BiDKT: Deep Knowledge Tracing with BERT

Weicong Tan[1](✉), Yuan Jin[1], Ming Liu[2], and He Zhang[3]

[1] Monash University, Victoria, Australia
{weicong.tan,yuan.jin}@monash.edu
[2] Deakin University, Victoria, Australia
m.liu@deakin.edu.au
[3] ZHONGTUKEXIN CO., LTD., Beijing 100020, China
zhanghe@cnpiec.com.cn

Abstract. Deep knowledge Tracing is a family of deep learning models that aim to predict students' future correctness of responses for different subjects (to indicate whether they have mastered the subjects) based on their previous histories of interactions with the subjects. Early deep knowledge tracing models mostly rely on recurrent neural networks (RNNs) that can only learn from a uni-directional context from the response sequences during the model training. An alternative for learning from the context in both directions from those sequences is to use the bidirectional deep learning models. The most recent significant advance in this regard is BERT, a transformer-style bidirectional model, which has outperformed numerous RNN models on several NLP tasks. Therefore, we apply and adapt the BERT model to the deep knowledge tracing task, for which we propose the model **BiDKT**. It is trained under a *masked correctness recovery* task where the model predicts the correctness of a small percentage of randomly masked responses based on their bidirectional context in the sequences. We conducted experiments on several real-world knowledge tracing datasets and show that BiDKT can outperform some of the state-of-the-art approaches on predicting the correctness of future student responses for some of the datasets. We have also discussed the possible reasons why BiDKT has underperformed in certain scenarios. Finally, we study the impacts of several key components of BiDKT on its performance.

Keywords: Educational data mining · Knowledge tracing · BERT

1 Introduction

The Intelligent Tutoring System (ITS) aims to provide students with personalised learning schemes based on their respective proficiency over different teaching concepts/subjects to help them achieve better learning outcomes. Hence, the efficacy of personalisation highly depends on the accurate estimate of students' proficiency. The ITS usually requires the students to become sufficiently knowledgeable about one concept before allowing them to proceed to study the next concept [23].

W. Bao et al. (Eds.): ADHOCNETS 2021/TridentCom 2021, LNICST 428, pp. 260–278, 2022.
https://doi.org/10.1007/978-3-030-98005-4_19

Alternatively, it has also attempted to place the questions/exercises in an optimal ordering such that students with increasing levels of proficiency can tackle them progressively without being discouraged or dropping out from the study [15]. The estimates of the student proficiency can also help the ITS monitor the skill development of the students implicitly and meanwhile, give them explicit feedback on their performance under different skills/subjects on time [2].

A well-known family of approaches that can effectively estimate the student's proficiency is knowledge tracing (KT) [11]. Corbett and Anderson [4] proposed the first knowledge tracing model based on Bayesian statistics and inference, referred to as the Bayesian knowledge tracing (BKT). It estimates the student's proficiency over different teaching concepts based on a student's previous history of performance on interactive exercises [4]. They proposed that if the model could accurately predict students' future behaviours based on their performance history, it can be considered able to capture the students' proficiency on different teaching concepts. They achieved this by modelling the historical performance sequences of each student as a Markov process which tracks the students' learning states on each subject as being either mastered or not mastered. The Markov process is primarily characterised by 1) a transition probability of the subject from being not mastered to mastered, but not vice versa, and 2) conditional probabilities of correctness given different states of the mastery. These two sets of probabilities are estimated using the Bayesian inference method.

After this pioneering work, a plethora of research that aimed to extend the BKT model had been proposed. For example, Pardos and Heffernan have proposed to introduce the difficulty of the questions into the BKT model by conditioning the probabilities of correctness on the specific questions [21]. Yudelson et al. proposed to personalise the two sets of probabilities by making them specific to each student [29]). These extended models have been shown to improve the prediction accuracy on the correctness of responses of the students compared to the original BKT model. However, despite the performance improvements, these traditional knowledge tracing models are developed under the constraints imposed by the Bayesian methods (e.g., the restricted Bayesian update rules on the parameters and the difficulty of being scaled up to handle large and datasets with longer sequences [8]). As a theoretical result, their performance improvements are limited due to the lack of flexibility.

The advent of deep neural networks granted the ITS a competitive alternative for knowledge tracing. In theory, leveraging deep learning techniques for knowledge tracing can 1) avoid the heavy engineering of the input features that are required by many classical models and 2) increase the flexibility and efficacy of the student proficiency and response correctness estimation. The pioneering work of applying deep learning to knowledge tracing is from [22] where a recurrent neural network (RNN) is employed for sequentially predicting the response correctness of each student on the current questions based on their response correctness on the previous questions. In their model, the student proficiency and its transition patterns (e.g. skill mastery transitions) are modelled by the flexible and sophisticated non-linear recurrent layers instead of some statistical models.

The authors reported a substantial gain in performance from this “deep” version of the knowledge tracing, referred to as DKT, compared to BKT models. Following the DKT paradigm, many extensions have been proposed which have focused on using recurrent neural networks for the sequential prediction of the response correctness [16,19,22,28,30]. Their performance, however, is mostly comparable to that of the original DKT model. This has cast a question to deep learning for knowledge tracing; that is whether the former has the potential to contribute to a further leap in the performance of the latter. In particular, Gervet et al. [8] has found that the DKT model tends to overfit smaller datasets and are less effective than a logistic regression model with hand-crafted features. For larger datasets, DKT tends to perform better than the logistic regression model.

Recently, transformer-style deep learning models start to become prominent and lead the performance in many natural language processing and computer vision tasks. One of the most popular transformer-style models is BERT [5], which leverages stacks of fully connected transformers (as hidden layers) and random masked token prediction (as the objective) for capturing the contextual information of each input token. Unlike the RNN models which endeavour to capture sequential contexts during the training, BERT focuses on the bidirectional contexts which tend to convey more information about each input token than the sequential ones. BERT has had many extensions [18,24,27]. Nonetheless, it remains to be the most popular and effective deep learning model whose potential has never been fully exploited in the knowledge tracing domain.

Therefore, in this paper, we strive for filling this research gap by adapting BERT to the domain of knowledge tracing. To achieve this, we seek to answer the following research questions:

- **RQ1**: How can BERT be adapted to 1) take in the knowledge tracing sequential data, which consists of the (correctness of) students’ responses, the responded questions and subjects, and 2) perform random masking on the input data, which needs to be specialised for knowledge tracing?
- **RQ2**: How does BERT perform compared to the state-of-the-art DKT models and the classical BKT and logistic regression models in terms of the prediction accuracy on the response correctness?
- **RQ3** Under what conditions does BERT yield better or worse prediction performance, possibly compared with the aforementioned competing models?

Therefore, in this paper, we first reviewed the research that had been done in the knowledge tracing domain especially in how recent new deep learning techniques have been applied to the deep knowledge tracing model to improve model performance. We then proceeded to introduce our proposed deep knowledge tracing with BERT. We introduced how we constructed our model layer by layer and the training and testing strategies for our model. We also introduced a plethora of experiments we conducted to evaluate the performance of our proposed model and discussed in what circumstance our model would perform better and how the changes of some of the important parameters of the model could affect the performance of the model. Finally, we concluded the result of our research and discussed how some of the improvement and future work could be done to the research and the deep knowledge tracing domain.

2 Related Work

2.1 Bayesian Knowledge Tracing and Extensions

Corbett and Anderson [4] proposed the Bayesian Knowledge Tracing model (i.e., BKT), which attempts to capture the knowledge states of students in an ITS. It has the following modelling assumptions:

- The knowledge state is binary for a subject, either "mastered" or "non-mastered", and the state can only change in one direction: from "non-mastered" to "mastered".
- The correctness of response is conditioned on the student's knowledge state on the corresponding subject (as a conditional probability table).

The knowledge tracing is then modelled by BKT as a Markov process. As a student responds to a sequence of questions, each belonging to a subject, BKT maintains the estimated probability that each subject is in the "mastered" state; when the student answers a question, this probability will be updated simultaneously.

Based on the BKT model, there has been further research on proposing extended models or studying the properties and limitations of BKT. Pardos and Hefferman [21] proposed to introduce difficulty (level) variables to different questions. Yudelson et al. proposed to have the probabilities of the knowledge state $P(L_t)$ and the mastery transition $P(T)$ specific to each student [29].

Khajah et al. [13] have studied the limitations of the classical BKT model. They found that the performance of BKT heavily rely on whether the Markov process modelling assumptions satisfy the particular scenario to which BKT is applied. Furthermore, they pointed out that due to the modelling limitations, BKT has failed to fully exploit the recency effects where a student who has (constantly) underperformed in recent timestamps tends to underperform in the current one. Correspondingly, Galyardt and Goldin [7] have shown that integrating features of recent history into their logistic regression model can improve its predictive performance on response correctness. BKT has also failed to capture the effects of the ordering patterns (e.g. interleaved ordering) of the subjects on the response correctness. Moreover, It ignores the inter-subject similarity and its effects on the response correctness; students are more likely to master more similar subjects altogether by practising on questions under these subjects [13].

2.2 Deep Knowledge Tracing and Its Extensions

To address the problems that BKT had, Piech et al. [22] proposed to apply recurrent neural networks (RNNs) [10] to exploit more of the complex characteristics of the sequential student-question interactions in knowledge tracing. They further employed a specialised case of RNN, long-short-term memory (LSTM) networks [12], which is more capable of capturing the long-term non-linear interactions in the sequences.

Ever since the proposal of the DKT model, many extensions with more deep learning capabilities and modelling of more characteristics of knowledge tracing have been proposed. Cheung and Yang [3] proposed to incorporate heterogeneous features, such as the number of hints used and the number of attempts, into the DKT model. They used the classification and regression tree (CART) to predict whether a student will answer a question correctly based on the heterogeneous features. This prediction will be concatenated with the ground-truth value of the response correctness and the result will be encoded into a four-digit one-hot vector. This vector will then be concatenated with the original one-hot vector of the pairwise input as the new input of the model. This model has been shown to have higher AUCs compared to the DKT model.

Minn et al. [19] proposed to incorporate the dynamic clustering of students into the DKT model. They achieved this by segmenting the sequences of students' responses into multiple equal-width intervals. The model will dynamically group the students based on their estimated proficiency in different subjects using the K-means clustering for each interval. The inputs of their proposed model then include the resulting group IDs, the subject IDs, and the responses' correctness. It has been shown to achieve higher AUCs than the DKT and BKT models. This paper has also investigated the impacts of the different number of clusters and the width of time intervals on the model performance.

More recently, the self-attention mechanism has attracted attention from the deep knowledge tracing domain. Pandey and Karypis proposed the first deep knowledge tracing that applied the self-attention mechanism [20]. Ghosh et al. proposed an attentive deep knowledge tracing model that applied monotonous self-attention in the encoder from Transformer to minimise the effect of unrelated subjects and interaction distant, in terms of time, from the position required to be predicted [9].

2.3 Transformer and BERT

A major problem of the RNN is that it performs sequential prediction, which hinders the parallelisation of its training and prediction. To address this issue, Vaswani et al. [26] proposed the transformer model which completely relies on the self-attention mechanism for the sequential prediction. A transformer inherits the classical encoder-decoder architecture. Both the encoder and decoder comprise a stack of composites of a multi-head self-attention component followed by a feed-forward network. In the encoder component, each input element will be used as a query for the self-attention in which the embedding of each of them is attended to the embeddings of all the others to obtain their final latent representations, which will be used in the decoder. To handle the problem that there is no convolution and recurrence in the transformer, a positional embedding specific to each input element is added/concatenated to their embeddings. The Transformer model has outperformed many state-of-the-art sequential sequence-to-sequence deep learning models at the time in several NLP tasks. More importantly, it has provided the foundation for many powerful state-of-the-art bidirectional deep learning models to date.

BERT [5] is one of the most successful bidirectional deep learning models based on the transformer encoder. It comprises the stack of composites of the multi-head self-attention component and the feed-forward network from the encoder part of the transformer model. The output from each layer of the composite serves as the input to the composite at the next layer. Another key feature of BERT is that it is trained to recover a small percentage of randomly masked input elements from the sequences. This training phase of BERT is known as the pre-training, which aims to learn coherent and meaningful latent representations for the data.

3 Proposed Model Architecture

3.1 Problem Formulation

The knowledge tracing problem can be formulated as a sequential prediction problem: given a sequence of a student's interactions $x_1, \dots, x_T$, a DKT model needs to predict the result of the next interaction x_{T+1}, which is the correctness of the $(T+1)$-th response. In this case, the t-th interaction is denoted as $x_t = (q_t, a_t)$ where $1 \leq t \leq T$. Here, q_t refers to the t-th subject the student was practising on, and $a_t \in \{0, 1\}$ is the correctness of the student's response to the question under the t-th subject with the value 1 standing for being correct [22].

A straightforward architecture for DKT is based on the RNN-type neural networks which model uni-directional sequential contexts and are trained to use the results of all the previous interactions to predict the result of the current interaction. However, we believe that modelling uni-directional sequential contexts is not sufficient for learning the complex dynamic patterns underlying the sequences of interaction results between the students and the subjects. Instead, we should model the bidirectional contexts surrounding each interaction to let the model better figure out what patterns underlying the preceding (or subsequent) interactions might have contributed to the current interaction result (Fig. 1).

Therefore, we propose to apply and adapt BERT, a transformer-style bidirectional deep learning model, to knowledge tracing. we name the adapted BERT model **BiDKT**. Unlike the current DKT models and the self-attentive knowledge tracing model [20] which are uni-directional and thus only make use of the preceding sequence $x_1, \dots, x_t$ while predicting a_{t+1}, BiDKT also leverages the subsequent sequence from x_{t+2} to x_T to predict a_{t+1}. In the following sections, we will further introduce the key components of the BiDKT model.

3.2 Input and Embedding Layer

The input layer of BiDKT takes in each interaction in the sequences specific to each student, which consists of two tokens: the correctness token (i.e., a_t) and

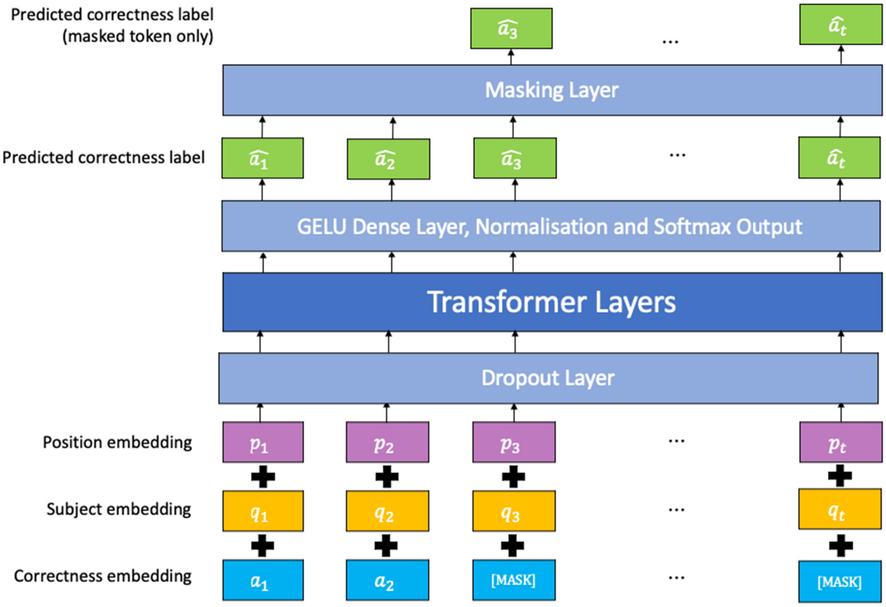

Fig. 1. The network architecture of our proposed BiDKT model.

the subject token (i.e., q_t). BiDKT inherits the transformer architecture which naturally ignores the position information of each interaction in the sequences. However, such information can be useful for revealing the knowledge states of the students. For example, students' earlier responses in their respective sequences are more likely to be erroneous, while their later responses are less likely to be so. Therefore, it is reasonable for the input layer of BiDKT to incorporate the positional information of each interaction. Therefore, the final embedding for the t-th interaction $\boldsymbol{x}_t$ is equal to the element-wise summation of three corresponding embeddings: the subject embedding $\boldsymbol{q}_t$, correctness embedding $\boldsymbol{a}_t$ and the position embedding $\boldsymbol{p}_t$. Mathematically, this can be formulated as:

$$\boldsymbol{x}_t = \boldsymbol{a}_t \bigoplus \boldsymbol{q}_t \bigoplus \boldsymbol{p}_t \tag{1}$$

In the following sections, we use $\boldsymbol{X}$ to denote the input matrix for BiDKT where:

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_T \end{bmatrix} \tag{2}$$

However, the introduction of the position embedding can limit the length of the sequence for the input layer [25]. When the sequence length exceeds the maximum length allowed by the model, it needs to be split into shorter sequences to fit it into the model [9,25]. More precisely, we denote N as the maximum length of the sequence input for BiDKT, for a sequence with the length $T > N$, we will split it into $\lceil T/N \rceil$ sequences. After the embedding layer, we apply a dropout layer to the output embeddings of each interaction to prevent the overfitting problem before feeding them to the core transformer layers.

3.3 Transformer Layers

The transformer layers of BiDKT are stacks of fully connected composites of two neural network modules: a multi-head self-attention module and a position-wise fully connected feed-forward neural network [26]. The first module is responsible for aggregating the contextual information towards each interaction from the other interactions in the same sequences. The second module takes in the aggregated information and transforms it non-linearly for the inputs of the next layer. We will elaborate on the details of both the modules in the following sections.

Multi-head Self-attention Layer. Self-attention [26] is a mechanism that can compute the embedding for each position in a sequence by relating the embeddings at all the other positions in the same sequence. More specifically, a multi-head attention mechanism with H heads refers to applying the self-attention mechanism to H consecutive chunks of the sequence separately with different sets of trainable parameters, which had been found beneficial to the performance of the model [26]. More specifically, each "head" is responsible for projecting the embeddings of the input matrix $\boldsymbol{X} \in \mathbb{R}^{T \times M}$ into a query matrix $\boldsymbol{Q} \in \mathbb{R}^{T \times M'}$, a key matrix $\boldsymbol{K} \in \mathbb{R}^{T \times M'}$ and a value matrix $\boldsymbol{V} \in \mathbb{R}^{T \times M'}$ respectively via the dot product with the corresponding trainable projection matrix, including $\boldsymbol{W_Q} \in \mathbb{R}^{M \times M'}$, $\boldsymbol{W_K} \in \mathbb{R}^{M \times M'}$ and $\boldsymbol{W_v} \in \mathbb{R}^{M \times M'}$, as follows:

$$\begin{aligned} \boldsymbol{Q} &= \boldsymbol{X}\boldsymbol{W_Q} \\ \boldsymbol{K} &= \boldsymbol{X}\boldsymbol{W_K} \\ \boldsymbol{V} &= \boldsymbol{X}\boldsymbol{W_V} \end{aligned} \tag{3}$$

In this case, the intermediate dimension for each head $M' = \frac{M}{H}$. For the i-th self-attention head where $1 \leq i \leq H$, its calculation can be formulated as follows:

$$\begin{aligned} \boldsymbol{A}_i &= \text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) \\ &= \text{Softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{M}})\boldsymbol{V} \end{aligned} \tag{4}$$

where the result $\boldsymbol{A}_i \in \mathbb{R}^{T \times M'}$. Afterwards, all the attention results across the H heads will be concatenated in the output layer of the multi-head self-attention module as follows:

$$\boldsymbol{Z} = \text{Concat}(\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_H)\boldsymbol{W_O} \tag{5}$$

where $\boldsymbol{W_O} \in \mathbb{R}^{M\times M}$ is a weight matrix for computing the final output embeddings $\boldsymbol{Z} \in \mathbb{R}^{T\times M}$ from the multi-head attention module. This module allows BiDKT to capture the bidirectional information from all the positions in each sequence during the training. Moreover, the attention computation of each head can be parallelised, which can reduce the computational complexity of the model.

Feed-Forward Neural Network (FNN) Layer. We then feed the output of the multi-head self-attention module to a position-wise fully-connected feed-forward neural network, which can be formulated as follows:

$$\mathrm{FNN}(\boldsymbol{Z}) = \mathrm{Max}(0, \boldsymbol{Z}\boldsymbol{\Phi}_1 + \boldsymbol{b}_1)\boldsymbol{\Phi}_2 + \boldsymbol{b}_2 \tag{6}$$

where $\boldsymbol{\Phi_1} \in \mathbb{R}^{M\times L}$ and $\boldsymbol{\Phi_2} \in \mathbb{R}^{L\times M}$ are the trainable weight matrices for the hidden and output layers of the FNN module, while $\boldsymbol{b}_1 \in \mathbb{R}^L$ and $\boldsymbol{b}_2 \in \mathbb{R}^M$ are the bias vectors for the two layers respectively. Notice that we set the above trainable weight matrices and bias vectors to be specific to each layer of the transformer component.

3.4 Output Layers

The output module of BiDKT starts with a dense layer with GELU (i.e. Gaussian Error Linear Unit) activation function. It is followed by a normalisation layer, whose result is passed onto the softmax function to obtain the predicted probability of response correctness corresponding to each interaction in the sequences. The output embeddings from the activated dense layer have a dimension of 4, where the indices 0 and 1 are reserved tokens respectively for the padding and the masked tokens, the index 2 represents the incorrect response, and index 3 represents the correct response. Finally, the softmax probability outputs for each interaction in the sequences will be multiplied element-wise with a binary masking layer. Its positions corresponding to the masked interactions are set to be 1 and the observed interactions are set to be 0, so that only the predictions for the "to-be-recovered" interactions will be considered in the calculation of the loss. In this case, BiDKT aims to minimise a sparse categorical cross-entropy between the correctness of the target (i.e. to-be-recovered) interactions and the corresponding softmax probability outputs.

3.5 Model Training and Testing

Training. Previous DKT models are primarily based on RNNs. Therefore, their training strategy focuses on predicting one interaction ahead. More specifically, with a sequential inputs $x_1, \ldots, x_t, 1 \leq t \leq T$ for the training, the corresponding outputs are $a_2, \ldots, a_{t+1}$ [22]. As for BiDKT, the ground-truth interactions will be masked at the input layer so that the corresponding predictions in the output layer will not be able to "see" the ground-truths but rather infer them using the surrounding bidirectional information from the sequences. Therefore, a straightforward training strategy for BiDKT is to simply predict the masked interactions at the current timestamps (rather than the ones ahead) based on the corresponding [MASK] tokens in the input layer.

More specifically, during the training, we will randomly substitute a small percentage of the correctness tokens with a [MASK] token, while the corresponding subject tokens are intact and input into BiDKT as they are. As an example, given an interaction sequence with the length of 4, i.e. $(q_1, a_1) \rightarrow (q_2, a_2) \rightarrow (q_3, a_3) \rightarrow (q_4, a_4)$, for the training, its corresponding random masked sequence to be input to the model will be in the form of $(q_1, a_1) \rightarrow (q_2, [\text{MASK}]) \rightarrow (q_3, a_3) \rightarrow (q_4, [\text{MASK}])$.

Testing. For testing, we adopted a method similar to the one in [25]. More specifically, for any sequence in the test data with the length being T', we generate T' sequences from it. Take a sequence with the length of 4 as an example. We will generate the following four sequences:

- Sequence 1: $(q_1, [\text{MASK}])$
- Sequence 2: $(q_1, a_1) \rightarrow (q_2, [\text{MASK}])$
- Sequence 3: $(q_1, a_1) \rightarrow (q_2, a_2) \rightarrow (q_3, [\text{MASK}])$
- Sequence 4: $(q_1, a_1) \rightarrow (q_2, a_2) \rightarrow (q_3, a_4) \rightarrow (q_4, [\text{MASK}])$.

In each of the above sequences, we mask only the correctness token in the last position for the model to predict, given all the previous interactions and the subject token at the current interaction.

It is worth noticing that the training and testing strategies of our model have some inconsistency in that the former one aims to predict the tokens masked at arbitrary positions in the sequences while the latter aims to predict the tokens masked at the last positions. Such inconsistency could possibly affect the performance of BiDKT adversely.

To address the above issue, during the training, we randomly sample a certain percentage of the sequences and only have their correctness tokens masked at the last positions. In other words, their masking strategy is now the same as that used for the test data. This method can be viewed as a fine-tuning step for BiDKT and can potentially improve the performance of the model.

4 Experiments

In this section, we evaluate the efficacy of our proposed model by comparing it with several state-of-the-art BKT and DKT models across 8 real-world datasets. The datasets are provided by Ghosh et al. (2020)[1] and Gervet et al. (2020)[2].

[1] https://github.com/arghosh/AKT/tree/master/data.

[2] https://github.com/theophilee/learner-performance-prediction.

4.1 Datasets

The details of these datasets are listed as follows:

- **The ASSISTment dataset in 2009, 2012, 2015, 2019.** The ASSISTment (**ASSIST**ing and assess**ment**) datasets are collected from a system utilised in the United States of America for high school mathematics classes. Each record in the dataset comprises the student's mastery status on the concept, timestamp of the response, the teaching concept associated with the question, etc. [6]. ASSISTment 2009 has been chosen to be the benchmark dataset for knowledge tracing problem in the past decade.
- **Statics 2012.** It is a dataset of the log data of ITS for a college-level engineering subject [14].
- **Algebra 2005 and Bridge to Algebra 2006.** These datasets are originally for KDD Cup 2010, a competition of data mining. The task of the competition was to predict students' correctness on mathematical exercises by learning from their log data from the Intelligent Tutoring Systems[3]. Each record comprises the hierarchy of curriculum levels containing the exercise, the identified concepts that are used in an exercise (where available), whether the student answered it right at the first go, etc.
- **Spanish.** It is a set of log data of high school students learning Spanish on an ITS [8,17]

Tables 1 and 2 summarise the key statistics of these datasets.

Table 1. Details of the data provided by Gervet et al. (2020); The average sequence length is abbreviated in the last row of the table.

	Statics	Assist09	Assist12	Assist15	Assist17	Spanish	Bridge06	Algebra05
Size	189,297	278,336	2,682,211	656,154	934,368	578,726	1,817,393	606,983
# of students	282	3,114	22,589	14,228	1,708	182	1,130	567
# of subjects	1,223	149	265	100	411	221	550	271
Avg_seq_len	636	32	59	31	440	2,924	1,373	581

Table 2. Information of the data provided by Ghosh et al. (2020); The average sequence length is abbreviated in the last row of the table.

	Statics	Assist09	Assist15	Assist17
Size	189,297	325,637	683,801	942,816
# of students	333	4,151	19,840	1,709
# of subjects	1,223	110	100	102
Avg seq len	568	78	34	551

[3] https://pslcdatashop.web.cmu.edu/KDDCup/rules.jsp.

4.2 Baselines and Metrics

The area under the receiver operating characteristic curve (AUC) has been widely used as the benchmark score for the comparison of model performance. Therefore, we used AUC as the performance score and compared the performance of our model with the results from Ghosh et al. (2020) and Gervet et al. (2020) by respectively testing our model on the pre-processed data they provided [8,9]. They also respectively re-implemented a plethora of baseline models by themselves. More specifically, the context-aware knowledge tracing model in Ghosh et al. (2020) was the state of the art [9]; and Gervet et al. conducted comprehensive experiments over different existing models and datasets [8]. We listed the datasets they provided and their chosen baselines in Table 3. Non-KT baseline models (e.g., models based on Item-Response Theory and Performance Factor Analysis) evaluated in Gervet et al. (2020) will not be listed, but we kept their proposed logistic regression model and compared it with our model in the experiments.

Table 3. Details and baseline models in Ghosh et al. (2020) and Gervet et al. (2020). Non-KT baseline models are not listed.

	Ghosh et al. (2020)	Gervet et al. (2020)
Dataset	ASSISTment 2009, 2015, 2017 and Statics 2012	ASSISTment 2009, 2015, 2012, 2017, Statics 2012, Bridge to Algebra 2006, Algebra 2005 and Spanish
Baseline models	BKT+ [29], DKT [22], DKT+ [28], SAKT [20], DKVMN [30]	BKT [4], BKT+ [29], DKT [22], SAKT [20]

4.3 Experiment Settings

As mentioned in Sect. 3.5, if a sequence is longer than a certain length, we will split it into several smaller sequences to fit in our model. To conduct 5-fold cross-validation, we have split each dataset into three parts: 60% of the data to be used as the training set, 20% to be used as the validation set for optimizing the hyper-parameters and for performing the early stopping, and the remaining 20% to be used as the test set to evaluate the competing models.

We have implemented BiDKT with Keras[4], and the structure of its transformer layers was adapted from Keras-BERT[5]. Adam optimiser was used for training the BiDKT model [1]. The implementations of all the baseline models are provided by Gervet et al. [8] and Ghosh et al. [9]. All the experiments are

[4] https://github.com/keras-team/keras.

[5] https://github.com/CyberZHG/keras-bert.

conducted on an NVIDIA V100 GPU with 16 GB memory on the M3 cluster (a high-performance computing cluster maintained by Monash University)[6].

We conducted a grid search across the hyper-parameter candidate sets specified in Table 4 to find the best one that can optimise the average model performance over the 5 validation folds of each dataset. We found the following best hyper-parameter set with 16 as the batch size, 200 as the maximum sequence length, 0.1 as the dropout rate, 1 as the number of self-attention heads, 2 as the number of transformer layers, 16 as the embedding dimension, 64 as the number of hidden neurons for the feed-forward networks, 0.15 as the masking rate (i.e. the probability of a correctness token being substituted by a [MASK] token) and 0.25 as the fine-tuning rate (i.e. the probability of a sequence only being masked at the last position in a training batch). In the later section, we will have a more detailed discussion about how the masking rate and fine-tuning rate will affect the model performance.

Table 4. Hyperparameters experimented

Hyperparameter	Values experimented
Batch size	8, 16, 24, 32, 64
Maximum sequence length	100, 200, 300
Dropout rate	0.1, 0.25, 0.5
Learning rate	1e−6, 5e−6, 1e−5, 5e−5, 1e−4
Number of self-attention heads	1, 2, 4, 8, 12, 16
Number of Transformer layer	1, 2, 4, 8, 12
Embedding dimension	16, 24, 64, 128, 192, 256
Hidden dimension	64, 96, 256, 512, 768, 1024
Mask rate	0.1, 0.50.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9
Fine-tune rate	0.1, 0.50.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9

4.4 Results and Discussion

In this section, we present the results of the competing models across the different datasets in Table 5 and 6. Ghosh et al. (2020) reported two AKT models similar in the core layers but applied different encoding mechanisms for the input (i.e. one with Rasch encoding and one without) [9]. On the ASSISTment 2009 and 2017 datasets, to which the Rasch encoding can be applied, the AKT model with such encoding had achieved better performance than the one without. Therefore, we only reported the results with the Rasch encoding on these two datasets.

It can be observed from Table 5 that BiDKT has outperformed the BKT model on the Statics 2012, the Algebra 2005 and the Spanish datasets. It has also outperformed DKT and SAKT on the Spanish dataset. It is also interesting

[6] https://www.massive.org.au/about/.

Table 5. Performance (AUC) comparison of BiDKT and the experiment result from Gervet et al. (2020). NA refers that the data is not provided or the experiment had not been conducted in the original paper. "LR" stands for "logistic regression".

Dataset	BKT	BKT+	LR	DKT	SAKT	BiDKT
Statics 2012	0.73	0.811	0.819	0.829	0.813	0.772
ASSISTment 2009	0.63	0.759	0.772	0.757	0.756	0.700
ASSISTment 2012	NA	NA	0.751	0.771	0.732	0.689
ASSISTment 2015	NA	0.701	0.702	0.731	0.730	0.674
ASSISTment 2017	NA	0.710	0.714	0.770	0.722	0.632
Bridge to Algebra 2006	NA	NA	0.803	0.790	0.784	0.763
Algebra 2005	0.62	NA	0.83	0.821	0.801	0.777
Spanish	0.83	0.851	0.863	0.832	0.831	0.835

Table 6. Performance (AUC) comparison of BiDKT and the experiment result from Ghosh et al. (2020). NA refers to that the result is not reported in the original paper

Dataset	BKT+	DKT	DKT+	DKVMN	SAKT	AKT	BiDKT
Statics 2012	0.75	0.8233	0.8301	0.8195	0.8029	0.8265	0.7785
ASSISTment 2009	0.69	0.817	0.8024	0.8093	0.752	0.8346	0.7651
ASSISTment 2015	NA	0.731	0.7313	0.7276	0.7212	0.7828	0.6766
ASSISTment 2017	NA	0.7263	0.7124	0.7073	0.6569	0.7702	0.5978

to see that BiDKT has outperformed SAKT on the ASSISTment 2009 dataset provided by Ghosh et al. (2020) but not on the same dataset provided by Gervet et al. (2020) (Table 6).

On the other datasets from the two sources, we can see that there is a notable performance gap between BiDKT and some of the state-of-the-art DKT models (e.g. AKT and SAKT). However, it is also worth noticing that in the original paper of SAKT [20], the authors reported an AUC of 0.848 on the ASSISTment 2009 dataset and 0.857 on the ASSISTment 2015 dataset. In comparison, both Ghosh et al. (2020) and Gervet et al. (2020) cannot reproduce the original performance.

Despite the performance gap on some of the datasets, we believe that BiDKT still bears the potential to further improve its performance. BERT has demonstrated its efficacy in the sequential recommendation, a similar domain to knowledge tracing [25]. The only difference is that the datasets used in this case contain hundreds of millions of responses and millions of users and items, which are much larger than popular knowledge tracing benchmark datasets. Both Gervet et al. (2020) and Ghosh et al. (2020) have pointed out that self-attentive models might require a large amount of data to be trained properly [8,9]. In comparison, the datasets used in our experiments are relatively small.

Furthermore, we hypothesise that the gap performance exists because the students' future performance is only dictated by their performance in the recent past but not by any longer one. Another possible reason is that the dynamic patterns underlying the interaction sequences are not sufficiently complex for our model to fully exploit to allow it to outperform simpler models.

4.5 The Impact of Masking Rate

The mask rate refers to the probability of whether a correctness token will be substituted by a [MASK] token. The mask rate will decide how many tokens in a training sequence the model should predict. On one hand, if it were too large, it would impose extra difficulty for the model to capture the pattern of the sequence; on the other hand, if it were too small, the robustness of the model would be impaired [25]. In this experiment, we kept fine-tune rate at 0.25 and changed the value of the mask rate to investigate how it affects the performance of the model.

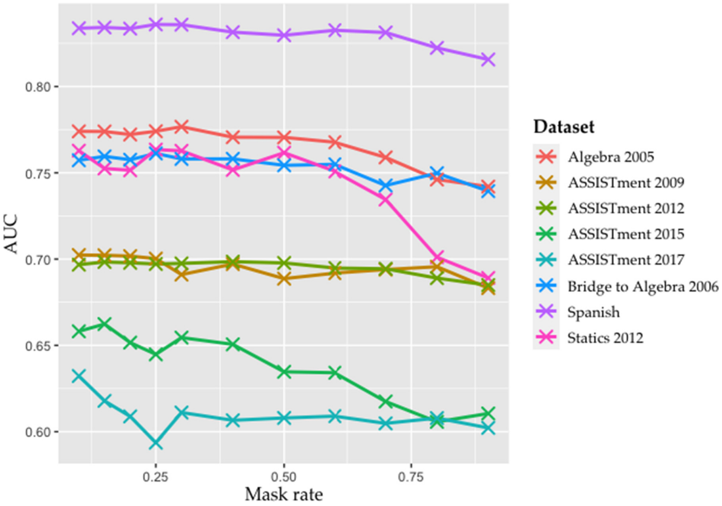

Fig. 2. The performance (AUC) of BiDKT with different masking rates across the different datasets.

As we can tell from Fig. 2, generally, the performance of BiDKT does not monotonously grow or decline within the domain of [0.1, 0.9], which can lead us to the same conclusion that the change of mask rate does not always result in performance improvement or decline, as per [25]. When the mask rate is larger than 0.3, generally speaking, the performance of BiDKT declines when the mask rate continues to grow.

4.6 The Impact of Fine-Tuning Rate

The fine-tune rate refers to the probability in which a sequence will have the correctness token masked only in the last position. Similar to the mask rate, we conjectured that it can either be too small or too large. On one hand, if it were too small, the discrepancy between the training task and the testing task would be large; on the other hand, if it were too large, we cannot fully leverage the power of BERT to capture the learning characteristics of the students by predicting correctness tokens from their upstream and downstream context.

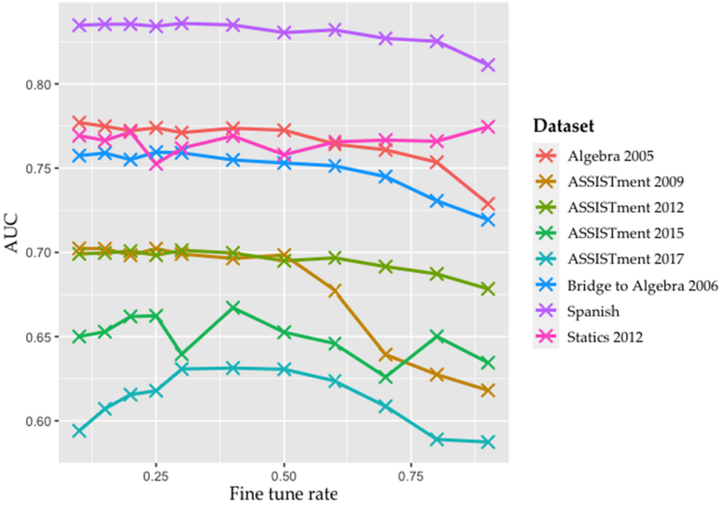

Fig. 3. The performance of BiDKT with different fine-tuning rates across the different datasets.

As we can tell from Fig. 3, when we changed the fine-tune rate from 0.1 to 0.9, the performance of the model did not monotonously grow or decline. This proved our aforementioned hypothesis.

4.7 Limitations of Our Study

Due to the time and resource limitation of this paper, we can only improve and evaluate our work within a certain scope. One of the limitations of this paper is that we did not investigate the root cause of the performance gap. We only empirically analysed why the gap exists. Another limitation of our research is that the granularity of the grid search for optimal hyperparameters was very high.

5 Conclusion and Future Work

In this paper, we proposed BiDKT, a deep knowledge tracing model based on BERT. We introduced the structure of BiDKT in details and how we implemented the model. We conducted a series of experiments to evaluate the overall performance of our model and analyse how some of the important parameters affect the performance of the model. Our model outperformed some of the current deep knowledge tracing models in certain scenarios. To our knowledge, even though a plethora of extensive BERT models have been proposed and have shown excellent performance in their respective settings, most of them are still models for natural language processing tasks. Our work extended the usage of the BERT model to the knowledge tracing domain, and more broadly, the non-NLP sequential prediction domain.

There are many possibilities for future research in the deep knowledge tracing domain. Currently, many DKT models have tried to incorporate more features of a student's response (e.g. the text of the exercise as side information [19]) or a more sophisticated method to encode the input (e.g. the Rasch encoding) [9]) We consider these research directions probable to be integrated with BERT models for higher performance. Another possible research direction could be training and testing the model on the EdNet dataset, which is larger in size and has a larger number of students but has not been widely used as a benchmark dataset.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint: arXiv:1607.06450 (2016)
2. Bull, S., Kay, J.: Open learner models. In: Nkambou, R., Bourdeau, J., Mizoguchi, R. (eds.) Advances in Intelligent Tutoring Systems. Studies in Computational Intelligence, vol. 308, pp. 301–322. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14363-2_15
3. Cheung, L.P., Yang, H.: Heterogeneous features integration in deep knowledge tracing. In: Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, E.S. (eds) Neural Information Processing. ICONIP 2017. Lecture Notes in Computer Science, vol. 10635, pp. 653-662. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70096-0_67
4. Corbett, A.T., Anderson, J.R.: Knowledge tracing: modeling the acquisition of procedural knowledge. User Model. User-Adapt. Interact. **4**(4), 253–278 (1994)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint: arXiv:1810.04805 (2018)
6. Feng, M., Heffernan, N., Koedinger, K.: Addressing the assessment challenge with an online system that tutors as it assesses. User Model. User-Adapt. Interact. **19**(3), 243–266 (2009)
7. Galyardt, A., Goldin, I.: Move your lamp post: recent data reflects learner knowledge better than older data. J. Educ. Data Mining **7**(2), 83–108 (2015)
8. Gervet, T., et al.: When is deep learning the best approach to knowledge tracing? JEDM—. J. Educ. Data Mining **12**(3), 31–54 (2020)

9. Ghosh, A., Heffernan, N., Lan, A.S.: Context-aware attentive knowledge tracing. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 2330–2339 (2020)
10. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
11. Hernández-Blanco, A., Herrera-Flores, B., Tomás, D., Navarro-Colorado, B.: A systematic review of deep learning approaches to educational data mining. Complexity **2019**, 1–22 (2019)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
13. Khajah, M., Lindsey, R.V., Mozer, M.C.: How deep is knowledge tracing? arXiv preprint: arXiv:1604.02416 (2016)
14. Koedinger, K.R., Baker, R.S., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J.: A data repository for the EDM community: The PSLC datashop. Handbook Educ. Data Mining **43**, 43–56 (2010)
15. Koedinger, K.R., Brunskill, E., Baker, R.S., McLaughlin, E.A., Stamper, J.: New potentials for data-driven intelligent tutoring system development and optimization. AI Mag. **34**(3), 27–41 (2013)
16. Lee, J., Yeung, D.Y.: Knowledge query network for knowledge tracing: how knowledge interacts with skills. In: Proceedings of the 9th International Conference on Learning Analytics and Knowledge, pp. 491–500 (2019)
17. Lindsey, R.V., Khajah, M., Mozer, M.C.: Automatic discovery of cognitive skills to improve the prediction of student learning. In: Advances in Neural Information Processing Systems, pp. 1386–1394 (2014)
18. Liu, Y., et al.: Roberta: a robustly optimized BERT pretraining approach. arXiv preprint: arXiv:1907.11692 (2019)
19. Minn, S., Yu, Y., Desmarais, M.C., Zhu, F., Vie, J.J.: Deep knowledge tracing and dynamic student classification for knowledge tracing. In: 2018 IEEE International Conference on Data Mining (ICDM), pp. 1182–1187. IEEE (2018)
20. Pandey, S., Karypis, G.: A self-attentive model for knowledge tracing. arXiv preprint: arXiv:1907.06837 (2019)
21. Pardos, Z.A., Heffernan, N.T.: KT-IDEM: introducing item difficulty to the knowledge tracing model. In: Konstan, J.A., Conejo, R., Marzo, J.L., Oliver, N. (eds.) UMAP 2011. LNCS, vol. 6787, pp. 243–254. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22362-4_21
22. Piech, C., et al.: Deep knowledge tracing. In: Advances in Neural Information Processing Systems, pp. 505–513 (2015)
23. Ritter, S., Yudelson, M., Fancsali, S.E., Berman, S.R.: How mastery learning works at scale. In: Proceedings of the Third (2016) ACM Conference on Learning@ Scale, pp. 71–79 (2016)
24. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In: The 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC2) co-located with NeurIPS 2019 (2019)
25. Sun, F., et al.: BERT4Rec: sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1441–1450 (2019)
26. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)

27. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: XLNet: generalized autoregressive pretraining for language understanding. In: Advances In Neural Information Processing Systems, pp. 5754–5764 (2019)
28. Yeung, C.K., Yeung, D.Y.: Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In: Proceedings of the Fifth Annual ACM Conference on Learning at Scale, pp. 1–10 (2018)
29. Yudelson, M.V., Koedinger, K.R., Gordon, G.J.: Individualized Bayesian knowledge tracing models. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 171–180. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39112-5_18
30. Zhang, J., Shi, X., King, I., Yeung, D.Y.: Dynamic key-value memory networks for knowledge tracing. In: Proceedings of the 26th International Conference on World Wide Web, pp. 765–774 (2017)