Tracing *Knowledge* Instead of *Patterns*: Stable Knowledge Tracing with Diagnostic Transformer

Yu Yin*
yxonic@mail.ustc.edu.cn
Anhui Key Lab. of Big Data Analysis and Application, University of Science and Technology of China & State Key Lab. of Cognitive Intelligence
Hefei, Anhui, China

Le Dai*
dl123@mail.ustc.edu.cn
Anhui Key Lab. of Big Data Analysis and Application, University of Science and Technology of China & State Key Lab. of Cognitive Intelligence
Hefei, Anhui, China

Zhenya Huang
huangzhy@ustc.edu.cn
Anhui Key Lab. of Big Data Analysis and Application, University of Science and Technology of China & State Key Lab. of Cognitive Intelligence
Hefei, Anhui, China

Shuanghong Shen
closer@mail.ustc.edu.cn
Anhui Key Lab. of Big Data Analysis and Application, University of Science and Technology of China & State Key Lab. of Cognitive Intelligence
Hefei, Anhui, China

Fei Wang
wf314159@mail.ustc.edu.cn
Anhui Key Lab. of Big Data Analysis and Application, University of Science and Technology of China & State Key Lab. of Cognitive Intelligence
Hefei, Anhui, China

Qi Liu
qiliuql@ustc.edu.cn
Anhui Key Lab. of Big Data Analysis and Application, University of Science and Technology of China & State Key Lab. of Cognitive Intelligence
Hefei, Anhui, China

Enhong Chen†
cheneh@ustc.edu.cn
Anhui Key Lab. of Big Data Analysis and Application, University of Science and Technology of China & State Key Lab. of Cognitive Intelligence
Hefei, Anhui, China

Xin Li
leexin@ustc.edu.cn
University of Science and Technology of China & iFLYTEK Co., Ltd.

ABSTRACT

Knowledge Tracing (KT) aims at tracing the evolution of the knowledge states along the learning process of a learner. It has become a crucial task for online learning systems to model the learning process of their users, and further provide their users a personalized learning guidance. However, recent developments in KT based on deep neural networks mostly focus on increasing the accuracy of predicting the next performance of students. We argue that current KT modeling, as well as training paradigm, can lead to models tracing patterns of learner's learning activities, instead of their evolving knowledge states. In this paper, we propose a new architecture, Diagnostic Transformer (DTransformer), along with a new training paradigm, to tackle this challenge. With DTransformer, we build the architecture from question-level to knowledge-level, explicitly diagnosing learner's knowledge proficiency from each question mastery states. We also propose a novel training algorithm based on contrastive learning that focuses on maintaining the stability of the knowledge state diagnosis. Through extensive experiments, we will show that with its understanding of knowledge state evolution, DTransformer achieves a better performance prediction accuracy and more stable knowledge state tracing results. We will also show that DTransformer is less sensitive to specific patterns with case study. We open-sourced our code and data at `https://github.com/yxonic/DTransformer`.

CCS CONCEPTS

• **Applied computing → Education**; • **Computing methodologies → Neural networks**.

KEYWORDS

knowledge tracing, contrastive learning, DTransformer

ACM Reference Format:
Yu Yin, Le Dai, Zhenya Huang, Shuanghong Shen, Fei Wang, Qi Liu, Enhong Chen, and Xin Li. 2023. Tracing *Knowledge* Instead of *Patterns*: Stable Knowledge Tracing with Diagnostic Transformer. In *Proceedings of the ACM Web Conference 2023 (WWW '23), April 30–May 04, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3543507.3583255

1 INTRODUCTION

With the advent of massive learning resources and online learning systems in the web, it becomes more and more popular for students and other general learners to choose web as a major media

[*] Both authors contributed equally to this research.
[†] The corresponding author.

WWW '23, April 30–May 04, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583255

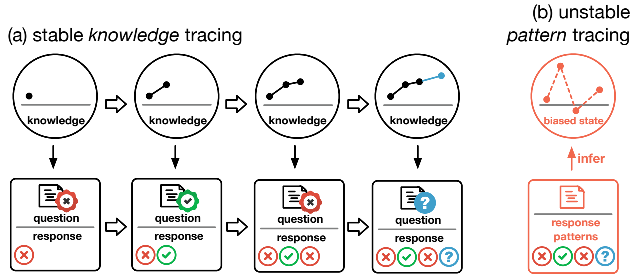

(a) stable *knowledge* tracing
(b) unstable *pattern* tracing
knowledge
question
response
biased state
infer
response patterns

Figure 1: An example of stable knowledge tracing. DTransformer traces knowledge evolution of a learner, then predicts responses on future questions (a), instead of predicting response patterns and then inferring knowledge states (b).

of learning [18, 29]. As online learning activity data begin to accumulate, understanding human learning process behind them plays an important role in online learning services [3, 17].

Generally speaking, human learning process involves two parts, *gaining knowledge* from previous experiences, and *applying knowledge* to solve future challenges [9]. During the process, each learner has an interior understanding, or technically speaking, a hidden knowledge state, that represents the acquisitions from their past experiences, and the evolution of the knowledge state represents the process of learning. In order to understand human learning process better, it is crucial to explicitly model and trace the evolution of the otherwise invisible knowledge states, which is exactly the focus of Knowledge Tracing [4, 22].

Researchers use the term Knowledge Tracing (KT) to refer to the task of tracing the knowledge state based on the learning activities of a learner [22]. For the importance of this problem, it has attracted much attention from both academia and industry in the past few years [15, 16, 31]. As the knowledge state cannot be directly learned or evaluated, almost all of the previous attempts model KT as the *next performance prediction* problem. Consequently, they either produce unstable tracing results that tend to overfit to memorize the exact sequence, or ignore student state tracing, only to gain prediction performance [8, 10, 30].

Through our investigation, we find that it is inherently difficult to trace states accurately in current KT paradigm. To understand this phenomena, we first ask a fundamental question: what does a model actually learn in their representation of student states? We can say that, with current KT models and the sequential prediction setup, most models are tracing **patterns** of the learners' activities, instead of **knowledge** that learners themselves acquire. Figure 1 gives an example of learning process under real scenario, where a learner answers questions sequentially and internalizes knowledge gradually along the process, as shown in Figure 1(a). With this example, we further elaborate our conclusion from two perspectives. First, current modeling of learning activity sequences fails to capture interior dynamics specific to the learning process, following or even reciting obvious patterns of the responses rather than the underlying knowledge states. Researchers have to introduce empirical constraints to match the psychological and educational background [30], while inferring stable and consistent knowledge states from the learning sequences remains challenging. Second, most of the recent models suffer from the *information bias* problem, i.e. the inferred knowledge state changes dramatically after the learner gets a different response [23]. This often makes their inference of knowledge states unstable, as demonstrated in Figure 1(b).

Trying to really trace *knowledge* instead of *patterns* in knowledge tracing, in this paper, we present a novel architecture, as well as a new paradigm, for more stable knowledge tracing. Specifically, to capture deeper learning dynamics along the learning process, we propose a new architecture for diagnosing knowledge states from sequential learning activities, the Diagnostic Transformer (**DTransformer**). Then, to alleviate information bias in the sequential prediction task, we design a new training algorithm based on contrastive learning that helps produce stable tracing results. In the end, we conduct experiments on 4 real-world datasets, showing that our proposed framework is able to trace knowledge state evolution more accurately. Furthermore, we present several visualization results that shows DTransformer can provide a more stable estimation of learner knowledge proficiency. We summarize the key innovations of our work as follows:

- We design a new architecture, DTransformer, to extract hidden knowledge state from sequential learning activities. As far as we know, this is the first Transformer-based architecture that focuses on stable knowledge state estimation and tracing, instead of solely next performance prediction.
- Inside DTransformer, we design a Temporal and Cumulative Attention (TCA), and build the architecture from question-level to knowledge-level. We specifically design a knowledge diagnosis extractor that explicitly diagnoses learner's knowledge proficiency from their question mastery states.
- We propose a novel training algorithm for KT that helps with stable tracing. Based on contrastive learning, our algorithm is able to maintain the stability of knowledge states that are diagnosed from similar learning histories, alleviating the information bias problem shared by most previous works.

2 RELATED WORK

2.1 Knowledge Tracing

Knowledge tracing is an important topic in all kinds of learning setups, including online learning [4]. This task is traditionally tackled from the perspective of cognitive science and psychology [4, 5] or probability [33]. Recently, more researchers incorporate deep learning into knowledge tracing, for its expressiveness and performance boost. Most recent work can be divided into two categories: recurrent neural network (RNN) based and attention (or Transformer) based. Both approaches view learning activities (exercising records) of a learner as a sequence, and focus on the task of the next performance prediction along the sequence [14, 22, 34].

Though effective, these methods experience inconsistencies with the actual learning process. The states these model are tracing often reflect the *patterns* of the learning activities of learners, instead of their evolving knowledge states. Works such as DKT+ [30] and LPKT [23] try to include constraints based on observations in human cognitive process to alleviate this problem. Nevertheless, it is still challenging to trace internal knowledge states from the next performance prediction task only.

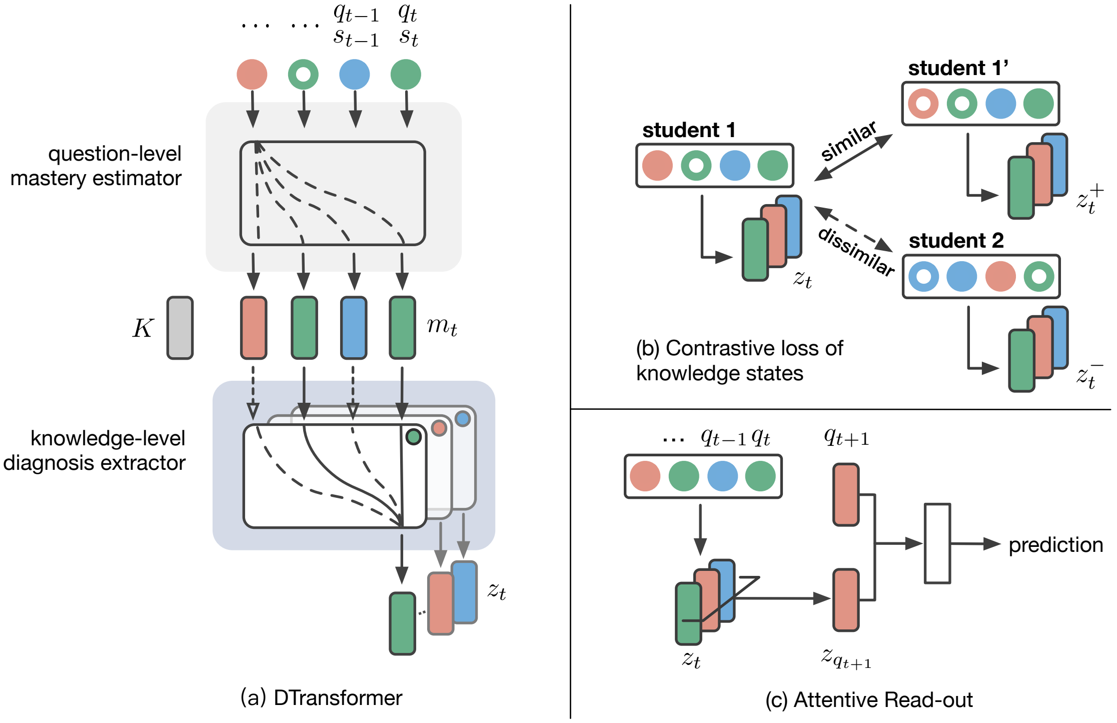


Figure 2: The overall architecture of DTransformer. Left shows the DTransformer model architecture from question-level to knowledge-level, based on Temporal and Cumulative Attention (TCA) mechanism. Right shows the contrastive loss of our proposed algorithm that helps maintain tracing stability, as well as the prediction loss for the task.

Inspired by Transformer [28] from natural language processing field, some recent works such as SAKT [19] and AKT [8] aim at using attention mechanism to model learner sequences more effectively. However, these attention based models mainly focus on sequential performance prediction accuracy, but ignores the need of tracing the evolution of knowledge states. In fact, through attention mechanism, these models mostly predict student performance according to learner's previous performance on relevant questions, instead of their current knowledge state.

2.2 Contrastive Learning

Contrastive learning (CL) is a technique that helps models learn high-level intrinsics in the final representations without relying on extreme details. We use this technique in this paper to assist our model with stability and consistency in knowledge tracing, without relying much on particular previous experiences and learning activity patterns. Previously, most work in contrastive learning focuses on representation learning in computer vision[2, 27]. Recently, representation learning in the fields such as graph embedding and sentence representation also incorporates contrastive learning in their workflows [7, 32]. A few recent works also try to incorporate contrastive learning in knowledge tracing. Song et al. [24] model the KT problem as bipartite graph to introduce graph contrastive learning techniques into KT, with the assistance of a external exercise-exercise graph. Lee et al. [13] propose CL4KT based on the AKT [8] architecture, to reveal semantically similar or dissimilar relations of the learning history. However, these works are certainly at an early stage, not explicitly taking the stability and consistency of the internal knowledge states into consideration.

3 METHODOLOGY

3.1 Problem Definition

In online learning systems, each learner's learning activities consist mostly of exercising records, i.e. a sequence of questions and the corresponding responses. For learner i at time step t, they will answer a question $q_t^i \in \mathbb{Q}$ drawn from a knowledge concept $c_t^i \in \mathbb{C}$, and get response $r_t^i \in \{0,1\}$, denoting whether the learner correctly answer the question ($r_t^i = 1$) or not ($r_t^i = 0$). Thus, for each learner, we have their exercising records as a sequence:

$$\{(q_1, c_1, r_1), \ldots, (q_T, c_T, r_T)\}, \quad q_t \in \mathbb{Q}, c_t \in \mathbb{C}, r_t \in \{0,1\}, \quad (1)$$

where T is the length of the learning sequence, $\mathbb{Q}$ is the set of all questions, $\mathbb{C}$ is the set of all knowledge concepts.

The learning sequence reflects the evolving knowledge proficiency of a learner, but the exact internal knowledge states of learners remain implicit. Therefore, aiming at tracing the evolving knowledge states at each time step t according to previous exercising records, we define our problem of Knowledge Tracing (KT) formally as follows:

Definition 3.1. **(Knowledge Tracing Problem)** Given the previous exercising records of a learner before time step t as a sequence $\{(q_1, c_1, r_1), \ldots, (q_t, c_t, r_t)\}$, the goal is to: (1) trace the internal knowledge state z_t at time step t of this learner; (2) predict the response $\hat{r}_{t+1}$ of them on the next question q_{t+1}.

In contrast with many of the previous works, our focus here is not only predicting future performance of each learner, but also keeping track of the evolution of their internal knowledge states. Specifically, one of our goals is to learn how well a learner master each knowledge concept along the whole learning process.

3.2 DTransformer Architecture

The overall architecture of DTransformer is shown in Figure 2. From a higher perspective, DTransformer architecture consists of two parts. The question-level mastery estimator takes the question q_t and score s_t sequences as input, and generates context-aware question mastery vectors m_t. The knowledge-level diagnosis extractor on the other hand, diagnose knowledge-level proficiency z_t at each time step t from all previous question-level mastery estimations. To break strong pattern dependencies experienced by most previous models, both modules are based on multi-head attention mechanism with temporal effects that align with learning behaviors.

3.2.1 Temporal and Cumulative Attention (TCA). The attention mechanism is the building block of DTransformer, for it breaks the strong sequential dependency between adjacent inputs mostly seen in RNN-based models. It helps the overall model to be less pattern-sensitive, focusing on learning the internal state transition. Inspired by the Monotonic Multi-head Attention first introduced in AKT [8], we design a Temporal and Cumulative Attention (TCA) mechanism, with explicit considerations of: (1) the temporal effect in the learning process, and (2) the cumulative effort. This better fits the learning activity sequence than the plain multi-head attention mechanism in the original Transformer [28], which empowers the knowledge state estimation of DTransformer later on.

Considering temporal effect in the learning process in the attention mechanism, following [8], we have:

$$\text{AttentionT}(Q, K, V) = \text{softmax}\left(\frac{QK^T \cdot e^{-\theta \cdot d(\Delta t)}}{\sqrt{d_k}}\right)V, \quad (2)$$

where θ is the parameter that controls how strong the temporal effect should add up to the final weights, and $d(\Delta t)$ is the distance function that measures how far away a query and a key are, sequentially and semantically (following Ghosh et al. [8]). The temporal effect decays exponentially as the distance go further, following the intuition that learning experiences long ago have less impact on current learning states.

While attention with temporal effect is able to aggregate relevant experiences, we argue that weighted sum of all these experiences ignores the cumulative effort in learning process. A learner can trial on a single concept or question for several times until they master it, but in current weighted aggregation, these repeated efforts are in no difference with a single experience. Therefore, we extend the current attention scores with a max-out operation, modeling cumulative learning gain in past experiences. Specifically, we have our new attention calculation:

$$\text{AttentionMax}(Q, K, V) = \text{MaxOut}\left(\frac{QK^T \cdot e^{-\theta \cdot d(\Delta t)}}{\sqrt{d_k}}\right)V, \quad (3)$$

$$\text{MaxOut}(x_i) = \frac{\text{softmax}(x_i)}{\max_j\{\text{softmax}(x_j)\}}. \quad (4)$$

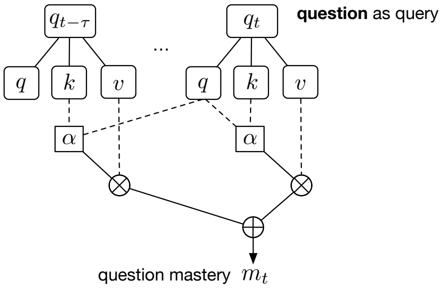


Figure 3: Question-level mastery estimator.

MaxOut scales the largest softmax score up to 1, and retains the proportion of all other scores. This allows the attention to scale the results when aggregating with regard to relevant sequence length.

Finally, we expand the attention into multiple heads to consider multiple aspects when aggregating and make the final result more robust. In multi-head attention, Q, K and V are linear transformed and then split into H heads:

$$\text{MultiHead}(Q, K, V) = \text{concat}(\text{head}_1, \ldots, \text{head}_H)W^O \quad (5)$$

$$\text{where head}_i = \text{AttentionMax}(QW_i^Q, KW_i^K, VW_i^V). \quad (6)$$

W^Q, W^K, W^V refers to the linear transform parameters. In the full DTransformer layer, we first conduct multi-head attention, then include residual connection [26] as well as layer normalization [1]. Formally, we define the DTransformer layer that we use across the whole architecture as:

$$\text{DTransformerLayer}(Q, K, V) = \text{LayerNorm}(\text{MultiHead}(Q, K, V)) + Q. \quad (7)$$

3.2.2 Question-level mastery. With TCA mechanism that we defined, we are able to estimate learner's internal knowledge states from past learning experiences. We will ask two questions along the process: (1) How well does the learner *actually* master each question they exercised? (2) How proficient is the learner with all the knowledge concepts at this step? This leads to a two-step solution of DTransformer, which estimate question-level mastery first, then diagnose knowledge-level proficiency.

To estimate learner's question-level mastery, we first obtain a more accurate understanding of how well the learner does on each question. Learner can correctly or incorrectly answer a question, but their mastery of this question is not determined by their response. To accurately estimate how well a learner master a question, we need to incorporate more context relevant to this question. We first incorporate Rasch embedding [8] to get question embedding sequence $\boldsymbol{q}$ and learning activity embedding sequence $\boldsymbol{a}$ from the original question and score sequence (see Appendix A.1). Then, we incorporate TCA that we described in the previous section to obtain a context-aware question mastery estimation. Within each head, we aggregate learner's performance on relevant questions for each question q_t at time step t as:

$$\alpha_{t,\tau} = \text{softmax}\left(\frac{Q_t^T K_\tau}{\sqrt{d_k}} \cdot e^{-\theta \cdot d(t-\tau)}\right), \tag{8}$$

$$= \frac{e^{Q_t^T K_\tau / \sqrt{d_k}} \cdot e^{-\theta \cdot d(t-\tau)}}{\sum_{i=1}^{t} e^{Q_t^T K_i / \sqrt{d_k}} \cdot e^{-\theta \cdot d(t-i)}}, \tag{9}$$

$$m_t = \sum_{\tau=1}^{t} \alpha_{t,\tau} \cdot V_\tau, \tag{10}$$

$$\text{where } Q_t = \boldsymbol{q}_t \cdot W^Q,\ K_t = \boldsymbol{q}_t \cdot W^K,\ V_t = \boldsymbol{a}_t \cdot W^V. \tag{11}$$

Finally, we combine each head m_t together following Eq (5), and apply residual connection and layer normalization afterwards. Together we obtain a question-level mastery vector $\boldsymbol{m}_t$ for each question q_t that takes previous experiences into consideration.

3.2.3 Knowledge-level diagnosis. After we obtained a more accurate question-level mastery for each learner, next we diagnose knowledge-level proficiency from them. To diagnose at knowledge-level, the intuition is to use *knowledge* instead of question as attention query. Knowledge queries should be used consistently through the whole sequence, so that we trace knowledge state transition of the learner themselves, instead of their mastery of each individual questions. To this end, we randomly initialize a set of knowledge parameters $K = \{K_1, \ldots, K_N\}$ in DTransformer, and use them as the query in the knowledge-level extractor. We use N knowledge parameters to represent N knowledge concepts that learners internalize during the learning process. Formally, for each knowledge concept n at time step t, the knowledge-level proficiency $z_{t,n}$ within each head is calculated as:

$$\alpha_{t,n,\tau} = \text{softmax}\left(\frac{Q_n^T K_\tau}{\sqrt{d_k}} \cdot e^{-\theta \cdot d(t-\tau)}\right), \tag{12}$$

$$= \frac{e^{Q_n^T K_\tau / \sqrt{d_k}} \cdot e^{-\theta \cdot d(t-\tau)}}{\sum_{i=1}^{t} e^{Q_n^T K_i / \sqrt{d_k}} \cdot e^{-\theta \cdot d(t-i)}}, \tag{13}$$

$$z_{t,n} = \sum_{\tau=1}^{t} \alpha_{t,n,\tau} \cdot V_\tau, \tag{14}$$

$$\text{where } Q_n = K_n \cdot W^Q,\ K_t = \boldsymbol{q}_t \cdot W^K,\ V_t = \boldsymbol{m}_t \cdot W^V. \tag{15}$$

At last, we also combine each head $z_{t,n}$ together as proficiency estimation $z_{t,n}$ for knowledge concept n following Eq (5). All the proficiency estimations of the N knowledge concepts then represent the internal knowledge state at time step t of a learner:

$$\boldsymbol{z}_t = \{z_{t,n} : n = 1, \ldots, N\}. \tag{16}$$

3.3 Model Training

In previous works, the training process only focuses on the task of next performance prediction, i.e. predicting the probability for a learner to answer the next question correctly. This leads to two consequences: (1) models tend to learn patterns of learner's answering sequence, instead of the internal knowledge transition. (2) learner's knowledge state often experience huge leaps after the model get new information of their performance, which is the information bias problem. Both of the problems cannot be tackled properly under the current training framework. Therefore, in the following subsections, we introduce a new training pipeline for more stable knowledge tracing.

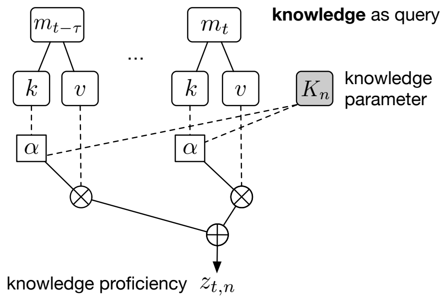


Figure 4: Knowledge-level diagnosis extractor.

Our training process consist of two main components, as shown on the right of Figure 2. First, we maintain a consistent diagnosis of a learner's knowledge state, through a contrastive loss function. Then, we trace learner's next performance directly from their knowledge state evolution. Further details are described below.

3.3.1 Contrastive loss of knowledge states. The first part of the training process is contrastive loss of knowledge states. Our goal is to maintain a consistent knowledge state for each learner that reflects their internal knowledge proficiency, and be less sensitive to individual responses they get. The intuition here is, if a learner had a different response on one particular question throughout the entire learning history, the learner's final knowledge state shouldn't see a huge difference. This inspire us to design a contrastive loss for knowledge states, to maintain the consistency and stability of knowledge states.

Following previous works on contrastive learning [2, 7], we first augment the learning activity sequence of a learner. Then we proceed to use DTransformer to extract knowledge states from both sequences. Finally we incorporate a contrastive loss to draw positive pairs of knowledge states (those from the sequences of the same learner) together, pushing away negative pairs. To reflect the intuition we mentioned earlier, we design three augmentation strategies. Specifically:

Flip Response: When a learner answers a question, they may have a probability of getting right answers wrong (slipping) or getting right answers accidentally (guessing). However, these occasional glitches of learner's response should not affect the internal knowledge states. To reflect this intuition, we design the Flip Response augmentation that randomly flips learner's responses in the learning sequence. Learner's states should remain similar after this augmentation, i.e. after they slip or guess some of the questions.

Drop Item: In this augmentation, we randomly drop questions in learner's sequence. This align with the intuition that learner's state is not affected much by only one question, but depends more on the whole learning process.

Swap Adjacent Items: In the learning process of a learner, the order of questions matters, but knowledge states should not be dependent on small order perturbations. Thus, instead of doing a full permutation as done by Lee et al. [13], we randomly swap adjacent question-response pairs to perform localized augmentation on order. Our model should learn similar knowledge states after a few swapping in the sequence.

We denote the input sequences after these augmentations as q^+ and a^+. Then we obtain the knowledge state of the augmented sequence as z_t^+. Following previous work [2], we calculate contrastive loss from a single batch. Specifically, for each knowledge state $z_{i,t}$ of the learner i at time step t, we construct positive and negative pairs within the current batch learner i is in. The contrastive loss for learner i at time step t is denoted as:

$$l_{t,i}^{CL} = -\log \frac{e^{\text{sim}(z_t^{(i)}, z_t^{+(i)})/\tau}}{\sum_{j \neq i} e^{\text{sim}(z_t^{(i)}, z_t^{+(j)})/\tau}}, \tag{17}$$

where τ is a temperature hyperparameter (set as 0.05 in our case), and $\text{sim}(z_i, z_j)$ is the cosine similarity function $\frac{z_i^T z_j}{\|z_i\| \cdot \|z_j\|}$.

3.3.2 Next performance prediction. To trace the evolution of the knowledge state, we do next performance prediction with current knowledge state of a learner. Different from previous work [8], here we only use knowledge state z_t from current time step t. The knowledge state z_t we obtain at each time step t consist of N knowledge concepts. To predict how well a learner answers the next question q_{t+1}, we should only focus on learner's proficiency levels on relevant knowledge concepts. To this end, we propose a new attentive read-out module for next performance prediction using knowledge state z_t. Specifically:

$$\alpha_i = \text{softmax}(q_{t+1}^T z_{i,t}) = \frac{e^{q_{t+1}^T z_{i,t}}}{\sum_{j=1}^{N} e^{q_{t+1}^T z_{j,t}}}, \tag{18}$$

$$z_{q_{t+1}} = \sum_{i=1}^{N} \alpha_i z_{i,t}, \tag{19}$$

$$\hat{r}_{t+1} = \sigma\left(\text{concat}(z_{q_{t+1}}, q_{t+1}) \cdot W^A\right), \tag{20}$$

where σ is the Sigmoid function.

Therefore, we have the prediction loss on each time step t as cross entropy loss:

$$l_t = -r_{t+1} \log \hat{r}_{t+1} - (1 - r_{t+1}) \log(1 - \hat{r}_{t+1}). \tag{21}$$

In the end, the overall objective function of DTransformer is defined as a linear combination of the two loss functions:

$$\mathcal{L}_t = l_t + \lambda \cdot l_t^{CL}, \tag{22}$$

where λ controls the influence of contrastive signals. We can then train the whole model using gradient descent based optimization algorithms, with regard of the objective function Eq (22). In our setup, we use Adam [11] to optimize the objective function, and set $\lambda = 0.1$ for an optimal performance.

4 EXPERIMENTS

In this section, we first describe the real-world datasets and experimental setup that we use across all the experiments. Then we demonstrate our results and observations aiming at answering the following research questions:

- **RQ1:** Does DTransformer outperform state-of-the-art knowledge tracing models on next performance prediction task?
- **RQ2:** Is DTransformer able to maintain the stability and consistency of knowledge states while tracing each learner?
- **RQ3:** How much does knowledge diagnostic module and contrastive learning contribute to the knowledge tracing performance of DTransformer?
- **RQ4:** How does DTransformer help tracing the evolution of knowledge states?

Statistics	assist09	assist17	algebra05	statics
Learners	4,217	1,079	574	331
Questions	26,688	3,162	1,084	-
Concepts	123	102	138	154
Responses	346,860	942,816	809,694	142,124
Avg. Length	82.25	873.79	1,410.62	429.38

Table 1: Statistics of 4 real-world datasets in this paper.

4.1 Datasets

We use 4 real-world datasets with diversity to evaluate the effectiveness of DTransformer in different learning scenarios. Table 1 shows the statistics of all the datasets. We introduce and compare each dataset as follows:

- **assist09**[1] refers to a dataset collected from ASSISTment, an online tutoring system created in 2004. We use the updated *skill-build* version of this dataset as it fixes data modeling issues and removes duplicated records [6].
- **assist17**[2] is another dataset collected from ASSISTment, but learners have much longer learning sequences. In this dataset, learners can trial many times on a single question until they get it right. Thus, it is essential for models to consider cumulative effect in the learning process [21].
- **algebra05**[3] was presented at the KDDcup 2010 Educational Data Mining challenge and contains student responses to Algebra questions from 2005 to 2006 [25].
- **statics**[4] dataset contains student interactions with questions related to the Engineering Statics course taught at the Carnegie Mellon University during Fall 2011. Compared with others, this dataset shows examples of higher education learning process [12].

4.2 Experimental Setup

To evaluate our model, we compare our model with state-of-the-art knowledge tracing models on all the datasets we mentioned earlier. The details of all the baseline methods that we use are described in Appendix A.2. For each dataset, we split 80% students as training set, and 20% students as test set. All hyper-parameters are tuned based on a standard 5-fold cross validation for a fair comparison.

We implemented our model, as well as most of the deep learning based baseline models, under a unified framework developed using PyTorch [20]. However, we did not implement CL4KT under our framework as it has a different data processing mechanism, and use results reported by the original paper. All models are trained on a cluster of Linux servers with NVIDIA TITAN V100 GPUs. All models are tuned to their best performance for a fair comparison.

[1] https://sites.google.com/site/assistmentsdata/home/2009-2010-assistment-data
[2] https://sites.google.com/view/assistmentsdatamining
[3] https://pslcdatashop.web.cmu.edu/KDDCup
[4] https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=507

Dataset	Metrics	DKT	DKT+	DKVMN	SAKT	AKT	CL4KT	DTransformer	– (w/o CL)
assist09	MAE	0.4105	0.3581	0.3132	0.3342	0.3459	-	0.3097	**0.3033**
	RMSE	0.4387	0.4172	0.4161	0.4194	0.4172	0.4333	**0.4007**	0.4062
	ACC	0.6482	0.7438	0.7458	0.7418	0.7418	-	**0.7632**	0.7579
	AUC	0.7908	0.7853	0.7891	0.7783	0.7868	0.7624	**0.8146**	0.8056
assist17	MAE	0.4143	0.4385	0.4064	0.4207	0.3867	-	**0.3760**	0.3831
	RMSE	0.4543	0.4655	0.4543	0.4609	0.4385	-	**0.4371**	0.4404
	ACC	0.6971	0.6633	0.6775	0.6735	0.7071	-	**0.7078**	0.7010
	AUC	0.6809	0.6540	0.7024	0.6726	0.7464	-	**0.7506**	0.7451
algebra05	MAE	0.3498	0.2774	0.2315	0.2304	0.2319	-	**0.2201**	0.2302
	RMSE	0.3798	0.3630	0.3424	0.3406	0.3454	0.3815	**0.3403**	0.3423
	ACC	0.8263	0.8337	0.8315	0.8337	0.8361	-	**0.8424**	0.8392
	AUC	0.7558	0.7138	0.7857	0.7819	0.7750	0.7891	**0.7946**	0.7878
statics	MAE	0.3828	0.3215	0.2635	**0.2557**	0.3017	-	0.2612	0.2797
	RMSE	0.4087	0.3956	0.3801	0.3786	0.3908	0.3945	**0.3621**	0.3632
	ACC	0.7355	0.7727	0.7910	0.7887	0.7757	-	**0.7962**	0.7948
	AUC	0.8182	0.7807	0.8265	0.8299	0.7938	0.7943	**0.8382**	0.8341

Table 2: The overall performance comparison on 4 real-world datasets. We compare our full model (DTransformer), as well as our model without contrastive loss (w/o CL), with state-of-the-art KT methods. We can see that DTransformer has the best performance on almost all metrics on 4 datasets. Without contrastive loss, it still outperforms other models on several datasets.

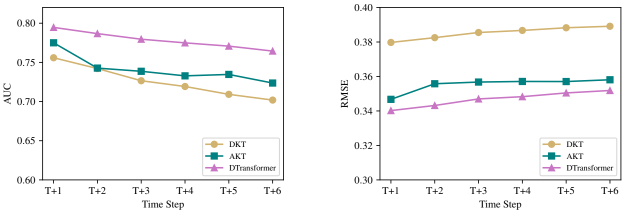


Figure 5: T+N prediction performance on algebra05.

4.3 Overall Performance (RQ1)

The main goal of KT is to trace the evolution of the internal knowledge states. However, it is not possible to directly measure how accurate the state estimation is. To evaluate the effectiveness of KT models, we will compare all the methods on the next performance prediction task. As shown in Table 2, we compare our model with all the baseline methods on all 4 datasets, under the metrics of Mean Absolute Error (MAE), Rooted Mean Squared Error (RMSE), Accuracy (ACC) and Area Under Curve (AUC). There are several important observations from the table. First, DTransformer with proper contrastive setup has the best performance under most metrics, while DTransformer without contrastive loss outperforms outperforms all baseline models on 3 of the datasets. For most of the datasets, the effectiveness of knowledge states estimated by DTransformer is strong enough to produce more accurate prediction. As for assist17 dataset where AKT shines, our model still outperforms it with the help of the contrastive loss. Second, compared with results reported by CL4KT [13], our contrastive loss boost the performance of the original model even more. Both CL4KT and ours incorporates attention mechanism, and the result proves our modeling to be more effective.

N	1	2	4	8	16	32
AUC	0.7759	0.7812	0.7842	0.7922	**0.7946**	0.7943

Table 3: Prediction AUC on algebra05 with different N.

λ	0.0001	0.001	0.01	0.1
AUC	0.7906	**0.7946**	0.7854	0.7821

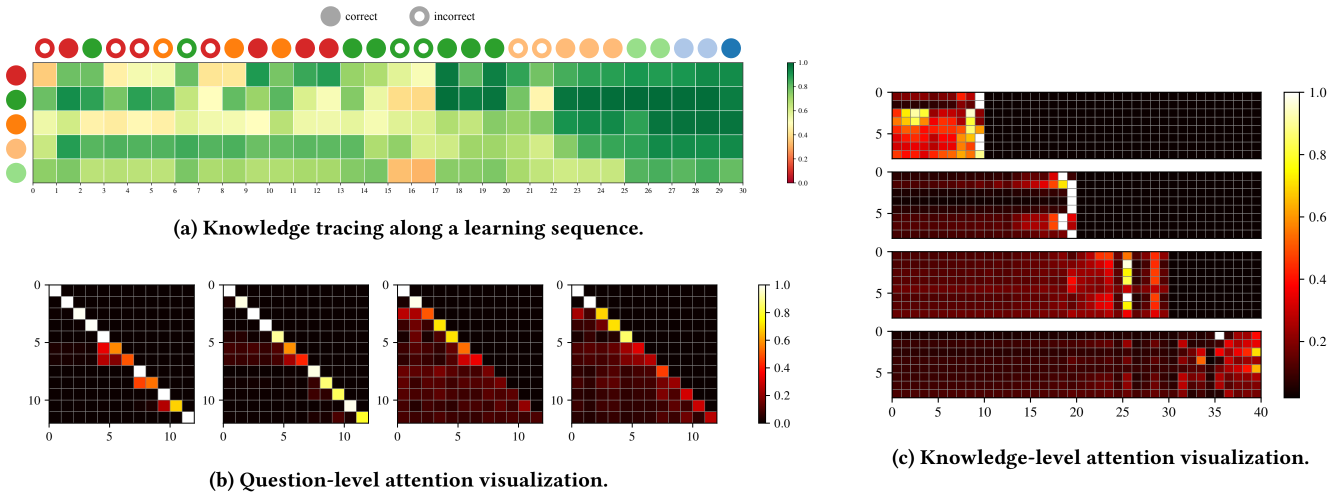
Table 4: Prediction AUC on algebra05 with different λ.

4.4 Stability of Knowledge Tracing (RQ2)

One of the main contributions of our model is being able to produce stable knowledge state tracing. We demonstrate this by conducting a novel $T+N$ prediction experiment. Specifically, at each evaluation step t, instead of predicting only next performance, we predict learner's response for questions at step $t+1, t+2, \ldots$, and evaluate how fast each method regress as duration gets longer. The result is shown in Figure 5. We can see that the prediction performance of DTransformer does not decline too quickly compared with DKT and AKT. This indicates that DTransformer maintains a relatively more stable knowledge state. Details of how this experiment is conducted are described in Appendix A.3.

4.5 Parameter Sensitivity (RQ3)

To demonstrate the effectiveness of the knowledge diagnostic module as well as the contrastive loss, we train our model with different setups and compare the outcome. We set the number of knowledge parameters N from 1 to 32, and show the prediction AUC of each setup on algebra05 dataset in Table 3. We can see that number of



(a) Knowledge tracing along a learning sequence.

(b) Question-level attention visualization.

(c) Knowledge-level attention visualization.

Figure 6: Visualization of DTransformer tracing knowledge of a learner along their learning sequence. We can see that the knowledge states traced by DTransformer is stable and not sensitive to response changes or patterns. In the attention visualization heat maps, we can also see where the tracing results come from and how DTransformer makes use of them.

knowledge parameters play an important role in extracting accurate diagnosis. With more knowledge parameters, the performance generally goes higher. The performance on $N = 16$ and $N = 32$ are similar, indicating model overfitting for larger N. We also set the λ hyperparameter for contrastive loss term in Eq (22) as different values and show the performance in Table 4. We can see that contrastive constraints that are either too large or too small, has a noticeable impact on the final performance. According to these results, we set $N = 16$ and $\lambda = 0.001$ across all the experiments.

4.6 Visualizations (RQ4)

To further understand how DTransformer traces the evolution of knowledge states, in this section, we demonstrate the tracing process with several visualizations of a learning sequence.

The most interesting and useful aspect of knowledge tracing may be to predict how much a learner masters each knowledge concept at any time step along a learning sequence. With DTransformer, we can use the explicitly extracted knowledge state z_t to estimate the mastery levels of each knowledge concepts. Specifically, we use embeddings of each knowledge concept as query and go through the attentive read-out module. The predicted probability should reflect the mastery level of this concept. More details about this are described in Appendix A.4. Following this scheme, we sample a real learning sequence and show our estimation of evolution of mastery levels on each knowledge concept in Figure 6a with DTransformer. Each row shows the estimation of the knowledge state evolution of a single concept along the learning sequence. The circles on top and left of the figure shows which knowledge concept this row or column is related to. Circles with white dot in it indicates that the learner answers this question wrongly. From the visualization, we can see that our model generates stable and consistent knowledge states, with considerations of temporal decay and cumulative effort. We can see the learner gains acquisition of a knowledge concept along with their exercising, even if the learner answers a question wrongly at first. This aligns with the intuition that human can learn from their mistakes. We can also see the learner forgets what they learned before as time goes by.

We also demonstrate the attention weights of both question-level and knowledge-level along the tracing process in Figure 6b and 6c. On question level, each position (i, j) shows the attention weight of question q_j when DTransformer estimates question mastery of question q_i. We list a few heads, showing each attention head either focus on the question q_i itself, or checks on recent similar questions. On knowledge level, we show attention weights of each knowledge parameter to each of the previous records, across the entire learning sequence ($t = 10, 20, 30, 40$) in Figure 6c. At each time step, this figure shows where the knowledge proficiency diagnosis is extracted from for each individual knowledge concept. We can see that our model tries to estimate question mastery more accurately by looking back to the sequence, and generates knowledge states based on relevant records across whole sequence of learning. We can also see the effect of Temporal and Cumulative Attention mechanism. On each figure, we can see the attention weights decay over time, matching the temporal effect in the learning process. It also reflects cumulative effort of a learner by accumulating weights throughout the whole sequence at knowledge-level.

5 CONCLUSION

In this paper, we discussed the instability of KT caused by tracing patterns instead of knowledge in current KT paradigm. We tackled this challenge with a novel architecture, DTransformer, along with a new training paradigm. Specifically, we designed a two-level framework to explicitly diagnose learner's knowledge states, and increased stability of knowledge state diagnosis by a new training algorithm based on contrastive learning. We hope our work has a positive impact on KT research as well as more general research areas such as user modeling and understanding.

ACKNOWLEDGMENTS

This research was supported by grants from the National Key Research and Development Program of China (No. 2021YFF0901003).

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer Normalization. *stat* 1050 (2016), 21.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[3] Konstantina Chrysafiadi and Maria Virvou. 2013. Student modeling approaches: A literature review for the last decade. *Expert Systems with Applications* 40, 11 (2013), 4715–4729.

[4] Albert T Corbett and John R Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction* 4, 4 (1994), 253–278.

[5] Susan E Embretson and Steven P Reise. 2013. *Item response theory*. Psychology Press.

[6] Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User modeling and user-adapted interaction* 19, 3 (2009), 243–266.

[7] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821* (2021).

[8] Aritra Ghosh, Neil Heffernan, and Andrew S Lan. 2020. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 2330–2339.

[9] Knud Illeris. 2018. A comprehensive understanding of human learning. In *Contemporary theories of learning*. Routledge, 1–14.

[10] Mohammad Khajah, Robert V Lindsey, and Michael C Mozer. 2016. How Deep is Knowledge Tracing?. *International Educational Data Mining Society* (2016).

[11] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[12] Kenneth R Koedinger, Ryan SJd Baker, Kyle Cunningham, Alida Skogsholm, Brett Leber, and John Stamper. 2010. A data repository for the EDM community: The PSLC DataShop. *Handbook of educational data mining* 43 (2010), 43–56.

[13] Wonsung Lee, Jaeyoon Chun, Youngmin Lee, Kyoungsoo Park, and Sungrae Park. 2022. Contrastive Learning for Knowledge Tracing. In *Proceedings of the ACM Web Conference 2022*. 2330–2338.

[14] Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, and Guoping Hu. 2019. EKT: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering* 33, 1 (2019), 100–115.

[15] Qi Liu, Shuanghong Shen, Zhenya Huang, Enhong Chen, and Yonghe Zheng. 2021. A survey of knowledge tracing. *arXiv preprint arXiv:2105.15106* (2021).

[16] Sein Minn, Yi Yu, Michel C Desmarais, Feida Zhu, and Jill-Jenn Vie. 2018. Deep knowledge tracing and dynamic student classification for knowledge tracing. In *2018 IEEE International conference on data mining (ICDM)*. IEEE, 1182–1187.

[17] Hossein Mohammadi. 2015. Investigating users' perspectives on e-learning: An integration of TAM and IS success model. *Computers in human behavior* 45 (2015), 359–374.

[18] Tuan Nguyen. 2015. The effectiveness of online learning: Beyond no significant difference and future horizons. *MERLOT Journal of Online Learning and Teaching* 11, 2 (2015), 309–319.

[19] Shalini Pandey and George Karypis. 2019. A self-attentive model for knowledge tracing. In *12th International Conference on Educational Data Mining, EDM 2019*. International Educational Data Mining Society, 384–389.

[20] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.

[21] Thanaporn Patikorn, Neil T Heffernan, and Ryan S Baker. 2018. Assistments longitudinal data mining competition 2017: A preface. In *Proceedings of the Workshop on Scientific Findings from the ASSISTments Longitudinal Data Competition, International Conference on Educational Data Mining*.

[22] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. *Advances in neural information processing systems* 28 (2015).

[23] Shuanghong Shen, Qi Liu, Enhong Chen, Zhenya Huang, Wei Huang, Yu Yin, Yu Su, and Shijin Wang. 2021. Learning process-consistent knowledge tracing. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1452–1460.

[24] Xiangyu Song, Jianxin Li, Qi Lei, Wei Zhao, Yunliang Chen, and Ajmal Mian. 2022. Bi-CLKT: Bi-graph contrastive learning based knowledge tracing. *Knowledge-Based Systems* 241 (2022), 108274.

[25] J Stamper, A Niculescu-Mizil, S Ritter, G Gordon, and K Koedinger. 2010. Algebra I 2005-2006 and Bridge to Algebra 2006-2007. Development data sets from KDD Cup 2010 Educational Data Mining Challenge.

[26] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*.

[27] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. 2020. What makes for good views for contrastive learning? *Advances in Neural Information Processing Systems* 33 (2020), 6827–6839.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[29] Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yuying Chen, Yu Yin, Zai Huang, and Shijin Wang. 2020. Neural cognitive diagnosis for intelligent education systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 6153–6161.

[30] Chun-Kit Yeung and Dit-Yan Yeung. 2018. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. 1–10.

[31] Chun-Kit Yeung and Dit-Yan Yeung. 2019. Incorporating features learned by an enhanced deep knowledge tracing model for stem/non-stem job prediction. *International Journal of Artificial Intelligence in Education* 29, 3 (2019), 317–341.

[32] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems* 33 (2020), 5812–5823.

[33] Michael V Yudelson, Kenneth R Koedinger, and Geoffrey J Gordon. 2013. Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education*. Springer, 171–180.

[34] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*. 765–774.

A APPENDICES

A.1 Learning Activity Embedding

Exercising records of a learner consist of discrete question and concept indexes, as well as binary responses. Before we feed the learning activity sequence into our model, we should first obtain real-valued embedding vectors from the exercising record sequence.

Embedding all questions and concepts into separate vectors can be expensive and overwhelming. Following previous work [8], we embed all question-concept pairs based on Rasch Model to avoid over-parameterization. Specifically, we construct the embedding of the question q_t from concept c_t as:

$$\boldsymbol{q}_t = \boldsymbol{c}_{c_t} + \mu_{q_t} \cdot \boldsymbol{d}_{c_t}, \tag{23}$$

where $\boldsymbol{c}$ embeds knowledge concepts into vectors, $\boldsymbol{d}$ summarizes the variations of questions within this concept, and μ is the scalar difficulty parameter that indicates the deviation of a particular question from its relevant concept.

In addition, we embed questions along with response signals r_t into the learning activity embedding $\boldsymbol{a}_t$:

$$\boldsymbol{a}_t = \boldsymbol{c}_{c_t} + \boldsymbol{g}_{r_t} + \mu_{q_t} \cdot \boldsymbol{f}_{c_t, r_t}, \tag{24}$$

where $\boldsymbol{g}$ embeds learner's response to a question, $\boldsymbol{f}$ summarizes the variations of learning activities within this concept, concept embedding $\boldsymbol{c}$ and question difficulty μ are as defined earlier.

Question-concept embedding $\boldsymbol{q}_t$ and activity embedding $\boldsymbol{a}_t$ play an important role in later sequential modeling. Using these embedding sequences as input, DTransformer is able to diagnose knowledge state $\boldsymbol{z}$ for each learner at each time step.

A.2 Baselines

In this paper, we compare DTransformer with several previous methods. The details of all the comparison methods are:

- **DKT** [22] leverages recurrent neural network to trace student knowledge states. For an up-to-date performance, we utilized LSTM in our implementation.
- **DKT+** [30] is an extended variant of DKT [30], which attempts to solve reconstruct and consistency problems in DKT. The second problem of DKT performance across time-steps being not consistent, is also addressed in DTransformer.
- **DKVMN** [34] takes advantage of memory network to get interpretable student knowledge state [34]. It defines key and value memory matrices and update the corresponding knowledge state through memory read and write operations over time.
- **SAKT** [19] is the first self-attentive knowledge tracing model. It exploits a Transformer architecture to capture long-term dependencies between learning interactions of learners.
- **AKT** [8] is the context-aware attentive knowledge tracing. It uses the two self-attentive encoders and a knowledge extraction module to predict future performance of students.
- **CL4KT** [13] is based on AKT, but include contrastive learning to overcome sparsity. It designed 4 data augmentation methods based on hypothesis about student learning process.

A.3 T+N Prediction

The $T + N$ prediction experiment that we propose is a way of showing how stable the knowledge state tracing is. In this experiment, we evaluate the knowledge states $\boldsymbol{z}_t$ that DTransformer diagnosed for each time step t. More stable knowledge states can be used to predict student performance not only on the next exercise, but to estimate more accurately on more exercises in the next period of time. Therefore, we first assume the knowledge state is stationary after time t, and predict the student's performance at time step $t + 1$, $t + 2$, ..., with the same knowledge state. Specifically, for prediction on time step $t + n$, we calculate the probability of the student answering question q_{t+n} correctly as:

$$\alpha_i = \mathrm{softmax}(q_{t+n}^T z_{i,t}) = \frac{e^{q_{t+n}^T z_{i,t}}}{\sum_{j=1}^N e^{q_{t+n}^T z_{j,t}}}, \tag{25}$$

$$z_{q_{t+n}} = \sum_{i=1}^N \alpha_i z_{i,t}, \tag{26}$$

$$\hat{r}_{t+n} = \sigma\left(\mathrm{concat}(z_{q_{t+n}}, q_{t+n}) \cdot W^A\right). \tag{27}$$

Afterwards, we aggregate performance for each n in $t + n$ prediction on all time steps. The results, shown in Figure 5, hence displays how fast the prediction performance decreases over n, reflecting the stability of the knowledge states.

A.4 Knowledge tracing with DTransformer

With DTransformer, we are able to do explicit knowledge proficiency tracing by predicting how much a learner masters each knowledge concept at any time step along a learning sequence. This is done by using knowledge parameters $\boldsymbol{K}$ to query the knowledge state $\boldsymbol{z}_t$ at each time step t. Each knowledge parameter K_n represents the latent knowledge concept that DTransformer learns along the process, and thus we use it as the query in the attentive read-out module defined in Sec 3.3.2. Specifically, for each knowledge concept n, we calculate their proficiency level of the student at time t as:

$$\alpha_{n,i} = \mathrm{softmax}(K_n^T z_{i,t}) = \frac{e^{K_n^T z_{i,t}}}{\sum_{j=1}^N e^{K_n^T z_{j,t}}}, \tag{28}$$

$$z_{n,t} = \sum_{i=1}^N \alpha_{n,i} z_{i,t}, \tag{29}$$

$$p_{n,t} = \sigma\left(\mathrm{concat}(z_{n,t}, K_n) \cdot W^A\right). \tag{30}$$

To display knowledge tracing results as in Figure 6a, we further match the latent knowledge concept with real knowledge concepts by matching knowledge parameters with the knowledge concept embeddings $\boldsymbol{c}$. In this way, we are able to visualize the internal evolution of knowledge states for students.