Rodrigo, M. M. T. et al. (Eds.) (2021). Proceedings of the 29th International Conference on Computers in Education. Asia-Pacific Society for Computers in Education

Prior Knowledge on the Dynamics of Skill Acquisition Improves Deep Knowledge Tracing

Qiushi PAN [a*] &Taro TEZUKA[b]**
[a]*Graduate School of Comprehensive Human Sciences, University of Tsukuba, Japan*
[b]*Faculty of Library, Information and Media Science, University of Tsukuba, Japan*
* han.qiushipan@gmail.com
** tezuka@slis.tsukuba.ac.jp

Abstract: Knowledge tracing (KT) is the task of modeling how students' academic skills change over time. Given a sequence of a student's learning history, one goal of KT is to predict how well he/she will perform in the next interaction. Unlike in BKT (Bayesian knowledge tracing), the models in DKT (Deep knowledge tracing) cannot be improved simply by introducing elaborate prior knowledge about the task domain. Instead, we need to observe how trained models behave and identify their shortcomings. In this paper, we examine a problem in existing models that have not been discussed previously: the *inverted prediction problem*, in which the model occasionally gives predictions that are opposite to a student's actual performance development. Specifically, given an input sequence where a student has solved several problems correctly in a row, the model will occasionally estimate his/her skills to be lower than when he/she could not solve them. To tackle this problem, we propose *pre-training regularization*, which incorporates prior knowledge by supplying synthetic sequences to the neural network before training it with real data. We provide regular, simplistic synthetic data to a sequence-processing neural network as a specific implementation of pre-training regularization. This method solves the inverted prediction problem and improves the performance of the model in terms of AUC. We observed its effect qualitatively and introduced a quantitative measure to assess the improvement also. For ASSISTments 2009, ASSISTments 2015, and Statics 2011, improvements in AUC scores were 0.2 ~ 0.7 %, which are significant considering the scores are already high (around 70~80%). We developed an open-source framework for DKT with pre-training regularization. It also contains user-friendly hyperparameter optimization functionality.

Keywords: Knowledge tracing, educational data mining, recurrent neural networks, prior knowledge, regularization

1. Introduction

Online e-learning systems have rapidly grown in popularity in recent years. Students with different levels of understanding no longer need to be taught the same material in a classroom and can instead learn what is best suited for their knowledge level as predicted by a computer, based on their answers to past questions. By analyzing the accumulated big data, courses and assignments can be tailored for the best learning efficiency. Two examples of widely used e-learning platforms are intelligent tutoring systems (ITSs) and massive open online courses (MOOCs). With these systems, students are recommended which courses or lessons to take on the basis of their interaction with the system (Adamopoulos, 2013; Feng et al., 2009).

Knowledge tracing (KT) is the task of modeling students' academic abilities (Corbett and Anderson, 1994). Given a sequence of a student's learning history, the task is to predict the probability that he/she will answer the next question correctly. Bayesian knowledge tracing (BKT) handles this task by using a stochastic model. One of its merits is that the obtained parameters are easy to interpret. However, BKT requires human professionals to design the stochastic model carefully. Deep knowledge tracing (DKT) is an alternative approach that has become very popular in recent years (Piech et al., 2015). One of its strengths is that it does not require professionals to design its model parameters, as in BKT. It uses a recurrent neural network (RNN) (Zaremba et al., 2014) to capture the underlying structure of the students' understandings.

In this paper, we examine a problem of DKT that has been overlooked: the inverted prediction problem, where the student's performance is predicted as low even when he/she is consecutively answering questions correctly for some time. Based on this observation, we propose *pre-training regularization* and its unique implementation. Experiments showed that our proposed method mitigates the inverted prediction problem and leads to a higher AUC score. Furthermore, we developed a framework for training DKT based on our proposed method, together with easily accessible hyperparameter optimization functionality. Such functionality is especially valuable since it has not been available in previous implementations of DKT. The main contributions of this paper are as follows.

- We identify a unique problem in knowledge tracing, the inverted prediction problem, that we observe when making predictions using DKT. We propose a quantitative way of measuring its severity.

- We propose to use pre-training regularization and its specific implementation to overcome the inverted prediction problem. The proposed method also improved the AUC of the prediction results.

- We developed a framework for training DKT with pre-training regularization. Unlike existing DKT frameworks, our system contains user-friendly tools for hyperparameter optimization. The framework is open source and can be downloaded from the following repository.

https://github.com/qqhann/KnowledgeTracing

2. Related Work

2.1 Bayesian Knowledge Tracing

Many authors have pointed out that considering academic skills acquired by solving problems is crucial in knowledge tracing (Shepard, 1991; Resnick and Resnick, 1992; Cen et al., 2006). Bayesian knowledge tracing (BKT) was initially introduced by Corbett et al., 1994. It uses a binary stochastic variable for each knowledge concept (KC): understanding or not understanding. KC is a general term that represents a unit of knowledge, skill, or ability. In BKT, the probability of stochastic variables representing a KC is updated over time using hidden Markov models (HMMs). BKT represents the process of learning using probabilities of already knowing, guessing, acquiring, or slipping related to KCs. Based on these probabilities, BKT estimates a student's knowledge acquisition process over time. However, commonly used models in BKT are often oversimplified and rely on unrealistic assumptions, such as assuming that students will never forget what they have learned or that KCs are mutually independent and acquiring one KC will not affect the ease of acquiring other KCs.

Extensions of BKT include contextually modeling guessing and slipping (d. Baker et al., 2008), considering item difficulty (Pardos and Hefferman, 2011), and using student-specific parameters for individualization (Yudelson et al., 2013). There have also been proposals to reconsider the representation parameters of student abilities (Ritter et al, 2009). Wilson et al., 2016 showed that Hierarchical Item Response Theory (HIRT) and Temporal Item Response Theory (TIRT) could improve the performance if contextual information is used correctly.

2.2 Deep Knowledge Tracing

Deep knowledge tracing (DKT) was proposed by Piech et al., 2015. The model uses a recurrent neural network (RNN), especially a long short-term memory (LSTM) network. Unlike BKT that required domain specialists to adopt pedagogical knowledge into the model structure, DKT can learn it based on data. Although the trained parameters are difficult to interpret, it achieves a higher prediction accuracy. The RNN model forms the basis of DKT and is formulated as

$$\boldsymbol{h}_t = \tanh(\boldsymbol{W}_{hx}\boldsymbol{x}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{b}_h), \qquad \widehat{\boldsymbol{y}}_t = \boldsymbol{\sigma}(\boldsymbol{W}_{yh}\boldsymbol{h}_t + \boldsymbol{b}_y), \tag{1}$$

where the initial $\boldsymbol{h}_0$ is a randomly generated hidden state vector, $\boldsymbol{W}$ is the weight, and $\boldsymbol{b}$ is the bias. $\boldsymbol{\sigma}$ is a Q-dimensional-vector-valued function whose components are the sigmoid function. Q is the number of knowledge concepts (KCs). t is a time step. The student answers one question at each time step. $\boldsymbol{x}_t$ is encoded vector of (q_t, a_t) at time step t, where q_t is the KC that the question at time t belongs, and a_t is the correctness of the answer, which is 0 when the student answered the question incorrectly and 1 if he/she answered it correctly. Figure 1 shows the structure of the DKT model.

DKT has been actively investigated as a state-of-the-art approach to knowledge tracing (Xiong et al., 2016; Khajah et al., 2016; Yang et al., 2017; Sapountzi et al., 2018). In recent works, various shortcomings of DKT have been pointed out. Yeung and Yeung (2018) observed the *waviness* problem and *reconstruction* problem. Some extensions of DKT try to use more detailed study logs to achieve better results Zhang et al., 2017. Although more recent models have shown improvements over vanilla DKT, many still use an LSTM-based DKT as their base model. Therefore, we chose the vanilla DKT model as the baseline in this study and examined it in detail to identify fundamental issues.

Since DKT models are based on neural networks, it is not clear how one can incorporate prior knowledge about a student's learning process into the models. Unlike BKT, there is no systematic way of using a prior distribution. The aim of the current paper is to propose the use of pre-training as a way of introducing prior knowledge into DKT models. For each t, $\boldsymbol{x}_t = (q_t, a_t)$ is the 2-dimensional input vector, $\boldsymbol{h}_t$ is the hidden state vector, and $\hat{\boldsymbol{y}}_t$ is the Q-dimensional output vector, where Q is the number of knowledge concepts present in the dataset. The k-th component of $\hat{\boldsymbol{y}}_t$ represents the estimated probability that the student answers correctly at time $t+1$ to a question that belongs to knowledge concept k. The LSTM cell contains the weight $\boldsymbol{W}$ that is optimized by training. The LSTM cell and the weight is shared across time t.

3. Method

3.1 Inverted Prediction Problem

When examining the DKT model's prediction results, we often found that when a student is consecutively answering all questions correctly, his/her predicted performance (the probability that he/she will answer the next question belonging to the same knowledge concept correctly) is lower than that of students consecutively answering all questions incorrectly. We can roughly assume that once a knowledge concept (which represents a skill) is acquired, it is not lost easily. Therefore, the phenomenon of the decreasing predicted performance for questions belonging to a knowledge concept is counterintuitive. It suggests that the trained model does not represent the real mechanism of the student's learning process. We named this phenomenon, the inverted prediction problem.

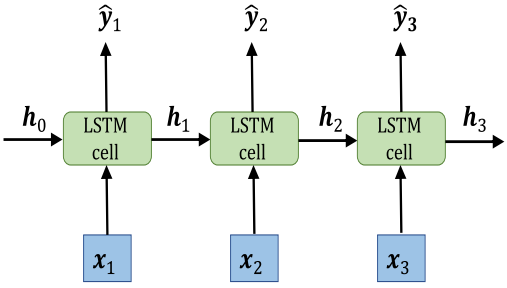

Figure 1. Schematic Diagram of DKT.

The opposite phenomenon maybe happening as well, where a DKT is unnecessarily optimistic and predicting that the student is more proficiently acquiring knowledge concepts than he/she really is. However, since such a phenomenon is harder to observe and quantify than the inverted prediction problem, we did not target it in the current paper.

Since the DKT model is a neural network, it is difficult to determine the cause of the phenomenon by analyzing its parameters. We chose to investigate it further by observing how the trained network responds to synthetic data. We created a synthetic sequence representing a student

answering all questions correctly for a specific knowledge concept (KC). The length of the sequence is set to T. We call this sequence the *oracle student*, following Ding et al., 2019. The opposite is the *totally failing student* who answers all questions incorrectly.

To interpolate between the oracle and totally failing students, we created sequences where all questions were answered incorrectly up to a certain point, then the later questions are all answered correctly. These sequences represent students failing up to a certain point but then acquiring the necessary knowledge (skill) to solve problems and start answering correctly. Note that in real data, elements in a sequence belong to different KCs. In contrast, we used a synthetic sequence where all questions belong to a single KC q.

$$\begin{gathered}
\boldsymbol{s}_0^q = ((q,0),(q,0),\ldots,(q,0),(q,0),(q,0)) \\
\boldsymbol{s}_1^q = \big((q,0),(q,0),\ldots,(q,0),(q,0),(q,1)\big) \\
\boldsymbol{s}_2^q = \big((q,0),(q,0),\ldots,(q,0),(q,1),(q,1)\big) \\
\ldots \\
\boldsymbol{s}_{T-1}^q = ((q,0),(q,1),\ldots,(q,1),(q,1),(q,1)) \\
\boldsymbol{s}_T^q = ((q,1),(q,1),\ldots,(q,1),(q,1),(q,1))
\end{gathered}$$

We can formalize such sequences as:

$$\begin{gathered}
\boldsymbol{s}_k^q = ((q,a_1),(q,a_2),\ldots,(q,a_T)) \\
a_t = 1 \text{ if } t > T-k, \text{and } a_t = 0 \text{ otherwise,}
\end{gathered} \tag{2}$$

where k is an integer in $[0,1,\ldots,T]$, and q represents a KC. a_t is a Boolean value representing whether the student answered the question correctly or not at time t. In other words, when $a_t = 0$, the student answered a question belonging to q incorrectly at time t, and when $a_t = 1$, he/she answered it correctly. $\boldsymbol{s}_0^q$ represents a totally failing student, and $\boldsymbol{s}_T^q$ represents an oracle student. We trained the LSTM baseline model using real data from ASSISTments 2009 (Feng et al., 2009). We then added synthetic data to the model and examined the predicted performances of the students. Synthetic sequences consist of $\boldsymbol{s}_0^q$ to $\boldsymbol{s}_T^q$ for each KC. We used $T = 20$ in the following experiment.

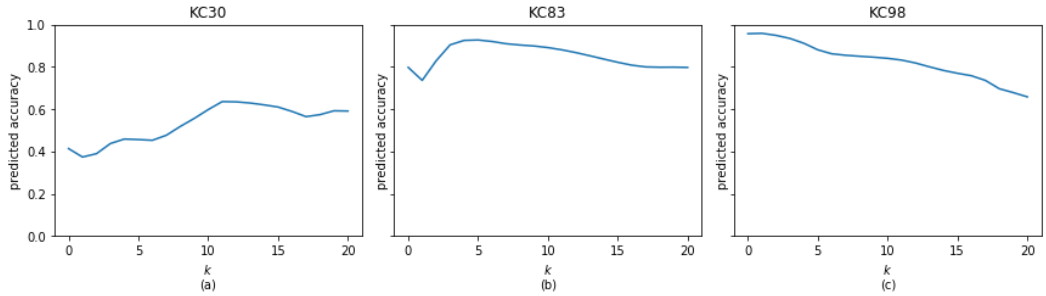

Figure 2. Predicted Performance Using Synthetic Sequences $\boldsymbol{s}_k^q$ Defined by Eq. 2.

The horizontal axes represent k and the vertical axes are the predicted performance $\hat{y}$. (a) is close to ideal dynamics since the predicted performance is nearly a monotonically increasing function of k. (c) is sub-optimal since the predicted performance is nearly a monotonically decreasing function of k. (b) is not as bad as (c) but is still sub-optimal.

In Figure 2, we use three values for KC q to explain the *inverted prediction problem*. The graph in (a) shows the result for KC with ID 30 ($q = 30$). When provided with the synthetic sequences $\boldsymbol{s}_0^q$ to $\boldsymbol{s}_T^q$ the model predicted that the longer the student correctly answered questions for the KC, the more likely he/she would be to answer the next question correctly. The result shown here matches our intuition about how students would perform. We call this property about students' predicted performance the *proportional prediction rule* since the predicted performance is proportional to how they performed earlier.

Unfortunately, the model did not learn this rule for all KCs. The graph shown in (c) is the worst-case example, where the predicted performance is a decreasing function of k. It means that when a student continued to answer questions correctly longer, the model predicted that his/her performance

would be lower. Graph (b) is not as bad as (c) but is still sub-optimal because the predicted performance is a partially decreasing function of k. Although the model can distinguish between the oracle student and the failing student, it does not predict well for students performing in between. The inverted prediction problem shows one limitation of DKT when the available samples are limited. We use the problem to show how pretraining by prior knowledge can lead to better results than vanilla DKT.

3.2 Quantification of the Inverted Prediction Problem

To measure the severity of the inverted prediction problem, we propose two measures r_1 and r_2 formulated as

$$r_1 = \frac{1}{Q}\sum_{q=1}^{Q} \mathbf{1}_{\{\hat{y}(\boldsymbol{s}_T^q, q) > \hat{y}(\boldsymbol{s}_0^q, q)\}} \tag{3}$$

$$r_2 = \frac{1}{Q}\sum_{q=1}^{Q} NDCG\left(\hat{y}(\boldsymbol{s}_0^q, q), \ldots, \hat{y}(\boldsymbol{s}_T^q, q), (0, \ldots, T)\right). \tag{4}$$

r_1 gives a rough approximation of how well the model avoids the inverted prediction problem. Here $\mathbf{1}_{\boldsymbol{P}}$ is the characteristic function whose value is 1 when a proposition $\boldsymbol{P}$ is true, and 0 otherwise. q represents a KC, and Q is the total number of KCs. $\boldsymbol{s}_k^q$ is a synthetic sequence defined in Equation 2. $\hat{y}(\boldsymbol{x}, q)$ is the output of the DKT model, representing the student's predicted performance when sequence $\boldsymbol{x}$ is the input for KC q. r_1 can be used to distinguish graphs (a) and (c) in Figure 2 but cannot be used distinguish (a) from (b).

Therefore, we propose r_2 that considers the overall dynamics of the predicted performance. Ideally, the predicted performance should be higher when the student answers more questions correctly. To quantify how well a model follows this pattern, we used the normalized discounted cumulative gain (NDCG) score, which is commonly used in evaluating ranking prediction (Wang et al., 2013). NDCG is a normalized measure of similarity between two orderings.

NDCG is calculated by dividing the discounted cumulative gain (DCG) by the ideal discounted cumulative gain (IDCG), that is, $NDCG = DCG/IDCG$.

Let n-dimensional vector $\boldsymbol{a}$ represent a sequence of numbers, $a_1, \ldots, a_n$, and n-dimensional vector $\boldsymbol{b}$ represent a sequence of numbers, $b_1, \ldots, b_n$. The numbers are not necessary in increasing or decreasing order. DCG is a similarity measure between two orderings defined as $\mathrm{DCG}(\boldsymbol{a}, \boldsymbol{b}) = \sum_{i=1}^{n} \frac{\rho_i}{\log_2(i+1)}$ where ρ_i is the graded relevance score (Sawade et al., 2013) of $\boldsymbol{a}$ and $\boldsymbol{b}$ at position i.

$\mathrm{IDCG}(\boldsymbol{a}, \boldsymbol{b})$ is equal to $\mathrm{DCG}(\boldsymbol{a}, \boldsymbol{b})$ when $\boldsymbol{a}$ and $\boldsymbol{b}$ can be sorted using the same permutation ρ, that is, both $\rho(\boldsymbol{a})$ and $\rho(\boldsymbol{b})$ are sequences in increasing order; in other words, $\mathrm{IDCG}(\boldsymbol{a}, \boldsymbol{b})$ equals to $\mathrm{DCG}(\boldsymbol{a}, \boldsymbol{b})$ when $a_i \leq a_j$ if and only if $b_i \leq b_j$ for all pairs of indices i and j. In our case, such a case corresponds to $\hat{y}(\boldsymbol{s}_i^q, q) \leq \hat{y}(\boldsymbol{s}_j^q, q)$ for all $i \leq j$.

When the sequence $\hat{y}(\boldsymbol{s}_0^q, q), \ldots, \hat{y}(\boldsymbol{s}_T^q, q)$ is monotonically increasing for all q, $DCG\left(\hat{y}(\boldsymbol{s}_0^q, q), \ldots, \hat{y}(\boldsymbol{s}_T^q, q), (0, \ldots, T)\right)$ is equal to $IDCG\left(\hat{y}(\boldsymbol{s}_0^q, q), \ldots, \hat{y}(\boldsymbol{s}_T^q, q), (0, \ldots, T)\right)$, so r_2 takes the maximum value, which is 1. r_2 is large when DKT predicts that students who answered correctly longer will perform better for the next problem than students who answered correctly for a shorter period. r_2 is small when the opposite happens, that is, when the inverted prediction problem occurs.

3.3 Pre-training Regularization

To avoid the inverted prediction problem, we want to use prior knowledge that when a student continues to answer questions correctly for some time, his/her predicted performance will not suddenly drop. In transfer learning, the model is first pre-trained using a large dataset for a common task. The model is then further trained using a more specific dataset or a loss function representing a more specific task (Pan and Yang, 2010). We extend the idea of transfer learning by pre-training the model using synthetic data. In our proposed *pre-training regularization*, we generate synthetic data using simple rules that represent our prior knowledge about the process in which a student acquires knowledge.

We can gain some insights about pre-training regularization by comparing it with Bayesian modeling. In Bayesian modeling, the parameters of a prior distribution often correspond to pseudo-observations. For example, consider using a multinomial distribution as a generative model $p(\boldsymbol{x}|\boldsymbol{\mu})$

where $\boldsymbol{x}$ and $\boldsymbol{\mu}$ are d-dimensional vectors. Its conjugate prior $p(\boldsymbol{\mu}|\boldsymbol{\alpha})$ is the Dirichlet distribution. The posterior distribution is

$$p(\boldsymbol{\mu}|\boldsymbol{x},\boldsymbol{\alpha}) = \frac{p(\boldsymbol{x}|\boldsymbol{\mu})p(\boldsymbol{\mu}|\boldsymbol{\alpha})}{p(\boldsymbol{x}|\boldsymbol{\alpha})} = \left(\prod_{h=1}^{d}\mu_h^{x_h}\right)\left(\frac{1}{B(\boldsymbol{\alpha})}\prod_{h=1}^{d}\mu_h^{\alpha_h-1}\right)\frac{1}{p(\boldsymbol{x}|\boldsymbol{\alpha})} = \frac{1}{p(\boldsymbol{x}|\boldsymbol{\alpha})B(\boldsymbol{\alpha})}\prod_{h=1}^{d}\mu_h^{x_h+\alpha_h-1}$$

where $B(\boldsymbol{\alpha})$ is the Beta function. The maximum a posteriori (MAP) estimate of $\boldsymbol{\mu}$ maximizes $p(\boldsymbol{\mu}|\boldsymbol{x},\boldsymbol{\alpha})$. Using the method of Lagrange multipliers to fulfill the constraint $\sum_{h=1}^{d}\mu_h = 1$, the estimate for μ_h can be obtained by

$$\hat{\mu}_h = \frac{x_h + \alpha_h - 1}{\sum_{u=1}^{d}(x_u + \alpha_u - 1)}.$$

In the above equation, the hyper-parameter α_h is added to the observation x_h which represents the number of occurrences of the event indexed by h. Incrementing α_h by one corresponds to increasing x_h by one, which means observing one extra occurrence of the event h. For this reason, α_h is called a *pseudo-observation*. Unlike real observations, pseudo-observations can take any non-negative real value. This correspondence between a hyper-parameter and a pseudo-observation is commonly seen in many Bayesian models. Since pre-training by synthetic data corresponds to adding extra observations to real data, it amounts to adding pseudo-observations to real data. In this way, pre-training corresponds to using prior knowledge. As DKT is based on deep learning rather than Bayesian modeling, its hidden layers and nodes can be considered as latent stochastic variables, and the probabilistic interpretation of deep learning can thus be a powerful analysis tool.

We propose to use a specific example of pre-training regularization as a remedy to the inverted prediction problem. We pre-trained DKT using monotonic synthetic sequences. Specifically, we generated synthetic sequences $\boldsymbol{s}_0^q$, $\boldsymbol{s}_1^q$, …, $\boldsymbol{s}_T^q$ (defined by Equation 2) for all KCs and used them to train the DKT network before training it with real data.

Another way to look at pre-training regularization is that it brings the output closer to a monotonically non-decreasing step function expressed as $\boldsymbol{s}_0^q$ to $\boldsymbol{s}_T^q$. For each KC, the model is pre-trained using sequences $\boldsymbol{s}_0^q$ to $\boldsymbol{s}_T^q$ repeatedly. We trained the model using synthetic data for all KCs for a certain number of epochs (e.g., 10 or 150) before training it with real data. By providing the network with sequences having a simple structure in pre-training, we expect the model to prefer simple dynamics when making predictions. Since this method does not require designing a prior distribution, it can be adapted to any model with ease. No domain knowledge is required, except the assumption that once a student acquires a knowledge concept, it is unlikely that he/she will lose it within a short time span.

4. Experiments

4.1 Datasets

ASSISTments 2009-2010: In the experiment, we used the ASSISTments *skill builder* dataset 2009-2010 (Feng et al., 2009). ASSISTments is an online tutor system that helps teachers access students' assessment information while studying math on the system. The dataset contains records of 4,417 students solving 328,291 problems. Each problem is manually tagged with a single skill concept required to solve it, out of 110 skill concepts in total.

ASSISTments 2015: ASSISTments 2015 is a dataset of 19,840 students solving 683,801 problems from 100 KCs.

Simulated-5: Simulated-5 is a dataset consisting of answer logs from simulated students based on the Item Response Theory (IRT) (Wilson et al., 2016). It contains records from 2,000 simulated students solving problems from 50 KCs.

Statics 2011: Statics 2011 is a dataset from students taking an engineering statics course. It has data of 333 students' 189,297 interactions on 1,223 KCs.

4.2 Results

Experimental results are summarized in Table 1. For comparison, we reconstructed the DKT model proposed by Piech et al., 2015. The baseline DKT is set up based on the previous research. We used a hidden dimension size of 200 and a batch size of 128. We set the learning rate to 0.1. We optimized hyperparameters using 5-fold cross-validation for ASSISTments 2009, ASSISTments 2015, and Statics 2011. We did not use cross-validation for Simulated-5 since we wanted to test it using the same condition as Piech et al., 2015. Cross-validation was carried out at the student level, that is, there is no student appearing in multiple folds.We searched for the best number of iterations for pre-training using validation data. We found that in most cases, 10 iterations were sufficient. We also tested 150 iterations to show that if there is a significant improvement by further pre-training.

Table 1 shows that the values of r_2 consistently increased with pre-training, suggesting it to be a better indicator than r_1. Pre-training improved AUC for most datasets. Pre 0 is without pre-training, pre 10 is with pre-training for 10 epochs, and pre 150 is with pre-training for 150 epochs. The values of r_1 are followed by the number of KCs in the corresponding dataset, which is the maximum possible value that r_1 can take. r_2 shows the mean value with standard deviation. Numbers in bold font are results that outperformed the baseline (DKT pre 0). The distribution of values for AUC comes from cross-validation. The distribution of values for r_2 represents the standard deviation of the terms in the sum in Equation 2.

Table 1. *Comparison of Area-under-the-Curve (AUC),* r_1 *and* r_2

Dataset	Model	Test AUC	r_1	r_2
ASSISTments 2009	DKT pre 0	0.802399 ± 0.002135	91 / 110	0.8858 ± 0.1210
ASSISTments 2009	DKT pre 10	**0.802763 ± 0.001107**	105 / 110	**0.9162 ± 0.1031**
ASSISTments 2009	DKT pre 150	**0.805398 ± 0.000801**	**106 / 110**	**0.8974 ± 0.1125**
ASSISTments 2015	DKT pre 0	0.703117 ± 0.001366	**99 / 100**	0.9243 ± 0.0914
ASSISTments 2015	DKT pre 10	**0.703355 ± 0.000890**	**99 / 100**	0.9211 ± 0.0967
ASSISTments 2015	DKT pre 150	**0.705414 ± 0.001263**	95 / 100	**0.9278 ± 0.0926**
Simulated-5	DKT pre 0	**0.777116**	26 / 50	0.7923 ± 0.1306
Simulated-5	DKT pre 10	0.776778	**44 / 50**	**0.8948 ± 0.1156**
Simulated-5	DKT pre 150	0.773313	**46 / 50**	**0.8908 ± 0.1139**
Statics 2011	DKT pre 0	0.787167 ± 0.000661	639 / 1223	0.7788 ± 0.1496
Statics 2011	DKT pre 10	**0.794149 ± 0.003777**	**934 / 1223**	**0.8855 ± 0.1267**
Statics 2011	DKT pre 150	**0.792491 ± 0.000426**	**716 / 1223**	**0.8095 ± 0.1582**

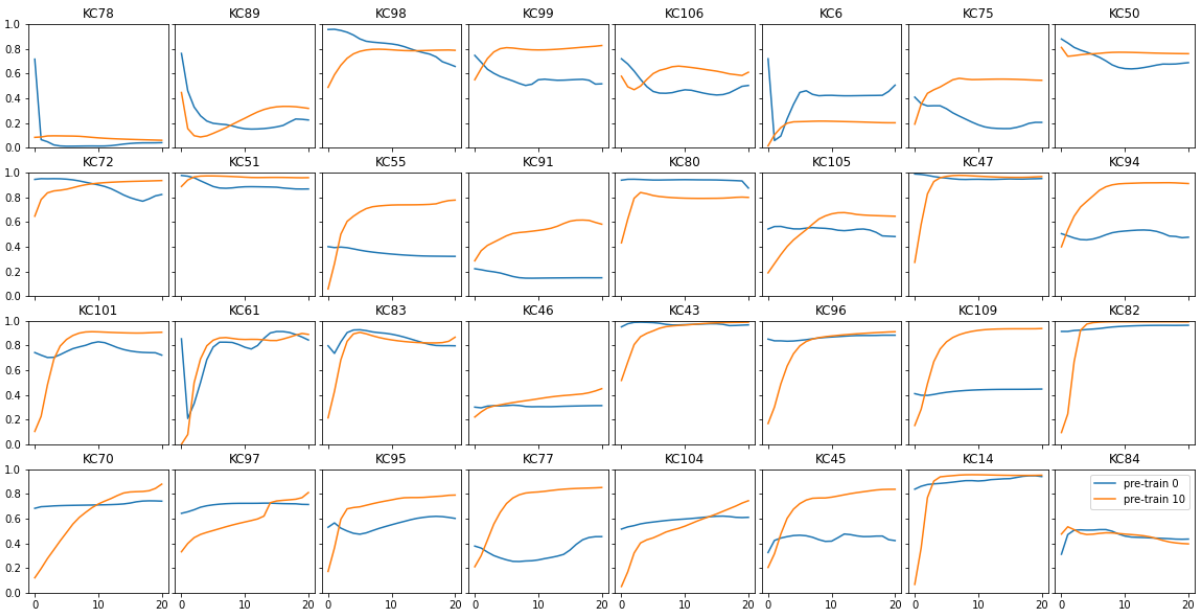

Figure 3. Prediction accuracy for KCs in ASSISTments 2009 when pre-trained using synthetic $\boldsymbol{s}_t^q$ defined in Equation 2. With pre-training, the inverted prediction problem occurred less. Blue is the baseline and orange is DKT with pre-training for 10 epochs. KCs are sorted by $\hat{y}(\boldsymbol{s}_T^q) - \hat{y}(\boldsymbol{s}_0^q)$. Vertical axis is the prediction accuracy for the same KC as used for the synthetic input. Horizontal axis represents knowledge concept k.

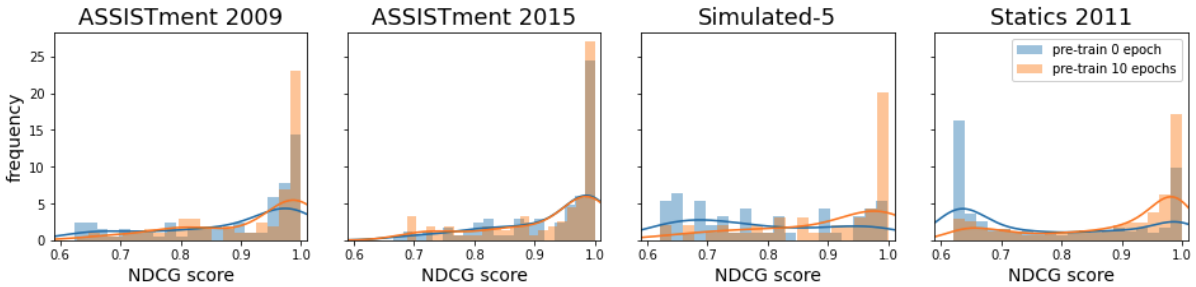

Figure 4. Quantitative inverted prediction (hard) effect with NDCG score distributions. Bars represent the histograms and solid lines are their kernel distribution estimates (KDE). Horizontal axis shows NDCG scores and vertical axis shows frequencies.

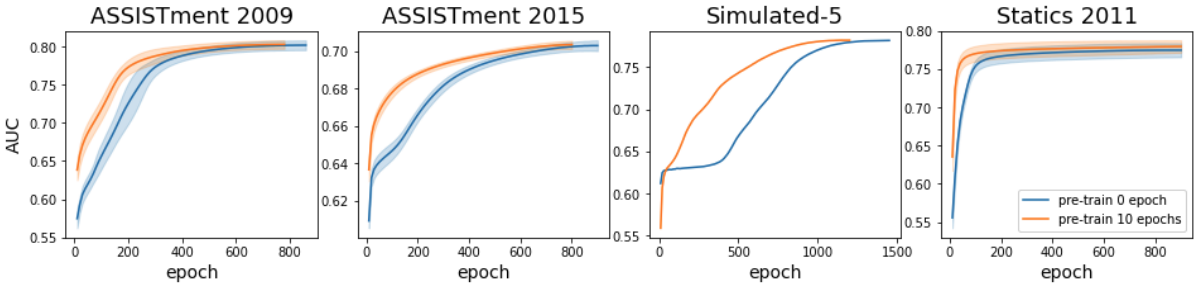

Figure 5. Learning curves of AUC for validation data. Solid lines represent the average of cross-validation. Shaded regions indicate 1-sigma intervals. Horizontal axis indicates the number of epochs and vertical axis the AUC scores.

4.3 Discussion

By pre-training, the model's parameters presumably reached a better starting position before being supplied with real data. The starting position has a property that when a student correctly answers several questions for a specific KC in a row, he/she is likely to answer another question with the same KC correctly. The model may learn this property if the dataset is large. However, the amount of data in educational data mining is often limited, so providing the property as prior knowledge is an effective strategy.

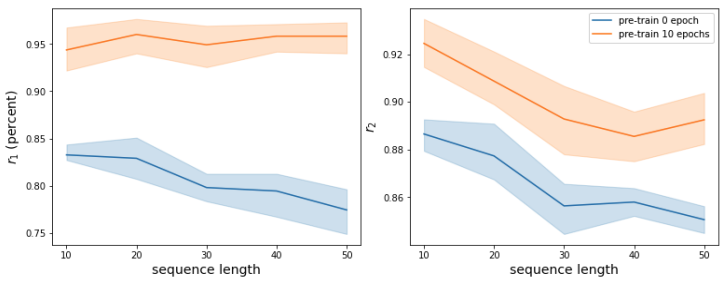

Figure 6. Dependence of r_1 and r_2 on sequence length in ASSISTments 2009. Horizontal axis is the sequence length and vertical axis is the value of r_1 and r_2. Solid lines show the average prediction probability for all KCs. Shaded regions represent $\mu \pm \sigma$ (i.e., mean $\pm$ standard deviation).

The results show that pre-training regularization had a positive effect on the predictive performance as well. With pre-training, the starting AUC for real data training is higher. The model also got a higher maximal validation AUC. For ASSISTments 2009, ASSISTments 2015, and Statics 2011, improvements in AUC scores were 0.2 ~ 0.7 %, which are significant considering the scores are already high (around 70 ~ 80 %). This result suggests that it had a better training starting point, resulting in a more global minimum and a better score. Simulated-5 is a simulated dataset and may not correctly reflect the learning dynamics of real students. Further investigation is needed to explain why the test AUC decreases with pretraining. For Statics 2011, pretraining 10 times produced better test AUC than pretraining 150 times. Such a decrease in test AUC can be explained by prior working too strong. Optimizing the amount of pretraining is a necessary step that we should explore.

We also examined the r_2 score further. We calculated the NDCG score using synthetic sequences for each KC, as described in Section 3. The NDCG scores can be visualized as a distribution. With pre-training, the number that equals or nearly equals 1 has increased. It means that the number of predictions having a perfect order has increased. Also, the distribution shifted toward 1. It suggests that our method made the prediction results better for most KCs.

Figure 3 shows the prediction results for synthetic inputs. It shows improvements for prediction results having counter-intuitive values of r_1 and r_2. The line plot gets smoother as the number of correct answers in the synthetic sequence increases from 0 to 20.

In Figure 4, the result for Simulated-5 indicates an interesting feature: without pre-training regularization, its r_1 score is only about half. It suggests that the dataset may have failed to reproduce the proportional prediction rule and that the DKT model can still perform well without the data coming from real human responses. In contrast, the model's AUC performance on Simulated-5 data dropped when using pre-training regularization.

Figure 5 shows the learning curves. When using pre-training, the model had a higher AUC at the start of the training iteration, and it continued to get a higher AUC than the baseline. It means that the model can get a considerable gain at an early stage. From another viewpoint, the result suggests that the vanilla DKT may be using the first few epochs just to acquire the proportional prediction rule. It would be more efficient to make the DKT model learn the rule using a few epochs of synthetic data since they're much smaller in size and take less time to train. Although our proposed method requires some epochs of pre-training, this is not a massive disadvantage in terms of the training time since it scales with the number of KCs, which is a much smaller number than the size of the training data. Also, the model becomes less likely to overfit, as indicated in the later epochs.

Figure 6 shows how r_1 and r_2 changed with respect to the sequence length T. The model with pre-training for 10 epochs generally had higher r_1 and r_2 than the model without pre-training. As T increased, both r_1 and r_2 decreased. This suggests that the inverted prediction problem becomes more significant for longer sequences.

5. Conclusion

In this paper, we pointed out a new difficulty in DKT: the inverted prediction problem. To alleviate the problem, we proposed pre-training regularization to train the model using synthetic data before providing real data to the model.

The method is easily adapted to any model since it does not require model adjustment or introducing a prior distribution, as done in Bayesian modeling. The experiments showed that pre-training regularization can reduce the effect of inversely proportional predictions quantified using two indicators, r_1 and r_2. It also made the model generalize better and resulted in higher AUC scores.

In the real world, there are various ways of utilizing the predicted academic abilities of students. It is not enough to evaluate the model by its accuracy when teachers or students ask to see the prediction scores. The prediction in such a situation should match the user's intuition, as otherwise, the system cannot gain much trust from practitioners of education. It is crucial to investigate the prediction in detail and check it to human intuition.

Our method's strength is that it can incorporate various types of prior knowledge through the design of simulation algorithms. It is advantageous when we want to use a prior distribution that has no efficient sampling algorithm. It is also easier for education researchers to incorporate their knowledge

on how students learn. Researchers only need to provide rules or specific examples and put them into the DKT model. They are not required to have in-depth knowledge of probability theory. For example, they can generate samples reflecting a tendency that a certain KC is likely to be acquired after another KC. They do not need to represent it as a stochastic model. If the prior knowledge is correct, it can increase the DKT model's performance, for example, in higher AUC, as it did in our case with the proportional prediction rule.

One limitation of our approach is that problems must be assigned to properly defined skills. If the assignments are unreliable, the underlining assumption of monotonic increase of the scores cannot be used. In future work, we plan to introduce knowledge state vectors (KS vectors) used in our previous work to represent the levels of skill acquisition (Pan and Tezuka, 2020). Since KS vectors can take continuous values, it can add more flexibility to the model than representing skill acquisition by binary values, as we did in this work. We also consider comparing how the order of supplying synthetic and real data may affect the performance. Preliminary experiments showed that supplying synthetic data before real data was more effective than supplying them after real data. We want to investigate this difference further and examine its validity for different types of tasks and models.

Acknowledgements

This work was supported by G-7 Scholarship Foundation and JSPS KAKENHI Grant Numbers JP16K00228, JP18KK0308.

References

Adamopoulos, P. (2013). What makes a great MOOC? An interdisciplinary analysis of student retention in online courses. *Proceedings of the 34th International Conference on Information Systems*.

Baker, R. S. J. d., Corbett, A. T., & Aleven V. (2008). More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian knowledge tracing. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 406-415. Springer.

Cen, H., Koedinger, K., & Junker, B. (2006). Learning factors analysis - A general method for cognitive model evaluation and improvement. *Proceedings of the 8th Int'l Conference on Intelligent Tutoring Systems*, 2006.

Corbett, A. T. & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-adapted Interaction*, 4(4), 253-278.

Ding. X. & Larson, E. C. (2019). Why deep knowledge tracing has less depth than anticipated. *Proceedings of the 12th International Conference on Educational Data Mining*.

Feng, M., Heffernan, N., & Koedinger, K. (2009). Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3), 243-266.

Kaplan, A. M. & Haenlein, M. (2016). Higher education and the digital revolution: About MOOCs, SPOC, social media, and the cookie monster. *Business Horizons*, 59(4), 441-450.

Khajah, M., Lindsey, R. V. & Mozer, M. C. (2016). How deep is knowledge tracing? *Proceedings of the 9th International Conference on Educational Data Mining*.

Pan, Q. & Tezuka, T. (2020). Accuracy-aware deep knowledge tracing with knowledge state vectors and an encoder-decoder architecture. *Proceedings of the 28th International Conference on Computers in Education*.

Pan, S. J. & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345-1359.

Pardos, Z. A. & Heffernan, N. T. (2011). KT-IDEM: Introducing item diffculty to the knowledge tracing model. *Proceedings of the 19th International Conference on User Modeling, Adaptation, and Personalization*, 243-254. Springer.

Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J., & Sohl-Dickstein, J. (2015). Deep knowledge tracing. *Advances in Neural Information Processing Systems*, 505-513.

Resnick, L. B. & Resnick, D. P. (1992). Assessing the thinking curriculum: New tools for educational reform. *Changing Assessments*, 37-75, Springer.

Ritter, S., Harris, T. K., Nixon, T., Dickison, D., Murray, R. C., & Towle, B. (2009). Reducing the knowledge tracing space. *International Working Group on Educational Data Mining*.

Sapountzi, A., Bhulai, S., Cornelisz, I., van Klaveren, C., Kardaras, D., & Semanjski, I. (2018). Dynamic models for knowledge tracing and prediction of future performance. *Proceedings of the 7th International Conference on Data Analytics*.

Sawade, C., Bickel, S., von Oertzen, T., Scheffer, T. & Landwehr, N. (2013). Active evaluation of ranking functions based on graded relevance. *Proceedings of the 23rd Int'l Joint Conference on Artificial Intelligence*.

Shepard, L. A. (1991). Psychometricians' beliefs about learning. *Educational Researcher*, 20(7), 2-16.

Wang, Y., Wang, L., Li, Y., He, D., & Liu, T.-Y. (2013). A theoretical analysis of NDCG type ranking measures. *Proceedings of the 26th Annual Conference on Learning Theory*, 25-54.

Wilson, K. H., Karklin, Y., Han, B., & Ekanadham, C. (2016). Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. *Proceedings of the 9th International Conference on Educational Data Mining*.

Xiong, X., Zhao, S., Van Inwegen, E. G., & Beck, J. E. (2016). Going deeper with deep knowledge tracing. *Proceedings of the 9th International Conference on Educational Data Mining*.

Yang, T.-Y., Brinton, C. G.. Joe-Wong, C., & Chiang, M. (2017). Behavior-based grade prediction for MOOCs via time series neural networks. *IEEE Journal of Selected Topics in Signal Processing*, 11(5).

Yeung, C.-K. & Yeung, D.-Y. (2018). Addressing two problems in deep knowledge tracing via prediction-consistent regularization. *Proceedings of the 5th Annual ACM Conference on Learning at Scale*, 5, ACM.

Yudelson, M. V., Koedinger, K. R., & Gordon, G. J. (2013). Individualized Bayesian knowledge tracing models. *Proceedings of the 16th International Conference on Artificial Intelligence in Education*.

Zaremba, W., Sutskever, I., & Vinyals, O. (2014). Recurrent neural network regularization, arXiv:1409.2329.

Zhang, L., Xiong, X., Zhao, S., Botelho, A., & Heffernan, N. T. (2017). Incorporating rich features into deep knowledge tracing. *Proceedings of the 4th ACM Conference on Learning at Scale*, 169-172. ACM.