# KT-Bi-GRU: Student Performance Prediction with a Bi-Directional Recurrent Knowledge Tracing Neural Network

Marina Delianidi
International Hellenic University
Greece
d.marina@iee.ihu.gr

Konstantinos Diamantaras
International Hellenic University
Greece
kdiamant@ihu.gr

Student performance is affected by their knowledge which changes dynamically over time. Therefore, employing recurrent neural networks (RNN), which are known to be very good in dynamic time series prediction, can be a suitable approach for student performance prediction. We propose such a neural network architecture containing two modules: (i) a dynamic sub-network including a recurrent Bi-GRU layer used for knowledge state estimation, (ii) a non-dynamic, feed-forward sub-network for predicting answer correctness based on the current question and current student knowledge state. The model modifies our previously proposed architecture and is different from all other existing models because it estimates the student's knowledge state considering only their previous responses. Thus the dynamic sub-network generates more stable knowledge state vector representations since they are independent of the current question. We studied both single-skill and multi-skill question scenarios and employed embeddings to represent questions and responses. In the multi-skill case the initialization of the question embedding matrix with pretrained word-embeddings is found to improve model performance. The experimental results showed that our current KT-Bi-GRU model and the previous one have similar performance while both surpassed the performance of previous state-of-the-art knowledge tracing models for five out of seven datasets where in some cases, the difference is quite noticeable.

**Keywords:** knowledge tracing, student performance prediction, dynamic neural networks, knowledge state

## 1. INTRODUCTION

Knowledge tends to change with time and this dynamic property has been a central subject of study in the educational data mining field in recent decades. Practically, the student's knowledge changes in a positive way when the student learns a learning object (i.e. a concept or skill) or in a negative way when the student forgets. Maintaining, over time, the probability that the student has mastered various learning objects is called Knowledge Tracing (KT) (Corbett and Anderson, 1994). The learning objects in a specific area, for example, "Algebra", "Geometry", "Physics" etc, are called knowledge components (KC) and compose the picture of the knowledge state (KS) of the student. The process of estimating students' performance helps to assess their knowledge state and can contribute to the future improvement of their learning performance by

tracing the current knowledge. During the learning process through an electronic Intelligent Tutoring System (ITS), students' activities are implicitly recorded in logs as they interact with the system. These log files can be used during the knowledge tracing process in order to assess the evolution of the student's knowledge state over time.

In general, according to (Liu et al., 2021), knowledge tracing approaches can be classified into three main categories: (i) Models based on statistical or probabilistic methods such as the BKT (Corbett and Anderson, 1994) which is implemented through the Hidden Markov model; (ii) Logistics models such as LFA (Cen et al., 2006) and PFA (Pavlik et al., 2009) and (iii) Deep Learning models such as DKT, DKMVN, Deep-IRT (Piech et al., 2015) (Zhang et al., 2017) (Yeung, 2019). Recently, methods from the first two categories were combined with deep learning methods, for example DBN (Käser et al., 2017) and DPFA (Pu et al., 2021), achieving improved results. Additionally, convolutional neural networks (Yang et al., ), (Wang et al., 2020), (Shen et al., ),(Ma et al., 2021) and graph neural networks models (Liu et al., 2021), (He et al., 2021) have been recently proposed for the KT task. The aim of all the works is the optimal representation of the knowledge state and the prediction of student performance so that the learning process can be improved and adapted to the student's learning needs.

In the basic setup, a student interacts with an ITS and gives answers to questions $q_i$. The variable $r_i \in \{0, 1\}$ indicates the correctness of the answer, where 1 means correct and 0 means wrong answer. At each time instance, the student's KS is formed in relation to the skill that is examined based on the correct or incorrect answer. The questions are related to one or more skills taken from some skill set $S = \{s_1, s_2, \ldots, s_n\}$. Our prediction task is formalized as follows: given for each student $j$ the past interaction sequence $X^j = (x_1^j, x_2^j, \ldots, x_i^j, \ldots, x_{t-1}^j)$, where $x_i^j = \{q_i^j, r_i^j\}$ is the interaction pair at the $i$-th time instance, we must accurately predict the correctness $r_t^j$ of the answer given by student $j$ to the current question $q_t^j$. Note that each student may have a different sequence of questions. An example of the prediction task based on students' interactions can be seen in Figure 1. A more accurate prediction $\hat{r}_t$ corresponds to a better assessment of the student's knowledge level in different skills at the time when the question $q_t$ will be asked.

The student knowledge state at time $t$ can be determined by the correctness $r_i$ of the responses to the previous questions $q_i, i = 1, \ldots, t-1$, and should not be a function of $q_t$ since we typically do not know the response to this question. However, in prior works (Corbett and Anderson, 1994), (Pavlik et al., 2009), (Piech et al., 2015), (Zhang et al., 2017), (Yeung, 2019), (Delianidi et al., 2021) employing neural networks for student performance prediction, the current question $q_t$ is not separated from the previous ones. Even though, using this approach, it is still possible to predict $r_t$, it is difficult to identify a part of the model that represents the current student knowledge state. In our earlier work (Delianidi et al., 2021) we followed this mainstream approach proposing a deep neural model based on a Bi-directional Gated Recurrent Unit layer.

In this work we propose to model $r_t$ taking into account the student's KS up to the time instance $t-1$. To this end, we take two steps: first, we separately estimate a representation vector $\mathbf{v}_t$ of the current KS using a recurrent neural layer using the student interaction history up to time $t-1$ and without knowing $q_t$. Then we combine $\mathbf{v}_t$ with the representation of $q_t$ and apply a feed-forward classification neural network to actually estimate $r_t$. The advantage of this approach is that having $\mathbf{v}_t$ we can combine with any question $q$ and predict the correctness of the student's answer to $q$ at any time $t$. This allows us to monitor the student's performance in questions relating to specific skills and identify student's weaknesses. Potentially this model

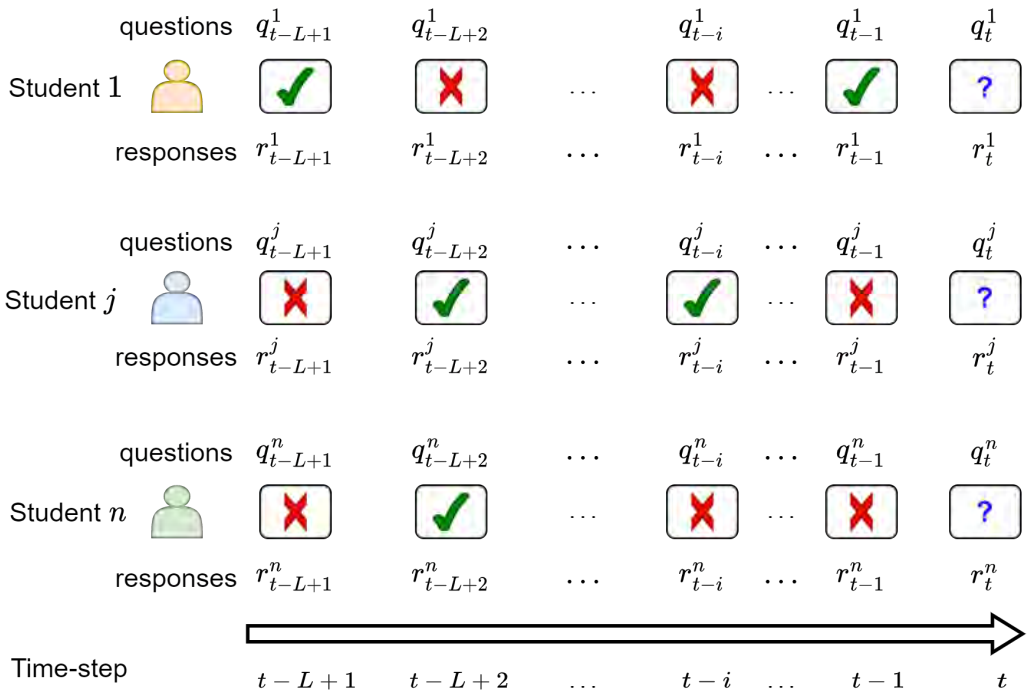



Figure 1: Prediction task: based on the history window with length $L$ of the previous interactions, the response correctness $r_t$ to the question $q_t$ at the current time instance $t$ must be predicted for every student.

could be used for other important tasks such as building an educational recommendation system. The proposed model is tested on seven public datasets with single-skill or multi-skill questions in the specific area of mathematics.

The contributions of this work can be summarized as follows:

1. We propose a new neural network architecture KT-Bi-GRU incorporating a bi-directional recurrent layer whose output represents the student's knowledge state based on his/her previous interactions. The recurrent layer output is used in combination with the current question to predict the correctness of the student response.

2. We studied the initialization of question embeddings by comparing pretrained vs. random vectors. We found that in the case of multi-skill questions, Word2Vec (W2V) initialization is advantageous compared to random initialization. We notice a differentiation in the utility of the W2V embedding initialization depending on whether the questions involve multiple skills or a single skill.

3. We compare the proposed model performance against earlier, state-of-the-art neural networks, including our own earlier Bi-GRU RNN model. The experimental evaluation is performed on seven different datasets.

4. The proposed architecture facilitates the estimation of the student knowledge state, a feature which could be potentially useful for tasks such as student clustering or educational content recommendation.

The rest of this paper is organized as follows. Section 2 provides the literature review of the existing student performance prediction models using the KT task specifically based on deep learning techniques. In Section 3, referring to our previous work (Delianidi et al., 2021), we present the new deep learning based knowledge tracing model with differentiated inputs. The datasets we used in our experiments are described in Section 4, while the experimental settings and parameters are presented in Section 5. The experimental results, in comparison to the deep learning based state-of-the-art models DTK, DKVMN, Deep-IRT and SAKT, are discussed in Section 6. We conclude the paper and present the future work in Section 7.

## 2. LITERATURE REVIEW

The tracing of student knowledge using Bayesian Networks has been introduced by (Corbett and Anderson, 1994) and it is referred to as Bayesian Knowledge Tracing (BKT). The method belongs to the probabilistic knowledge modeling techniques. Due to the continuous development and wide use of e-learning, there has been an increasing interest in this topic which resulted in the development of a variety of approaches. The so-called logistic KT methods including Learning Factor Analysis (LFA) (Cen et al., 2006) and Performance Factor Analysis (PFA) (Pavlik et al., 2009) have been shown to achieve better performance compared to the BKT model. In logistic KT models the probability of a correct answer is represented by a logistic function involving student and knowledge components (KC) parameters.

Deep Neural Networks (DNN) and, especially, Recurrent Neural Networks (RNN) have contributed to the development of the most efficient knowledge tracing models to date. The first knowledge tracing model utilizing RNNs, and specifically the LSTM model, was DKT (Piech et al., 2015) introduced in 2015. Having as input the one-hot encoded skill tags and the associated responses, the neural network is trained to predict the correctness of the next student's response. The hidden state of LSTM can be considered as the latent state of a student's knowledge and can transfer the information of previous interactions to the output level. The output level of DKT, (depending on the question or the skill), estimates the probability of answering the question related to a specific knowledge component correctly.

The Dynamic Key Value Memory Network (DKVMN) (Zhang et al., 2017), is another approach that uses a modified memory augmented neural network (MANN) (Miller et al., 2016) for knowledge tracing, attempting to capture the relationship between different concepts. The DKVMN model outperforms the DKT model. To encode students' knowledge state, DKVMN uses memory slots as key-value pairs in which learning or forgetting a particular skill are controlled through read and write operations. The concepts are stored in the key component which is fixed during testing, while the value component is updated when a concept state changes. This means that when a student masters a concept in a test, the value component is updated based on the correlation between the corresponding concept and the exercises.

Recently the DKVMN model has been extended by the Deep-IRT model (Yeung, 2019). In Deep-IRT the capabilities of the DKVMN are combined with the Item Response Theory (Hambleton et al., 1991) in order to measure both student ability and question difficulty. Another model, named Sequential Key-Value Memory Networks (SKVMN) (Abdelrahman and Wang, 2019), combines DKVMN with Hop-LSTM, a variation of the LSTM architecture. SKVMN utilizes the Hop-LSTM ability to discover sequential dependencies between exercises but it skips some LSTM cells to approach previous concepts that are considered relevant. In this way, SKVMN tried to overcome the problem of DKVMN to capture long-term dependencies on

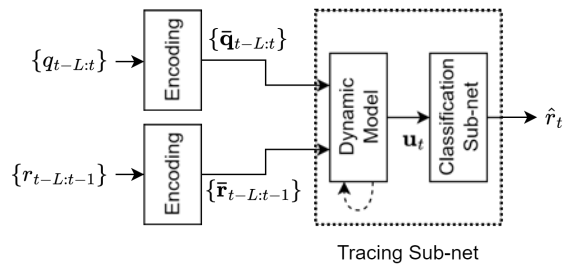
the sequences of exercises and generally on sequential data. Authors in (Liu et al., 2019) used the text content of the exercises and the students' responses to the exercises to predict students' performance. The embedding vector of each exercise is constructed from the exercise content and is trained with a Bidirectional LSTM recurrent network. To predict a student's performance, two variants are applied using an attention mechanism (EERNNA) and a Markovian property (EERNNM). The experiments have been performed on a dataset with mathematical content from an online learning system.

The attention mechanism (Vaswani et al., 2017) has been also used in other KT models. One of such model is the Self Attentive Knowledge Tracing (SAKT) (Pandey and Karypis, 2019). SAKT consists of three layers. An embedding layer is used for student interactions and question representation. Given a KC, the relevant KCs are identified from the student's past activities using the second layer which employs a self-attention mechanism. Then the third feed-forward layer is used for predicting the student response. Another model based on the attention mechanism, called Attentive knowledge tracing (AKT), was proposed in (Ghosh et al., 2020). AKT couples flexible attention-based neural network models with cognitive and psychometric models. AKT consists of two self-attentive encoders: (a) the question encoder with contextualized representations of each question, given the sequence of questions the learner has previously practiced on, and (b) the knowledge encoder, which produces modified, contextualized representations of the knowledge the learner acquired while responding to past questions.

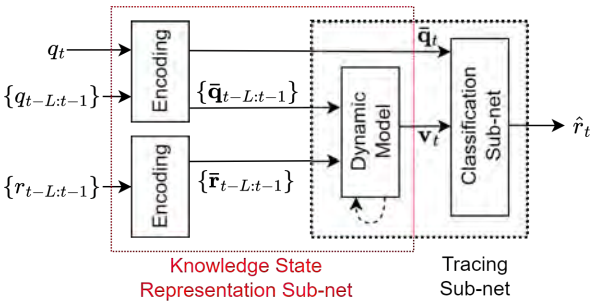
Other KT approaches combine earlier proposed models with deep learning techniques to enhance them with the ability to dynamic knowledge modeling. For example, the Dynamic BKT (DBKT) (Käser et al., 2017), introduced in 2017, models knowledge by taking into account the correlations between KCs and predicting students' performance based on performance in previous relevant KCs. Similarly, the Deep Performance Factors Analysis (DPFA) (Pu et al., 2021) model published in 2021, combines the PFA with deep learning models in order to improve it and is summarized as a logistic regression model based on the affinity of previous and future items.

Recent research has presented knowledge tracing models using convolutional neural networks (CNN) (Yang et al., ), (Wang et al., 2020), (Shen et al., ), (Ma et al., 2021) and graph neural networks (GNN) (Liu et al., 2021), (He et al., 2021). Other works (Yang et al., 2021), (Song et al., 2021) employ CNN and GNN models incorporating recurrent neural layers, such as LSTM, in order to achieve better performance.

In our previous work (Delianidi et al., 2021) we proposed a dynamic KT model using a special type of Recurrent Neural Network called Bidirectional Gated Recurrent Unit (Bi-GRU). Comparing the GRU network (Cho et al., 2014) with the LSTM network, the GRU has fewer parameters although it typically has similar performance. Bidirectional RNNs (Schuster and Paliwal, 1997), such as the Bi-GRU, connect two hidden layers of opposite directions at the same output. This structure provides information to the output layer from the future states (backward direction) and from the past state (forward directions) at the same time. The output of the Bi-GRU network combines and normalizes the outputs of the forward and backward hidden layers at each instant. In this paper, we give a short description of our previous work and present a new dynamic KT model named KT-Bi-GRU. The Bi-GRU neural network is a basic component of our both models' architecture.




(a) The early Bi-GRU model architecture (Delianidi et al., 2021).




(b) The KT-Bi-GRU model architecture.

Figure 2: The Bi-GRU models' general architectures. The vector $\mathbf{v}_t$ represents the student's Knowledge State at the current time instance.

## 3. Dynamic Neural Network Architecture

As mentioned earlier, knowledge is formed dynamically over time. Recurrent neural networks have been known to be very good at modeling and predicting dynamic processes and so they can be used to model the students' knowledge state. Student response prediction should take into account information regarding the estimated student knowledge state. Since responses can be either correct (1) or wrong (0) we approach the task of predicting them as a binary classification problem. Our model architecture consists of the following parts:

- the input sequences $\{q_{t-L}, \ldots, q_t\}$ and $\{r_{t-L}, \ldots, r_{t-1}\}$ where $q_i$ denotes the question id at time $i$ and $r_i$ denotes the correctness of the response at $i$. The hyper-parameter $L$ indicates the length of the time window used to predict the response $r_t$.
- an *Encoding* component incorporating two embedding layers which produce vector representations $\bar{\mathbf{q}}_i$, $\bar{\mathbf{r}}_i$ for $q_i$ and $r_i$, respectively.
- a *Dynamic* component which traces the student's knowledge state using the time sequences $\{\bar{\mathbf{q}}\}$ and $\{\bar{\mathbf{r}}\}$
- a *Classification* component which predicts the student response based on the output of the dynamic component.

The above architecture does not specify an important detail: whether the current question, $q_t$, should participate in the sequence that feeds the dynamic tracing component. In our earlier work (Delianidi et al., 2021) we followed the above design philosophy with $q_t$ actually feeding the dynamic component (Fig. 2a). However, in this case, it is difficult to associate the output of the dynamic component with the student's knowledge state at time $t$ since the current KS should not depend on the current question $q_t$ which is not answered yet.

For this reason, we propose here an alternative architecture depicted in Figure 2b where the current question does not feed the dynamic tracing component but is only used as input for the classification component in order to predict $r_t$. The architectures of the two models are described in detail below.

### 3.1. EARLY DYNAMIC BI-GRU MODEL

The architecture of our earlier proposed recurrent model Bi-GRU[1] (Delianidi et al., 2021) is shown in Figure 2a. The correctness $r_t$ of the answer at time $t$ is defined as a function of the student's previous interactions $(q_i, r_i), i = t-1, t-2, \ldots$, and the current question $q_t$:

$$r_t = \phi(q_t, q_{t-1}, q_{t-2}, \ldots, r_{t-1}, r_{t-2}, \ldots) + \epsilon_t \tag{1}$$

where $\epsilon_t$ is the prediction error.

In order for the model to remember arbitrarily long sequences the Dynamic component of the architecture incorporates a recurrent Bidirectional GRU (Bi-GRU) layer. A Bi-GRU layer consists of two Gated Recurrent Unit (GRU) layers, one for the forward time direction and another for the backward time direction (Fig. 3).

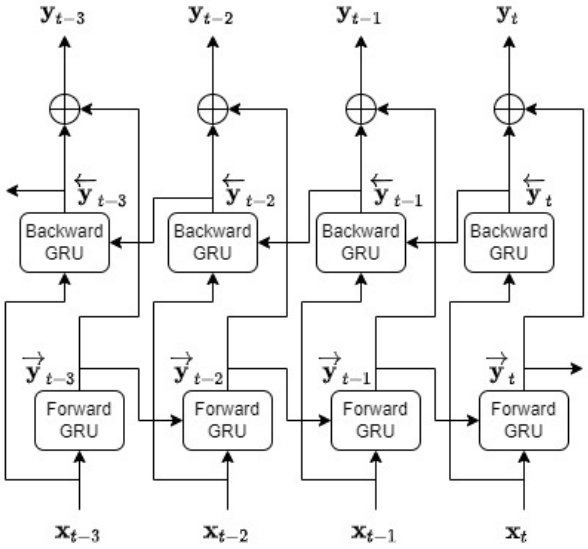



Figure 3: A Bidirectional GRU layer.

The forward GRU layer is described by the following equations:

$$\overrightarrow{\mathbf{y}}_t = (1 - \mathbf{z}_t) \circ \overrightarrow{\mathbf{y}}_{t-1} + \mathbf{z}_t \circ \mathbf{h}_t \tag{2}$$

$$\mathbf{h}_t = \tanh(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h [\mathbf{r}_t \circ \overrightarrow{\mathbf{y}}_{t-1}] + \mathbf{b}_h) \tag{3}$$

$$\mathbf{z}_t = \sigma(\mathbf{W}_z \mathbf{x}_t + \mathbf{U}_z \overrightarrow{\mathbf{y}}_{t-1} + \mathbf{b}_z) \tag{4}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r \mathbf{x}_t + \mathbf{U}_r \overrightarrow{\mathbf{y}}_{t-1} + \mathbf{b}_r) \tag{5}$$

where $\mathbf{z}$ is the update gate vector, $\mathbf{r}$ is the reset gate vector, $\mathbf{h}$ is the output of the hidden layer, $\mathbf{W}_{h,z,r}$, $\mathbf{U}_{h,z,r}$ are weight matrices, $\mathbf{b}_{h,z,r}$ are bias vectors, $\circ$ denotes element-wise vector multiplication and $\sigma$ is the logistic sigmoid function. Similar equations hold for the backward GRU layer except that the $t-1$ index is replaced by $t+1$. The final output of the Bi-GRU layer is

$$\mathbf{y}_t = \overrightarrow{\mathbf{y}}_t + \overleftarrow{\mathbf{y}}_t \tag{6}$$

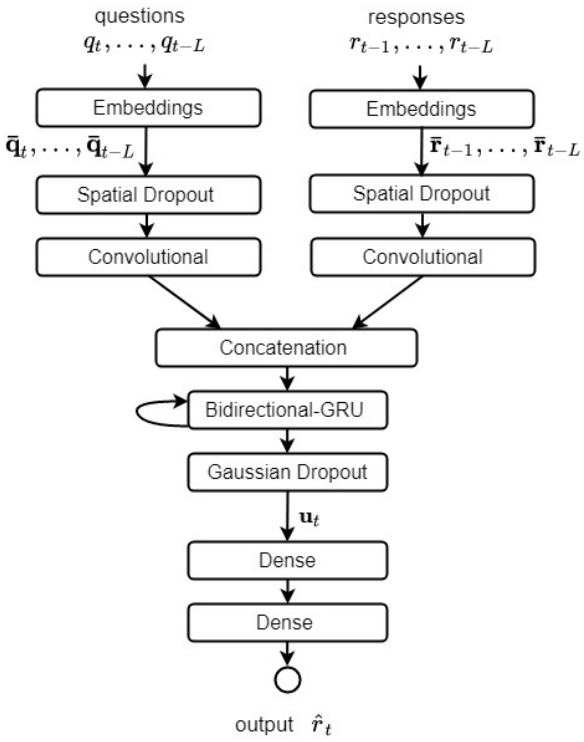



Figure 4: The originally proposed Bi-GRU Model (Delianidi et al., 2021).

The BiGRU layer in this architecture contains 32 units.

In our case, the input vector is the concatenation of the question and response embeddings filtered using 1-D convolutional operations:

$$\mathbf{x}_t = [\mathbf{F}_Q * \bar{\mathbf{q}}_{t-L:t}] \oplus [\mathbf{F}_R * \bar{\mathbf{r}}_{t-L:t-1}] \tag{7}$$

where $\mathbf{F}_Q$ and $\mathbf{F}_R$ are the convolution masks, the operation $*$ denotes 1-D convolution and $\oplus$ denotes concatenation. Regarding the experiments demonstrated in (Delianidi et al., 2021) the combination of an embeddings layer followed by a 1-D convolutional layer increases performance, presumably because it captures meaningful local interactions in the input sequences. The Convolutional layer consists of 100 filters, with kernel size 3, stride 1, and ReLU activation function. Spatial dropout is used on the embedding prior to filtering in order to reduce overfitting. The value of the dropout percentage depends on the size of the examined dataset. The smaller the dataset size, the bigger the dropout percentage parameter value.

Depending on the values of the update and reset gates the Bi-GRU layer can have arbitrarily long memory. Therefore the output of the layer is a function of a potentially very long sequence of previous questions and responses:

$$\mathbf{u}_t = f(\bar{\mathbf{q}}_t, \bar{\mathbf{q}}_{t-1}, \bar{\mathbf{q}}_{t-2}, \ldots, \bar{\mathbf{r}}_{t-1}, \bar{\mathbf{r}}_{t-2}, \ldots) \tag{8}$$

The prediction of the correctness of the current answer $r_t$ is a function of the current output of the dynamic unit:

$$\hat{r}_t = g(\mathbf{u}_t) \tag{9}$$

[1]https://github.com/delmarin35/Dynamic-Neural-Models-for-Knowledge-Tracing

Since the target value $r_t$ is binary, the function $g$ corresponds to a classifier. This is implemented by a fully connected network which includes three dense layers with 50, 25 units with ReLU activation function and one output unit with sigmoid activation which is used to make the final prediction $\hat{r}_t \in (0, 1)$. Note that Gaussian dropout (Srivastava et al., 2014) is applied to the output of the Bi-GRU layer before feeding the classification sub-network.

The overall layer structure of this early Bi-GRU model is depicted in Fig. 4.

### 3.2. Modified dynamic KT-Bi-GRU model

A drawback of the previous model is that the output $\mathbf{u}_t$ of the dynamic component cannot be easily associated with the knowledge state of the student at time $t$ since it depends on the current question $q_t$ for which we have no answer yet. Motivated by this observation, we propose an alternative KT-Bi-GRU model[2] depicted in Fig. 2b where the output of the dynamic component $\mathbf{v}_t$ is defined as a function of the previous student interactions $(q_i, r_i)$, $i = t-1, t-2, \ldots$ *excluding* the current question $q_t$.

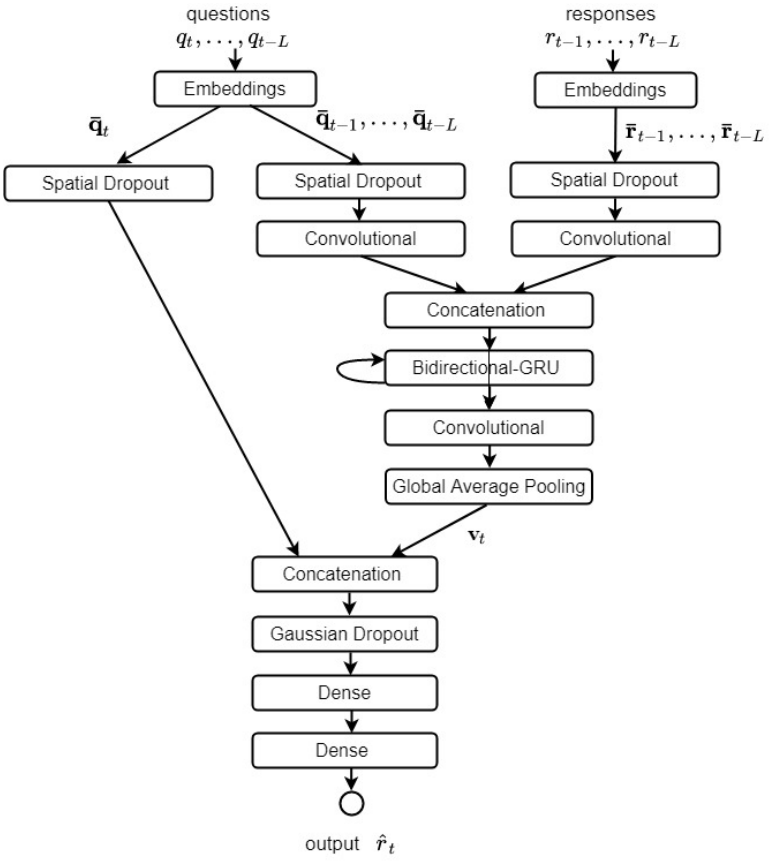



Figure 5: The KT-Bi-GRU Model.

This is described by equation (10) where $q_i$ and $r_i$ are involved through their embeddings $\bar{\mathbf{q}}_i$ and $\bar{\mathbf{r}}_i$, respectively:

$$\mathbf{v}_t = f'(\bar{\mathbf{q}}_{t-1}, \bar{\mathbf{q}}_{t-2}, \ldots, \bar{\mathbf{r}}_{t-1}, \bar{\mathbf{r}}_{t-2}, \ldots) \tag{10}$$

[2]https://github.com/delmarin35/KT-Bi-GRU

In order to predict the correctness $r_t$ of the student response to $q_t$, a new classifier function is required which combines $\mathbf{v}_t$ and $\bar{\mathbf{q}}_t$

$$\hat{r}_t \quad = \quad g'(\mathbf{v}_t, \bar{\mathbf{q}}_t) \tag{11}$$

Similar to our previous model, the KT-Bi-GRU model consists of three components: the Encoding, the Dynamic (BiGRU) component, and the Classification component (Fig. 5). The three components create two sub-networks: the Knowledge State Representation sub-net and the Tracing sub-net for the student's performance prediction. The common component of both sub-nets is the Dynamic model. The Bi-GRU layer contains either 32 or 64 units. Batch normalization and the ReLU activation function are applied to the output of the Bi-GRU layer before generating $\mathbf{v}_t$. The classification component takes the input from two branches: the vector $\mathbf{v}_t$ which estimates the student's knowledge state and the representation vector $\bar{\mathbf{q}}_t$ of the current question. As with the previous model, the classifier contains three dense layers. The first two layers contain 50 and 25 units, respectively, with the ReLU activation function, while the final layer has 1 unit with the sigmoid function which predicts the student's response.

## 4. DATASETS

To evaluate the models we used six publicly available benchmark datasets for the knowledge tracing task and generated a 7th one using the data of task 1 of the NeurIPS 2020 Educational Challenge (Wang et al., 2020). All the datasets are related to the examination of mathematical problems. Each student's interactions are recorded in the 3-line format. The first line shows the number of student interactions. The 2nd line contains the skill IDs or question IDs to which the student answers. The 3rd line contains the student's answers, with values 1 or 0 for the correct answer or for the wrong answer correspondingly. In all datasets, the sequence and the number of questions are not the same for each student. We worked with two types of datasets, depending on the complexity of the questions. In all datasets except for NeurIPS, each question relates to only one skill, while in NeurIPS each question is related to two or more skills. The datasets are described in detail below.

### 4.1. DATASETS DESCRIPTION

Three of the datasets were provided by the ASSISTments platform (AssistmentsData, 2015). These are, the *ASSISTment09*, the *ASSISTment09 corrected*[3] and the *ASSISTment12*[4]. The fourth Assistments dataset, named *ASSISTment17*, was obtained from 2017 Data Mining competition page[5]. The fifth dataset, *FSAI-F1toF3* is provided by Find Solution Ai Limited and is collected using data from the 4LittleTrees[6] adaptive learning application. The sixth dataset *STATICS2011*[7] (Koedinger et al., 2010), is provided by a college-level engineering statics course. The seventh dataset called *NeurIPS-2020-small* was generated from the NeurIPS 2020 educational

[3]https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data/skill-builder-data-2009-2010
[4]https://sites.google.com/site/assistmentsdata/home/2012-13-school-data-with-affect
[5]https://sites.google.com/view/assistmentsdatamining/data-mining-competition-2017
[6]https://www.4littletrees.com
[7]https://pslcdatashop.web.cmu.edu

challenge[8] data provided by Eedi[9] team platform[10]. For each student in the original NeurIPS 2020 dataset a percentage of his/her interactions appear in the train set while the remaining interactions appear in the test set. However, in all other datasets, the interactions of any student appear either in the train set or in the test set, but not in both. In order to comply with this paradigm, we created *NeurIPS-2020-small* as follows: (i) we combined the train and the test sets into a single dataset (ii) for practical reasons, due to the very large size of the original dataset, we kept only about 10% of the original data and removed the rows with missing values (iii) we split the dataset into train and test sets at a 70% / 30% ratio, in such a way that all student interactions appear exclusively either in the train set or in the test set.

In order to compare with previous works in the ASSISTment datasets we followed the common protocol where skill ids are used as inputs instead of questions in order to predict the student response. For the rest of the datasets, we use question ids as inputs. Those questions, of course, are associated with one or more skills as explained above.

All the datasets, except for *NeurIPS-2020-small*, have been used to evaluate previous state-of-art knowledge tracing models including DKT, DKVMN, Deep-IRT, and SAKT. The datasets differ from each other in the number of participating students, responses, skills, questions, and the number of records. We also used those benchmark datasets to evaluate the performance of all models in comparison to each other. In six of seven datasets, there is a one-to-many relationship between skills and questions. In other words, each question corresponds to a single skill but the same skill may be associated with many questions. We refer to those datasets as single-skill datasets. In contrast, each question in *NeurIPS-2020-small* corresponds to a list of skills. At the same time, one skill is part of the list of skills in many questions. Thus *NeurIPS-2020-small* is referred to as a multi-skill dataset.

### 4.2. Data pre-processing

Before proceeding with our evaluation experiments, we corrected grammatical errors in the skill names and replaced mathematical symbols with the corresponding words. For example, the word "*Polnomial*" in the skill name "*Parts of a Polnomial Terms Coefficient Monomial Exponent Variable*" was corrected to "*Polynomial*". Furthermore, the math symbols found in the skill names were converted to the corresponding words. For example, the symbols "*+,-,/,* ()*" in the skill name "*Order of Operations +,-,/,*() positive reals*" have been replaced with the words that express these symbols, ie. "*addition subtraction division multiplication parentheses*". This preprocessing action was preferred over the total removal of the math symbols since the skills refer to mathematical operations and deleting them, would have a destructive impact on the meaning of the skill. Moreover, the "*ASSISTment09 corrected*" and "*ASSISTment12*" datasets contained skills of the form of "*skill1_skill2*" and "*skill1_skill2_skill3*" which actually correspond to the same skill names. In this case, we merged them into the first skill id, found before the first underscore symbol.

We observed that for all data sets the number of student interactions differed. There are cases with only one interaction and those with hundreds of interactions. We have removed the students' records with only one question-answer pair since they include no past interactions to base our predictions. This is also the approach used by all models we compare against. The

[8] https://eedi.com/projects/neurips-education-challenge
[9] https://eedi.com/
[10] https://diagnosticquestions.com/

overview of all the datasets used in our experiments is shown in Table 1.

Before the experimentation procedure, the datasets were split to train and test sets, with a splitting percentage of 70% for training and 30% for testing.

Table 1: Datasets Overview.

| Dataset | Skills | Questions | Students | Responses |
|---|---|---|---|---|
| ASSISTment09 corrected | 101 | 101* | 4,009 | 274,448 |
| ASSISTment09 | 110 | 110* | 4,029 | 325,515 |
| ASSISTment12 | 196 | 196* | 27,849 | 2,629,095 |
| ASSISTment17 | 101 | 101* | 1,709 | 864,713 |
| FSAI-F1toF3 | 99 | 2,266$^{+}$ | 309 | 51,282 |
| STATICS2011 | 98 | 1,223$^{+}$ | 333 | 189,297 |
| NeurIPS-2020-small | 388 | 27,570$^{+}$ | 7,733 | 1,696,689 |

*skills used as inputs to the prediction model.

+questions used as inputs to the prediction model.

## 5. EXPERIMENTS

We experimentally compare the performance of our dynamic Bi-GRU models against the performance of the state-of-art knowledge tracing models DKT (Piech et al., 2015), DKVMN (Zhang et al., 2017), Deep-IRT (Yeung, 2019) and SAKT (Pandey and Karypis, 2019) on the seven datasets described above. Note that the python code[11] used for the DKT model experiments requires that the train/test split is performed during code execution, thus the data files have been converted to the appropriate format required for the experimentation process.

In the experiments we performed, we re-examined the effectiveness of our previous Bi-GRU model on the same datasets as we did for all other models. The reason is that in our previous paper (Delianidi et al., 2021), the results we presented concerned the data that also contained records with only one interaction of students and so the results are not completely comparable. In addition, we also ran experiments for "*STATICS2011*" and "*NeurIPS-2020-small*" datasets. We used the same parameter values as mentioned in (Delianidi et al., 2021) (see Table 2).

Next, we describe the process of conducting the experiments in terms of the input representation, the parameter settings, and the characteristics that affected the performance of the proposed model.

### 5.1. EMBEDDING VECTOR INITIALIZATION

The initialization of the response embeddings in our both Bi-GRU models, is done exclusively using random vectors. However, the questions are represented using embedding vectors which are initialized either randomly, or using pretrained vectors, based on the verbal description of the skill(s) corresponding to the questions. In our earlier model (Delianidi et al., 2021), for the pretrained initialization of the question embeddings we used the text files from Wikipedia2Vec[12] (Yamada et al., 2020) that is based on Word2Vec method (Mikolov et al., 2013) and contains

[11] https://github.com/lccasagrande/Deep-Knowledge-Tracing

[12] https://wikipedia2vec.github.io/wikipedia2vec/

pretrainable embeddings for the word representation vectors in English language in 100 and 300 dimensions. For the same model we also used pretrained embeddings based on FastText (Joulin et al., 2016) in 300 dimensions using the "*SISTER*" (SImple SenTence EmbeddeR)[13] library. Due to the fact that in all the datasets a skill name consists of one or more words, in the case of pre-trained question embeddings initialization, the skill name embedding vector is created by the additive aggregation of the separate word embeddings in the skill name.

For the KT-Bi-GRU model, introduced in this work, we only apply the W2V method for skill names with the initialization of vectors in 100 dimensions. We used the dimension size 100 based on the conclusions of our previous research regarding our Bi-GRU model.

The pre-trained embedding vectors for each examined question in the *NeurIPS-2020-small* multi-skill dataset are generated based on the list of skill names corresponding to the question. The process is completed in two steps:

- the representation vector of each skill name is calculated by the W2V method as in the single skills data described above
- the question embedding vector representation is generated based on the average of the skills embedding vectors related to the question.

For the random embeddings initialization, we assigned a random initial embedding vector to each skill id in the case of ASSISTment datasets, while we assigned random initial embedding vectors to the question ids in all other datasets. This difference comes from the datasets' nature: for the ASSISTment datasets each student interaction $x_i = \{q_i, r_i\}$ contains a skill id $q_i$, while in the other datasets $q_i$ is a question id.

We notice a differentiation in the utility of the W2V embedding initialization depending on whether the questions involve multiple skills or a single skill. We found that in the case of multi-skill questions, W2V initialization is advantageous compared to the random initialization. We postulate that this is because the W2V representations are not specifically trained in a mathematical context while all studied datasets involve mathematical problems. Therefore, in the single-skill scenario, initializing the embeddings with the average embeddings of the skill text, for example "Table" or "Scientific notation" or "Pattern finding" (in ASSISTment datasets), seems to offer practically no advantage compared to the random initialization. In the multi-skill scenario, the combination of the corresponding skill representations with the pretrained embeddings offers richer information and seems to offer an advantage. For example, combining the W2V representations of "Maths"+"Numbers"+"Factors Multiples and Primes"+"Factors and Highest Common Factor"+"Prime Numbers and Prime Factors", (skills of a question) generates a rich initialization vector which improves the performance. In fact, the skills are related to each other as shown in Figure 6. This gives a better focus on the question and adds more detailed context.

## 5.2. Experimental settings

We performed experimental tuning of our proposed model hyper-parameters with the aim to construct models with similar parameter settings for all the datasets we used.

Table 2 shows the best parameter settings for both the earlier Bi-GRU and the KT-Bi-GRU models. Both models have been trained using the cross-entropy loss and the Adam optimization

[13] https://pypi.org/project/sister/

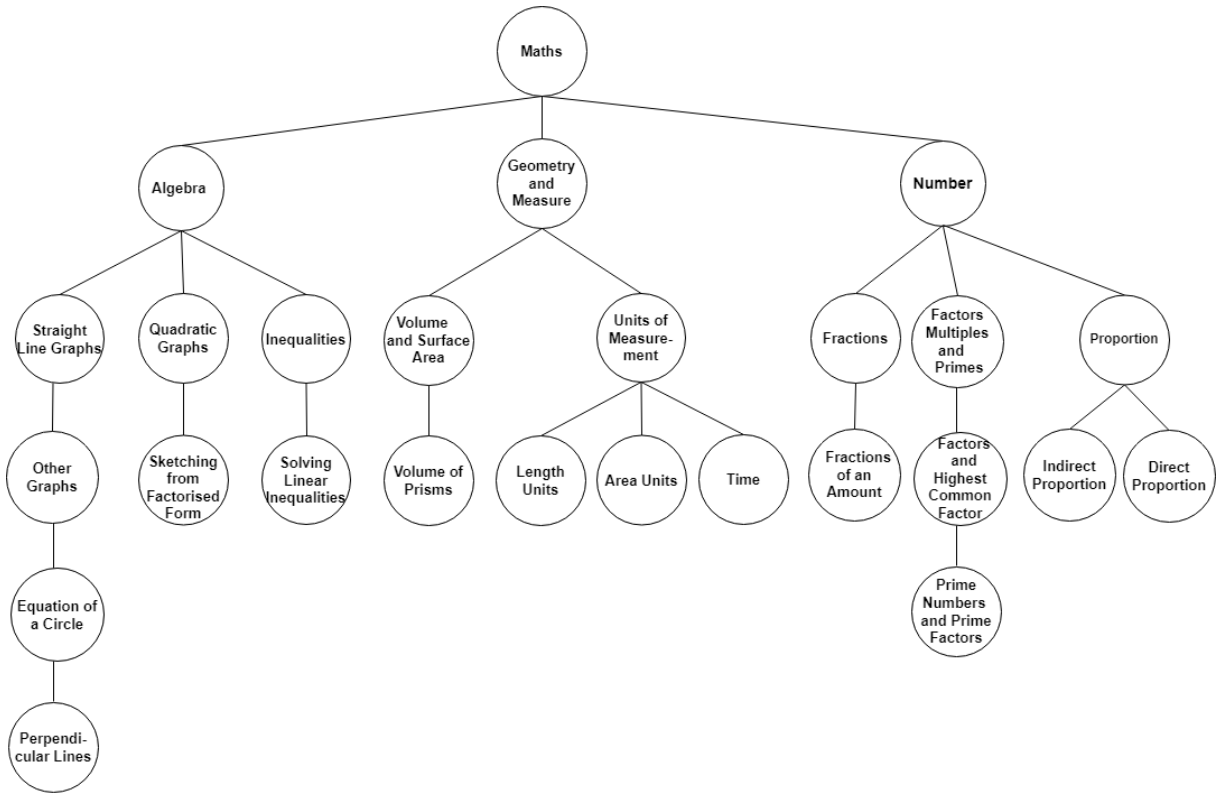



Figure 6: A part of the skill tree structure of the NeurIPS 2020 dataset.

algorithm (Kingma and Ba, 2015). Additionally, the learning rate $lr$ was scheduled to begin from a starting value and decrease according to the epoch number $n$:

$$lr = \begin{cases} r_{init} & \text{if } n < 15 \\ r_{init} \times e^{(0.5 \cdot (15-n))} & \text{otherwise} \end{cases}$$

As can be seen from Table 2, values of the parameters in the models are varied according to the data sets. Specifically, in the new KT-Bi-GRU model, the differences concern the parameters of learning rate and recurrent units. For all the single-skill datasets the value of learning rate is 0.001 and the value of recurrent units is 64, while in *NeurIPS-2020-small* multi-skill dataset the corresponding values of the parameters are 0.0001 and 32. Most variations concern dropout parameters values according to the number of skills or questions embeddings vectors and the size of database. In the case of ASSISTment datasets, the number of parameters to be trained by the neural network is less compared to other datasets. We consider this to be due to a combination of two characteristics, the input to the model during training is the representation of the skills and not the questions and the number of records in each dataset. We use dropout layers to reduce the number of trainable parameters and avoid overfitting. The dropout values for all the datasets were set as follows:

- *ASSISTment09 corrected* and *ASSISTment09* : spatial and gaussian dropouts = 0.6
- *ASSISTment12* and *ASSISTment17*: spatial and gaussian dropouts = 0.5
- *FSAI-F1toF3*: spatial dropout = 0.9, gaussian dropout = 0.8,

Table 2: The parameter settings of the both models.

| Parameter | Early Bi-GRU | KT-Bi-GRU |
|---|---|---|
| $L$ | 50 | 50 |
| batch size | 100 or 300$^{(*)}$ | 100 |
| skill/question emb. init. | random, W2V | random, W2V |
| responses embed. init. | random | random |
| embeddings vector dim. | 100 | 100 |
| learning rate | 0.001 & 0.0001$^{(*)}$ | 0.001 & 0.0001$^{(*)}$ |
| hidden layers | 2 layers: 50 and 25 units | 2 layers: 50 and 25 units |
| recurrent units | 32 | 64 & 32 $^{(*)}$ |
| dropout rate | 0.2-0.9 depending on the dataset | 0.5-0.9 depending on the dataset |
| training epochs | 30 | 30 |

$^{(*)}$ In the case of the *NeurIPS-2020-small* dataset

- *STATICS2011*: spatial and gaussian dropout = 0.8,
- *NeurIPS-2020-small*: spatial and gaussian dropout = 0.5.

Corresponding settings in parameter values were made for our earlier Bi-GRU model. The values of the parameters are shown in above Table 2 and are specifically described in more detail in (Delianidi et al., 2021).

The evaluation metric that is used for the prediction of the probability correctness of students' responses is the Area Under the ROC Curve (AUC) (Ling et al., 2003). This metric was used for the evaluation of all the state-of-art knowledge tracing models for the student performance prediction task.

## 6. Results and discussion

The experimental results are shown in Table 3. The two dynamic Bi-GRU models, in general, outperform the previous state-of-art models in five of seven datasets. The SAKT model achieves the best performance in the cases of the *ASSISTment09 corrected* and *ASSISTment12* datasets, although it performs poorly in the case of the *ASSISTment17*, *FSAI-F1toF3*, *STATICS2011*, and *NeurIPS-2020-small* datasets. It also achieves reasonable performance in the *ASSISTment09* dataset. As seen in Fig. 7 the datasets where SAKT performs well are the ones where the majority of the students have short interaction histories while it has low performance when most interaction histories are long. For example, in the *ASSISTment17* dataset, SAKT achieves performance quite lower than the second worst model, DKT. Our proposed Bi-GRU models work complementary to SAKT in the sense that they perform better when the student interaction data are long. Also, note that with the exception of *ASSISTment12*, the datasets where our proposed Bi-GRU models outperform the competition have a large number of distinct questions (more than 1000 as seen in table 1). We conclude that the Bi-GRU models work better with rich data (lengthy history and many questions).

Table 3: The best results of AUC metric per dataset and per model (in percentages).

| Dataset | Models | | | | | |
|---|---|---|---|---|---|---|
| | **DKT** | **DKVMN** | **Deep-IRT** | **SAKT** | **Bi-GRU** | **KT-Bi-GRU** |
| ASSISTment09 corrected | 74.27 | 74.06 | 73.41 | **81.82** | 75.17 | 74.84 |
| ASSISTment09 | 81.56 | 81.61 | 81.65 | 81.31 | **82.42** | 82.22 |
| ASSISTment12 | 69.40 | 69.26 | 69.73 | **77.91** | 68.46 | 68.30 |
| ASSISTment17 | 66.85 | 70.25 | 70.54 | 58.62 | 73.13 | **73.35** |
| FSAI-F1toF3 | 69.42 | 68.40 | 68.69 | 67.35 | 70.47 | **72.63** |
| STATICS2011 | 82.71 | 83.17 | 83.09 | 79.50 | **83.29** | 82.80 |
| NeurIPS-2020-small | - | 75.14 | 74.78 | 69.77 | 78.05 | **78.45** |

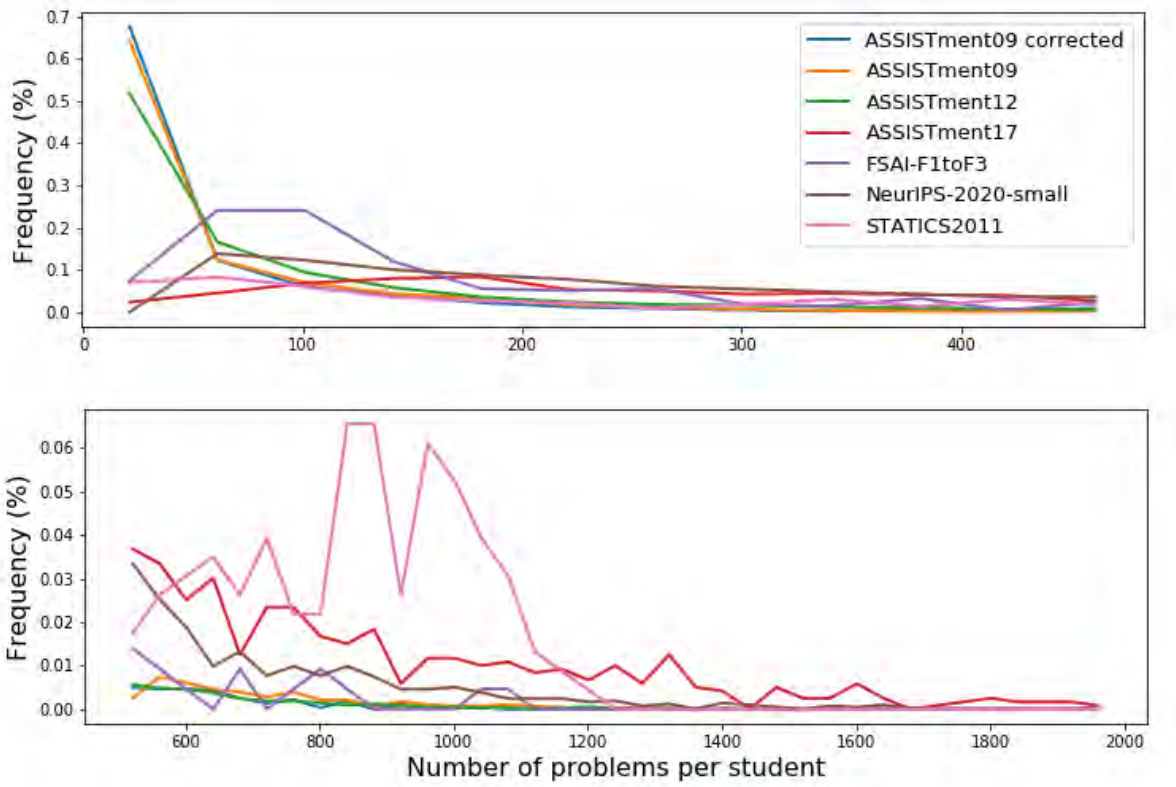
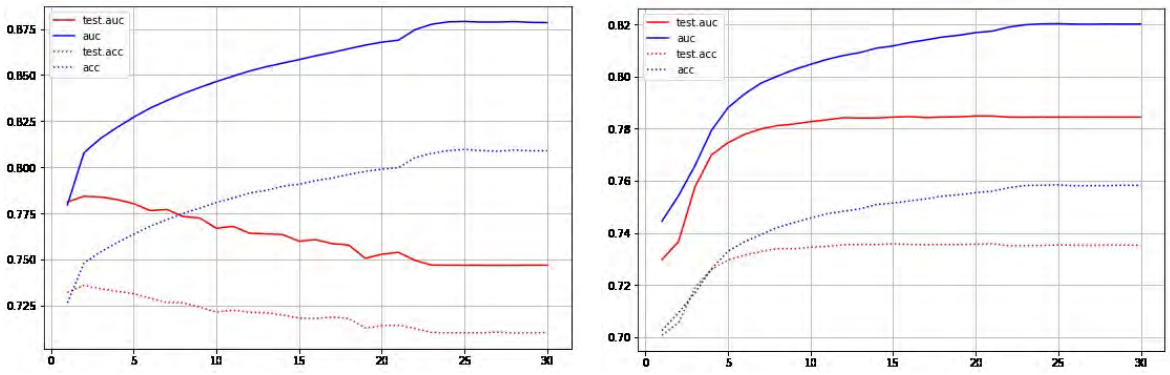



Figure 7: The histogram of students' responses per dataset. For improved visualization, we have split the histogram into two subplots: one for the range $N \in [2, 500]$ (top figure) and another one for $N \in [501, 2000]$ (bottom figure).

Comparing the recurrent Bi-GRU models with each other, we see that in the single-skill datasets the KT-Bi-GRU model performance is not far from our earlier model, while it is noticeably better in the case of the *FSAIF-F1toF3* –the smaller of all the datasets– as well as in the case of *NeurIPS-2020-small* which is a multi-skill dataset. Moreover, we should note that the KT-Bi-GRU model has an advantage for tasks that require student knowledge estimation. For example, an important task is clustering students based on their knowledge state similarity (Khayi and Rus, 2019), (Omar et al., 2020). Our early Bi-GRU model is not suitable for this

task since the internal representation vector $\mathbf{u}_t$ is not unique for a student with a specific history of question-response pairs since it also depends on the current question $q_t$. On the contrary, the output $\mathbf{v}_t$ of the recurrent layer in the KT-Bi-GRU model, is unique for a specific interaction sequence, and therefore it can be used to represent the student knowledge state after he/she has completed this sequence. This is a key difference between the KT-Bi-GRU model compared to the other state-of-the-art models as well. The exploitation of this novel feature of the proposed model is beyond the scope of the present paper which focuses on student performance prediction, however, it deserves further investigation in the future.

The experiments showed that in all single-skills datasets, the random question embeddings initialization method offers slightly better results. Therefore, the initial representation of the skills embeddings with the corresponding pretrained vectors did not contribute much to the model's performance. On the other hand, in the case of the *NeurIPS-2020-small* multi-skills dataset, the random question embedding initialization led to poor model training, with the test performance decreasing and after a few epochs stabilizing at a relatively low value (Figure 8a). Using the W2V question embedding initialization method, the model achieved an AUC value equal to 78.45 presented in Table 3 with the performance smoothly increasing in every epoch until convergence (Figure 8b). Thus, the questions were initially placed in the vector space in relation to the skills that concern them. Thus, the representation of the questions in the space, based on the parent-child tree structure of the included skills, contributed to the achievement of the model's better performance results.




(a) Random question embeddings initialization. (b) W2V question embeddings initialization.

Figure 8: Training/testing AUC and accuracy curves of the multi-skill *NeurIPS-2020-small* dataset.

The performance of the DKT model could not be tested for *NeurIPS-2020-small* data due to insufficient computing resources. While all the experiments for all the models and all the datasets were performed normally, in the DKT model with the *NeurIPS-2020-small* dataset there was a memory overload and inability to complete even the first training epoch of the model using the same code as in the rest of the datasets. One possible explanation for the lack of resources lies in the nature of the dataset itself. In contrast to all of the above datasets, *NeurIPS-2020-small* records numerous responses per student and a very large number of different questions. Another possible explanation is that there is a bug in the code or that the model is not expected to support data with *NeurIPS-2020-small* characteristics (a large number of different questions

or a large number of answers from each user). Probably this volume of data can not be managed easily and we considered that there should be no intervention in the code of the DKT model that we borrowed for the research purpose.

## 7. Conclusion and future work

In this paper we proposed a new recurrent Bidirectional GRU (KT-Bi-GRU) model for knowledge tracing and student performance prediction. The reference point of the proposed model is our earlier recurrent neural model, which surpassed the performance of the state-of-the-art models in most of the tested data sets. The KT-Bi-GRU model introduces a modified architecture with two sub-network parts. The first sub-network is for estimating the student knowledge state based on his/her interaction history using a recurrent neural network and the second sub-network predicts the student performance using multi-layer neural network.

The input data, ie. the student's interaction history, are encoded using embedding layers followed by 1-D convolutional layers. This layer combination was found to improve performance. We also investigated different methods for initializing the question embedding layer with random or pretrained embedding vectors. Our experiments showed that in the single-skill questions datasets there is no noticeable difference with respect to the initialization method used. On the contrary, in the *NeurIPS-2020-small* dataset where a question involves multiple skills, the initialization using pretrained Word2Vec embeddings obtained from the verbal description of the skills contributed significantly to the performance improvement of the model.

Both of the Bi-GRU models are suitable either for single- or multi-skills datasets with the prospect of taking advantage of the fact that there is an assessment of the student's knowledge state. This information can play a key role in expanding our research to the production of personalized educational recommendations. In particular, in the KT-Bi-GRU model, where the estimation of the student knowledge state is based solely on the previous interaction history, the production of recommendations would not be affected by the current subject to be tested, which we consider more appropriate.

We found that the proposed models are more efficient in cases of a long history of interactions and at the same time a large number of different questions, regardless of the order of the questions. The limitation of our proposed Bi-GRU models is that the performance is diminished when the history of student interactions is short. This finding is reinforced by the fact that recurrent networks, by their construction, rely on historical data to achieve the best performance. On the other hand, the attention-based SAKT model has inversely proportional results on datasets with a large number of interactions and a small number of different questions.

In future work, we intend to address the above limitation to achieve better performance with a smaller amount of interaction history data. Furthermore, we plan to exploit the proposed architecture by extending it to tasks that require student knowledge representation such as recommendation of educational content and student clustering. We also intend to investigate the effect of pretrained vector representations on datasets with different themes (other than mathematics) and explore different initialization methods.

## Acknowledgments

We would like to thank NVIDIA Corporation for the kind donation of an Titan Xp GPU card that was used to run our experiments.

We acknowledge the valuable contribution of Mr. George Chrysogonidis and Mr. Vasileios Nikiforidis in the development of the original neural network model and the corresponding experimental evaluations.

## REFERENCES

ABDELRAHMAN, G. AND WANG, Q. 2019. Knowledge tracing with sequential key-value memory networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 175–184. https://doi.org/10.1145/3331184.3331195.

ASSISTMENTSDATA. 2015. Assistments data. https://sites.google.com/site/assistmentsdata/.

CEN, H., KOEDINGER, K., AND JUNKER, B. 2006. Learning factors analysis – a general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems*, M. Ikeda, K. D. Ashley, and T.-W. Chan, Eds. Springer Berlin Heidelberg, 164–175.

CHO, K., VAN MERRIËNBOER, B., GULCEHRE, C., BAHDANAU, D., BOUGARES, F., SCHWENK, H., AND BENGIO, Y. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 1724–1734.

CORBETT, A. T. AND ANDERSON, J. R. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction 4,* 4, 253–278.

DELIANIDI, M., DIAMANTARAS, K., CHRYSOGONIDIS, G., AND NIKIFORIDIS, V. 2021. Student performance prediction using dynamic neural models. In *Proceedings of the Fourteenth International Conference on Educational Data Mining (EDM 2021)*, S. Hsiao, S. Sahebi, F. Bouchet, and J.-J. Vie, Eds. Educational Data Mining Society, 46–54.

GHOSH, A., HEFFERNAN, N., AND LAN, A. S. 2020. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. Association for Computing Machinery, 2330–2339.

HAMBLETON, R. K., SWAMINATHAN, H., AND ROGERS, H. J. 1991. Fundamentals of item response theory. Vol. 2. Sage Publications.

HE, Z., LI, W., AND YAN, Y. 2021. Modeling knowledge proficiency using multi-hierarchical capsule graph neural network. *Applied Intelligence 52*, 7230–7247. https://doi.org/10.1007/s10489-021-02765-w.

HOCHREITER, S. AND SCHMIDHUBER, J. 1997. Long short-term memory. *Neural computation 9,* 8, 1735–1780.

JOULIN, A., GRAVE, E., BOJANOWSKI, P., DOUZE, M., JÉGOU, H., AND MIKOLOV, T. Dec 2016. Fasttext.zip: Compressing text classification models. *arXiv, arXiv:1612.03651*. https://doi.org/10.48550/arXiv.1612.03651.

KÄSER, T., KLINGLER, S., SCHWING, A. G., AND GROSS, M. H. 2017. Dynamic bayesian networks for student modeling. *IEEE Transactions on Learning Technologies 10*, 450–462.

KHAYI, N. A. AND RUS, V. 2019. Clustering students based on their prior knowledge. In *Proceedings of the 12th International Conference on Educational Data Mining (EDM 2019)*, C. F. Lynch, A. Merceron, M. Desmarais, and R. Nkambou, Eds. Educational Data Mining Society, 246–251.

KINGMA, D. P. AND BA, J. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego,*

*CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds. https://doi.org/10.48550/arXiv.1412.6980.

KOEDINGER, K. R., BAKER, R. S., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., AND STAMPER, J. 2010. A data repository for the edm community: The pslc datashop. In *Handbook of educational data mining*, C. Romero, S. Ventura, M. Pechenizkiy, and R. S. Baker, Eds. Vol. 43. CRC Press, Boca Raton, FL, 43–56. https://doi.org/10.1201/b10274.

LI, P., LUO, A., LIU, J., WANG, Y., ZHU, J., DENG, Y., AND ZHANG, J. 2020. Bidirectional gated recurrent unit neural network for chinese address element segmentation. *ISPRS International Journal of Geo-Information 9,* 11. https://doi.org/10.3390/ijgi9110635.

LING, C. X., HUANG, J., AND ZHANG, H. 2003. Auc: a statistically consistent and more discriminating measure than accuracy. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. IJCAI'03, vol. 3. Morgan Kaufmann Publishers Inc., 519–524.

LIU, Q., HUANG, Z., YIN, Y., CHEN, E., XIONG, H., SU, Y., AND HU, G. 2019. Ekt: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering 33,* 1, 100–115.

LIU, Q., SHEN, S., HUANG, Z., CHEN, E., AND ZHENG, Y. 2021. A survey of knowledge tracing. *CoRR abs/2105.15106*. https://doi.org/10.48550/arXiv.2105.15106.

LIU, Y., YANG, Y., CHEN, X., SHEN, J., ZHANG, H., AND YU, Y. 2021. Improving knowledge tracing via pre-training question embeddings. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, C. Bessiere, Ed. IJCAI'20. International Joint Conferences on Artificial Intelligence, 1577–1583.

MA, R., ZHANG, L., LI, J., MEI, B., MA, Y., AND ZHANG, H. 2021. Dtkt: An improved deep temporal convolutional network for knowledge tracing. In *Proceedings of the 2021 16th International Conference on Computer Science & Education (ICCSE)*. IEEE, 794–799.

MIKOLOV, T., CHEN, K., CORRADO, G., AND DEAN, J. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations, Workshop Track Proceedings*, Y. Bengio and Y. LeCun, Eds. ICLR 2013. Scottsdale, Arizona, USA. https://doi.org/10.48550/arXiv.1301.3781.

MILLER, A., FISCH, A., DODGE, J., KARIMI, A.-H., BORDES, A., AND WESTON, J. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, J. Su, K. Duh, and X. Carreras, Eds. Association for Computational Linguistics, 1400–1409.

OMAR, T., ALZAHRANI, A., AND ZOHDY, M. 2020. Clustering approach for analyzing the student's efficiency and performance based on data. *Journal of Data Analysis and Information Processing 8,* 03, 171–182. https://doi.org/10.4236/jdaip.2020.83010.

PANDEY, S. AND KARYPIS, G. 2019. A self-attentive model for knowledge tracing. In *Proceedings of the 12th International Conference on Educational Data Mining, EDM 2019*, C. F. Lynch, A. Merceron, M. Desmarais, and R. Nkambou, Eds. International Educational Data Mining Society, 384–389.

PAVLIK, P. I., CEN, H., AND KOEDINGER, K. R. 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*, V. Dimitrova, R. Mizoguchi, B. du Boulay, and A. Graesser, Eds. IOS Press, 531–538.

PIECH, C., BASSEN, J., HUANG, J., GANGULI, S., SAHAMI, M., GUIBAS, L., AND SOHL-DICKSTEIN, J. 2015. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds. Vol. 28. Curran Associates, Inc., 505–513.

PU, S., CONVERSE, G., AND HUANG, Y. 2021. Deep performance factors analysis for knowledge tracing. In *Proceedings of the Artificial Intelligence in Education: 22nd International Conference, AIED 2021*. Springer-Verlag, Berlin, Heidelberg, 331–341.

SCHUSTER, M. AND PALIWAL, K. K. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing 45,* 11, 2673–2681.

SHEN, S., LIU, Q., CHEN, E., WU, H., HUANG, Z., ZHAO, W., SU, Y., MA, H., AND WANG, S. Convolutional knowledge tracing: Modeling individualization in student learning process. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '20. Association for Computing Machinery, 1857–1860. https://doi.org/10.1145/3397271.3401288.

SONG, X., LI, J., TANG, Y., ZHAO, T., CHEN, Y., AND GUAN, Z. 2021. Jkt: A joint graph convolutional network based deep knowledge tracing. *Information Sciences 580*, 510–523.

SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., AND SALAKHUTDINOV, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research 15,* 1, 1929–1958.

VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, L., AND POLOSUKHIN, I. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, U. von Luxburg, I. Guyon, S. Bengio, H. Wallach, and R. Fergus, Eds. NIPS'17. Curran Associates Inc., 6000–6010.

WANG, W., LIU, T., CHANG, L., GU, T., AND ZHAO, X. 2020. Convolutional recurrent neural networks for knowledge tracing. In *Proceedings of the 2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*. IEEE, 287–290.

WANG, Z., LAMB, A., SAVELIEV, E., CAMERON, P., ZAYKOV, Y., HERNÁNDEZ-LOBATO, J. M., TURNER, R. E., BARANIUK, R. G., BARTON, C., JONES, S. P., WOODHEAD, S., AND ZHANG, C. 2020. Diagnostic questions: The neurips 2020 education challenge. *arXiv preprint arXiv:2007.12061*.

YAMADA, I., ASAI, A., SAKUMA, J., SHINDO, H., TAKEDA, H., TAKEFUJI, Y., AND MATSUMOTO, Y. 2020. Wikipedia2Vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, 23–30. https://wikipedia2vec.github.io/wikipedia2vec/.

YANG, S., ZHU, M., HOU, J., AND LU, X. Deep knowledge tracing with convolutions. In *12th International Conference on Educational Data Mining, EDM 2019*, C. F. Lynch, A. Merceron, M. Desmarais, and R. Nkambou, Eds. International Educational Data Mining Society, 683–686.

YANG, Y., SHEN, J., QU, Y., LIU, Y., WANG, K., ZHU, Y., ZHANG, W., AND YU, Y. 2021. Gikt: a graph-based interaction model for knowledge tracing. In *Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2020*, F. Hutter, K. Kersting, J. Lijffijt, and I. Valera, Eds. Springer International Publishing, Cham, 299–315.

YEUNG, C.-K. 2019. Deep-irt: Make deep learning based knowledge tracing explainable using item response theory. In *Proceedings of the 12th International Conference on Educational Data Mining, EDM 2019*, C. F. Lynch, A. Merceron, M. Desmarais, and R. Nkambou, Eds. International Educational Data Mining Society, 683–686.

ZHANG, J., SHI, X., KING, I., AND YEUNG, D.-Y. 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, 765–774. https://doi.org/10.1145/3038912.3052580.