed them before conducting our experiments. The resulting dataset contains 4,417 students with 328,291 question-answering interactions from 124 skills. Some of the students in this dataset are used for visualizing the prediction result.

ASSISTment 2015 (ASSIST2015)

This dataset contains 19,917 student responses for 100 skills with a total of 708,631 question-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.

ASSISTment Challenge (ASSISTChall)

This dataset has been made available for the 2017 ASSISTments data mining competition. It is richer in terms of the average number of records per student as there are 686 students with 942,816 interactions and 102 skills.

Statics2011

This dataset is obtained from an engineering statics course with 189,927 interactions from 333 students and 1,223 exercise tags. We have adopted the processed data provided by [21] with a train/test split of ratio 70:30, and exercise tags are used.

Simulated-5

Piech et al. [12] also simulated 2000 virtual students' answering trajectories in which the exercises are drawn from five virtual concepts. Each student answers the same sequence of 50 exercises each of which has a single concept $k \in \{1, \ldots, 5\}$ and a difficulty level β, with an assigned ability α_k of solving the task related to the skill k. The probability of a student answering an exercise correctly is defined based on the conventional item response theory as $p(\text{correct}|\alpha, \beta) = c + \frac{1-c}{1+\exp(\beta-\alpha)}$, where c denotes the probability of guessing it correctly and it is set to 0.25.

Results

The experiment results are shown in Table 3 which gives a comparison of DKT models with and without regularization with respect to all of the evaluation measures. For clarity, here the DKT model without regularization is simply denoted as DKT, while the DKT model with regularization is denoted as DKT+.

For the ASSIST2009 dataset, the DKT achieves an average test AUC(N) of 0.8212, while the DKT+ performs slightly better with an AUC(N) of 0.8227. However, for the DKT+, there is a considerable improvement in AUC(C) with an increase from 0.9044 to 0.9625. The waviness quantities also decrease significantly, from 0.0830 to 0.0229 for w_1 and from 0.1279 to 0.0491 for w_2. Moreover, although the DKT has already made half of the predictions change in the right direction, the DKT+ further uplifts the values of m_1 and m_2 from 0.3002 to 0.4486 and from 0.0156 to 0.0573, respectively.

Similar improvements in the evaluation measures are observed in ASSIST2015 as well. The DKT+ retains a similar AUC(N) of 0.7371, compared to that of the DKT. The values of AUC(C), m_1 and m_2 are boosted to 0.9233, 0.8122 and 0.0591, respectively. Moreover, the values of w_1 and w_2 in the DKT+ are only half of those for the DKT.

As for ASSISTChall, although the performance of AUC(N) slightly decreases from 0.7343 to 0.7285 in the DKT+, improvement with respect to the other evaluation criteria is very significant. The DKT+ pushes the AUC(C) from 0.7109 to

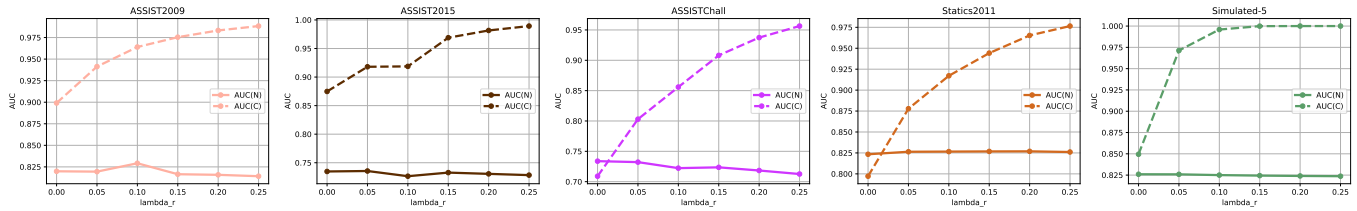


Figure 3: Average test AUC(N) and AUC(C) for different values of λ_r over different combinations of λ_{w_1} and λ_{w_2}.

0.8570 and reduces the w_1 from 0.0690 to 0.0147 and the w_2 from 0.1045 to 0.0301. Moreover, the DKT+ also improves the performance in m_1 and m_2, from 0.1151 to 0.3052 and from −0.0055 to 0.0441, respectively.

For statics2011, a noticeable increase is observed in both AUC(N) and AUC(C), from 0.8159 to 0.8349 and from 0.7404 to 0.9038, respectively. Moreover, w_1 and w_2 shrink from 0.1358 to 0.0074 and from 0.1849 to 0.0130, respectively. This substantial decrease in w_1 and w_2 would be ascribed to the large number of exercises in the dataset since the waviness regularizers act to confine the prediction changes on those exercises which are unrelated to the input. With a potentially substantial amount of unrelated exercises, w_1 and w_2 shrink significantly as a result. The DKT+ also ameliorates the situation that DKT makes more than half of the predictions change in the wrong direction. The values of m_1 and m_2 surge from −0.25952 to 0.47597 and from −0.05090 to 0.05712, respectively.

With regard to Simulated-5, the DKT and the DKT+ result in a similar AUC(N), of 0.8252 and 0.8264, respectively. However, the DKT+ gives a huge improvement in AUC(C), m_1 and m_2. The DKT+ boosts the values of AUC(C) from 0.8642 to 0.9987, m_1 from −0.1512 to 0.9064 and m_2 from −0.0134 to 0.1659. This means the DKT+ model makes almost all of the predictions and the prediction changes for the input exercise correct. Moreover, the waviness in the prediction transition is also reduced.

In summary, the experiment results reveal that the regularization based on r, w_1 and w_2 effectively alleviates the reconstruction problem and the wavy transition in prediction without sacrificing the prediction accuracy for the next interaction. Furthermore, for some combinations of λ_r, λ_{w_1} and λ_{w_2}, the DKT+ even slightly outperforms the DKT in AUC(N).

DISCUSSION

Apart from the experiment results, we plot Figures 3 and 4 to better understand how the regularizers based on reconstruction and waviness affect the performance with respect to different evaluation measures.

In Figure 3, we plot the average test AUC(N) and AUC(C) of different values of λ_r over all combinations of λ_{w_1} and λ_{w_2}. It is observed that the higher the λ_r is, the higher the AUC(C) is achieved for all of the datasets. On the other hand, the AUC(N) generally decreases when the λ_r increases, but the downgrade in AUC(N) is not significant compared with the upgrade in AUC(C). This reveals that the reconstruction regularizer r robustly resolves the reconstruction problem, without sacrificing much of the performance in AUC(N). Moreover, from the result in Table 3, we are usually able to find a combination of λ_r, λ_{w_1} and λ_{w_2} that gives a comparable or even

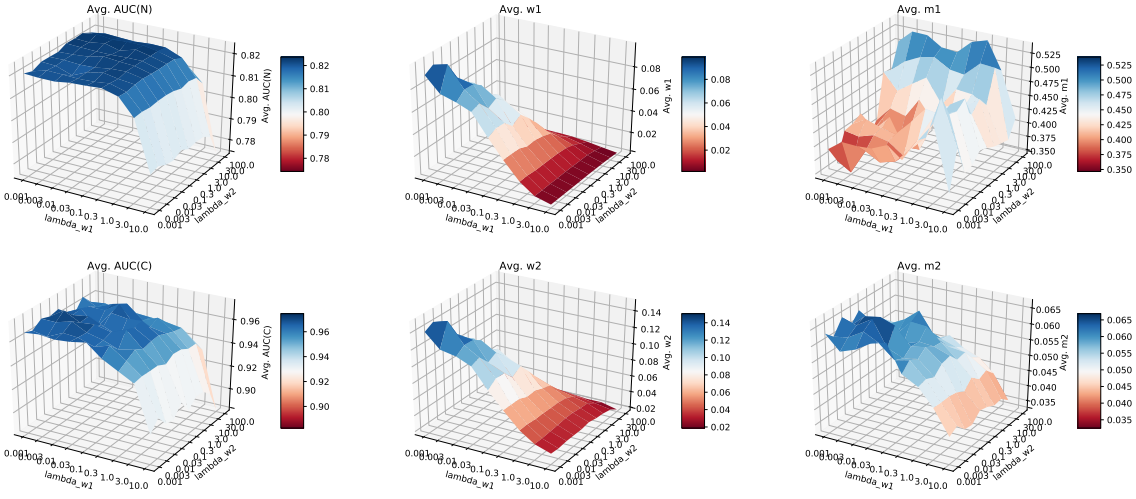


Figure 4: 3-D mesh surface plots of AUC(N), AUC(C), w_1, w_2, m_1 and m_2 of different combinations of λ_{w_1} and λ_{w_2}, with a fixed $\lambda_r = 0.1$ from the ASSIST2009 dataset.

better AUC(N). This implies that the waviness regularizers can help to mitigate the slight degradation in AUC(N) incurred by the reconstruction regularizer.

To also see how the regularization parameters λ_{w_1} and λ_{w_2} affect the evaluation measures, their 3-D mesh plots, with a fixed $\lambda_r = 0.1$, for the ASSIST2009 dataset are shown in Figure 4. The AUC(N) has a relatively flat and smooth surface when λ_{w_1} lies between 0.0 and 1.0 and λ_{w_2} lies between 0.0 and 10.0. Within this region, the DKT+ model also results in a higher AUC(C) value between 0.94 and 0.96. The performance of AUC(N) and AUC(C) starts to decline when λ_{w_1} and λ_{w_2} are larger than 1.0 and 10.0, respectively. It suggests that the model performance has a low sensitivity in AUC(N) and AUC(C) with respect to the hyperparameters λ_{w_1} and λ_{w_2}. As for the waviness measures w_1 and w_2, they decrease in a bell-like shape when λ_{w_1} and λ_{w_2} increase. With regard to the consistency measures, even though the mesh surface is a bit bumpy, m_1 increases along with larger values of λ_{w_1} and λ_{w_2} within the same range mentioned above. This observation implies that both the reconstruction regularizer and the waviness regularizers help to improve the prediction consistency for the current input. On the other hand, m_2 has a decreasing trend with larger values of λ_{w_1} and λ_{w_2}. This is reasonable because the waviness regularizers will reduce the prediction change between the outputs and thus the value of m_2, which takes the change in magnitude into account, is reduced. All in all, the robustness of the regularizers w_1 and w_2 is ascertained thanks to the low sensitivity in the prediction accuracy (AUC(N) and AUC(C)), the observable reduction in the waviness measures (w_1 and w_2), and the improvement in the consistency measures (m_1 and m_2).

In addition to the overall goodness in terms of the evaluation measures, the prediction results of DKT and DKT+ for the student (id-1) are visualized in Figure 5 in order to give a better sense of the regularization's effect on the prediction. Specifically, the prediction results are visualized in two line plots, in addition to the heatmap plot. The first line plot illustrates the change in prediction of all the answered skills of the DKT/DKT+ model, while the second line plot emphasizes the change in prediction of each skill between DKT and DKT+, showing their differences in prediction. Concretely, as for the DKT, it shows a relatively wavy transition of knowledge state across time-steps (Figure 5a). Moreover, the predicted knowledge states of most of the skills share the same directional change in prediction in DKT (Figure 5b). This means that when a student answers a question wrongly, most of the predicted skills' mastery level will decrease simultaneously, and vice versa. However, it is intuitively untrue, as answering skill s_i wrongly does not necessarily lead to a knowledge fade in other skills. On the other hand, the DKT+ demonstrates a smooth prediction transition notably. For example, as seen from Figures 5a and 5c, when the DKT+ receives the inputs $(s_{32}, 0)$ or $(s_{33}, 0)$, the changes in prediction for s_{45}, s_{55} and s_{98} across time-steps are not as wavy as those in the DKT, revealing that DKT+ retains latent knowledge state in RNN for s_{45}, s_{55} and s_{98} from the previous time-steps. This consistent prediction will alleviate the misinterpretation of the student knowledge state caused by the wavy transition problem, and enhance the interpretability of the knowledge state in DKT.

CONCLUSION AND FUTURE WORK

This paper points out two problems which arise when interpreting DKT's prediction results: (1) the reconstruction problem, and (2) the wavy transition in prediction. Both problems are undesirable because it would mislead the interpretation of students' knowledge states. We thereby proposed three regularization terms for enhancing the consistency of prediction in DKT. One of them is the reconstruction error r, which is evaluated in AUC(C), m_1 and m_2. The other two are waviness measures w_1 and w_2, which are the norms for measuring the changes between two consecutive prediction output vectors and are used directly as evaluation measures. Experiments show that these regularizers effectively alleviate the two problems without sacrificing the prediction accuracy (AUC(N)) on the original task which is to predict the next interaction performance.

Although the reconstruction regularizer improves the performance with respect to AUC(C) and the waviness regularizers reduce the waviness in the prediction transition, it is hard to say how low the values of w_1 and w_2 should be in order to qualify for a good KT model. Ideally, a KT model should only change those prediction components which are related to the current input while keeping the other components unchanged or only changed slightly due to some other subtle reasons, e.g., forgetting. Nevertheless, the KC-dependency graphs are different from one dataset to another dataset, so different values of w_1 and w_2 are expected in their ideal KT models.

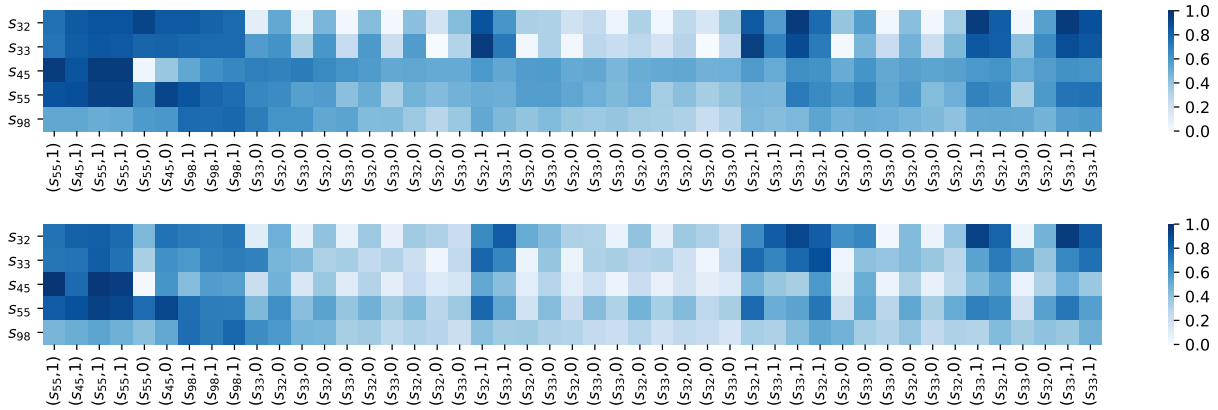
Moreover, more work should be done on improving the stability and accuracy of the prediction for unseen data, more specifically the unobserved KCs. The objective function and the evaluation measures for the DKT+ take only the current and next interactions into account. There is no consideration for interactions in the further future, let alone the evaluation measures for the prediction precision of the unobserved KCs. Yet, unobserved KCs are of vital importance because an ITS should make personalized recommendation on the learning materials for students over not only the observed KCs, but also the unobserved ones. An accurate estimation on the unobserved KCs will help an ITS provide more intelligent pedagogical guidance to students. One of the possible extensions of this work is to take the further future interaction into account when training the DKT model:

$$\mathcal{L} = \frac{1}{c}\left(\sum_{i=1}^{n}\sum_{t=1}^{T_i-1}\sum_{j=1}^{T_i-t}\gamma^{j-1} l\left(\mathbf{y}_t^i \cdot \boldsymbol{\delta}(q_{t+j}), a_{t+j}\right)\right) \tag{13}$$

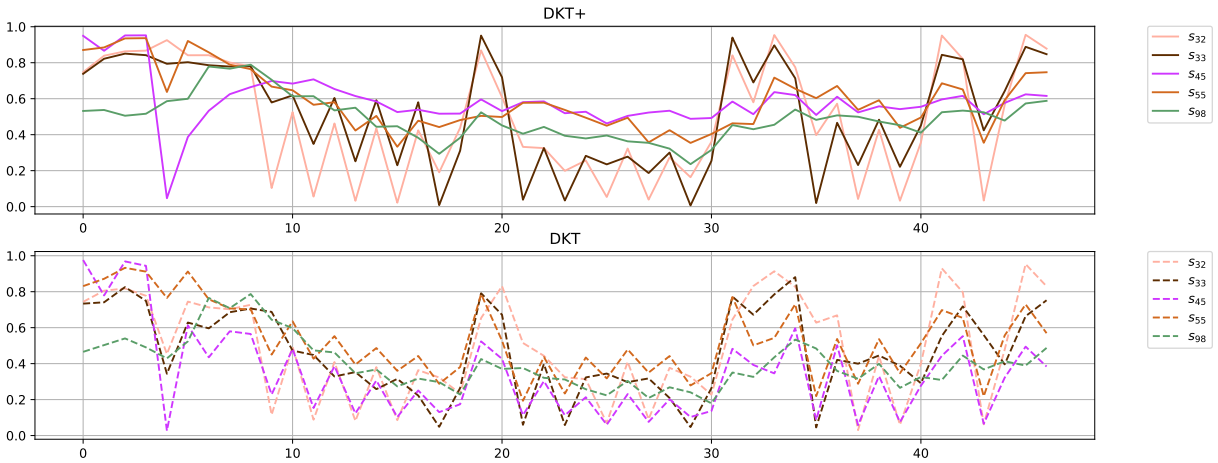
where $c = \sum_{i=1}^{n}\sum_{t=1}^{T_i-1}\sum_{j=1}^{T_i-t}\gamma^{j-1}$ is the normalizing term, γ is the decay factor similar to that in reinforcement learning. This potentially leads the DKT model to learn a more robust representation of the latent knowledge state.

ACKNOWLEDGMENTS

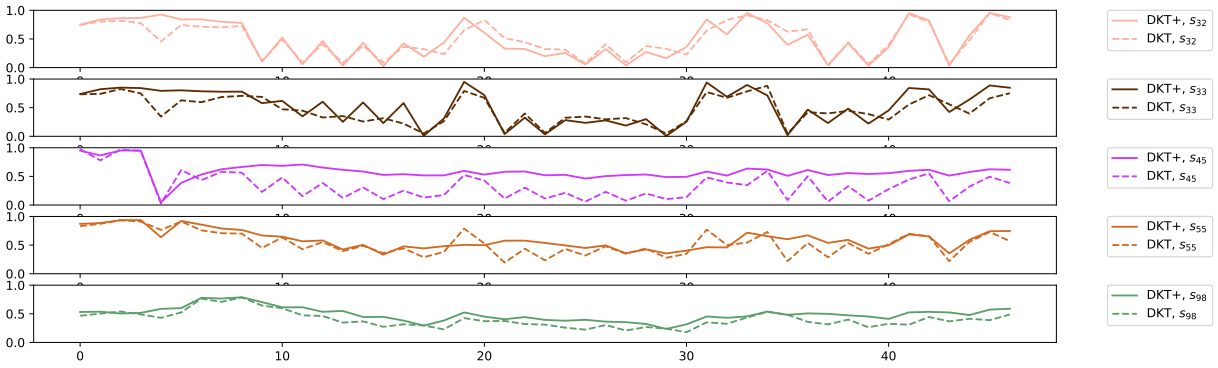
This research has been supported by the project ITS/205/15FP from the Innovation and Technology Fund of Hong Kong.



(a) Heatmap (upper: DKT+; lower: DKT) for the predicted probability of correctly answering each skill answered by the student.



(b) Line plot (upper: DKT+; lower: DKT) for the predicted probability of correctly answering each skill answered by the student. It aims to show the directional change in prediction of each skill in the DKT/DKT+ model.



(c) Line plot for the each skill's prediction, which is visualized separately with regard to the skill tag. It aims to compare the prediction result of the same skill between DKT and DKT+.

Figure 5: Visualization of a subset of DKT's output layer using the student's interaction sequences (id-1) extracted from ASSIST2009. The DKT+ model used is trained with $\lambda_r = 0.10$, $\lambda_{w_1} = 0.003$, $\lambda_{w_2} = 3.0$. We note that s_{32} is "Ordering Positive Decimals", s_{33} is "Ordering Fractions", s_{45} is "Subtraction Whole Numbers", s_{55} is "Absolute Value", and s_{98} is "Equation Solving Two or Fewer Steps".

REFERENCES

1. Hao Cen, Kenneth Koedinger, and Brian Junker. 2006. Learning factors analysis – a general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference in Intelligent Tutoring Systems*. Springer, Berlin, Heidelberg, 164–175.
2. Albert T. Corbett and John R. Anderson. 1995. Knowledge tracing: modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 4 (March 1995), 253–278.
3. José González-Brenes, Yun Huang, and Peter Brusilovsky. 2014. General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge. In *Proceedings of the 7th International Conference on Educational Data Mining*. 84–91.
4. William J. Hawkins, Neil T. Heffernan, and Ryan S.J.D. Baker. 2014. Learning Bayesian knowledge tracing parameters with a knowledge heuristic and empirical probabilities. In *Proceedings of the 12th International Conference on Intelligent Tutoring Systems*. Springer, Cham, 150–155.
5. Tanja Kaeser, Severin Klingler, Alexander G. Schwing, and Markus Gross. 2017. Dynamic Bayesian networks for student modeling. *IEEE Transactions on Learning Technologies* 10 (March 2017), 450–462.
6. Mohammad Khajah, Robert V. Lindsey, and Michael C. Mozer. 2016. How deep is knowledge tracing. In *Proceedings of the 9th International Conference on Educational Data Mining*. 94–101.
7. Zachary C. Lipton, John Berkowitz, and Charles Elkan. 2015. A critical review of recurrent neural networks for sequence learning. *ArXiv e-prints 1506.00019* (May 2015).
8. Christopher Olah. 2015. Understanding LSTM networks. Colah. github. io. (Augest 2015). Retrieved December 10, 2017 from `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`.
9. Zachary A. Pardos and Neil T. Heffernan. 2010. Modeling individualization in a Bayesian networks implementation of knowledge tracing. In *Proceedings of the 18th International Conference on User Modeling, Adaptation, and Personalization*, Vol. 6075. Springer, Berlin, Heidelberg, 255–266.
10. Zachary A. Pardos and Neil T. Heffernan. 2011. KT-IDEM: introducing item difficulty to the knowledge tracing model. In *Proceedings of the 19th International Conference on User Modeling, Adaption and Personalization*, Vol. 6787. Springer, 243–254.
11. Philip I. Pavlik, Hao Cen, and Kenneth R. Koedinger. 2009. Performance factors analysis – a new alternative to knowledge tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*. IOS Press, Amsterdam, Netherlands, 531–538.
12. Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*. 505–513.
13. Lutz Prechelt. 1998. Automatic early stopping using cross validation: quantifying the criteria. *Neural Networks* 11, 4 (June 1998), 761–767.
14. Siddharth Reddy, Igor Labutov, and Thorsten Joachims. 2016. Learning student and content embeddings for personalized lesson Sequence Recommendation. In *Proceedings of the 3rd ACM Conference on Learning @ Scale*. ACM, New York, NY, USA, 93–96.
15. Steven Tang, Joshua C. Peterson, and Zachary A. Pardos. 2016. Modelling student behavior using granular large scale action data from a MOOC. *ArXiv e-prints 1608.04789* (Augest 2016).
16. Lisa Wang, Angela Sy, Larry Liu, and Chris Piech. 2017. Learning to represent student knowledge on programming exercises using deep learning. In *Proceedings of the 10th International Conference on Educational Data Mining*. 324–329.
17. Yutao Wang and Neil T. Heffernan. 2013. Extending knowledge tracing to allow partial credit: using continuous versus binary nodes. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education*. Springer, Berlin, Heidelberg, 181–188.
18. Kevin H. Wilson, Yan Karklin, Bojian Han, and Chaitanya Ekanadham. 2016. Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. In *Proceedings of the 9th International Conference on Educational Data Mining*. 539–544.
19. Xiaolu Xiong, Siyuan Zhao, Eric Van Inwegen, and Joseph Beck. 2016. Going deeper with deep knowledge tracing. In *Proceedings of 9th International Conference on Educational Data Mining*. 545–550.
20. Michael V. Yudelson, Kenneth R. Koedinger, and Geoffrey J. Gordon. 2013. Individualized Bayesian knowledge tracing models. In *Proceedings of the 16th International Conference on Artificial Intelligence in Education*, Vol. 7926. Springer, Berlin, Heidelberg, 171–180.
21. Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 765–774.
22. Hui Zou and Trevor Hastie. 2004. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 2 (September 2004), 301–320.