Augmenting Knowledge Tracing by Considering Forgetting Behavior

Koki Nagatani
Fuji Xerox Co., Ltd.
Yokohama, Japan
nagatani.koki@fujixerox.co.jp

Qian Zhang
Fuji Xerox Co., Ltd.
Yokohama, Japan
qian.zhang@fujixerox.co.jp

Masahiro Sato
Fuji Xerox Co., Ltd.
Yokohama, Japan
sato.masahiro@fujixerox.co.jp

Yan-Ying Chen
FX Palo Alto Laboratory
Palo Alto, California
yanying@fxpal.com

Francine Chen
FX Palo Alto Laboratory
Palo Alto, California
chen@fxpal.com

Tomoko Ohkuma
Fuji Xerox Co., Ltd.
Yokohama, Japan
ohkuma.tomoko@fujixerox.co.jp

ABSTRACT

Computer-aided education systems are now seeking to provide each student with personalized materials based on a student's individual knowledge. To provide suitable learning materials, tracing each student's knowledge over a period of time is important. However, predicting each student's knowledge is difficult because students tend to forget. The forgetting behavior is mainly because of two reasons: the lag time from the previous interaction, and the number of past trials on a question. Although there are a few studies that consider forgetting while modeling a student's knowledge, some models consider only partial information about forgetting, whereas others consider multiple features about forgetting, ignoring a student's learning sequence. In this paper, we focus on modeling and predicting a student's knowledge by considering their forgetting behavior. We extend the deep knowledge tracing model [17], which is a state-of-the-art sequential model for knowledge tracing, to consider forgetting by incorporating multiple types of information related to forgetting. Experiments on knowledge tracing datasets show that our proposed model improves the predictive performance as compared to baselines. Moreover, we also examine that the combination of multiple types of information that affect the behavior of forgetting results in performance improvement.

CCS CONCEPTS

• **Applied computing → Learning management systems**; • **Social and professional topics** → *Student assessment*;

KEYWORDS

knowledge tracing, deep neural network, forgetting behavior

ACM Reference Format:
Koki Nagatani, Qian Zhang, Masahiro Sato, Yan-Ying Chen, Francine Chen, and Tomoko Ohkuma. 2019. Augmenting Knowledge Tracing by Considering Forgetting Behavior. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313565

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313565

1 INTRODUCTION

Computer-aided education (CAE) systems are widespread. Because a huge amount of learning logs can be collected from CAE systems, they can provide personalized materials to students based on their individual knowledge. This helps to reinforce knowledge by repeating materials that are difficult for a student and reducing learning costs by skipping materials that a student already knows.

To recommend suitable learning materials to each student, we should understand each student's knowledge accurately. The knowledge tracing task, which consists of: (1) modeling a student's knowledge through their interactions with the learning system contents and (2) predicting how a student will perform on future interactions, is a well-established and challenging problem in the field of CAE. The more precise the knowledge tracing is, the more satisfactory and suitable the contents provided by the system are [4]. On the contrary, a user study has shown that providing contents and questions to students at an inappropriate level of difficulty causes a decrease in their engagement [1]. Therefore, it is important to model and predict a student's knowledge precisely. This, in turn, is directly related to how efficiently a student can learn and how satisfied a student is with the CAE system.

For knowledge tracing, Bayesian knowledge tracing (BKT) [3] and performance factors analysis (PFA) [14] have been explored widely and applied to intelligent tutoring systems [10, 19, 20]. Recently, as deep learning models are outperforming the conventional models in a range of domains such as pattern recognition and natural language processing, the deep knowledge tracing (DKT) model [17], based on deep learning, can model a student's knowledge more precisely as compared to the conventional models mentioned above. The DKT model represents a student's knowledge state using the hidden variable of a recurrent neural network (RNN), which is a low-dimensional and dense vector. The BKT model represents a student's knowledge state as a binary variable for each skill. A skill is a class of related material, such as by grammar, and is identified with a question tag or ID. Hence, the DKT model can represent a student's knowledge more richly than the BKT model, which leads to improvements in knowledge tracing performance.

Predicting a student's knowledge precisely is a difficult task because students do forget. The recall rate five minutes after the previous learning may be different as compared to that after five days. Besides, the number of times students learn a target skill changes the behavior of forgetting: the more number of times they

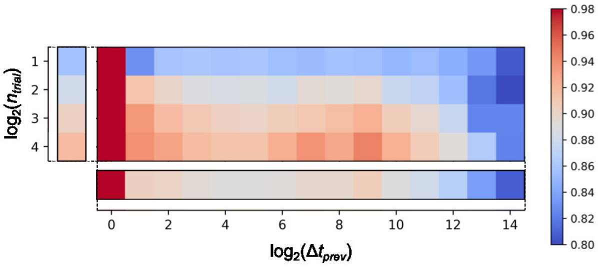


Figure 1: The quantized distribution map of the probability of a student correctly answering a question against the lag time from the previous interaction with the same skill id, Δt_{prev} and the number of past trials, n_{trial}. The distribution averaged over either the lag time or the number of trials are on the left and bottom of the distribution map, respectively. Note that single feature consideration does not capture the complex behavior of forgetting.

learn, the less they are likely to forget. To confirm the complexity of human forgetting, we analyzed the slepemapy.cz dataset, which is real-world learning logs acquired from an online learning platform. Figure 1 shows how the probability of answering correctly depends on both the lag time from the previous interaction with the same skill id and the number of past trials of that skill id. We aggregated interactions that students answered the previous question correctly. The marginal (averaged) distribution over the number of trials and the lag time are shown at the bottom and to the left of the distribution map, respectively. The forgetting behavior for a given lag time differs among the number of trials: as the number of trials increases, the student is less likely to forget. Such complex forgetting behavior cannot be captured if only lag time information is considered. Hence, this comprehensive context related to forgetting should be incorporated to achieve an accurate knowledge modeling.

There are only a few studies that have addressed forgetting in the field of knowledge tracing [11, 15, 18, 21], which either simplify the forgetting behavior or are independent of past sequences. Extensions of BKT only consider a fragment of the information related to forgetting: either the number of past trials [11], or the lag time from the previous interaction with the same skill. [18]. Regression models for knowledge tracing incorporate both the number of trials and the lag time from the previous interaction with the same skill. However, these regression models use features obtained from the interactions with the same skill [15, 21]. Considering a student's whole sequence makes it possible to modify the student's knowledge state on a skill with the student's interactions on other related skills. Otherwise, a model does not acquire any benefits from interactions on other skills.

In this study, we focus on modeling and predicting a student's knowledge considering the forgetting behavior from large amounts of learning logs to support providing adaptive learning materials in CAE systems. We propose an extension of the DKT model which can incorporate the information related to forgetting in the model. We have conducted experiments to examine the predictive performance of our model using two knowledge tracing datasets: Assistments 2012 dataset [7] and slepemapy.cz dataset [13]. In addition, we have investigated the properties of our model by analyzing the output of the model.

The contributions of the paper are summarized as follows.

- We propose a knowledge tracing model that extends the DKT model to consider both a learning sequence and the forgetting behavior by explicitly modeling the forgetting behavior using multiple features.
- We have conducted experiments showing that our proposed model outperforms conventional methods in terms of the predictive performance on the knowledge tracing datasets.
- We have also examined how the combination of multiple types of forgetting information influences the performance.

2 RELATED WORK

2.1 Knowledge tracing

Conventional knowledge tracing models such as BKT [3] and PFA [14] have been explored widely and applied to actual intelligent tutoring systems [10, 19, 20]. The BKT models a student's knowledge using a hidden Markov model (HMM). Each student's knowledge is represented as a set of binary latent variables, and the value of each variable corresponds to whether the student has mastered a skill or not. The PFA is a logistic regression model that predicts the probability of answering correctly based on the number of previous successes and failures. Although the PFA can easily incorporate several features, it basically cannot consider the whole sequence of interactions; instead, it considers the interactions only from the same skill.

As deep learning models are outperforming the conventional models in a range of domains such as pattern recognition and natural language processing, the DKT model [17] has shown that deep learning can model a student's knowledge more precisely as compared to the conventional models mentioned above. The DKT models a student's knowledge by an RNN which is often used for sequential processing. Although the DKT model is a sequence model as well as the BKT model, the DKT model represents a student's knowledge state using the hidden variable of a RNN, which is a low-dimensional and dense vector; the BKT model represents a student's knowledge state as a binary variable for each skill. In doing so, the DKT model can represent a student's knowledge more richly than the BKT model, which leads to an improvement of the predictive accuracy for the student's future performance. Following the DKT model, there are increasing amounts of research on deep learning-based knowledge tracing models [2, 12, 23, 24]. Zhang et al. improved DKT by incorporating additional features into it [24]. Although their approach has similarity to ours in terms of incorporating additional features to the DKT model, they neither focused on modeling forgetting and modeling across time, nor used time-based features such as lag time from the previous interaction.

2.2 Considering forgetting for student's knowledge tracing

There are several research branches that focus on improving knowledge tracing tasks [16]. Considering the forgetting behavior is one of the directions to better understand a student's knowledge. One of the early studies about forgetting [5] revealed that the retention

Table 1: Comparison between our proposed approach and previous work. Sequence indicates the method used to model the sequence of interactions. Time-based forgetting information is the lag time between interactions, and the count-based forgetting information is the number of trials in the past.

Method	Sequence	Forgetting info.	
		Time-based	Count-based
BKT [3]	HMM	-	-
PFA [14]	-	-	-
DKT [17]	RNN	-	-
Qiu et al. [18]	HMM	✓	-
Khajah et al. [11]	HMM	-	✓
Pelánek [15]	-	✓	✓
Settles and Meeder [21]	-	✓	✓
Ours	RNN	✓	✓

rate decreases exponentially as time passes by, and also, increasing the number of repetitions helps a user avoid forgetting. Without considering forgetting behavior in a knowledge tracing model, a student's knowledge state cannot be changed as time passes by. This assumption is not reasonable in reality.

The original BKT model does not consider forgetting behavior. Qiu et al. [18] extended the BKT model to consider forgetting behavior for the first time. A new day flag was added to BKT to model forgetting one day after the previous interaction. However, this system could not model forgetting that occurred on a much shorter time scale. Khajah et al. then extended the BKT model to estimate the probability of forgetting by applying the number of trials in the past [11]. However, the lag time between the current and previous interactions was not considered.

For the PFA and the related regression models, there are two studies that focused on incorporating forgetting behavior into these models. Pelánek incorporated a time factor into the memory activation, which determines the probability of answering correctly [15]. In this research, applying a staircase function, which boosts the memory activation when the lag time is in a certain bound, resulted in the best performance. On the other hand, Settles and Meeder proposed a half-life regression model which extends the PFA model with the essence of Ebbinghaus's forgetting curve [21]. Although these models incorporated information related to forgetting, the problem still remains. These regression models consider only the information about the skill and neglect the interaction sequence of other skills.

The difference between our proposed approach and the previous works is that we extended DKT, which is a sequential knowledge model, to consider a student's whole sequence of interactions, and incorporate multiple types of information to represent the complex forgetting behavior. Table 1 shows the comparison between our proposed approach and previous works.

3 PRELIMINARIES

In this section, we describe the knowledge tracing task and briefly explain the DKT model, which is the base component of our model.

3.1 Knowledge Tracing Task

The knowledge tracing task aims to model a student's knowledge through interactions with questions and to predict how a student will perform on the future interactions. The task can be formulated as a supervised learning problem: given a student's past interactions $\mathbf{x}_0, \cdots, \mathbf{x}_t$, the student's performance is predicted in the next interaction. An interaction $\mathbf{x}_t = (q_t, a_t)$ is defined as a tuple containing the skill id q_t of a question that a student attempts at time step t, and the label a_t which represents whether the student answers correctly or not. We define the number of skills as Q. In this study, the skill id q_t is identified by either a question tag or an ID, and a_t is a binary variable (one represents correct and zero represents incorrect answers). The values of a_{t+1} are predicted for the given values of q_{t+1}.

3.2 Deep Knowledge Tracing

The deep knowledge tracing model is the first deep neural network-based knowledge tracing model [17]. The structure of the DKT model is shown in Fig. 2 (a). DKT models a student's knowledge transition by using various RNNs such as Elman RNN [6] and long short term memory (LSTM) [9]. The student knowledge state is represented by a hidden state vector of RNN $\mathbf{h}_t \in \mathbb{R}^k$, where k is the number of hidden state dimensions. The DKT model can be separated into two processes: modeling a student's knowledge and predicting a student's performance.

In the process of modeling a student's knowledge, for a given input $\mathbf{x}_t = (q_t, a_t)$, the model updates a student's knowledge state $\mathbf{h}_t$ at each time step t. The input $\mathbf{x}_t$ is a one-hot vector, which is a Cartesian product of q_t and a_t. The input vector $\mathbf{x}_t$ is then transformed into a low-dimensional and dense real-valued vector $\mathbf{v}_t$. In [17], a random vector $\mathbf{n}_{q,a} \sim N(0, \mathbf{I})$ is assigned to each input tuple as v_t. With the embedding vector $\mathbf{v}_t$ and the previous student knowledge state vector $\mathbf{h}_{t-1}$, the student knowledge vector $\mathbf{h}_t$ is updated as:

$$\mathbf{h}_t = \phi(\mathbf{v}_t, \mathbf{h}_{t-1}),$$

where ϕ is the cell module of RNN.

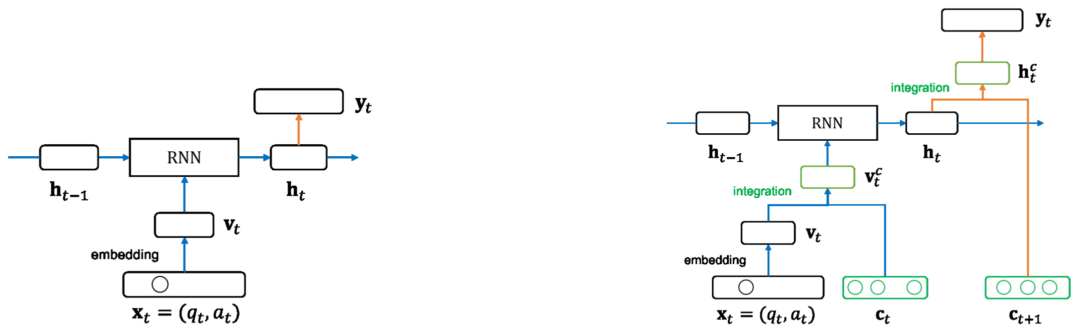
In the process of predicting a student's performance, the probabilities of answering correctly for all skills $\mathbf{y}_t \in \mathbb{R}^Q$ based on the updated student's knowledge state $\mathbf{h}_t$ is calculated as:

$$\mathbf{y}_t = \sigma(\mathbf{b}^{\text{out}} + \mathbf{W}^{\text{out}}\mathbf{h}_t),$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathbf{W}^{\text{out}} \in \mathbb{R}^{Q\times k}$ is the weight matrix and $\mathbf{b}^{\text{out}} \in \mathbb{R}^Q$ is the bias vector of output.

4 PROPOSED APPROACH

We have utilized the DKT model as our base model because it is a state-of-the-art model in the knowledge tracing task, and a deep neural network that can easily incorporate multiple input sources and capture the nonlinear dynamics among them. By extending DKT so that it can consider forgetting, the model can adapt the student performance to the student's forgetting.



(a) Architecture for deep knowledge tracing. (b) Architecture for proposed model.

Figure 2: Architectures for (a) deep knowledge tracing and (b) proposed model. The blue arrows describe a process of modeling a student's knowledge. The orange arrows describe a process of predicting a student's performance. Our proposed model incorporates the additional information related to forgetting $\mathbf{c}_t$, $\mathbf{c}_{t+1}$ and an integration component in both processes, which are colored by green. $\mathbf{c}_t$, $\mathbf{c}_{t+1}$ are multi-hot vectors which quantize the information related to forgetting: repeated time gap, sequence time gap, and past trial counts.

4.1 Information related to forgetting

Earlier research has revealed that the retention rate differs depending on the combination of two aspects: the lag time from the previous interaction and the number of times a student learns the learning material. The lag time can be calculated for the repetition of same skill ids or for sequence of any skill ids. Hence, in this study, we have considered the following three features:

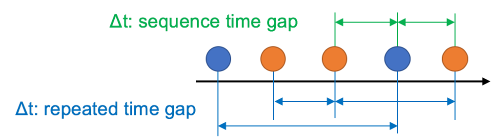
- repeated time gap: the lag time between an interaction and the previous interaction with the same skill id,
- sequence time gap: the lag time between an interaction and the previous interaction in the sequence; the skill id of an interaction does not matter, and
- past trial counts: the number of times a student answers questions with the same skill id.

Figure 3 illustrates these features. As for time gap, the repeated time gap has been commonly used in the previous researches [15, 21]. The reason we have used the sequence time gap is if both the skill of the question that student attempts and the skill of the previous question are related to each other, the lag time between these interactions may affect the performance of the question. Incorporating the sequence time gap into the model may capture this effect. The two time gap features are used by the minute and all three features are discretized at $\log_2$ scale.

4.2 Model

We extended DKT by incorporating information related to forgetting into two spaces of the model: the input and the output space of the RNN. The overall model architecture is shown in Fig. 2 (b).

To model a student's knowledge process, we have used a trainable embedding matrix $\mathbf{A}$ to calculate the embedded vector for $\mathbf{v}_t$ for the interaction vector $\mathbf{x}_t$ instead of assigning the random values. In addition, we have added the information related to forgetting, $\mathbf{c}_t$, discussed in Sect. 4.1, as the input as well as the interaction at time step t. $\mathbf{c}_t$ is a multi-hot vector because the three features in the information are represented as one-hot vectors and then concatenated. Before passing into the RNN module, the embedding



Time step	t_1	t_2	t_3	t_4	t_5
Repeated time gap	-	-	Δt_{32}	Δt_{41}	Δt_{53}
Sequence time gap	-	Δt_{21}	Δt_{32}	Δt_{43}	Δt_{54}
Past exercise counts	0	0	1	1	2

Figure 3: The additional information related to forgetting from a student's sequence of interactions. Each point corresponds to an interaction and the same color represents the same skill. In the table, the time gap $\Delta_{ij} = t_i - t_j$.

vector $\mathbf{v}_t$ and the additional information vector $\mathbf{c}_t$ are integrated with each other:

$$\mathbf{v}_t^c = \theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t),$$

where θ^{in} is the input integration function that incorporates additional information into the student's knowledge state. We describe this integration function in Sect. 4.3. Next, the student's knowledge state vector $\mathbf{h}_t$ is updated with the integrated input $\mathbf{v}_t^c$ and the previous student's knowledge state $\mathbf{h}_{t-1}$:

$$\mathbf{h}_t = \phi(\mathbf{v}_t^c, \mathbf{h}_{t-1}).$$

Thereby, a student's knowledge state is updated considering both the response of a student for the skill and the information related to forgetting. If a student answers correctly with a long time gap, the model updates the skill in the direction of acquiring more mastery.

Similarly in the process of predicting a student's performance, we have integrated the additional information at the next time step $\mathbf{c}_{t+1}$ with the updated student's knowledge state vector $\mathbf{h}_t$:

$$\mathbf{h}_t^c = \theta^{\text{out}}(\mathbf{h}_t, \mathbf{c}_{t+1}),$$

where θ^{out} is the output integration function that brings additional information into the prediction. Thereby, the model can predict

Table 2: Statistics of the data.

Dataset	#interactions	#users	#items
ASSISTments 2012	5,818,868	45,675	266
slepemapy.cz	10,087,305	87,952	1,458

a student's performance considering forgetting: if the student attempts a question with a short time lag, the model revised the probability of answering correctly upward. This integration function is described in detail in Sect. 4.3. Subsequently, the model predicts the probability of answering correctly for all skills $\mathbf{y}_t \in \mathbb{R}^Q$ based on the updated student's knowledge state $\mathbf{h}_t^c$:

$$\mathbf{y}_t = \sigma(\mathbf{b}^{\text{out}} + \mathbf{W}^{\text{out}}\mathbf{h}_t^c).$$

4.3 Integration Methods

In both processes of modeling a student's knowledge and predicting a student's performance, we have incorporated the integration functions, θ^{in} and θ^{out}, into our model to consider the forgetting behavior. We have explored the following four integration methods:

- concatenation: $\theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t) = [\mathbf{v}_t; \mathbf{c}_t]$,
- multiplication: $\theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t) = \mathbf{v}_t \odot \mathbf{C}\mathbf{c}_t$,
- concatenation and multiplication: $\theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t) = [\mathbf{v}_t \odot \mathbf{C}\mathbf{c}_t; \mathbf{c}_t]$, and
- bi-interaction: $\theta^{\text{in}}(\mathbf{v}_t, \mathbf{c}_t) = \sum_i \sum_{j \neq i} \mathbf{z}_i \odot \mathbf{z}_j, \mathbf{z}_i \in \{\mathbf{v}_t, \mathbf{C}_i c_t^i \mid c_t^i \neq 0\}$,

where $\mathbf{C}$ is the trainable transformation matrix, and $\odot$ denotes the element-wise multiplication. For the output integration function θ^{out}, we have used the same integration methods but applied to the student's knowledge vector $\mathbf{h}_t$ instead of $\mathbf{v}_t$. We have shared the transformation matrix C for both the input integration function θ^{in} and the output integration function θ^{out}. Concatenation stacks the interaction vector with the feature vector. Hence, this integration does not change the interaction vector itself. On the other hand, multiplication modifies an interaction vector by the contextual information. Concatenation and multiplication are common ways to integrate vectors. The first three integration methods have been used in [22]. Bi-interaction, which has been proposed in [8], encodes the second-order interactions between the interaction vector and context information vector, and between context information vectors.

4.4 Training

The learning parameters of our proposed model are the embedding matrix $\mathbf{A}$ for the interaction vector $\mathbf{x}_t$, the weights in the RNN module, the weight matrix $\mathbf{W}^{\text{out}}$ and the bias $\mathbf{b}^{\text{out}}$ for the prediction, and the transformation matrix $\mathbf{C}$ for additional information $\mathbf{c}_t$ if we use multiplication or bi-interaction integration. These parameters are jointly learned by minimizing a standard cross entropy loss between the predicted probability of correctly answering the next question for the skill id q_{t+1} and the true label a_{t+1}:

$$\mathcal{L} = -\sum_t (a_{t+1}\log(\mathrm{y_t^T}\delta(\mathrm{q_{t+1}})) + (1 - a_{\mathrm{t+1}})\log(1 - \mathrm{y_t^T}\delta(\mathrm{q_{t+1}}))),$$

where $\delta(q_{t+1})$ is the one-hot encoding for which skill id is answered in the next time step $t + 1$.

5 EXPERIMENTS

In this section, we present evaluation results for our model. Firstly, we evaluated the predictive performance by comparing our proposed model with other baselines on two real-world public datasets collected from online learning platforms. Next, we conducted comparative experiments to examine the contribution of features related to forgetting in our proposed model for further model exploration.

5.1 Datasets

We used the two datasets: ASSISTments 2012[1] [7] and slepemapy.cz[2] [13]. The statistics of the two datasets are shown in Table 2.

ASSISTments 2012: This dataset is gathered from the ASSISTments, which is an online tutor system that simultaneously teaches and assesses students in mathematics. For the dataset, we defined *skill_id* as the identifier of a skill id. We also removed users with only one interaction. After preprocessing, the dataset includes 5,818,868 interactions of 45,675 users and 266 skills.

slepemapy.cz: This dataset is from an online system used for practicing geography. We defined *place_asked* as the identifier of a skill id. After removing the users with only one interaction, the dataset includes 10,087,305 interactions of 87,952 users and 1,458 skills.

5.2 Experimental setup

We evaluated the knowledge tracing models using five-fold cross validation in which each dataset was split based on students. We extracted 10% of the students from the training set for a validation set that was used to tune hyperparameters and perform early stopping. On each dataset we used area under the curve (AUC) as an evaluation metric, which ranges from 0 (worst) to 1 (best). We reported the average test AUC.

We used an LSTM for both DKT and our model. To examine the effectiveness of incorporating the forgetting features, we used the same embedding technique for the DKT model, which is described in Section 4.2. The performance of DKT with trainable embedding was almost the same as the original DKT model using random embedding (an AUC of 0.7235 for the DKT with trainable embedding was comparable with an AUC of 0.7233 for the original DKT on ASSISTments dataset). We used the DKT with trainable embedding as our baseline. Both the size of the embedding vector and the dimension of the LSTM hidden state were set to 100. We used the Adam algorithm to optimize the models. The mini batch size was set to 100 for ASSISTments 2012 dataset and 30 for slepemapy.cz dataset. We explored the optimal hyperparameters in terms of weight decay in a validation set.

5.3 Performance Prediction

We compared our proposed models with different integration methods against the DKT and the HLR model, which is the regression model considering forgetting. Table 3 shows the results on the ASSISTments and slepemapy.cz datasets. Because the HLR model can predict only the interactions of questions which a student has attempted in the past, we also evaluated on the subset. In our experiments, the regression-based HLR model performed worse than

[1] https://sites.google.com/site/assistmentsdata/home/2012-13-school-data-with-affect

[2] https://www.fi.muni.cz/adaptivelearning/?a=data

Table 3: **Test AUC results for two datasets. We conducted five-fold cross-validation and calculated the average test AUC. *all* is all interactions except the first interaction of each student in a test set. *sub* is the subset of the interactions with questions which a student has attempted even once in the past. The standard deviation values are 0.0014 and 0.0034 on ASSISTments and slepemapy.cz datasets, respectively.**

Model	ASSISTments		slepemapy.cz	
	all	sub	all	sub
HLR	-	0.5846	-	0.6317
DKT	0.7235	0.7213	0.7871	0.7929
DKT + forgetting (Ours)				
concat	0.7283	0.7257	0.7999	0.8061
multi	0.7301	0.7280	0.8028	0.8093
concat + multi	**0.7309**	**0.7287**	**0.8046**	**0.8115**
bi-interaction	0.7289	0.7267	0.8037	0.8103

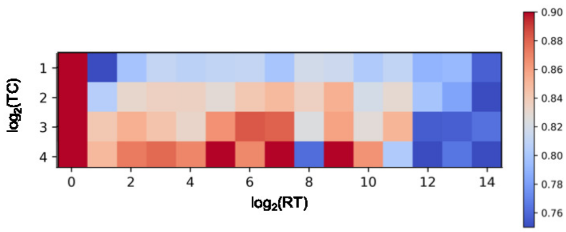


Figure 4: The quantized distribution map of the prediction of our model against the repeated time gap (RT) and the past trial counts (TC) on slepemapy.cz dataset. This figure is similar to Fig. 1, which indicates that our model can reproduce the complex forgetting behaviors.

any of the deep-learning based models. DKT achieved an average test AUC of 0.7235 on the ASSISTments dataset, and an AUC of 0.7871 on the slepemapy.cz dataset. Compared with DKT, our proposed model, which considers the information related to forgetting, performed better ($p < 0.01$ for both datasets). For reference, the Pelánik model [15], which incorporated a time factor into the PFA model, performed with an AUC of 0.7614 on the slepemapy.cz dataset. Although their experimental condition differs from ours, since they filtered the data to remove items with low counts, our model outperforms their model. We also investigated how the prediction of our model differs depending on both the repeated time gap and the number of past trials of the skill id, which is shown in Fig. 4. As in Fig. 1, we aggregated the interactions that students answered the previous question correctly. Our model reproduced the complex forgetting behavior shown in Fig. 1: as the number of trials increases, the student is less likely to forget.

Among the different integration methods, the combination of concatenation and multiplication (hybrid) achieved the best AUC on both the ASSISTments and slepemapy.cz datasets, with values of of 0.7309 and 0.8046, respectively. The multiplication integration achieved the second-best AUC score on ASSISTments dataset while the bi-interaction integration achieved the second-best AUC score on slepemapy.cz. The better result of the multiplication integration method than the concatenate integration method indicates that modifying an interaction vector using the information related to forgetting is better.

Table 4: Comparative analysis of the features related to forgetting. We used one or more features and compared the AUC performance. We used the hybrid integration of concatenation and multiplication. In the table, RT is repeated time gap, ST is sequence time gap, and TC is past trial counts.

Model	RT	ST	TC	ASSISTments	slepemapy.cz
DKT				0.7235	0.7871
DKT + forgetting	✓			0.7288	0.7972
+ one feature		✓		0.7284	0.7940
			✓	0.7258	0.7976
+ two features	✓	✓		0.7292	0.7992
	✓		✓	0.7304	0.7989
		✓	✓	0.7301	0.7993
+ three features (Ours)	✓	✓	✓	0.7309	0.8046

5.4 Comparative Study: Forgetting features

We investigated the effectiveness of incorporating the combination of features related to forgetting in the DKT model. Specifically, we used one or multiple forgetting features for modeling and observed how that affects performance.

Table 4 shows the results of the experiments. During performance comparison, while changing the number of features, we obtained the same trend on both ASSISTments and slepemapy.cz datasets: the more features used for modeling, the better the predictive performance. Adding two features resulted in a better performance than adding one feature with statistical significance ($p = 0.012$). Adding three features had a better performance than adding two features, though not statistically significant. These results support that considering multiple aspects about forgetting is important for modeling a student's knowledge.

The effectiveness of incorporating the combination of the features was different in two datasets. For the ASSISTments dataset, the combination of repeated time gap and trial count (RT+TC) was superior to other combinations. For the slepemapy.cz dataset, the combination of sequence time gap and trial count (ST+TC) seemed slightly more effective. As seen from the result of one feature in Table 4, these combinations do not comprise the top two features that had a good performance. This indicates that these features correlate with each other.

6 CONCLUSION

In this paper, we focused on considering forgetting behavior in knowledge tracing. We extended the DKT model so that the information related to forgetting can be incorporated into the model. The proposed solution achieved better predictive performance compared with the DKT model on two real-world datasets for knowledge tracing. We also revealed that the combination of several types of information which affect the behavior of forgetting resulted in improved performance.

In future work, we intend to extend this study to model forgetting behavior for each student since the tendency of forgetting differs among students. We also consider to use this model for the knowledge tracing with side information.

REFERENCES

[1] Panagiotis Adamopoulos. 2013. What Makes a Great MOOC? An Interdisciplinary Analysis of Student Retention in Online Courses. In *34th International Conference on Information Systems*. Association for Information Systems, United States. The most heavily-cited paper from the ICIS 2013 proceedings (as of August 15th, 2016).

[2] Haiqin Cheung, Lap Pongand Yang. 2017. Heterogeneous Features Integration in Deep Knowledge Tracing. In *Neural Information Processing*. Springer International Publishing, 653–662.

[3] Albert T. Corbett and John R. Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 4 (01 Dec 1994), 253–278. https://doi.org/10.1007/BF01099821

[4] Yossi Ben David, Avi Segal, and Ya'akov (Kobi) Gal. 2016. Sequencing Educational Content in Classrooms Using Bayesian Knowledge Tracing. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge (LAK '16)*. ACM, New York, NY, USA, 354–363. https://doi.org/10.1145/2883851.2883885

[5] Hermann Ebbinghaus. 1885. *Memory: A Contribution to Experimental Psychology*. Teachers Collage, Columbia University, New York.

[6] Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science* 14, 2 (1990), 179 – 211. https://doi.org/10.1016/0364-0213(90)90002-E

[7] Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction* 19, 3 (01 Aug 2009), 243–266. https://doi.org/10.1007/s11257-009-9063-7

[8] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. ACM, New York, NY, USA, 355–364. https://doi.org/10.1145/3077136.3080777

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[10] Jussi Kasurinen and Uolevi Nikula. 2009. Estimating Programming Knowledge with Bayesian Knowledge Tracing. In *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE '09)*. ACM, New York, NY, USA, 313–317. https://doi.org/10.1145/1562877.1562972

[11] Mohammad Khajah, Robert Lindsey, and Michael Mozer. 2016. How Deep is Knowledge Tracing?. In *Proceedings of the ninth International Conference on Educational Data Mining (EDM'16)*. 94–101.

[12] Byung-Hak Kim, Ethan Vizitei, and Varun Ganapathi. 2018. GritNet: Student Performance Prediction with Deep Learning. *CoRR* abs/1804.07405 (2018).

[13] Jan Papouek, Radek Pelánek, and Vit Stanislav. 2016. Adaptive Geography Practice Data Set. *Journal of Learning Analytics* 3, 2 (2016), 317–321. https://doi.org/10.18608/jla.2016.32.17

[14] Philip I. Pavlik, Hao Cen, and Kenneth R. Koedinger. 2009. Performance Factors Analysis –A New Alternative to Knowledge Tracing. In *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 531–538. http://dl.acm.org/citation.cfm?id=1659450.1659529

[15] Radek Pelánek. 2015. Modeling Students' Memory for Application in Adaptive Educational Systems. In *Proceedings of eighth International Conference on Educational Data Mining*. 480–483. https://files.eric.ed.gov/fulltext/ED560907.pdf

[16] Radek Pelánek. 2017. Bayesian Knowledge Tracing, Logistic Models, and Beyond: An Overview of Learner Modeling Techniques. *User Modeling and User-Adapted Interaction* 27, 3-5 (Dec. 2017), 313–350. https://doi.org/10.1007/s11257-017-9193-2

[17] 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 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 505–513. http://papers.nips.cc/paper/5654-deep-knowledge-tracing.pdf

[18] Yumeng Qiu, Yingmei Qi, Hanyuan Lu, Zachary A. Pardos, and Neil T. Heffernan. 2011. Does Time Matter? Modeling the Effect of Time with Bayesian Knowledge Tracing. In *Proceedings of the eighth International Conference on Educational Data Mining (EDM '11)*. 139–148.

[19] Jonathan P. Rowe and James C. Lester. 2010. Modeling User Knowledge with Dynamic Bayesian Networks in Interactive Narrative Environments. In *Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE'10)*. AAAI Press, 57–62. http://dl.acm.org/citation.cfm?id=3014666.3014678

[20] Thorsten Schodde, Kirsten Bergmann, and Stefan Kopp. 2017. Adaptive Robot Language Tutoring Based on Bayesian Knowledge Tracing and Predictive Decision-Making. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction (HRI '17)*. ACM, New York, NY, USA, 128–136. https://doi.org/10.1145/2909824.3020222

[21] Burr Settles and Brendan Meeder. 2016. A Trainable Spaced Repetition Model for Language Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1848–1858. https://doi.org/10.18653/v1/P16-1174

[22] Elena Smirnova and Flavian Vasile. 2017. Contextual Sequence Modeling for Recommendation with Recurrent Neural Networks. In *Proceedings of the 2Nd Workshop on Deep Learning for Recommender Systems (DLRS 2017)*. ACM, New York, NY, USA, 2–9. https://doi.org/10.1145/3125486.3125488

[23] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris H. Q. Ding, Si Wei, and Guoping Hu. 2018. Exercise-Enhanced Sequential Modeling for Student Performance Prediction. In *AAAI*.

[24] 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 (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 765–774. https://doi.org/10.1145/3038912.3052580