Multi-task Transformer with LSTM Model for Question Set Generation

Yeoun Chan Kim
AI R&D
TagHive Inc.
Seoul, Republic of Korea
ychan@tag-hive.com

Su Bin Jo
AI R&D
TagHive Inc.
Seoul, Republic of Korea
lizzy@tag-hive.com

Min Ah Kim
AI R&D
TagHive Inc.
Seoul, Republic of Korea
minah@tag-hive.com

Radhika Makharia
AI R&D
TagHive Inc.
Kolkata, India
radhika@tag-hive.com

Pankaj Agarwal
TagHive Inc.
Seoul, Republic of Korea
pankaj@tag-hive.com

***Abstract*—Amidst the COVID-19 pandemic, Educational Technology (EdTech) has emerged as a vital option for remote learning. This transition has ushered in technical innovations while revealing challenges, including the widening educational gaps attributed to educators' struggles in assessing student comprehension in self-directed learning. The extensively researched field of knowledge tracing, which models a student's knowledge state promises personalized education for effective learning. Originally reliant on statistical analysis, this approach has evolved with technological advances and extensive datasets. This research introduces a multi-task transformer with Long Short-Term Memory (LSTM) architecture capable of knowledge tracing and problem-solving time prediction. The model's adaptability facilitates task customization yielding superior performance in knowledge tracing. Our study explores problem-solving time prediction to motivate students and support educators in assessment design. Detailed implementation is provided in this paper.**

***Index Terms*—deep knowledge tracing, problem-solving time prediction, transformer, long short-term memory**

I. Introduction

In light of the COVID-19 outbreak, Educational Technology (EdTech) has garnered global attention as remote learning became essential while maintaining social distancing. This shift to remote learning has spurred technical innovations. However, it has also brought new issues to the forefront such as an increase in educational (learning outcomes) gaps. The reasons for this rising trend include the lack of access to in-person educational resources and personalized guidance, as well as the digital divide, characterized by disparities in access to technology and internet connectivity. As a result, it has become crucial for instructors to identify educational gaps and provide students with the necessary support.

The most extensively researched area in EdTech is how to model a student's knowledge state, referred to as knowledge tracing. Accurately modeling a student's knowledge enables personalized education, leading to more effective learning for students. Initially, knowledge tracing relied on statistical analysis of limited class-wise data. With technological advancements, a large dataset of student performance has become available, allowing machines to learn representations of students' skill acquisition from this data.

Inspired by the potential of machine learning in knowledge tracing, we shifted our focus to another aspect aimed at enhancing students' learning experiences: problem-solving time prediction. During examinations, students are required to solve questions within a limited time frame, testing both their knowledge and time management skills. The purpose of problem-solving time prediction is to motivate students to improve their fast problem-solving skills by comparing their own solving speed with what the machine predicts. Additionally, teachers can utilize these predictions to design assessments that consider individual students' abilities to solve questions quickly.

In this research, we introduce a multi-task transformer with a Long Short-Term Memory (LSTM) architecture capable of performing both knowledge tracing and problem-solving time prediction. The tasks can be easily modified by changing the target feature, evaluation metric, and the architecture's final activation function. In knowledge tracing, our model has outperformed the current state-of-the-art model while increasing the model's efficiency by more than 80%. The implementation details of our model are described in this paper.

II. Related Works

The term, knowledge tracing, was introduced by Corbett and Anderson in 1994 and has since been the subject of intense study in the field of EdTech. Initial knowledge tracing used Bayesian statistics to model students' skill acquisition, as exemplified by BKT [1]. BKT proposed an innovative method for modeling student learning; however, this approach required professional's assistance.

In the early stages of adopting big-data analysis and machine learning, researchers attempted to predict students' performance using collaborative filtering (CF) [2, 3]. With

advancements in machine learning, deep learning began to play a significant role in knowledge tracing, leading to the development of deep knowledge tracing (DKT) [4]. DKT employed recurrent neural network (RNN) to analyze patterns in sequences of student question-solving. The publication of DKT spurred the application of RNN in knowledge tracing, as demonstrated in works such as [5, 6].

In recent research, DKT models inspired by the transformer architecture [7] have outperformed previous models. The self-attention mechanism was first introduced in the SAKT model [8], marking a remarkable shift from topic-based prediction to question-based prediction. The goals of [5, 6] were to predict the correctness of the next interaction on a topic, regardless of the specific questions. SAKT [8] aimed to predict correctness on each individual question. DKT, utilizing the entire transformer architecture, was introduced in SAINT [9], and an improved version was introduced in SAINT+ [10]. After the publication of SAINT+, an online competition on deep knowledge tracing was held [11], and the first-ranked model [12] implemented the encoder of the transformer architecture and the LSTM layer. This model used only the last interaction of a student as a query to reduce $O(L^2)$ to $O(L)$ in a query and key matrix multiplication within attention calculation. Since the dataset used to validate the performance of SAINT+ is widely adopted, SAINT+ is currently considered the state-of-the-art model in deep knowledge tracing.

III. Model Architecture

The transformer with LSTM model is constructed of encoder layers from the transformer architecture, the LSTM layer, and a deep neural network layer. The encoder layers contain a multi-head self-attention network structure from [7], and the LSTM layer is based on [13]. The entire model architecture is demonstrated in Figure 1.

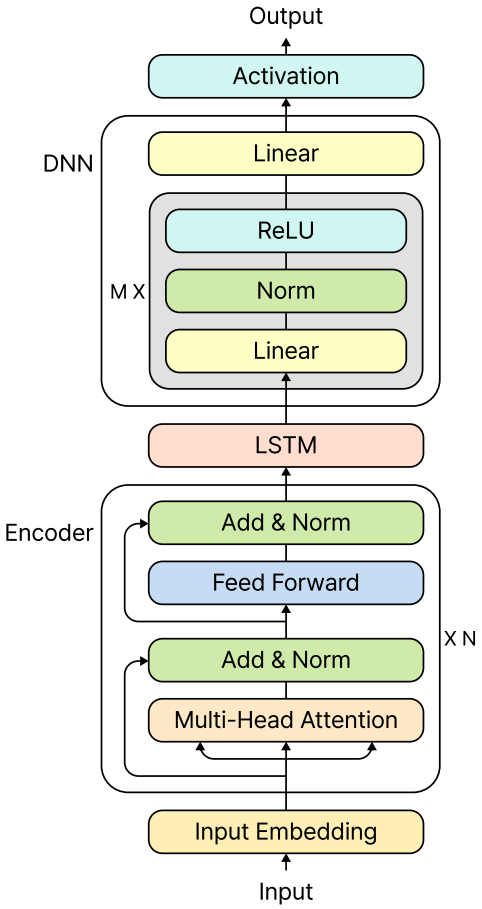


Fig. 1: The Transformer with LSTM model architecture.

A. Input Representation

We adopted the same input features as SAINT+ [10] to facilitate model performance comparison. SAINT+ introduced five key input features: exercise ID, part, answer correctness, elapsed time, and lag time. Each exercise is uniquely identified by an exercise ID, while exercises are categorized into subject-based groups, with the part feature serving as the representative category identifier. The answer correctness feature is assigned a binary value of 1 if a student answered an exercise correctly and 0 otherwise. The elapsed time indicates the duration, in seconds, a student took to respond, with any response times exceeding 300 seconds being capped at 300 seconds. The lag time represents the duration between the response to the previous exercise and the initiation of the current exercise, measured in minutes, with durations exceeding 1440 minutes being capped at 1440 minutes.

To process these features, we applied categorical embedding for exercise ID, part, and answer correctness, while continuous embedding was employed for elapsed time and lag time. Notably, when predicting the first interaction within a sequence, obtaining information regarding answer correctness, elapsed time, and lag time for the initial interaction is not possible. To address this, we utilized a start token embedding for these specific features. Differing from the approach in [12], we integrated a position embedding vector to capture the sequential order of exercises and student responses effectively. A visual representation of our input embeddings is provided in Figure 2.

B. Encoder

The encoder consists of stacked layers, each containing two sub-layers. The first sub-layer uses multi-head self-attention mechanism, and the second employs a feed-forward network. Residual connections and layer normalization are applied to enhance learning across sub-layers and maintain consistent dimensions throughout the model.

1) Multi-Head Self-Attention: The Multi-Head Self-Attention mechanism, a vital component of the transformer architecture, enhances understanding relationships within sequences by allowing the model to simultaneously focus on different parts of the input. It achieves this through multiple attention heads, each learning distinct patterns and dependencies, ultimately improving the model's ability to capture relationships and dependencies of the sequence. In our implementation, we employed an upper triangular mask in

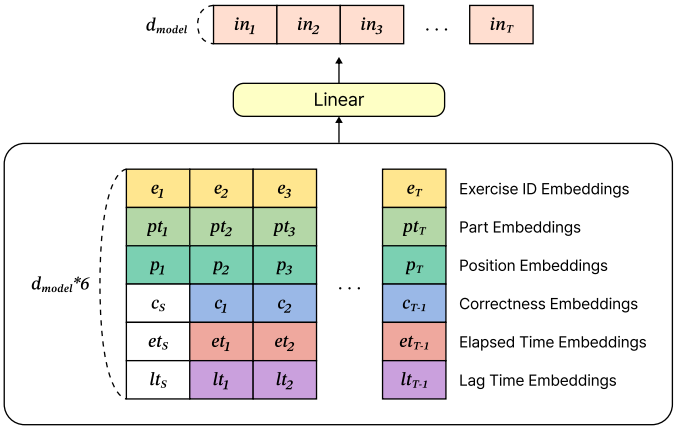


Fig. 2: Visual representation of input embeddings. This figure illustrates the structured embeddings used to encode input features, including exercise ID, part, answer correctness, elapsed time, and lag time, enhancing the model's understanding of sequential student interactions and exercise attributes.

the self-attention mechanism, ensuring that the model attends to information only from previous or current positions, which is essential for sequence data processing.

2) Feed-Forward Network: The feed-forward network is a critical component responsible for processing input data. It consists of two layers of linear transformations and ReLU activation function to capture intricate patterns and dependencies within the data. These layers enable the model to extract and combine features effectively, allowing it to understand both local and global relationships in sequential data. The feed-forward network's design is a key to generate coherent sequences and model complex interactions.

3) Residual Connection: The residual connection [14] is an element that tackles the vanishing gradient problem and boosts the model's depth and learning capacity. It adds the input to the model's output, creating a shortcut that allows gradient flow during training. By preserving the original input and combining it with the intermediate layer's output, the residual connection mitigates the issue of gradient vanishing, thereby enabling deeper learning.

C. Long Short-Term Memory

LSTM is designed for tasks involving sequential data like natural language processing, speech recognition, and time series analysis. Unlike traditional RNN, LSTM excels at capturing long-range dependencies and mitigating the vanishing gradient problem. They achieve this through a sophisticated gating mechanism that regulates information flow, allowing them to retain essential information over extended time steps while forgetting less relevant data.

D. Deep Neural Network

In the Deep Neural Network (DNN) layer, we have incorporated non-linearity to enhance its capabilities. The DNN architecture is composed of three fundamental components: a linear layer, a layer normalization, and a ReLU activation function. Additionally, to mitigate the risk of overfitting, we have strategically incorporated dropout after the layer normalization process. The output of the DNN is versatile, catering to different aspects of our model's functionality. In the context of deep knowledge tracing, the DNN output yields a sequence of probability values. This is achieved through the application of a sigmoid activation layer, facilitating the model's ability to assess students' knowledge states. Conversely, in the problem-solving time prediction, the DNN produces numerical predictions by passing through a ReLU activation layer.

IV. Experiments

A. Dataset

We utilized two datasets throughout our experiments in deep knowledge tracing and problem-solving time prediction: the EdNet dataset [15] and the dataset from [11], referred to as the RAIEd2020 dataset. Both datasets, provided by Riiid Labs, a South Korean EdTech company, have been intricately curated to enhance the effectiveness of their adaptive learning platform, known as "Santa". They capture a wide range of student interactions, including timestamps, content IDs for educational materials, user IDs, correctness indicators, and supplementary metadata.

These comprehensive datasets serve as essential foundations for the development and evaluation of machine learning models targeting answer correctness prediction and personalized learning enhancement within the field of EdTech. When comparing the statistics between these datasets, the EdNet dataset features 131,441,538 user interactions, 784,309 students, and an average of 441.20 interactions per student. In contrast, the RAIEd2020 dataset boasts over 101 million records, 99,271,300 question interactions from 393,646 students. This underscores the substantial scale and richness of both resources, rendering them invaluable for educational research and analysis. The detailed statistics can be found in Table I.

TABLE I: Statistics of Datasets

	EdNet	**RAIEd2020**
Interactions	131,441,538	99,271,300
Students	784,309	393,656
Questions	13,169	13,523
Skills	293	188

B. Training Details

We trained our model and comparison models using a single RTX 3060 GPU. For our model, we employed a sequence length of 100, a batch size of 64, embedding dimensions of 128, and 8 parallel attention layers for the multi-head attention mechanism. In the case of encoder layers, we utilized a stack of $N = 4$. Regarding the DNN architecture, we found that setting the number of DNN layers (M) to 2 yielded the best performance. As illustrated in Figure 3, we applied the Adam optimizer with a learning rate that varied over time based on the equation from [7] and maintained warm-up steps of 4,000.

We conducted a comparative training run with two different sets of steps, 100,000 and 300,000. To ensure reproducibility, we controlled randomness in the datasets by fixing their seed values.

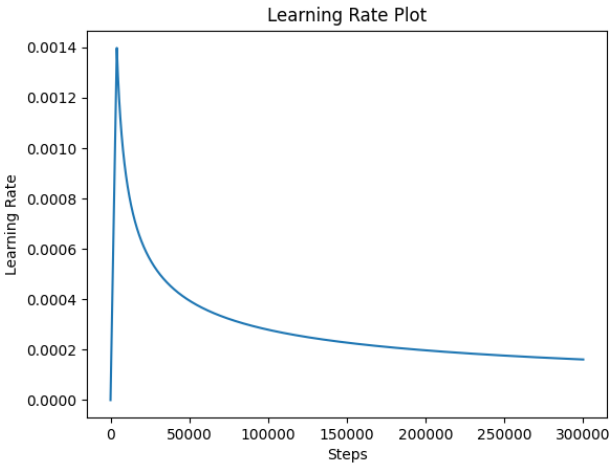


Fig. 3: Learning rate variation over 300,000 steps.

C. Deep Knowledge Tracing

We trained our model with a sigmoid activation output layer to predict the probability of a student correctly answering each question, using answer correctness as the target feature. To evaluate our model's performance, we compared it to the SAINT+ model [10] in terms of area under the receiver operating characteristic curve (AUC) and accuracy (ACC). Both models were trained to minimize binary cross-entropy between the target and input features.

1) Experimental Results: As shown in Table II, on both the EdNet and RAIEd2020 datasets, our model outperformed the SAINT+ model by an average of 0.24% in AUC. The best performance was observed on the EdNet dataset with 300K steps. In terms of a step time, our model required only 0.02 seconds per step, while the SAINT+ model took 0.17 seconds per step, representing a remarkable 88.24% increase in efficiency.

TABLE II: Test Result

MODEL	DATASET	STEPS	AUC	ACC
SAINT+	EdNet	100K	.7705	.7284
		300K	.7743	.7311
	RAIEd2020	100K	.7694	.7216
		300K	.7736	.7306
Transformer with LSTM	EdNet	100K	.7715	.7289
		300K	**.7761**	**.7333**
	RAIEd2020	100K	.7726	.7310
		300K	.7749	.7313

D. Problem-Solving Time Prediction

In comparison, we modified the final activation layer to a ReLU activation during the training of our model for predicting students' problem-solving times. We used elapsed time as the target feature, and the model aimed to minimize the mean squared error. To assess the performance of our model, we conducted evaluations using the XGBoost and the SAINT+ model. All models were tested based on the root mean square error (RMSE).

1) Experimental Results: As shown in Table III, our model is observed to perform 9.12% better on average in terms of RMSE compared to the XGBoost and SAINT+ models on both datasets. The best performance was observed on the RAIEd2020 dataset with 300K steps. In line with the deep knowledge tracing experiment, our model was 88.24% more efficient than the SAINT+ model. Since XGBoost operates using a boosting algorithm and decision trees, it does not rely on the concept of steps for training, and as a result, we could not directly compare its efficiency with our model and SAINT+.

TABLE III: Test Result

MODEL	DATASET	STEPS	RMSE
XGBoost	EdNet		15.4413
	RAIEd2020		14.6764
SAINT+	EdNet	100K	13.6601
		300K	13.4592
	RAIEd2020	100K	13.5348
		300K	13.3188
Transformer with LSTM	EdNet	100K	13.2108
		300K	12.4589
	RAIEd2020	100K	12.9369
		300K	**12.3425**

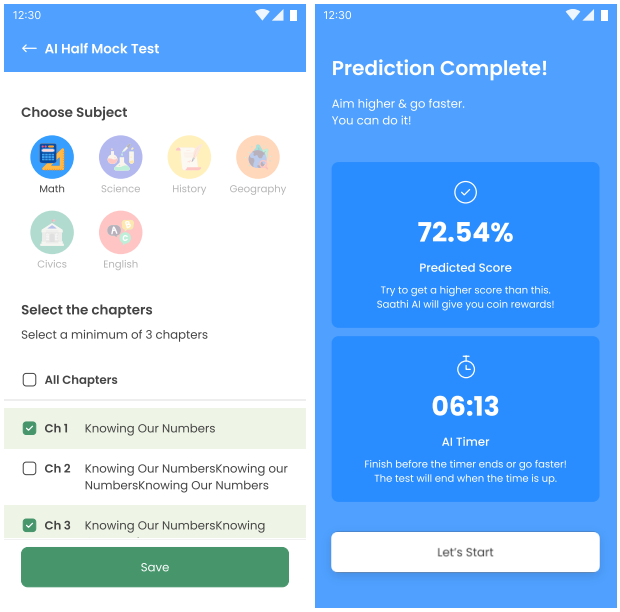
V. Model Implementation

Our research team operates Class Saathi, a comprehensive platform serving students from grades 3 to 10 across a multitude of subjects. At its core, the transformer with LSTM model plays a pivotal role in generating tailored question sets, optimizing the learning experience for students.

Among various use cases in deep knowledge tracing and problem-solving time prediction, Figure 4 demonstrates the model's implementation in the creation of predictive scores and timers. When students select a subject and chapters of interest, as shown in Figure 4a, our system generates a question set that evenly distributes chapters of interest and question difficulties (easy, medium, and hard). The model predicts the correctness of each question, and an appropriate timeline for completing the question set, thereby enhancing the efficiency and effectiveness of study sessions. After completing the calculations, we present the average predicted answer correctness as a percentage and the total time required to complete the question set, as depicted in Figure 4b.

VI. Conclusion

In this research, we introduced a model architecture that combines the encoder of the transformer with an LSTM layer to perform deep knowledge tracing and problem-solving time prediction. Our model outperformed the current state-of-the-art model in deep knowledge tracing and the baseline models



(a) The interface where students select their subject and chapters of interest.

(b) The average predicted answer correctness as a percentage and the total time required to complete the generated question set are displayed.

Fig. 4: An overview into model implementation.

in problem-solving time prediction by 0.24% and 9.12%, respectively. Additionally, we improved the model's efficiency by 88.24% compared to the SAINT+ model while maintaining superior performance. We also provided an implementation of the model in a real-world service to illustrate its potential use-cases. Both deep knowledge tracing and problem-solving time prediction hold significant promise in assessing students, personalizing education, and generating tailored learning paths. Our future research will aim to maximize the potential of technology in the EdTech field to better support both instructors and students.

Acknowledgment

This work was supported by the Technological Innovation R&D Program (S3280745) funded by the Ministry of SMEs and Startups (MSS, Korea).

References

[1] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User modeling and user-adapted interaction*, vol. 4, pp. 253–278, 1994.

[2] N. Thai-Nghe, L. Drumond, A. Krohn-Grimberghe, and L. Schmidt-Thieme, "Recommender system for predicting student performance," *Procedia Computer Science*, vol. 1, no. 2, pp. 2811–2819, 2010.

[3] K. Lee, J. Chung, Y. Cha, and C. Suh, "Machine learning approaches for learning analytics: Collaborative filtering or regression with experts," in *NIPS workshop, Dec*, 2016, pp. 1–11.

[4] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein, "Deep knowledge tracing," *Advances in neural information processing systems*, vol. 28, 2015.

[5] J. Zhang, X. Shi, I. King, and D.-Y. Yeung, "Dynamic key-value memory networks for knowledge tracing," in *Proceedings of the 26th international conference on World Wide Web*, 2017, pp. 765–774.

[6] C.-K. Yeung and D.-Y. Yeung, "Addressing two problems in deep knowledge tracing via prediction-consistent regularization," in *Proceedings of the fifth annual ACM conference on learning at scale*, 2018, pp. 1–10.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[8] S. Pandey and G. Karypis, "A self-attentive model for knowledge tracing," *arXiv preprint arXiv:1907.06837*, 2019.

[9] Y. Choi, Y. Lee, J. Cho, J. Baek, B. Kim, Y. Cha, D. Shin, C. Bae, and J. Heo, "Towards an appropriate query, key, and value computation for knowledge tracing," in *Proceedings of the seventh ACM conference on learning@ scale*, 2020, pp. 341–344.

[10] D. Shin, Y. Shim, H. Yu, S. Lee, B. Kim, and Y. Choi, "Saint+: Integrating temporal features for ednet correctness prediction," in *LAK21: 11th International Learning Analytics and Knowledge Conference*, 2021, pp. 490–496.

[11] A. Howard, bskim90, C. Lee, D. H. P. T. Jeon, J. J. Baek, K. Chang, kiyoonkay, NHeffernan, seonwooko, S. Dane, and Y. Lee, "Riiid answer correctness prediction," 2020. [Online]. Available: https://kaggle.com/competitions/riiid-test-answer-prediction

[12] S. Jeon, "Last query transformer rnn for knowledge tracing," *arXiv preprint arXiv:2102.05038*, 2021.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[15] Y. Choi, Y. Lee, D. Shin, J. Cho, S. Park, S. Lee, J. Baek, C. Bae, B. Kim, and J. Heo, "Ednet: A large-scale hierarchical dataset in education," in *Artificial Intelligence in Education: 21st International Conference, AIED 2020, Ifrane, Morocco, July 6–10, 2020, Proceedings, Part II 21*. Springer, 2020, pp. 69–73.