Broader and Deeper: A Multi-Features with Latent Relations BERT Knowledge Tracing Model

Zhaoxing Li[1(✉)], Mark Jacobsen[1], Lei Shi[2], Yunzhan Zhou[1], and Jindi Wang[1]

[1] Department of Computer Science, Durham University, Durham, UK
{zhaoxing.li2,mark.jacobsen,yunzhan.zhou,jindi.wang}@durham.ac.uk
[2] Open Lab, School of Computing, Newcastle University, Newcastle upon Tyne, UK
lei.shi@ncl.ac.uk

Abstract. Knowledge tracing aims to estimate students' knowledge state or skill mastering level over time, which is evolving into an essential task in educational technology. Traditional knowledge tracing algorithms generally use one or a few features to predict students' behaviour and do not consider the latent relations between these features, which could be limiting and disregarding important information in the features. In this paper, we propose MLFBK: A Multi-Features with Latent Relations BERT Knowledge Tracing model, which is a novel BERT based Knowledge Tracing approach that utilises multiple features and mines latent relations between features to improve the performance of the Knowledge Tracing model. Specifically, our algorithm leverages four data features (*student_id*, *skill_id*, *item_id*, and *response_id*, as well as three meaningful latent relations among features to improve the performance: individual *skill mastery*, *ability profile* of students (learning transfer across skills), and *problem difficulty*. By incorporating these explicit features, latent relations, and the strength of the BERT model, we achieve higher accuracy and efficiency in knowledge tracing tasks. We use t-SNE as a visualisation tool to analyse different embedding strategies. Moreover, we conduct ablation studies and activation function evaluation to evaluate our model. Experimental results demonstrate that our algorithm outperforms baseline methods and demonstrates good interpretability.

Keywords: Knowledge Tracing · BERT · Multi-Features · Latent Relations

1 Introduction

In recent years, Technology Enhanced Learning (TEL) has become an essential research field, mainly due to the increasing need for innovative solutions in the education sector. One of the most promising approaches to this problem is Intelligent Tutoring Systems (ITS), which could provide personalised learning experiences for each student. To achieve this personalisation, ITS requires a reliable method for estimating students' knowledge state and learning progress. This

O. Viberg et al. (Eds.): EC-TEL 2023, LNCS 14200, pp. 183–197, 2023.
https://doi.org/10.1007/978-3-031-42682-7_13

method is known as Knowledge Tracing (KT). KT estimates students' knowledge state or skill mastering level based on the student's interaction data collected from ITS [15]. Accurate and efficient KT models are essential for ITS and educators to provide personalised learning experiences and support to students, such as tailored feedback, targeted hints, and relevant additional learning resources.

Generally, there are three kinds of KT models. Bayesian Knowledge Tracing (BKT), Logistic KT models, and Deep Learning based Knowledge Tracing (DKT) [16]. BKT is one of the earliest and most influential KT models. It uses a probabilistic framework to model student knowledge and learning state over time [2]. Logistic KT models are developed based on the concept of logistic regression, a statistical technique utilised to model the probability of a binary outcome by utilising one or more predictor variables. While BKT and Logistic models have achieved significant success in predicting student performance, they have also been criticised for their inability to capture the complex relationships between different skills and concepts [18]. To address this limitation, the more recent DKT models have utilised deep learning techniques to capture the complex interactions between student responses, skills, and questions [23]. DKT models have achieved state-of-the-art performance on benchmark datasets but require a large amount of training data to achieve good results.

Previous KT models have often been limited by their reliance on a single or few features, which could fail to capture the complexity of student learning behaviour data. Numerous studies have demonstrated that incorporating one or two additional features could enhance the performance of KT models [8,32]. Additionally, Minn *et al.* suggested that identifying latent features could further improve the performance of KT [20]. However, to date, there has been no research that has investigated the effectiveness of combining multiple features and latent relations to improve the performance of KT models.

Therefore, the research question of this paper is: *Whether incorporating multiple features and mining the latent relations between features together could improve the accuracy and efficiency of KT models?*

In this paper, we present the **M**ulti-**L**atent **F**eature **B**ERT **K**nowledge Tracing model that is both "broader" and "deeper" than the previous models to address the above-mentioned limitation by incorporating multiple features with mined latent relations that provide richer and more diverse contextual information. By "broader", we incorporate four different types of features into our model: *student_id*, *skill_id*, *question_id*, and *response_id*. These features provide additional contextual information to help the model better capture individual differences in learning and problem-solving strategies. By "deeper", we employ a feature engineering method to extract three meaningful latent relations important for representing student's behaviour: *skill mastery*, *ability profile* of students (learning transfer across skills), and *problem difficulty*. We utilise a monotonic convolutional multi-head self-attention mechanism to combine the above explicit features and latent relations. By incorporating these explicit features and latent relations, our model could better account for the nuances and complexities of student learning and achieve superior performance compared to existing KT models. The experimental results show that MLFBK outperforms the four base-

line models on five benchmark datasets. Furthermore, the t-SNE as the visualisation tool was used to analyse the interpretability of MLFBK and the embedding strategies. The experimental results show that MLFBK outperforms the baseline models and could effectively enhance the interpretability of deep learning based KT models.

The main contributions of our paper lie in the following three aspects:

1. We propose MLFBK, a novel **M**ulti- **F**eatures with **L**atent relations **B**ERT **K**nowledge Tracing model, which not only considers the multiple explicit features but also deeply mines the latent relations between the features by using a feature engineering method.[1]
2. Our model achieves state-of-the-art performance, outperforming four existing state-of-the-art models on five benchmark datasets. Moreover, we conduct ablation experiments and demonstrate different embedding strategies with a visualisation tool to investigate the contribution of different latent relations.
3. MLFBK exhibits good interpretability as a deep learning based KT method and has advantages in training efficiency.

2 Related Work

This paper aims to present a novel BERT-based KT model that incorporates multi-features and latent relations. Therefore, we first review the cornerstone of the BERT model – the Transformer based models and their applications. Then we review the development of KT methods in general. At last, we review existing KT methods from the perspective of the number of integrated features.

2.1 Transformer-Based Models and Application

The Transformer architecture, proposed by Vaswani *et al.* [28], is a type of neural network that has gained widespread popularity in natural language processing (NLP), and other domains due to its ability to effectively model long-range dependencies and capture complex patterns in sequential data. Transformers have been used in various NLP tasks, including language translation, question answering, and text classification, and have achieved state-of-the-art performance on many benchmarks [13].

In addition to NLP, Transformers have also been applied in other domains, such as computer vision [10], speech recognition [22], and recommendation systems [30]. For example, the Vision Transformer (ViT) has recently been proposed as an alternative to convolutional neural networks (CNNs) for image classification tasks, achieving competitive performance on several benchmark datasets.

Besides the basic Transformer models, many powerful evolutions of Transformer-based methods were proposed, such as the GPT [6] and BERT [3]. The well-known ChatGPT originated from GPT [19]. BERT (Bidirectional

[1] Source code and datasets are available at https://github.com/Zhaoxing-Li/MLFBK.

Encoder Representations from Transformers), introduced by Devlin *et al.* [3], is a pre-trained Transformer-based language model that has achieved state-of-the-art performance on various NLP tasks. BERT utilises self-attention and masked language modelling (MLM) techniques to train the Transformer bidirectionally. Its remarkable ability to process natural language text effectively and generate high-quality embeddings has made it a popular choice and a superior performer in many Deep Learning tasks [14]. BERT has also been adapted to various other fields with excellent results. For instance, ConvBERT utilises the original BERT architecture in image processing task [9], BERT4Rec enhances the performance recommendation systems [25], and LakhNES improves the quality of music generation by incorporating BERT [4].

2.2 Knowledge Tracing

Knowledge Tracing is a technique utilised in educational data mining that aims to model students' knowledge state and mastering level of the learning concepts or subjects [31]. Generally, the KT models could be classified into three categories based on the different structures of the modelling approach that the model used: probabilistic models, logistic regression KT models, and deep learning-based models [18].

Probabilistic KT models assume a student's learning process follows a Markov process. They use a probabilistic graphical model such as Hidden Markov Model (HMM) or Bayesian Belief Network to track their changing learning states. Bayesian Knowledge Tracing (BKT) is a classic probabilistic model that has been used for this purpose, but it has several limitations: BKT does not account for the complexity or difficulty of concepts and skills and assumes that each question requires only one skill. This makes it difficult to process complex problems involving multiple skills and complex relationships between concepts, questions, and skills. To address these limitations, researchers have proposed models including Dynamic BKT (DBKT), which uses Dynamic Bayesian Network (DBN) to model prerequisite hierarchies and dependencies of multiple skills [5]. The logistic KT models are based on the principle of logistic regression, which is a statistical method used to estimate the probability of a binary outcome by using one or more predictor variables. However, both BKT and logistic KT models struggle to process multiple topics or skills and fail to account for other features that may impact student learning.

To overcome these limitations, researchers have turned to deep learning technologies to develop Deep Knowledge Tracing (DKT) [14]. DKT models a knowledge tracing task as a sequence prediction problem and has shown promise in achieving better performance than BKT and logistic KT models. The self-attention mechanisms were widely used in deep learning architectures, which have also been applied to KT models, resulting in models such as SAKT and SAINT+ [24]. These methods have achieved higher performance than traditional DL-based methods. More recently, several BERT-based methods were proposed that achieved state-of-the-art performance. BEKT [27] is a deep knowledge

tracing with bidirectional encoder representations from transformers. MonaconBERT [12] utilised the monotonic attention based ConvBERT to improve the knowledge tracing.

2.3 KT Models with Different Feature Numbers

Single feature KT models Single-feature KT models use only one feature, usually exercise or skill, to predict a student's knowledge or mastery of a particular skill or concept. Deep Knowledge Tracing (DKT) and Self-Attentive Knowledge Tracing (SAKT) [26] are examples of single-feature models that have been proposed to improve performance by using different techniques, such as LSTM networks and attention mechanisms to deal with the sparsity of exercise data.

Double-feature KT models Double-feature KT models use both exercise and skill features, resulting in significant performance gains compared to single-feature models. Deep Hierarchical Knowledge Tracing (DHKT) [29], Bi-Interaction Deep Knowledge Tracing (BIDKT) [11], and Attentive Knowledge Tracing (AKT) [7] are examples of double-feature models that have been proposed to improve performance by modelling the hierarchical relations between skills and exercises and proposing new attention mechanisms and embedding methods.

Multi-feature KT models Multi-factor KT models integrate multiple learning-related factors into the model to improve performance. Exercise-aware Knowledge Tracing (EKT) [17] and Relation-aware self-attention Knowledge Tracing (RKT) [21] are examples of multi-feature models that have been proposed to integrate information such as the exercise-making sequence, the relations between skills and time delay since the last interaction, and the text information of the exercise content.

3 Methodology

3.1 Problem Statement

The goal of knowledge tracing is to use a series of interaction data from Online Learning Systems (OLS) or Intelligent Tutoring Systems (ITS) to predict the correctness of a student's next answers. The student's interactions are represented by a data sequence, denoted as $x_1, ..., x_t$, where $t-th$ is represented as $x_t = (qt, at)$. Here, q_t refers to the $t-th$ question and indicates whether the student's answer is correct (1) or not (0).

3.2 Proposed Model Architecture

We propose a novel Knowledge Tracing model, Multi Features with Latent Relations BERT Knowledge Tracing (MLFBK), to improve the traditional KT models by incorporating multi-features and mining latent relations between different

features in the student historical interaction data. Figure 1 shows the architecture of MLFBK, which consists of three parts: embedding on the left; BERT-based architecture in the middle; the correctness sequence output on the right. The embedding part on the left further contains two components: the Multi-Features embedding on the top, and the Latent-Relations embedding at the bottom.

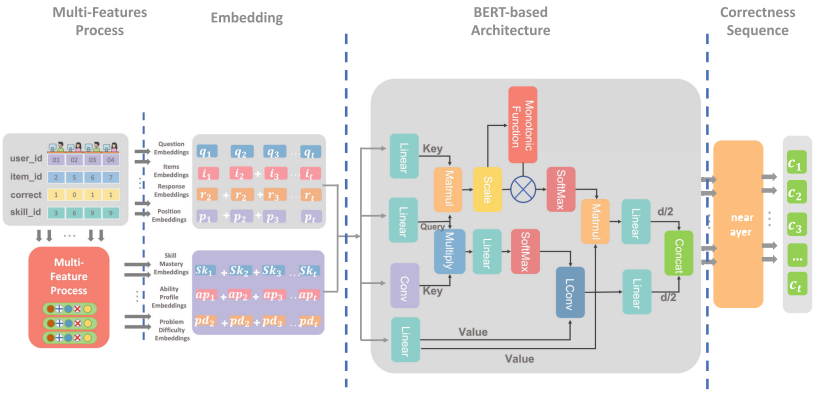

Fig. 1. The architecture of MLFBK. MLFBK consists of three parts: 1) the multi-features process (on the left), 2) the BERT-based architecture (in the middle), and 3) the correctness sequence output part(on the right).

Multi-Features Embedding. In the Multi-Features Embedding part, we incorporate four different types of features into our model: *student_id*, *skill_id*, *question_id*, and *response_id*. Particularly, *student_id* is utilised to generate the interaction sequences. It is also used in the Latent Relation Embedding component for calculating the *skill mastery* embedding and the *ability profile* embedding. These features need *student_id* to keep track of a single student.

Latent Relations Embedding. In the Latent Relations Embedding component, we use a feature engineering method proposed by work [20]. Using this method, we mine three different meaningful latent relations among the behaviour data of individual students: *skill mastery*, *ability profile* (learning transfer across skills), and *problem difficulty*.

Skill Mastery. The formulation of skill mastery is based on the Bayesian Knowledge Tracing (BKT) model, which uses four parameters to represent probabilities related to a student's mastery of a skill. These parameters include $P(L_o)$, the probability that a student masters the skill before attempting the first problem associated with it; $P(T)$, the probability that a student will master the skill after the next practice opportunity; $P(G)$, the probability that a student guesses the correct answer to a question despite not knowing the skill; and $P(S)$, the probability that a student answers a question incorrectly despite knowing the

skill. Skill mastery is the probability of learning a skill rather than the probability that a student applies the skill correctly. A BKT model is trained for each skill, and the inputs to each skill model are the binary responses of a student on that single skill. Figure 2 shows the Skill Mastery mining process.

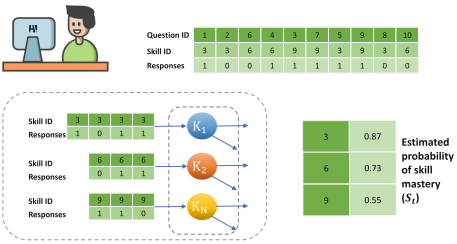

Fig. 2. Estimated the probability of skill mastery at each timestamp.

Ability Profile. Students' interactions are divided into multiple time intervals, and past performance is encoded to estimate their ability profile. The ability profile is encoded as a cluster ID and updated after each time interval using all previous attempts on each skill. The K-means algorithm is used to evaluate the temporal long-term learning ability of students in both training and testing at each time interval. Figure 3 shows the ability profile extraction process.

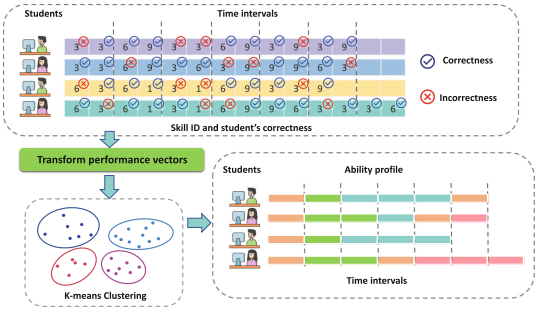

Fig. 3. Estimated the probability of skill mastery at each timestamp.

Problem Difficulty. This is calculated on a scale between 1 and 10, with 1 being the easiest and 10 being the most difficult. We use function 1 to map the average success rate of a problem onto the 10-level scale. The problem difficulty (p_j) could be calculated as:

$$\delta\left(p_j\right) = \left\lfloor \frac{\sum_i^{|N_j|} O_i\left(p_j\right)}{|N_j|} \cdot 10 \right\rfloor \tag{1}$$

where P_j is the j^{th} problem. N_j is the set of the students who tried to solve problem p_j. $O_i\left(p_j\right)$ is the first attempt of student i to solve problem p_j. Problems with a higher success rate are considered easier, while problems with a lower success rate are considered more difficult.

Overall, in the first part of our model, we incorporate question embedding E_q, items embedding E_i, response embedding E_r, learnable positional embedding E_{pos}, skill mastery embedding E_{SK}, ability profile embedding E_{AP}, and problem difficulty embedding E_{PD}. The final input embedding is denoted as:

$$E_{\text{input}} = E_q + E_i + E_r + E_{pos} + E_{SK} + E_{AP} + E_{PD}. \tag{2}$$

BERT Based Architecture. The second part of our proposed architecture is a BERT-based method (shown in Fig. 1). First, the encoder blocks use the pre-LN Transformer architecture with 12 layers to normalise the input vectors E_{input}. The pre-LN can be formulated as follows:

$$z = LN_{\text{pre}}\left(E_{\text{input}}\right) \tag{3}$$

The normalised value z is then transformed into the query, key, and value of monotonic convolutional multi-head attention. This result is passed through a dropout layer and added to the embedding vectors as a residual connection.

$$a = x + D(\text{MonoConvMulAttn}(z, z, z)) \tag{4}$$

The output is normalised and passed through fully connected layers with a LeakyReLU activation function. The results are again normalised through a dropout layer, and the second result is added as a residual connection.

$$fc = W_{fc2}\left(\text{LeakyReLU}\left(W_{fc1}\right)\right) \tag{5}$$

Moreover, we utilise a monotonic convolutional multi-head attention proposed by [12], which is combined with mixed-attention and monotonic attention, to represent forgetting in sequence data. Monotonic multi-head attention uses an exponential decay mechanism to measure the distance between sequences, while span-based dynamic convolution uses a lightweight convolution to combine query and key vectors.

3.3 Experiment Setting

Datasets. We adopted four benchmark datasets to validate the performance of the MLFBK model, including EdNet [1][2], assist09[3], assist12[4], algebra06[5].

[2] https://github.com/riiid/ednet.
[3] https://sites.google.com/site/assistmentsdata/home.
[4] https://sites.google.com/site/assistmentsdata/home.
[5] https://pslcdatashop.web.cmu.edu/KDDCup.

Baseline Models. In this study, we evaluated the performance of our MLFBK model by comparing it with three state-of-the-art models: MonaCoBERT [12], BEKT [27], and AKT [7], as well as the top two baseline models (SSAKT and LTMTI) in the Riiid Answer Correctness Prediction Competition hosted on Kaggle[6].

Evaluation Metrics and Validation. We used the area under the curve (AUC) as the evaluation metric to compare the model's performance on four benchmark datasets. After that, we conducted an activation function evaluation to compare the different activation functions. We also conducted an ablation study to identify the contribution of different latent relations. Furthermore, we applied t-SNE as the visualisation tool to evaluate our method's embedding strategies and interpretability.

Hyperparameters for Experiments. For a fair comparison, all baseline models were trained using the same set of parameters. Specifically, training was conducted with a batch size of 64 and a train/test split ratio of 0.8/0.2. The model was trained for 100 epochs with the Adam optimiser and a learning rate of 0.001. The loss function used was binary cross-entropy, and the model utilised a total of 12 encoder layers with a hidden size of 512 and 8 attention heads. The data was preprocessed by splitting it into interaction sequences with a maximum length of 100. In cases where a student had less than 100 interactions, the remaining sequence was padded with zeros. For students with more than 100 interactions, the sequence was split into multiple subsequences of length 100.

4 Results and Discussion

4.1 Overall Performance

Table 1 presents the comparison results of MLFBK with five other KT models, including MonaCoBERT, BEKT, AKT, SSAKT, and LTMTL, on four benchmark datasets, including EdNet, assist09, algebra06, and assist12. It is clear from Table 1 that MLFBK outperforms the other five KT models on all four datasets in terms of AUC, indicating that MLFBK is a promising method for KT. Take the algebra06 dataset as an example: MLFBK achieves an AUC of 0.8327, which is 1.4%, 2.9%, 3.9%, 5.2%, and 2.2% higher than the AUC values of MonaCoBERT, BEKT, SSAKT, LTMTI, and AKT, respectively. The average improvement on this dataset is 3.12%. MonaCoBERT and BEKT also perform relatively well, with AUC values close to those of MLFBK on some datasets. SSAKT and LTMTI, on the other hand, have lower AUC values, indicating weaker performance. The results suggest that MLFBK is a competitive method for knowledge tracing and could potentially improve the accuracy of student modelling by incorporating more student action features and mining latent relations.

[6] https://www.kaggle.com/code/datakite/riiid-answer-correctness.

Table 1. Comparison of different KT models on five benchmark datasets. The best performance is denoted in bold.

Dataset	Metrics	MLFBK	Monaco	BEKT	SSAKT	LTMTI	AKT
EdNet	AUC	**0.8278**	0.7336	0.8204	0.7981	0.8023	0.7982
assist09	AUC	**0.8524**	0.8059	0.8227	0.6754	0.8132	0.7691
algebra06	AUC	**0.8412**	0.8201	0.8165	0.7937	0.7915	0.8143
assist12	AUC	**0.8350**	0.8132	0.7167	0.7356	0.6834	0.8034

4.2 Ablation Study

In order to identify the contribution of each latent relation in the MLFBK model to the overall performance, we conducted an ablation study. The results are summarised in Table 2. MLFBK* in the table indicates the basic model structure with explicit features. *ap* represents ability profile, *sm* represents skill mastery, and *pd* represents problem difficulty. Table 2 also shows the AUC values for different versions of MLFBK* that were trained with different combinations of latent relations. It is clear that the performance of the MLFBK model is influenced by the different embedding strategies used for different relations. The models incorporating all three latent relations achieved the highest AUC values on three of the four datasets, except the assist09. Nevertheless, it also achieved the second-highest score in the assist09.

The *problem difficulty* contributed significantly to the model's performance, with the models that used only the problem difficulty embedding achieving the highest AUC values on four datasets compared to other single latent relation embeddings. The combination of *problem difficulty* and *ability profile* achieved the best performance on the assist09 dataset and the second-highest performance on EdNet, indicating that the combination of these two latent relations has more weight in the predictions. The *skill mastery* feature had a comparatively lower impact on the model's performance, with the models that used only the *skill mastery* feature achieving the lowest AUC values on four datasets. However, the models that used a combination of features achieved higher AUC values than the models that used a single feature, indicating that the three latent relations are complementary to each other.

Overall, the ablation study results suggest that the MLFBK model's performance could be effectively improved by incorporating multi-features and multiple latent relations. The more features and/or latent relations embeddings were incorporated, the higher AUC scores could be achieved.

4.3 Activation Function Evaluation

To investigate the impact of activation functions on the performance of our MLFBK model, we conducted a study where we tested our model with three different activation functions: Leaky ReLU, Sigmoid, and Linear. Figure 6 shows the results of activation function evaluation. We trained and validated the models

Table 2. Ablation Study of MLFBK. The abbreviations used in there are as follows: *ap* for ability profile, *sm* for skill mastery and *pd* for problem difficulty. The best performance is denoted in bold.

Model	EdNet	assist09	algebra06	assist12
MLFBK*	0.7221	0.8002	0.7997	0.8065
MLFBK* + *ap*	0.7503	0.7922	0.8139	0.7713
MLFBK* +*sm*	0.7454	0.7891	0.7983	0.7611
MLFBK* +*pd*	0.8194	0.8411	0.8256	0.8304
MLFBK* +*ap* + *sm*	0.7429	0.8078	0.8201	0.7989
MLFBK* +*ap* + *pd*	0.8270	**0.8560**	0.8344	0.8287
MLFBK* +*sm* + *pd*	0.8233	0.8445	0.8362	0.8216
MLFBK* +*ap* + *sm* + *pd*	**0.8278**	0.8524	**0.8412**	**0.8350**

for 50 epochs with early stopping. Upon analysing the results, we found that all three activation functions produced similar results in terms of both training and validation behaviour. However, the Linear activation function performed slightly better than the other two. Specifically, it had the highest accuracy and AUC score on the validation set, which indicates that it may be the most suitable activation function for our MLFBK model. It is worth noting that the Leaky ReLU activation function stopped early during the training process, which may be due to its high learning rate. Overall, our findings suggest that the choice of an activation function has a relatively minor impact on the performance of our MLFBK model, but using the Linear activation function may lead to slightly better results.

4.4 Analysis of Embedding Strategy

We conducted a t-SNE analysis to visualise the entire embedding vector created in our MLFBK model. The results show the good interpretability of our methods' embedding strategies. Figure 4 shows the comparison of general embedding and MLFBK embedding strategies, utilising t-SNE as the visualisation method. Here, we take the embedding strategy of AKT as an example (on the left) and our MLFBK embedding strategy (on the right) on the assistments09 dataset. Each data point in the plot represents a learning interaction associated with a student, question, response, correctness, item, ability profile, skill mastery, and problem difficulty. The data points were coloured based on the ability profile value associated with them, specifically the transfer across skills value for the relevant student at the relevant time.

The left part of Fig. 4 shows the general embedding could not distinguish different features as all the features mixed together. In contrast, the right part of Fig. 4 shows that the MLFBK embedding strategy could distinguish different embedding with different colours well. The t-SNE plot shows that the students with small ability profile values at the current interaction were grouped together

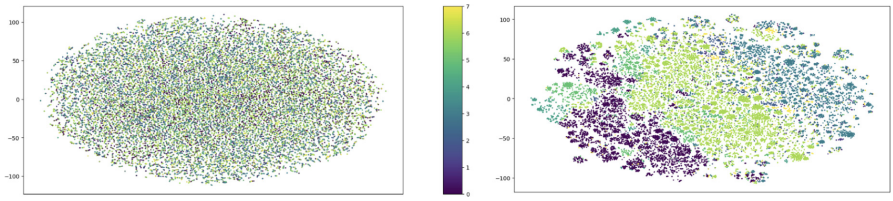

Fig. 4. The comparison of general embedding (Left) and MLFBK embedding (Right) strategies, utilising t-SNE as the visualisation method.

by the embedding, as were students with large values. This grouping could be used by the model to differentiate between interactions with correct responses and incorrect responses. The ability profile values provide additional information about students' performance, which could be useful for predicting their future performance. Overall, the t-SNE analysis demonstrated the effectiveness of the MLFBK model in capturing and utilising complex student interaction data.

Figure 5 shows the different embedding strategies of different single latent relations. The left is the embedding for the ability profile. It is the embedding without the additional features and then coloured according to the problem difficulty. It is easy to see that the problem difficulty feature is heavily considered in the feature embedding. The middle is embedding for problem difficulty. It only colours the learning interactions based on the problem difficulty of the relevant question. In this figure, the embedding doesn't seem to generate groupings, but more of a constant gradient, where the more difficult problems are in the top left and the easier problems are in the bottom right. The right image is the embedding for skill mastery. Here the skill mastery of the student on the relevant *item* is highlighted. This feature is multiplied by 100 and rounded to convert it to a categorical feature instead of a continuous one. The embedding also seems to be a gradient instead of groupings.

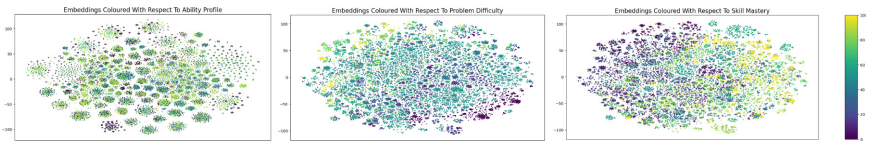

Fig. 5. Different Embedding Strategies.

4.5 Analysis of Estimating Problem Difficulty

We compared our model with MonaCoBERT regarding estimating problem difficulty for specific questions in the assistments2009 dataset. While MonaCoBERT uses a classical test theory (CTT) approach to estimate difficulties, our model calculates problem difficulty as a feature to use as input for the BERT model.

The comparison is visually represented in Fig. 7, with difficulty levels on the x-axis and the number of students answering correctly (green) or incorrectly (red) on the y-axis. Common sense dictates that harder questions should have more incorrect answers, although there may be exceptions. Surprisingly, the MonaCoBERT method showed that many students answered easy questions incorrectly but answered more difficult questions correctly, which seemed unlikely given the number of students evaluated. In contrast, our model revealed that as question difficulty increased, fewer students answered correctly, aligning with expectations. The results show that our method of estimating problem difficulty is far superior to the CTT difficulty estimation used by MonaCoBERT. Our method provides much more predictive value in estimating problem difficulty. This highlights the effectiveness of using our MLFBK model in predicting student performance in educational settings.

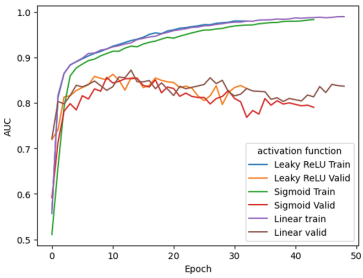

Fig. 6. Activation Function evaluation.

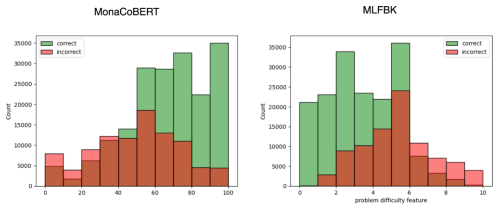

Fig. 7. Estimating Problem Difficulty.

5 Conclusion and Future Work

In this paper, we have proposed the MLFBK, which employs a BERT-based architecture incorporating multi-features and latent relations to improve the performance of Knowledge Tracing models. Experimental results show that MLFBK outperforms the five baseline models in every benchmark dataset on the metric of AUC. Moreover, we conducted an ablation study for different embedding strategies. The results indicate that combining different features and latent relations could improve performance effectively. Incorporating additional embeddings resulted in increased AUC scores. Moreover, we utilise the t-SNE as the visualisation tool to compare different embedding strategies. The results show that our method not only improves the performance of the models but also improves the model's interpretability. In future work, we plan to improve the architecture to process more comprehensive features and latent relations to satisfy the sustainable development requirement of Knowledge Tracing. Moreover, we will develop a model that could enhance memory efficiency as it handles growing amounts of multi-features and latent relations.

References

1. Choi, Y., et al.: EdNet: a large-scale hierarchical dataset in education. In: Bittencourt, I.I., Cukurova, M., Muldner, K., Luckin, R., Millán, E. (eds.) AIED 2020. LNCS (LNAI), vol. 12164, pp. 69–73. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-52240-7_13
2. Corbett, A.T., Anderson, J.R.: Knowledge tracing: modeling the acquisition of procedural knowledge. User Model. User-Adap. Inter. **4**(4), 253–278 (1994)
3. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
4. Donahue, C., Mao, H.H., Li, Y.E., Cottrell, G.W., McAuley, J.: Lakhnes: improving multi-instrumental music generation with cross-domain pre-training. arXiv preprint arXiv:1907.04868 (2019)
5. Drass, J.A., Muir-Nash, J., Boykin, P.C., Turek, J.M., Baker, K.L.: Perceived and actual level of knowledge of diabetes mellitus among nurses. Diabetes Care **12**(5), 351–356 (1989)
6. Floridi, L., Chiriatti, M.: Gpt-3: its nature, scope, limits, and consequences. Mind. Mach. **30**(4), 681–694 (2020)
7. Ghosh, A., Heffernan, N., Lan, A.S.: Context-aware attentive knowledge tracing. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 2330–2339 (2020)
8. He, L., Tang, J., Li, X., Wang, P., Chen, F., Wang, T.: Multi-type factors representation learning for deep learning-based knowledge tracing. World Wide Web **25**(3), 1343–1372 (2022)
9. Jiang, Z.H., Yu, W., Zhou, D., Chen, Y., Feng, J., Yan, S.: Convbert: improving BERT with span-based dynamic convolution. Adv. Neural. Inf. Process. Syst. **33**, 12837–12848 (2020)
10. Kalyan, K.S., Rajasekharan, A., Sangeetha, S.: Ammus: a survey of transformer-based pretrained models in natural language processing. arXiv preprint arXiv:2108.05542 (2021)
11. Krishnan, R., Singh, J., Sato, M., Zhang, Q., Ohkuma, T.: Incorporating wide context information for deep knowledge tracing using attentional bi-interaction. In: L2D@ WSDM, pp. 1–13 (2021)
12. Lee, U., Park, Y., Kim, Y., Choi, S., Kim, H.: Monacobert: monotonic attention based convbert for knowledge tracing. arXiv preprint arXiv:2208.12615 (2022)
13. Li, Z., Shi, L., Cristea, A., Zhou, Y., Xiao, C., Pan, Z.: SimStu-transformer: a transformer-based approach to simulating student behaviour. In: Rodrigo, M.M., Matsuda, N., Cristea, A.I., Dimitrova, V. (eds.) Artificial Intelligence in Education. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners' and Doctoral Consortium. AIED 2022. Lecture Notes in Computer Science, vol. 13356, pp. 348–351. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-11647-6_67
14. Li, Z., Shi, L., Cristea, A.I., Zhou, Y.: A survey of collaborative reinforcement learning: interactive methods and design patterns. In: Designing Interactive Systems Conference 2021, pp. 1579–1590 (2021)
15. Li, Z., Shi, L., Zhou, Y., Wang, J.: Towards student behaviour simulation: a decision transformer based approach. In: Frasson, C., Mylonas, P., Troussas, C. (eds.) Augmented Intelligence and Intelligent Tutoring Systems. ITS 2023. Lecture Notes in Computer Science, vol. 13891, pp. 553–562. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-32883-1_49

16. Lin, T., Wang, Y., Liu, X., Qiu, X.: A survey of transformers. AI Open (2022)
17. Liu, Q., et al.: Ekt: exercise-aware knowledge tracing for student performance prediction. IEEE Trans. Knowl. Data Eng. **33**(1), 100–115 (2019)
18. Liu, Q., Shen, S., Huang, Z., Chen, E., Zheng, Y.: A survey of knowledge tracing. arXiv preprint arXiv:2105.15106 (2021)
19. Lund, B.D., Wang, T.: Chatting about chatGPT: how may AI and GPT impact academia and libraries? Library Hi Tech News **40**, 26–29 (2023)
20. Minn, S., Vie, J.J., Takeuchi, K., Kashima, H., Zhu, F.: Interpretable knowledge tracing: simple and efficient student modeling with causal relations. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 12810–12818 (2022)
21. Pandey, S., Srivastava, J.: Rkt: relation-aware self-attention for knowledge tracing. In: Proceedings of the 29th ACM International Conference on Information and Knowledge Management, pp. 1205–1214 (2020)
22. Parmar, N., et al.: Image transformer. In: International Conference on Machine Learning, pp. 4055–4064. PMLR (2018)
23. Piech, C., et al.: Deep knowledge tracing. Adv. Neural Inf. Process. Syst. **28** (2015)
24. Shin, D., Shim, Y., Yu, H., Lee, S., Kim, B., Choi, Y.: Saint+: integrating temporal features for EdNet correctness prediction. In: LAK21: 11th International Learning Analytics and Knowledge Conference, pp. 490–496 (2021)
25. Sun, F., et al.: Bert4rec: sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1441–1450 (2019)
26. Tan, W., Jin, Y., Liu, M., Zhang, H.: BiDKT: deep knowledge tracing with BERT. In: Bao, W., Yuan, X., Gao, L., Luan, T.H., Choi, D.B.J. (eds.) ADHOCNETS/TridentCom -2021. LNICST, vol. 428, pp. 260–278. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-98005-4_19
27. Tiana, Z., Zhengc, G., Flanaganb, B., Mic, J., Ogatab, H.: Bekt: deep knowledge tracing with bidirectional encoder representations from transformers. In: Proceedings of the 29th International Conference on Computers in Education (2021)
28. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
29. Wang, T., Ma, F., Gao, J.: Deep hierarchical knowledge tracing. In: Proceedings of the 12th International Conference on Educational Data Mining (2019)
30. Wu, L., Li, S., Hsieh, C.J., Sharpnack, J.: SSE-PT: sequential recommendation via personalized transformer. In: Proceedings of the 14th ACM Conference on Recommender Systems, pp. 328–337 (2020)
31. Yudelson, M.V., Koedinger, K.R., Gordon, G.J.: Individualized Bayesian knowledge tracing models. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 171–180. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39112-5_18
32. Zhang, C., Jiang, Y., Zhang, W., Gu, C.: Muse: multi-scale temporal features evolution for knowledge tracing. arXiv preprint arXiv:2102.00228 (2021)