

28th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2024)

ACKT : Self-Attentive Convolutional Model for Knowledge Tracing

Mahsa Abazari Kia, Alexandros Koliouris

Northeastern University London, London, United Kingdom

Abstract

The task of knowledge tracing involves establishing a model that characterizes the mastery level of individual students in relation to knowledge concepts (KCs), as they engage with a sequence of learning activities. Each student's knowledge is modelled by estimating the student's performance on the learning activities. This domain of research holds significant importance in the development of personalized learning platforms for students. However, within the current landscape, the majority of existing Knowledge Tracing (KT) methods exhibit a gap in their approach, as they often fail to incorporate two pivotal factors simultaneously: the individualization of student characteristics (i.e., the prior knowledge and learning rates) and the interconnectedness between KCs. In essence, these existing approaches tend to disregard the intrinsic variations in students' prior knowledge and learning paces, which invariably differ from one student to another. In this study, our focus lies in predicting student performance through modelling his/her knowledge state, while also extracting dependencies between KCs and his/her prior learning interactions. To this end, we propose a novel self Attentive Convolutional Knowledge Tracing (ACKT) method for analyzing continuous learning interactions of students. Extensive experiments on several real-world benchmark datasets show that ACKT could obtain better knowledge tracing results and it outperforms existing KT methods for predicting future learner responses.

© 2024 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<https://creativecommons.org/licenses/by-nc-nd/4.0>)

Peer-review under responsibility of the scientific committee of the 28th International Conference on Knowledge Based and Intelligent information and Engineering Systems

Keywords: Intelligent education; Knowledge tracing; Convolution Neural Networks; Individualized learning

1. Introduction

The availability of extensive datasets documenting students' learning trajectories about their knowledge concepts (KCs), encompassing exercises, skills, and conceptual domains, has enticed the data mining community to craft tools aimed at predicting students' performance and delivering tailored feedback [20]. In the pursuit of constructing personalized learning platforms, the task of knowledge tracing (KT) stands out as an important task. It revolves around the systematic tracking of a student's knowledge state, reflecting their mastery levels across diverse KCs, drawing insights from their historical learning activities. Employing KT holds the promise of enhancing students' learning efficiency

E-mail address: mahsa.abazari@nulondon.ac.uk

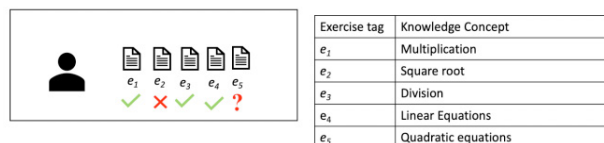


Fig. 1: The left subfigure depicts the sequence of exercises attempted by the learner, while the right subfigure depicts the knowledge concepts to which each of the exercises belongs.

and providing them with heightened insight into their educational progression [3]. Various deep learning models have recently been presented, and those based on Recurrent Neural Networks (RNN) suffer the problem of not generalizing when dealing with sparse data [10]. Furthermore, most earlier KT techniques assessed the students' knowledge state without combining the individualization of learning rates of different students and the relevance between the KCs. Actually, prior knowledge and learning rates varies from student to student, which has been shown to have a significant impact on the learning process [25]. Given the student's prior exercise interactions $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t)$, predict some feature of his/her upcoming interaction $\mathbf{x}_{t+1}$. The interactions are represented on the question-answering platform as $\mathbf{x}_t = (e_t, r_t)$, where e_t is the exercise that the student attempts at timestamp t and r_t is the correctness of the student's answer. The goal of KT is to forecast whether the learner will be able to correctly answer the next exercise, i.e., predict $p(r_{t+1} = 1 | e_{t+1}, \mathbf{X})$. In this paper, we present a model based on a purely attention mechanism, a transformer [23], and a Convolutional Neural Network (CNN) [16]. The suggested Self-Attentive Convolutional Knowledge Tracing (ACKT) approach is the first method that seamlessly integrates the power of attention mechanisms and the structural insights offered by CNNs in the context of Knowledge Tracing, providing a novel and comprehensive framework for capturing intricate patterns in students' learning trajectories while offering accurate forecasts of their performance outcomes. Attention mechanisms are more adaptable than recurrent and memory-based neural networks and outperform them in natural language processing tests. Furthermore, the multi-layer convolutional architecture extracts deep features and builds hierarchical representations of the input sequence. In KT task, the skill sets that a student develops while going through the sequence of learning activities are related to each other, and his/her performance on a specific exercise is based on his/her performance on previous exercises related to that exercise [2]. For example, in Figure 1, a student must know how to calculate "square roots" (exercise 3) and "linear equations" (exercise 4) in order to successfully complete an exercise on "quadratic equations" (exercise 5), which belongs to the knowledge concept "Equations". As we show later, ACKT constructs the individualized knowledge state and learning rate for each student based on the previously answered exercises and their relevance. Then the model anticipates the student's performance on a particular exercise based on the extracted knowledge state and individualised learning rate. ACKT assigns weights to the previously answered exercises while extracting their relevance and dependencies by employing the attention mechanism. CNN is applied to generate the student's knowledge state based on the connections drawn by the multi-head self-attention mechanism between this question and every question the learner has responded to in the past. We summarize our key innovations below:

- Generating exercise representations by taking into account the connection between exercises through the computation of their embedding similarity and student performance data.
- Employing multi-head self-attention mechanism to analyze the student's past interactions and identify interdependencies between KCs and interactions by assigning weights to the previously answered exercises.
- Designing a convolutional layer to reveal the knowledge state and individualized learning rate by consuming the information about student's past interactions and their correlation obtained from attention layer. The student's performance on a particular exercise is predicted in this layer by utilizing the captured knowledge state.

2. Related work

The KT task assesses a student's understanding level by analyzing her performance data. The prevalent approaches for addressing this task can be grouped into two categories: the traditional Bayesian Knowledge Tracing (BKT) [3] and methods based on deep learning. The BKT is based on Hidden Markov Models and it represents a student's latent

knowledge as a series of binary variables, where each variable signifies comprehension or lack thereof for an individual concept. The trend of using deep learning for the KT task is pioneered by the work of DKT [18]. Deep Knowledge Tracing (DKT) leverages the capabilities of Long Short-Term Memory (LSTM) networks [6] to model a student's cognitive state. To address certain limitations inherent in DKT, such as challenges in reconstructing inputs and achieving temporal smoothness in predicted KCs, an extended variant known as DKT+ [24] was devised. Subsequently, drawing inspiration from Memory-Augmented Neural Networks [7], Zhang et al. [26] introduced the concept of Dynamic Key-Value Memory Networks (DKVMN) as an enhancement to the DKT framework. The motivation behind this proposition arises from the recognition that the expressive capacity of the LSTM's hidden state might be confined in representing latent knowledge states. To mitigate this, two distinct memory matrices are employed. Following this work, Abdelrahman et al. present Sequential Key-Value Memory Networks (SKVMN) model [1], aimed at capturing intricate interdependencies inherent in questions. To facilitate this, they augment the existing architecture by incorporating an auxiliary hop-LSTM network prior to the DKVMN's output layer. Specifically, the LSTM units within this hop-LSTM network establish connections exclusively amongst hidden states pertaining to sequentially dependent questions. Nakagawa et al. [15] incorporate Graph Neural Networks into their KT model and propose Graph based Knowledge Tracing (GKT). They build a graph with nodes representing knowledge concepts and edges connecting nodes if the corresponding concepts have dependency links. When a student answers a question linked with a specific concept, GKT first collects the node features relevant to the replied concept, and then updates the students' knowledge state for both the responded concept and the related concepts at the same time.

Recently, the Self-Attentive Knowledge Tracing (SAKT) [17] methodology is introduced, which first determines the KCs related to the target KC for which performance is to be predicted from the student's past activities. SAKT then uses information from previous KC performance to estimate student mastery at the future KC. AKT [5] employs a monotonic attention mechanism that compares a learner's future responses to assessment questions to their previous responses; attention weights are calculated using exponential decay and a context-aware relative distance metric, in addition to question similarity. Furthermore, the Rasch model [13, 19] is used to regularise the idea and question embeddings.

3. Methodology

Our model is designed to predict the student's performance to tackle the upcoming exercise e_{t+1} . This prediction is based on the student's preceding interaction sequence $\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t$. We can transform the problem into a sequential modeling problem. We consider $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{t-1}$ and the exercise sequence with one position ahead, $e_2, e_3, \dots, e_t$ as the model's inputs and the output is the correctness of the response to exercises $r_2, r_3, \dots, r_t$. The interaction tuple $\mathbf{x}_t = (e_t, r_t)$ is presented to the model as a number $y_t = e_t + r_t \times E$, where E is the total number of exercises. In the interaction tuple, e_t is the exercise that the student attempts at timestamp t and r_t is the correctness of the student's answer.

For predicting student performance, we utilize past student interactions and identify relevant knowledge concepts from them and the attention layer is designed for this purpose, to assign weights to the previously answered exercises. We propose CNN to measure the individualized prior knowledge, learning rates and generate the knowledge state. At the final stage, based on the obtained knowledge state and the current exercise representation the correctness of the response to the current exercise will be predicted. Figure 2 visualizes the ACKT method and its components.

Also, we develop a method to learn the underlying relations between exercises to be used for the exercise representation, then the interaction and exercises embedding matrices are generated to be the inputs for the attention layer.

3.1. Embedding Layer

We transform the acquired input sequence $\mathbf{y} = (y_1, y_2, \dots, y_t)$ into a new sequence denoted as $s = (s_1, s_2, \dots, s_n)$, wherein n represents the upper limit of the model's sequence processing capacity. Given the model's proficiency in handling inputs of fixed sequence lengths, instances arise where the length of the sequence, denoted as t , is below the predetermined value of n . In such instances, a repetitive augmentation of padding, in the form of question-answer pairs, is applied to the sequence's left end. However, in cases where the length of the sequence, t , exceeds the threshold

Table 1: Notations

Notations	Descriptions
$\mathbf{X}$	Interaction sequence of a student: $(x_1, x_2, \dots, x_t)$
x_i	i th exercise-answer pair of a student
n	maximum length of sequence
d	latent vector dimensionality
$\hat{\mathbf{H}}$	Interaction embedding matrix
$\mathbf{P}$	Positional embedding matrix
$\mathbf{A}$	exercise-exercise relation matrix
e	Sequence of exercises solved by the student
E	total number of exercises
$\hat{\mathbf{E}}$	Exercise embedding matrix

n , we execute a segmentation procedure, dividing the sequence into multiple subsequences, each adhering to the length n . To be specific, when t surpasses n , the y_t component is partitioned into t/n subsequences, each comprising n units. All of these partitioned subsequences are subsequently employed as inputs for the model.

In our training procedure, we construct an Interaction embedding matrix, represented as $\mathbf{H} \in \mathbf{R}^{2E \times d}$, where d denotes the latent dimensionality. This matrix serves as a tool for deriving embeddings, denoted as $\mathbf{H}_{s_i}$, corresponding to each constituent element s_i present within the sequence. We train the exercise embedding matrix, $\mathbf{E} \in \mathbf{R}^{E \times d}$, such that each exercise in the set e_i is embedded in the e_i th row.

3.1.1. Exercise-Relation Matrix Computation

A notable innovation within our model involves the investigation of methodologies aimed at the identification of latent interconnections among exercises, with the exercises themselves being represented through embedding vectors which are trained. Given the absence of explicit knowledge concerning inter-exercise relationships, a process of deductive inference is undertaken, leading to the establishment of an exercise relation matrix denoted as $\mathbf{A} \in \mathbf{R}^{E \times E}$. In this matrix, each $\mathbf{A}_{i,j}$ element denotes the extent to which the performance achieved in exercise j influences the performance outcomes of exercise i . We draw on two primary sources of information to unearth these exercise relationships: data concerning student performance and the similarity of the exercises embeddings. The former aids in capturing the significance of knowledge acquired while solving exercise j in the context of solving exercise i . On the other hand, the latter aspect captures the semantic similarity between the two exercises.

We will proceed to elucidate the manner in which learner performance data can be effectively utilized to ascertain the significance of the acquired knowledge from exercise j in the resolution of exercise i . Initially, we construct a contingency table, as depicted in Table 2, by exclusively considering instances where exercise j precedes exercise i within the sequential learning process. In cases where multiple instances of exercise j appear before exercise i , our analysis only incorporates the most recent occurrence. Subsequently, we calculate the Phi coefficient, a well-established metric for measuring the association between two binary variables. The mathematical computation of the Phi coefficient, which delineates the relationship from exercise j to exercise i , is $\phi = \frac{ad-bc}{\sqrt{efgh}}$.

The $\phi_{i,j}$ value ranges between -1 and 1, with a higher score indicating the significance of students' performance in exercise j in influencing their performance in exercise i . We have selected the Phi coefficient, out of various correlation metrics, for calculating exercise relations due to the following reasons: 1) its ease of interpretation, and 2) its explicit penalization of disparities between the two variables.

Furthermore, we use additional data for evaluating the relationship between the two exercises, cosine similarity between the trained exercises' embeddings. Initially, we derive the exercise embeddings for exercises i and j , denoted as E_i and E_j respectively. We then measure exercise similarity by employing the cosine similarity metric on these embeddings. In a formal sense, the computation of exercise similarity is calculated by $\text{sim}_{i,j} = \frac{\mathbf{E}_i \mathbf{E}_j}{\|\mathbf{E}_i\|_2 \|\mathbf{E}_j\|_2}$.

Table 2: A contingency table for two exercises i and j .

		exercise i		total
		incorrect	correct	
exercise j	incorrect	a	b	e
	correct	c	d	f
total		g	h	n

Finally, the relation of exercise j with exercise i is calculated as :

$$\mathbf{A}_{i,j} = \begin{cases} \phi_{i,j} + \text{sim}_{i,j}, & \text{if } \text{sim}_{i,j} + \phi_{i,j} > \theta \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where θ is a threshold that controls sparsity of relation matrix.

3.1.2. Position Encoding

constitutes a foundational layer within the architecture of the self-attention neural network. This specific layer plays a pivotal role in encoding positional information, however, its significance is notably pronounced in the domain of knowledge tracing due to the gradual and consistent evolution of a student's knowledge state over time. Maintaining a coherent and seamless progression of knowledge state is of paramount importance, where abrupt fluctuations are undesirable [24]. To address this, a parameter referred to as the “position embedding,” denoted as $\mathbf{P} \in \mathbf{R}^{n \times d}$. The i th row of position embedding matrix, P_i is then added to the interaction embedding vector of the i th element of the interaction sequence.

3.1.3. Interaction Embedding

The output from the embedding layer is interaction embedding matrix, $\hat{\mathbf{H}}$ and exercise embedding matrix, $\hat{\mathbf{E}}$. The exercise embedding matrix is a concatenation of the exercise embeddings matrix and exercise-relation matrix.

$$\hat{\mathbf{H}} = \begin{bmatrix} \mathbf{H}_{s_1} + \mathbf{P}_1 \\ \mathbf{H}_{s_2} + \mathbf{P}_2 \\ \vdots \\ \mathbf{H}_{s_n} + \mathbf{P}_n \end{bmatrix}, \hat{\mathbf{E}} = \begin{bmatrix} \mathbf{E}_{s_1} \\ \mathbf{E}_{s_2} \\ - \\ \mathbf{E}_{s_n} \end{bmatrix} \oplus \begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{n1} & \cdots & A_{nn} \end{bmatrix} \quad (2)$$

3.2. Self Attention

In our model, we employ the scaled dot product attention mechanism as an integral element [23]. This mechanism serves the crucial purpose of discerning the relative importance assigned to each previously solved exercise, subsequently influencing the prediction of the correctness associated with the current exercise. The acquisition of query and key-value pairs is executed through the following mathematical formulations:

$$\mathbf{Q} = \hat{\mathbf{E}}\mathbf{W}^Q, \mathbf{K} = \hat{\mathbf{H}}\mathbf{W}^K, \mathbf{V} = \hat{\mathbf{H}}\mathbf{W}^V, \quad (3)$$

Here, $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbf{R}^{d \times d}$ are the query, key and value projection matrices, respectively, which serve the function of linearly mapping the corresponding vectors into distinct spaces [23]. The significance of prior interactions with the

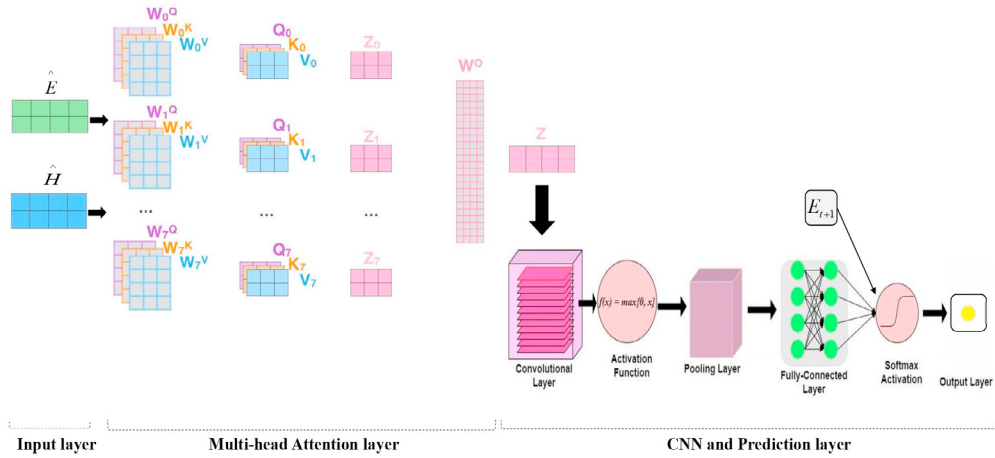


Fig. 2: The architecture of ACKT model.

current exercise is established by using attention weights. These attention weights are computed using the scaled dot product approach [23], which is defined as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right) \mathbf{V} \quad (4)$$

Multiple heads: To collectively focus on information from diverse representative subspaces, we conduct linear projections on queries, keys, and values h times, employing distinct projection matrices for each operation.

$$\text{Multihead}(\hat{\mathbf{H}}, \hat{\mathbf{E}}) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) \mathbf{W}^O, \quad (5)$$

where $\text{head}_i = \text{Attention}(\hat{\mathbf{E}}\mathbf{W}_i^Q, \hat{\mathbf{M}}\mathbf{W}_i^K, \hat{\mathbf{H}}\mathbf{W}_i^V)$ and $\mathbf{W}^O \in \mathbf{R}^{hd \times d}$.

Causality: Within our model, when it comes to predicting the outcome of the next exercise (at position $t + 1$), it becomes imperative to exclusively factor in the initial t interactions. Consequently, in the context of a given query Q_i , it is essential to disregard keys K_j for which the index $j > i$. To facilitate this constraint, we incorporate a causality layer which effectively conceals the weights learned from forthcoming interaction keys.

3.3. Convolutional and Prediction Layer

The self-attention layer described above results in weighted sum of values, V_i of the previous interactions. The rows of the matrix obtained from the multihead layer is $\mathbf{Z} = \text{Multihead}(\hat{\mathbf{H}}, \hat{\mathbf{E}})$. Subsequently, our focus shifts to extracting learning rate attributes from the matrix $\mathbf{Z}$. Individualized learning rates represent the absorptive capacities for knowledge of different students. The succession of student learning interactions may offer insights into learning rates, whereby students with elevated learning rates exhibit rapid mastery of knowledge concepts, whereas others encounter longer periods of trial and error. As a result, we formulate hierarchical convolutional layers to distill the learning rate features contained in continuous learning interactions of students and their relations. In our design, we

Table 3: Statistics of all datasets

Datasets	Statistics		
	Students	Concepts	Records
ASSIST2009	4,151	110	325,637
ASSIST2012	46,674	265	6,123,270
ASSIST2015	19,860	100	683,801
ASSISTchall	1,709	102	942,816
Statics2011	333	1,223	189,297

employ one-dimensional convolution as a key mechanism. The sliding window employed is parametrized as $\mathbf{W} \in \mathbf{R}^{2m \times d}$, $\mathbf{b} \in \mathbf{R}^d$. Thus a filter with size m in the convolution layer works on the interdependencies between previous interactions to generate deep features in order to construct the individualized knowledge state. The number of feature maps is also set to be d .

To enhance the efficiency of the training process, we introduce residual connections [8] from the input to the output of the convolutional layer. Subsequently, we apply L identical convolutional layers consecutively, forming a hierarchy of convolutional layers. This hierarchical structure serves the purpose of capturing the learning rates over distinct temporal spans. Lower layers focus on recent periods, while higher layers encompass a broader temporal scope, effectively monitoring a wider range of learning rate dynamics. The output matrix of the hierarchical convolutional layers $\mathbf{S} \in \mathbf{R}^{n \times d}$ represents the student knowledge state. We leverage the dot product between the current student knowledge state and the embedding of the forthcoming exercise to predict student performance:

$$\begin{cases} y_{t+1} = s_t \cdot \mathbf{E}_{t+1}, \\ p_{t+1} = \sigma(y_{t+1}). \end{cases} \quad (6)$$

For parameter learning within our model, we opt for the cross-entropy log loss as the objective function. This function operates between the prediction p_t and the actual answer a_t . Minimization of this objective is accomplished using the Adam optimizer [11]: $L = -\sum_{t=1}^N (a_t \log p_t + (1 - a_t) \log (1 - p_t))$

4. Experiments

4.1. Dataset Description

Our model's efficacy has been evaluated through the utilization of four publicly available real-world datasets, Table 3 shows the statistics of all datasets.

- **ASSIST2009** originates from the ASSISTments [4] platform, an online educational tool established in 2004. The dataset is gleaned from problem sets focused on skill building, where students engage with analogous exercises to attain mastery.
- **ASSIST2012** is the ASSISTments data for the school year 2012-2013 with affect predictions. The dataset consists of 2,541,201 interactions, 27,066 students, and 45,716 questions.
- **ASSIST2015** also emanates from the ASSISTments platform, encompassing response data recorded during the year 2015.
- **ASSISTChall** is harnessed from the 2017 ASSISTments data mining competition. It was compiled through a longitudinal study that tracks students' progress through their usage of ASSISTments.
- **Statics 2011** is sourced from an engineering statics course at the college level [12]. We have concatenated problem name and step name as the knowledge concept.

Table 4: Student Performance prediction comparison.

	ASSIST2009		ASSIST2012		ASSIST2015		ASSISTChall		STATICS	
	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
DKT [18]	0.74	0.708	0.701	0.722	0.701	0.68	0.691	0.712	0.815	0.723
DKVMN [26]	0.739	0.618	0.675	0.624	0.705	0.68	0.689	0.614	0.814	0.722
SAKT [17]	0.735	0.679	0.703	0.692	0.721	0.647	0.701	0.657	0.803	0.797
JKT [22]	0.798	0.753	—	—	0.765	0.757	0.745	0.737	0.856	0.832
Bi-CLKT [21]	0.857	0.802	—	—	0.765	0.757	0.775	0.764	0.865	0.835
KET [9]	0.802	0.796	0.846	0.826	—	—	—	—	—	—
ACKT (ours)	0.861	0.827	0.870	0.842	0.873	0.791	0.754	0.814	0.869	0.856

4.2. Evaluation Methodology

Metrics: The prediction task is framed as a binary classification scenario, specifically focused on whether an exercise is answered correctly or not. Accordingly, we assess performance by employing the Area Under the Curve (AUC) and accuracy metrics for comparison.

Baseline methods: We compare our model against the state-of-the-art KT methods.

- DKT [18]: employed RNN [14] and a Long Short Term Memory (LSTM) to predicate the probability of correctly answering a question at each time step.
- DKVMN [26]: exploited Memory Augmented Neural Network for KT. Using two matrices, key and value, it learns the correlation between the exercises and the underlying knowledge concept and student's knowledge state, respectively.
- SAKT [17]: developed an approach that identifies the knowledge concepts from the student's past activities that are relevant to the given knowledge concept and predicts his/her mastery based on the relatively few knowledge concepts that it picked.
- JKT [22]: employed a joint graph convolutional network-based approach for deep Knowledge Tracing, utilizing an advanced inference-generating framework.
- Bi-CLKT [21]: leveraged contrastive learning to effectively utilize large volumes of unlabeled data. The model comprises three key components: subgraph establishment, contrastive learning, and performance prediction.
- KET [9]: proposed an input representation by using graph embedding methods, providing richer information compared to traditional one-hot encoding. It incorporates two attention-based encoders specifically designed to account for students' knowledge states and engagement levels, resulting in a more comprehensive understanding of their learning processes.

Model Training and parameter selection.: We trained the model using 80% of the dataset and then evaluated its performance on the remaining 20%, employing a five-fold cross-validation technique to ensure robust performance metrics and to mitigate overfitting. We conducted experiments employing a configuration of 50 filters with size 2, 100 filters with size 3, and 100 filters with size 4 and the number of head in the attention layer is 8. For initialization of weights and optimization, we used a similar procedure as [17]. Our model, ACKT, was implemented using TensorFlow, and optimization was carried out through the ADAM optimizer with a learning rate of 0.001. Batch sizes of 256 were employed for the ASSISTChall and ASSIST2012 datasets and 128 for the remaining datasets. Notably, for datasets with a larger volume of records, such as ASSISTChall and ASSIST2012, a dropout rate of 0.3 was used, whereas for the other datasets, a dropout rate of 0.2 was applied. The maximum sequence length, n , was established in proportion to the average count of exercise tags per student. Specifically, n was set to 500 for ASSISTChall, ASSIST2012 and STATICS datasets, 100 for ASSIST2009, and 50 for the ASSIST2015 dataset.

Table 5: Ablation Study based on the AUC score.

Architecture	ASSIST2009	ASSIST2012	ASSIST2015	ASSISTChall	STATICS	Average
Default	0.861	0.870	0.873	0.754	0.869	0.845
No ER	0.855	0.856	0.871	0.752	0.860	0.838
No Dropout	0.860	0.849	0.866	0.758	0.862	0.839
Single head	0.852	0.833	0.857	0.740	0.849	0.826
Attention (No CNN)	0.852	0.829	0.860	0.741	0.855	0.827
CNN (No Attention)	0.843	0.840	0.858	0.739	0.857	0.827

5. Results and Discussion

Our model, ACKT, demonstrates superior performance compared to other baseline methods across multiple datasets, as illustrated in Table 4. The effectiveness of ACKT is primarily attributed to the fact that the attention mechanism can efficiently model a student's interaction history and identify the relevance of past interactions. Moreover, CNN is designed to generate the student's knowledge state based on the information acquired from the attention layer. It extracts deep features from this information to generate students' individualized knowledge state, learning rate and the final layer predicts the students' performance on the next exercise. In comparison, Bi-CLKT secures the second position, showcasing strong results particularly in datasets such as STATICS, where its use of contrastive learning frameworks aids in understanding implicit exercise relationships. JKT follows closely in third place, benefiting from its joint graph convolutional network that effectively learns conceptual embeddings of exercises, though it lags slightly behind Bi-CLKT in overall performance. The robust performance of ACKT highlights its advanced capability in integrating various aspects of student interaction and knowledge state, setting it apart from the other models. None of the existing works combined the multi-head attention mechanism with CNN and the latent interconnections among exercises, contributing to ACKT's superior performance.

5.1. Ablation Study

Table 5 shows the performance of default ACKT architecture and all the variants on all the datasets.

- No exercise relation (No ER): In this variation of ACKT, the exercise relation will not be considered for exercise representation, and the exercise embeddings are trained alone which drops the performance of the CAKT on all the datasets. Computing exercise relations using the student performance data and embedding similarity improves exercise representation which affects the overall performance of the ACKT.
- No Dropout: Dropout is used in neural networks to improve the model's generalization capability. This regularization method becomes notably influential when addressing datasets characterized by a scarcity of data points relative to the model's parameter complexity. As a result, role of dropout is more effective for ASSIST2012 dataset and ASSIST2015 dataset.
- Single head: Instead of using 8 heads as is the case in default architecture, we tried a variant of using only one head. Multiple heads help in capturing the attention weights in different subspaces. Using single head consistently drops the performance of ACKT on all the datasets.
- Attention (No CNN): Instead of generating the knowledge state using CNN, the student's interactions are only processed by the multi-head attention and the performance prediction is based on the interactions connections retrieved by the attention layer.
- CNN (No Attention): In this variation of ACKT, the multi-head attention is removed and the performance prediction only relies on the CNN layer, the interdependencies between student interactions and knowledge concepts are neglected which caused the performance to drop on all datasets.

6. Conclusion

In this work, we proposed a self-attentive convolutional based knowledge tracing model, ACKT. It analyzes a student's interaction history to generate the individualized knowledge state and predicts his/her performance on the next exercise. Extensive experimentation on a variety of real-world datasets shows that our model outperforms the state-of-the-art methods for assessing each student's mastery of KCs. Our model extracts the dependencies between the student's historical learning interactions by attention mechanism and then constructs the individualized prior knowledge and learning rate using CNN to predict the students' performance on the next exercise. Future work will focus on hyperparameter optimization and addressing data sparsity to enhance model performance and accuracy for diverse educational datasets.

References

- [1] Abdelrahman, G., 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, pp. 175–184.
- [2] Chen, P., Lu, Y., Zheng, V.W., Pian, Y., 2018. Prerequisite-driven deep knowledge tracing, in: 2018 IEEE International Conference on Data Mining (ICDM), pp. 39–48. doi:[10.1109/ICDM.2018.00019](https://doi.org/10.1109/ICDM.2018.00019).
- [3] Corbett, A.T., Anderson, J.R., 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction* 4, 253–278.
- [4] Feng, M., Heffernan, N., Koedinger, K., 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User modeling and user-adapted interaction* 19, 243–266.
- [5] Ghosh, A., Heffernan, N., Lan, A.S., 2020. Context-aware attentive knowledge tracing, in: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, pp. 2330–2339.
- [6] Graves, A., Graves, A., 2012. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, 37–45.
- [7] Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., Colmenarejo, S.G., Grefenstette, E., Ramalho, T., Agapiou, J., et al., 2016. Hybrid computing using a neural network with dynamic external memory. *Nature* 538, 471–476.
- [8] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.
- [9] Jiang, H., Xiao, B., Luo, Y., Ma, J., 2023. A self-attentive model for tracing knowledge and engagement in parallel. *Pattern Recognition Letters* 165, 25–32.
- [10] Kang, W.C., McAuley, J., 2018. Self-attentive sequential recommendation, in: 2018 IEEE international conference on data mining (ICDM), IEEE, pp. 197–206.
- [11] Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- [12] Koedinger, K.R., Baker, R.S., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J., 2010. A data repository for the edm community: The pslc datashop. *Handbook of educational data mining* 43, 43–56.
- [13] Lord, F.M., 1980. Applications of item response theory to practical testing problems. lawrence eribaum associates. Inc, Hillsdale, NJ.
- [14] Medsker, L.R., Jain, L., 2001. Recurrent neural networks. *Design and Applications* 5, 2.
- [15] Nakagawa, H., Iwasawa, Y., Matsuo, Y., 2019. Graph-based knowledge tracing: modeling student proficiency using graph neural network, in: IEEE/WIC/ACM International Conference on Web Intelligence, pp. 156–163.
- [16] O'Shea, K., Nash, R., 2015. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.
- [17] Pandey, S., Karypis, G., 2019. A self-attentive model for knowledge tracing, in: 12th International Conference on Educational Data Mining, EDM 2019, International Educational Data Mining Society, pp. 384–389.
- [18] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L.J., Sohl-Dickstein, J., 2015. Deep knowledge tracing. *Advances in neural information processing systems* 28.
- [19] Rasch, G., 1993. Probabilistic models for some intelligence and attainment tests. ERIC.
- [20] Self, J., 1990. Theoretical foundations for intelligent tutoring systems. *Journal of Artificial Intelligence in Education* 1, 3–14.
- [21] Song, X., Li, J., Lei, Q., Zhao, W., Chen, Y., Mian, A., 2022. Bi-clkt: Bi-graph contrastive learning based knowledge tracing. *Knowledge-Based Systems* 241, 108274.
- [22] Song, X., Li, J., Tang, Y., Zhao, T., Chen, Y., Guan, Z., 2021. Jkt: A joint graph convolutional network based deep knowledge tracing. *Information Sciences* 580, 510–523.
- [23] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. *Advances in neural information processing systems* 30.
- [24] Yeung, C.K., Yeung, D.Y., 2018. 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.
- [25] Yudelson, M.V., Koedinger, K.R., Gordon, G.J., 2013. Individualized bayesian knowledge tracing models, in: Artificial Intelligence in Education: 16th International Conference, AIED 2013, Memphis, TN, USA, July 9–13, 2013. Proceedings 16, Springer, pp. 171–180.
- [26] Zhang, J., Shi, X., King, I., Yeung, D.Y., 2017. Dynamic key-value memory networks for knowledge tracing, in: Proceedings of the 26th international conference on World Wide Web, pp. 765–774.