Knowledge Tracing: A Survey

GHODAI ABDELRAHMAN, QING WANG, and BERNARDO NUNES, School of Computing, Australian National University

Humans' ability to transfer knowledge through teaching is one of the essential aspects for human intelligence. A human teacher can track the knowledge of students to customize the teaching on students' needs. With the rise of online education platforms, there is a similar need for machines to track the knowledge of students and tailor their learning experience. This is known as the ***Knowledge Tracing*** **(KT)** problem in the literature. Effectively solving the KT problem would unlock the potential of computer-aided education applications such as intelligent tutoring systems, curriculum learning, and learning materials' recommendation. Moreover, from a more general viewpoint, a student may represent any kind of intelligent agents including both human and artificial agents. Thus, the potential of KT can be extended to any machine teaching application scenarios which seek for customizing the learning experience for a student agent (i.e., a machine learning model). In this paper, we provide a comprehensive survey for the KT literature. We cover a broad range of methods starting from the early attempts to the recent state-of-the-art methods using deep learning, while highlighting the theoretical aspects of models and the characteristics of benchmark datasets. Besides these, we shed light on key modelling differences between closely related methods and summarize them in an easy-to-understand format. Finally, we discuss current research gaps in the KT literature and possible future research and application directions.

CCS Concepts: • **Computing methodologies** → **Reasoning about belief and knowledge**; **Knowledge representation and reasoning;**

Additional Key Words and Phrases: Knowledge tracing, memory networks, deep learning, sequence modelling, key-value memory, Bayesian knowledge tracing (BKT), intelligent education, factor analysis, survey

ACM Reference format:
Ghodai Abdelrahman, Qing Wang, and Bernardo Nunes. 2023. Knowledge Tracing: A Survey. *ACM Comput. Surv.* 55, 11, Article 224 (February 2023), 37 pages.
https://doi.org/10.1145/3569576

1 INTRODUCTION

Teaching is a vital activity to facilitate transfer of knowledge. It is well-known that one key factor of teaching is the ability of human teachers to track the learning progress of their students. This ability allows human teachers to adjust their teaching pace, materials, and methodology to maximize the knowledge growth of each individual student. Over the past 30 years, a variety of online education platforms such as **Massive Open Online Courses (MOOCs)** [114], intelligent

This research is supported by an Australian government higher education scholarship (Ghodai Abdelrahman), ANU Vice-Chancellor's Teaching Enhancement Grant.
Authors' address: G. Abdelrahman, Q. Wang, and B. Nunes, Australian National University, Acton, Canberra, ACT, Australia, 2601; emails: {ghodai.abdelrahman, qing.wang, Bernardo.Nunes}@anu.edu.au.

0360-0300/2023/02-ART224
https://doi.org/10.1145/3569576

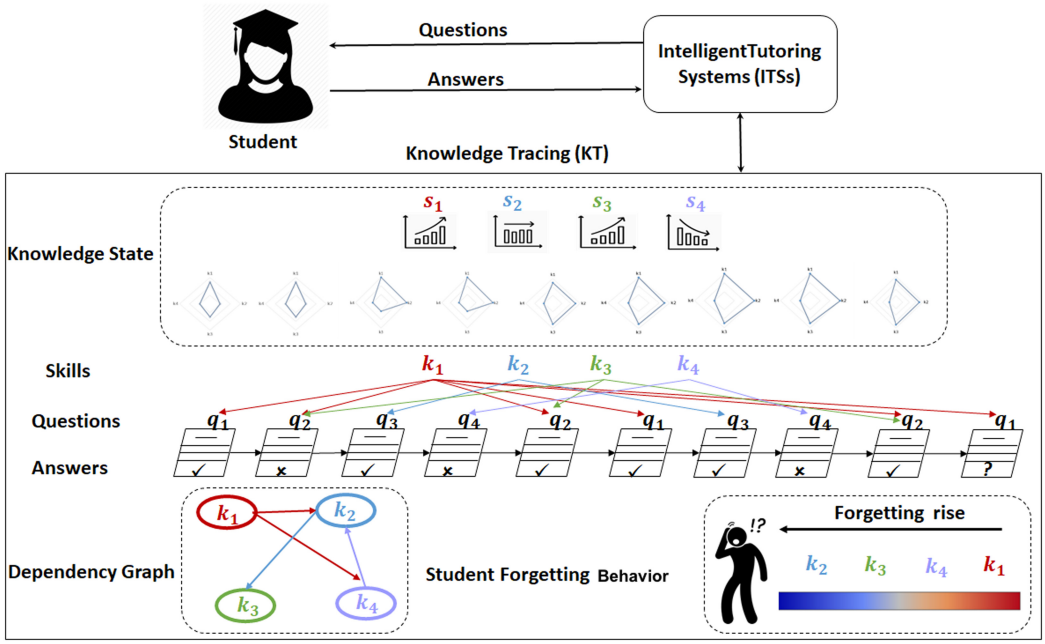

Fig. 1. An example scenario for knowledge tracing in an Intelligent Tutoring System (ITS).

tutoring systems [90], and educational games [69] have emerged to complement and sometimes completely replace conventional education systems. The COVID-19 pandemic, for example, has challenged conventional classroom-based teaching and sped up digital transformation in education systems. To alleviate the disruption of COVID-19, teachers and students around the world had to rapidly adjust to an online education mode. However, while there is a pressing need for teaching using computer technologies, technology-enhanced teaching has also posed new challenges. One of such challenges is to determine *how to effectively track the learning progress of a student through their online interaction with teaching materials* – known as the ***Knowledge Tracing* (KT)** problem [5, 6, 117]. Generally speaking, knowledge tracing aims to observe, represent, and quantify a student's knowledge state, e.g., the mastery level of skills underlying the teaching materials.

To better understand the KT problem, let us consider the learning activity depicted in Figure 1. The figure shows an interaction scenario between a student and an **Intelligent Tutoring System (ITS)** in which the student is given a sequence of questions taken from a question set $\{q_1, q_2, q_3, q_4\}$ and asked to answer these questions. During the interaction, the ITS estimates the student's knowledge states over the skills $\{k_1, k_2, k_3, k_4\}$ (e.g., math skills such as addition, subtraction, and multiplication) that are required to answer these questions. However, capturing a student's knowledge state is a challenging task due to several reasons:

- Each question might require more than one *skill*, which adds complexity to trace knowledge states. For instance, as shown by the arrows going from skills to questions in Figure 1, question q_2 requires the skills $\{k_1, k_3\}$. It is worth noting that *skill* is also referred to as *knowledge component* in some previous studies [65].
- Dependency among skills is another important factor to consider when tackling the KT problem. For example, although q_3 requires only skill k_2, skills k_1 and k_4 are prerequisites

for skill k_2 according to the dependency graph shown in Figure 1. Thus, the mastery level of skills required by the question q_3 should also consider k_1 and k_4, in addition to skill k_2.
- A student's forgetting behavior [29] may result in decaying their knowledge over skills. By modeling forgetting features, skills can be ranked by their relevance to forgetting. For example, the bottom of Figure 1 shows that skill k_1 is least affected by forgetting when the latest question q_1 is reached, whereas skill k_2 is the most affected one.

Historically, the notion of knowledge tracing was introduced by Anderson et al. in a technical report [5] for cognitive modeling and intelligent tutoring in 1986, which was later published in the *Artificial Intelligence* journal in 1990 [6]. Since then, many attempts have been made to design machine learning models for solving the KT problem. Early attempts [27, 117] followed Bayesian inference approaches, which usually relied on oversimplifying the model assumptions (e.g., assuming only one skill) to make the posterior computation tractable. Later, with the rise of classic machine learning methods such as logistic regression models [123], another direction followed by KT is to use parametric factor analysis approaches which trace a student's knowledge states and perform the answer prediction based on modeling a variety of factors [16, 17, 87], including: (1) aspects about students such as prior knowledge, learning capacity, or learning rate; (2) aspects about learning materials such as familiarity, number of previous practices, or difficulty; (3) aspects about a learning environment itself such as the nature of the learning channel (paper- or computer-based) and the temporal context of the practice time (within an examination period or a regular study period). In addition to these, psychological studies about the learning behavior [77] and forgetting behavior [52] of students have also suggested additional factors, such as the time lapse between a student's different interactions and the number of times on practicing learning materials, to be considered when tracing knowledge states. It is worth noting that this direction of KT is still active [34, 116] and is considered as an alternative to the recent state-of-the-art approaches based on deep learning.

Motivated by breakthroughs achieved by deep learning techniques [61], deep learning KT models have emerged rapidly. Piech et al. [89] has pioneered this direction of research and revealed the power of deep learning techniques for knowledge tracing. They proposed a model called **Deep Knowledge Tracing (DKT)** which applies **Recurrent Neural Networks (RNNs)** [63] to capture temporal dynamics in a sequence of interactions between questions and answers by a student, and based on that, to predict the student's answer for a new question. Empirical results showed that DKT outperformed traditional KT models on several benchmark datasets. This attempt highlighted the potential of using deep learning models in addressing the KT problem. In recent years, an increasing number of studies have exploited the development of deep learning KT models from different perspectives, including:

- **Memory structures.** Inspired by memory-augmented neural networks [39, 74], deep learning KT models have been extended by augmenting more powerful memory structures, typically key-value memory, for capturing knowledge states dynamically at a finer granularity such as the mastery level of each individual skill (e.g., [1, 131]).
- **Attention mechanisms.** Inspired by the *Transformer* architecture [115] and some further developments in natural language processing applications, attention mechanisms have been incorporated into deep learning KT models for capturing the relationships among questions and their relevance to a student's knowledge states (e.g., [22, 35, 81, 82, 102]).
- **Graph representation learning.** Inspired by the representational power of graph learning techniques such as Graph Neural Networks [58, 94], deep learning KT models have been equipped with graph learning techniques to leverage the rich structural information from graphs that can flexibly model relationships among questions and skills (e.g., [80, 112, 126]).

- **Textual features.** Question text may potentially contain a great wealth of information such as skills required by questions, difficulty of questions, and relationships between questions. Several deep learning KT models have leveraged textual features from question text for learning question representations and tracing a student's knowledge states (e.g., [64, 104, 128]).
- **Forgetting features.** Motivated by the *learning curve theory* [77], a recent trend in developing deep learning KT models is to incorporate forgetting features so that a student's forgetting behavior can be taken into consideration for knowledge tracing (e.g., [3, 20, 79]).

These studies facilitate the translation of breakthroughs in deep learning techniques into the KT domain. So far, deep learning KT models have achieved the state-of-the-art results on the majority of benchmark datasets for knowledge tracing (a summary of the results obtained by different KT models is presented in Table 7).

1.1 Contributions

This paper performs a detailed survey that summarizes, classifies, and analyzes KT methods both from the traditional machine learning perspective and the recent deep learning perspective. It also presents KT benchmark datasets and applications. The main contributions are as follows:

- We highlight the key categories of KT methods and compare their architectures over multiple aspects including model design, knowledge state representation, assumptions for relationships between questions and skills, and consideration of student's forgetting behavior.
- We present the chronic evolution for each KT category and discuss how each method is extended on the previous work.
- We summarize the characteristics of well-known KT datasets and compare the performance of key KT methods on each dataset by consolidating results from the relevant literature.
- We discuss application areas of KT that are not well explored currently to help derive future research directions in new venues.

Note that a recent KT survey has been presented by Liu et al. [65]. Despite their contributions to the field, our survey differs in depth and topics covered. The most notable differences are the comprehensive coverage of KT models; an important discussion on the differences in representation learning techniques related to key aspects of KT (such as knowledge state, forgetting behaviors, and knowledge components); an in-depth coverage of the KT datasets used by the relevant literature highlighting their similarities and issues; and, an extensive report of the results obtained by different KT models which allows future and accurate comparisons between them.

1.2 Survey Methodology

We follow a specific search criterion to select relevant articles in the KT literature for this survey. Our criterion goes as follows. First, we curated two search queries that match with the majority of KT methods over the last decade and executed them using the well-known scientific search engines including Google scholar, IEEE explore, ScienceDirect, Springer Link, and ACM Digital Library, i.e., (*"[intitle:knowledge AND intitle:tracing]"* and *"[intitle:student* AND (intitle:modeling OR intitle:predict*)]"*). Following that, we took the union of their results, which was around 3,000 articles, and reduced the volume of these initial search results by filtering out ones that were published before 2017 to focus on the recent methods. This reduced the volume to 478 articles. Then, we sorted the results by multiple factors including novelty of the contribution (e.g., proposing a new technique versus extending an existent one), relevance of the contributions (i.e., whether or not KT is the main research problem addressed), quality of the publication venue, and depth of evaluation (e.g., number of evaluated datasets, performing ablation study, and number of

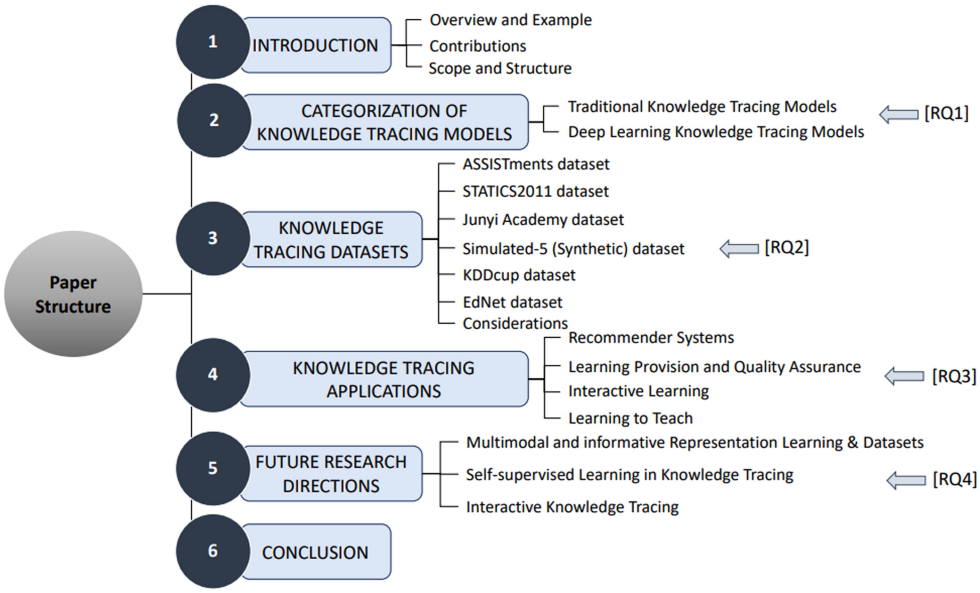

Fig. 2. Scope and structure of the survey.

comparison baselines). Finally, we took the top 80 articles of the sorted list to be included in this survey. We further performed backward snowballing to find the historical methods using references from the above included articles, and this process added around 45 additional papers into the set of articles covered by this survey.

1.3 Scope and Structure

This paper surveys the knowledge tracing literature to answer the following research questions:

- **[RQ1]:** What are the key categories of KT methods in the literature and how they differ from each other?
- **[RQ2]:** What are the datasets collected, pre-processed, and used for benchmarking KT tasks in the literature?
- **[RQ3]:** What are the application areas that can benefit from the research in the field of KT?
- **[RQ4]:** What are the future research opportunities and challenges in the field of KT?

In the rest of the article, we address each of these research questions in detail, as illustrated in Figure 2. We answer RQ1 in Section 2 by presenting a categorization of KT methods and discussing the characteristics, limitations, and assumptions of each category. After that, in Section 3, we conduct a comprehensive analysis on the datasets in terms of data collection, pre-processing, characteristics, and the ground truth information, which answers RQ2. For RQ3, we discuss several types of KT applications in Section 4, particularly in relating to how KT techniques can enhance students' personalized learning experience and performance. Finally, we explore future research directions for KT which may enrich the field of study and provide a broad understanding of opportunities and limitations of existing KT methods with respect to RQ4 in Section 5. The article is concluded in Section 6.

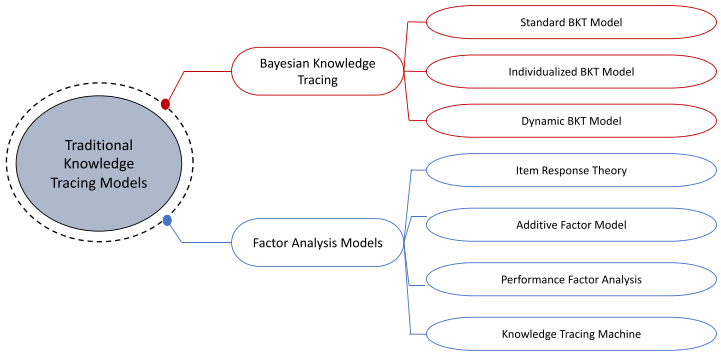

Fig. 3. An overview of traditional knowledge tracing models.

2 CATEGORIZATION OF KNOWLEDGE TRACING MODELS

This section introduces a comprehensive categorization for the KT models according to the related works found in the literature. Generally speaking, there are two broad categories: (1) *Traditional Knowledge Tracing Models*; and (2) *Deep Learning Knowledge Tracing Models*. Table 5 summarizes the main characteristics of different KT models across these two broad categories.

2.1 Traditional Knowledge Tracing Models

Traditionally, there are two popular lines of research for knowledge tracing: the *Bayesian Knowledge Tracing* and the *Factor Analysis Models*. Figure 3 provides an overview for major traditional knowledge tracing models that have been developed in the KT literature.

2.1.1 Bayesian Knowledge Tracing. ***Bayesian Knowledge Tracing*** **(BKT)** was motivated by the concepts of *mastery learning* [24]. Mastery learning assumes that all students can practise on a skill such that it may lead to mastery of that skill if two conditions are satisfied: (a) knowledge is appropriately described as a hierarchy of skills; and (b) learning experiences are structured to ensure that students master skills lower than those higher in the hierarchy [25].

BKT models often use a probabilistic graphical model such as Hidden Markov Model [27] and Bayesian Belief Network [117] to trace students' changing knowledge states as they practise skills. Central to these models is the Bayes' theorem that, for two events A and B, the following holds:

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}. \tag{1}$$

In what follows we discuss the standard Bayesian approaches and their variations.

- **Standard BKT Model**
  The first BKT model was introduced by Corbett and Anderson in 1994 [25]. The proposed model associates a skill with a binary knowledge state: $\{unlearned, learned\}$. This model only considers transitions from the unlearned state to the learned state, overlooking the forgetting theory (i.e., the probability of transition from a learned state to an unlearned state is always zero). Additionally, note that a student may make a mistake while in a learned state or guess correctly in an unlearned state. We refer to this model as the *standard BKT model* from now on. Table 1 summarizes the main parameters for the BKT model.
  There are two types of variables in the standard BKT model: (1) the *Binary latent variables* which represent the knowledge states of a given student (i.e., a single variable per skill indicating *learned state* and *unlearned state*); and (2) the *Binary observed variables* which

Table 1. Four Types of Model Parameters used in BKT

Parameter	Description
$p(L_0)$	Probability of skill mastery by a student before learning
$p(T)$	Probability of transition from an unlearned state to a learned state
$p(S)$	Probability of slipping by a student in a learned state
$p(G)$	Probability of guessing correctly by a student in an unlearned state

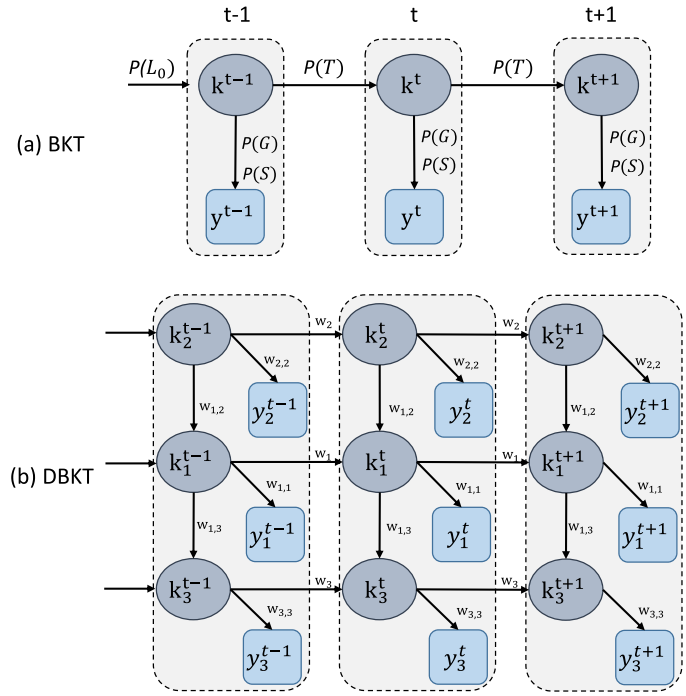

Fig. 4. Comparing the model architectures between (a) BKT and (b) DBKT, where latent variables for skills at the time step t are denoted as k_i^t and their correspondingly observed variables for answers at the time step t are denoted as y_i^t.

represent how students attempt questions (i.e., a variable per question indicating whether a question is answered correctly or not).

Figure 4 shows four types of model parameters in the standard BKT model. For each skill, there is one set of four corresponding parameters. At each time step $n \geq 1$, the model estimates the probability $p(L_n)$ of skill mastery by a student by:

$$p(L_n) = Posterior(L_{n-1}) + (1 - Posterior(L_{n-1})) * p(T), \tag{2}$$

where Posterior(L_{n-1}) is the posterior probability of being in a learned state given the observation to the n-th attempt by a student, calculated as:

$$Posterior(L_{n-1}) = \begin{cases} \dfrac{p(L_{n-1}) * (1 - p(S))}{p(L_{n-1}) * (1 - p(S)) + (1 - p(L_{n-1})) * p(G)} & \text{if the } n\text{-th attempt is } \textit{correct}; \\ \dfrac{p(L_{n-1}) * p(S)}{p(L_{n-1}) * p(S) + (1 - p(L_{n-1})) * (1 - p(G))} & \text{otherwise.} \end{cases}$$

Table 2. Model Parameters of Individualized BKT Models

Parameter	[25]		[85]		[62]		[129]		[56]	
	skill	student	skill	student	skill	student	skill	student	skill	student
$p(L_0)$	✓	✓	✓	✓	-	✓	✓	✓	✓	-
$p(T)$	✓	✓	✓	-	-	✓	✓	✓	✓	-
$p(S)$	✓	✓	✓	-	-	✓	✓	-	✓	✓
$p(G)$	✓	✓	✓	-	-	✓	✓	-	✓	✓

Hence, the probability of a student to correctly answer a question at each time step n is the sum of the probability of either mastering the skill but making a "slip" or not mastering the skill but making a correct "guess", as formulated by:

$$p(L_n) * (1 - p(S)) + (1 - p(L_n)) * p(G) \tag{3}$$

Figure 4(a) shows the standard BKT model with one skill node k. Starting with the prior probability $p(L_0)$ of skill mastery, the latent variable for skill k is transitioned from one time step $t-1$ to the next time step t based on the probability $p(T)$. The corresponding observed variable y represents the answer node, i.e., how a student attempts questions that require skill k, which is based on the probabilities $p(G)$ and $p(S)$.

- **Individualized BKT Model**
  One limitation of the standard BKT model is that it has no model parameters specific to students. All students are assumed to have the same prior knowledge and the same learning rate for any skill. As a result, the standard BKT model may underestimate the learning performance for above-average students, but overestimate the learning performance for below-average students [25].
  To alleviate this limitation, several attempts [25, 56, 62, 85, 129] have been made to extend the standard BKT model by introducing student-specific parameters. Corbett and Anderson [25] considered to add individual weights for each student, one weight corresponding to each of the four parameter types in the standard BKT model. Pardos and Heffernan [85] focused on individualizing the prior probability of skill mastery $p(L_0)$ heuristically for each student. Lee and Brunskill [62] also individualized the four parameter types in the standard BKT model for students; nonetheless, the combined effects of skill-specific and student-specific parameters were unexplored. Yudelson et al. [129] introduced an approach for individualizing BKT models that account for student differences with respect to two types of model parameters, the prior probability of skill mastery $p(L_0)$ and the probability of transition $p(T)$. The idea was to first define student-specific parameters and skill-specific parameters. Then, the gradients were explicitly computed in terms of both student-specific and skill-specific parameters. The underlying Hidden Markov Model remains unchanged. It turns out that adding student-specific parameter for $p(T)$ is more beneficial for the model accuracy than adding student-specific parameter for $p(L_0)$. Later, Khajah et al. [56] proposed to extend the standard BKT model by personalizing the guess and slip probabilities $p(G)$ and $p(S)$ based on student ability and problem difficulty.

  Compared to the standard BKT model, the individualized models can provide better correlation between actual and expected accuracy among students, leading to more effective decisions or improving the accuracy of predicting student performance. Table 2 summarizes the skill- and student-specific parameters used in the individualized BKT models.

- **Dynamic BKT Model**
  In the early years, the BKT models assumed that each question requires only one skill and different skills are independent from each other [25, 62, 85, 129]. Thus, these models cannot handle questions that require multiple skills, nor represent relationships between different skills. To address this limitation, Käser et al. [54] proposed to jointly model multiple skills and dependencies between different skills using ***Dynamic Bayesian Network*** **(DBN)**. They aimed to capture prerequisite skill hierarchies within a single model, e.g., one skill is conditionally dependent on another skill if the former is a prerequisite for mastering the latter. Similar to the standard BKT model, DBN considers the same two types of variables: binary latent variables and binary observed variables.

  At each time step, a latent variable for each skill is associated with an observed variable. A forget probability $p(F)$ is introduced, in addition to the four types of model parameters $\{p(L_0), p(T), p(S), p(G)\}$ in the standard BKT model. Dependencies between different skills are learnt as weights $\mathbf{w}$ of a log-linear model. Let $f : X \times O \rightarrow \mathbb{R}^d$ denote a mapping from a latent space X and an observed space O to d-dimensional feature vectors, and c be a normalizing constant. The objective of the log-linear model is to find the model parameters $\{p(L_0), p(T), p(S), p(G), p(F), \mathbf{w}\}$ that maximize the likelihood of the joint probability of $x_i \in X$ and $y_i \in O$ as formulated below:

$$L(\mathbf{w}) = \sum_i ln\left(\sum_{x_i} exp(\mathbf{w}^\top f(x_i, y_i) - c)\right).$$

  Figure 4(b) shows the **Dynamic Bayesian Network (DBKT)** with three skill nodes (denoted as k_1, k_2 and k_3). As indicated by the directed arrows, the latent variable for skill k_2 depends on the latent variable for skill k_1, and the latent variable for skill k_1 depends on the latent variable for skill k_3. Further, at each time step t, latent variables for skills depend on their latent variables in the previous time step $t-1$, while y_i is the corresponding observed answer nodes.

2.1.2 Factor Analysis Models. Factor analysis models are theoretically supported by the ***Item Response Theory*** **(IRT)** [30], which has played a large role in educational assessment and measurement. The key idea is to estimate student performance by learning a function, usually a logistic function, based on various factors in a population of students who solve a set of problems. It is important to note that, although an item in the original IRT corresponds to a question involving a single skill, later works in this line have been generalized to considering an item that may involve multiple skills. The mapping between items and skills is often represented in the form of Q-matrix, i.e., an entry q_{jk} in a Q-matrix is 1 if an item j involves a skill k; or 0, otherwise. Q-matrix is commonly assumed as the side information.

- **Item Response Theory (IRT)**
  The history of IRT can be traced back to Thurston's pioneering work in the 1920s [71] and several other works in the 1950s and 1960s [7, 13, 40, 72]. IRT is built upon the following assumptions: (a) The probability that a student correctly answers an item can be formulated as an *item response function* based on the parameters of the student and the item; (b) The item response function monotonically increases with respect to the ability θ_i of a student i; and (c) For a student with the ability θ_i, items are considered conditionally independent.
  The *Rasch* model [92] is often referred to as the simplest IRT model, in which the item response function is defined by a one-parameter logistic regression (1PL) model. Let $\mathcal{L}(\cdot)$ be a logistic function. By taking into account a difficulty parameter b_j that models the difficulty

of an item j, the probability p_{ij} that a student i correctly answers an item j is defined as:

$$p_{ij} = \mathcal{L}(\theta_i - b_j) = \frac{exp^{(\theta_i - b_j)}}{1 + exp^{(\theta_i - b_j)}}. \tag{4}$$

Several multiple parameter logistic regression models have also been developed for IRT, e.g., a **four-parameter logistic (4PL)** model introduced by Barton and Lord [9]:

$$p_{ij} = c_j + (d_j - c_j)\mathcal{L}(a_j(\theta_i - b_j)) = c_j + (d_j - c_j)\frac{exp^{a_j(\theta_i - b_j)}}{1 + exp^{a_j(\theta_i - b_j)}}, \tag{5}$$

where a_j is a discrimination parameter to model how well an item j can differentiate students, c_j is a guessing parameter to model the effect of guessing, and d_j is a slipping parameter to model the effect of careless errors.

IRT has been extended in many different directions. Wilson et al. [122] proposed ***Hierarchical IRT*** **(HIRT)** and ***Temporal IRT*** **(TIRT)**. HIRT exploits structure among questions by assuming that related questions (i.e., questions sharing similar skills) have difficulty parameters drawn from the same distribution. Different questions might vary in difficulty, but questions for trivial skills tend to be easier while ones for difficult skills tend to be harder. TIRT models each parameter in the logistic model (e.g., 4PL) as a time-varying stochastic process such as a *Wiener random* process [108]. Zhou et al. [132] proposed the ***Educational context-aware Cognitive Diagnosis (ECD)*** model that builds on the IRT theory through combining a student's educational context features (e.g., gender or parents' educational level) with exercise answering features using a two-stage hierarchical attentive architecture. Zhuang et al. [134] proposed a question recommendation framework for enhancing a student's proficiency level, named ***Neural Computerized Adaptive Testing (NCAT)***, which consists of two components: a question selection agent following a Q-learning setup [121] and an IRT-based knowledge tracing model [120] for answer prediction. The NCAT framework firstly selects a support set (i.e., a train mini-batch of question-answer samples) that is used to train the IRT-based knowledge tracing model, then uses a query set (i.e., a test mini-batch) to evaluate the performance of the model, and generates the reward signal for the question selection agent as the inverse of the prediction error on the query set.

- **Additive Factor Model (AFM)**
  The *Additive Factor Model* [17], originated from ***Learning Factors Analysis*** **(LFA)** [16], is a logistic regression model under four assumptions: (1) the prior knowledge of students may vary; (2) students learn at the same rate; (3) some skills are more likely to be known than others; and (4) some skills are easier to be learnt than others. In this model, a difficulty parameter β_k and a learning rate parameter γ_k are assigned for each skill k, respectively.

  The key idea of AFM is that the probability of answering an item correctly by a student is proportional to an additive combination of the ability of the student, the difficulty of skills involved in the item, and the amount of learning gained from each attempt. Let $K(j)$ be the set of skills involved in an item j, which can be obtained from a Q-matrix, and T_{ik} be the number of times that a student i has attempted on an item involving a skill k. AFM defines the probability of answering an item correctly by a student i on an item j as:

$$p_{ij} = \mathcal{L}\left(\theta_i + \sum_{k \in K(j)} (\beta_k + \gamma_k T_{ik})\right). \tag{6}$$

- **Performance Factor Analysis (PFA)**
  ***Performance Factor Analysis* (PFA)** [87] overcomes the limitation of AFM which ignores the evidence of learning in the successful and unsuccessful attempts on items by a student.

  The key idea of PFA is to discard the parameter θ_i used in the previous models and instead count for success and unsuccessful attempts separately. PFA has three parameters for each skill, including: (1) β_k is for the difficulty of a skill k; (2) γ_k^S is for the effect of learning a skill k after successful attempts; and, (3) γ_k^F is for the effect of learning a skill k after unsuccessful attempts. Conceptually, γ_k^S and γ_k^F reflect the learning rate for a skill k when being applied successfully and unsuccessfully.

  Let T_{ik}^S and T_{ik}^F be the number of successful attempts and the number of unsuccessful attempts made by a student i on a skill k, respectively. Then PFA calculates the probability that a student i correctly answers an item j as:

$$p_{ij} = \mathcal{L}\left(\sum_{k \in K(j)} \left(\beta_k + \gamma_k^S T_{ik}^S + \gamma_k^T T_{ik}^F \right) \right). \tag{7}$$

- **Knowledge Tracing Machine (KTM)**
  ***Knowledge Tracing Machine* (KTM)** was recently proposed by Vie and Hisashi [116], which generalizes ***Factorization Machine* (FM)** [107, 109] for student modeling.

  KTM allows to consider an arbitrary number of factors about students, items, skills, successful and unsuccessful attempts, or extra information about the learning environment, such as using a mobile or a laptop. For a number N of factors, we denote all factors involved in an event by a sparse vector x of length N such that $x_k > 0$ if a factor $k \in [1, N]$ is involved in the event; or 0, otherwise. Then KTM estimates the probability p_{ij} of a correct answer on an item j by a student i with an event involving x as follows:

$$p_{ij} = \mathcal{L}\left(\sum_{k=1}^{N} w_k x_k + \sum_{1 \leqslant k < l \leqslant N} x_k x_l \langle v_k, v_l \rangle + \mu \right) \tag{8}$$

  where μ is a global bias, and each factor k is modeled by both a weight $w_k \in \mathbb{R}$ and an embedding vector $v_k \in \mathbb{R}^d$ for some dimension d. The first term models the logistic regression of all factors and the second term models pairwise interactions between different factors. When $\mathcal{L}$ is a logistic function, KTM includes IRT, AFM and PFA as special cases [116].

  Wenbin et al. [34] extended the KTM model through their ***Knowledge Tracing Machine by modeling cognitive item Difficulty and Learning and Forgetting (KTM-DLF)*** model by adding factors related to the forgetting behavior of students. They represented the forgetting by time lapse since the last successful attempt for the involved skills.

2.1.3 Discussion. Both Bayesian knowledge tracing and factor analysis models have strengths and weaknesses. We discuss their connections and differences from three aspects: model parameters, model inference, and temporal analysis.

- **Model parameters:** Most recent models in BKT and FAM have taken into account both student- and skill-specific model parameters. Early BKT models were primarily centered on the four parameters: prior learning parameter $p(L_0)$, learning rate parameter $p(T)$, guess parameter $p(G)$, and slip parameter $p(S)$ [25] and their student-specific variants. These early BKT models usually assume that there is no forgetting parameter. Only some recent works

have incorporated the forgetting parameter [54]. For factor analysis models, most of them have been developed with similar or more flexible model parameters than BKT models. Particularly, recent works such as KTM [116] consider a wide range of factors, enabling a flexible way to incorporate the side information into student modelling.

- **Model inference:** To keep the Bayesian inference tractable, BKT models typically assume a first-order Markov chain when making inference based on a sequence of past question-answering history, i.e., only considering the most recent observation. This assumption however limits their ability to model complex dynamics on student learning behaviors. Some recent dynamic BKT models such as DBN [54] are often computationally intractable, and thus they trade off predication accuracy for computational efficiency. On the other hand, factor analysis models usually do not explicitly make inferences on knowledge states of a student (e.g., decide whether a student has achieved a certain level of skill mastery by tracing knowledge states). Instead, they target to maximize other model parameters such as the learning rate.
- **Temporal analysis:** BKT models essentially deal with a sequence prediction problem based on the history of student learning. In contrast, factor analysis models do not consider the order of questions in which a student's answers are observed. For example, given two questions and their corresponding answers from a student, whether one question is answered before the other is not important for factor analysis models. Nonetheless, by incorporating extra temporal features of student learning behaviors, factor analysis models can be enhanced to analyze temporal aspects of student learning.

A number of attempts have been made to leverage the best from both worlds of Bayesian knowledge tracing and factor analysis models. For example, [47, 56, 57] extended the IRT using the Bayesian inference to customize the estimation of question difficulty based on observations of each student. Further work has been done through incorporating factors that reflect the characteristics of individual students [27, 85], or the characteristics of specific items of assessment within skills [86].

2.2 Deep Learning Knowledge Tracing Models

Inspired by the success of deep learning [37, 38, 61, 96, 105], recent researches on knowledge tracing have applied deep learning techniques. Figure 5 presents a taxonomy of deep learning knowledge tracing models.

2.2.1 Sequence Modeling for Knowledge Tracing. A knowledge tracing task is typically modeled as a sequence prediction problem from a machine learning perspective. Let $Q = \{q_1, \ldots, q_{|Q|}\}$ be the set of all distinct questions in a dataset. Each $q_i \in Q$ may have a different level of difficulty, which is not explicitly provided. When a student interacts with the questions in Q, a sequence of interactions $X = \langle x_1, x_2, \ldots, x_{t-1} \rangle$ undertaken by the student can be observed, where $x_i = (q_i, y_i)$ consisting of a question q_i and answer $y_i \in \{0, 1\}$. $y_i = 0$ means that q_i is incorrectly answered and $y_i = 1$ means that q_i is correctly answered.

Definition 2.1. Given a sequence of interactions X that contains the previous question answering of a student, the *knowledge tracing* problem is to predict the probability p_t of correctly answering a new question q_t at the time step t by the student, i.e., $p_t = (y_t = 1 | q_t, X)$.

***Deep Knowledge Tracing* (DKT)** [89] pioneered the use of deep learning for knowledge tracing. It employs ***Recurrent Neural Network* (RNN)** [63] and a **Long Short Term Memory**

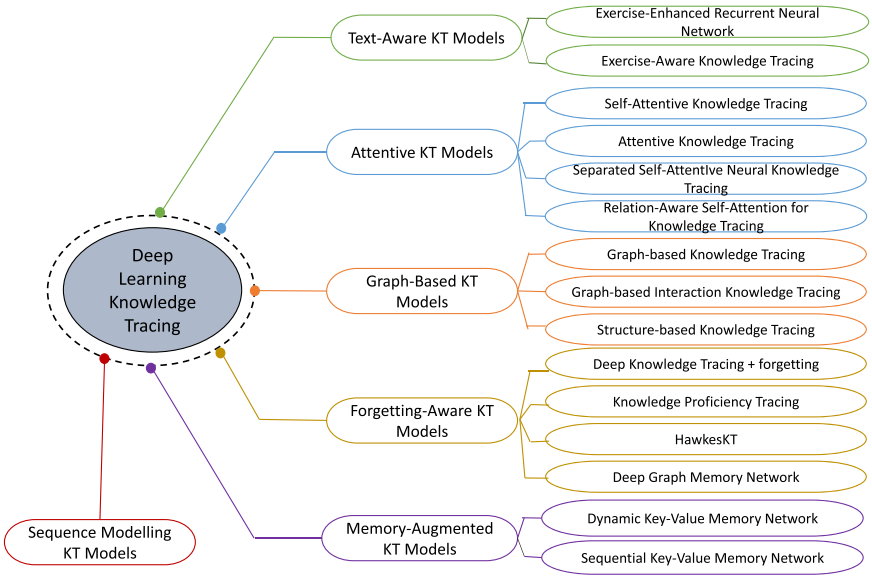

Fig. 5. A taxonomy of deep learning knowledge tracing models.

(LSTM) [45] to predicate the probability of correctly answering a question at each time step. A sequence of hidden states $\langle h_1, h_2, \ldots, h_n \rangle$ is computed which encodes the sequence information obtained from previous interactions. At each time step t, the model calculates the hidden state h_t and the student's response p_t as follows:

$$h_t = Tanh(W_{hx}x_t + W_{hh}h_{t-1} + b_h) \tag{9}$$

$$p_t = \sigma(W_{hy}h_t + b_p) \tag{10}$$

where the $Tanh(u_i) = (e^{u_i} - e^{-u_i})/(e^{u_i} + e^{-u_i})$ and $\sigma(u_i) = 1/(1 + e^{-u_i})$ are activation functions, W_{hx}, W_{hh} and W_{hy} are weight matrices, and b_h and b_p are bias vectors.

Despite the promising performance, DKT has several limitations. First, it assumes only one hidden KC (i.e., a skill) in a student's knowledge state h_t. Second, it cannot model the relationships among multiple KCs. Third, it assumes that all questions are equally likely related to each other, which may not hold in many scenarios as some questions may be more relevant to each other than the remainder of the sequence. Thus, various attempts have been made on extending DKT with the aim of enhancing the model capacity for tackling the KT problem. Below, we review the related work in the following areas of extension.

A number of KT models have extended DKT [89] to address its limitations. For example, Xiong et al. [125] proposed *Extended-Deep Knowledge Tracing* that extends DKT by adding auxiliary student features such as previous knowledge, question answering rates and time spent on learning and practice; and, exercise features, such as textual information, question difficulty, skill hierarchies and skill dependencies. A variant of DKT, called (DKT+) [127], was proposed to augment the original DKT loss function with two additional regularization terms to address the limitations in DKT's ability to reconstruct an answer input and reduce inconsistency of answer prediction for questions sharing similar KCs. Minn et al. [75] proposed an extension of DKT, named ***Deep Knowledge Tracing with Dynamic Student Classification*** **(DKT-DSC)**, that uses K-means to cluster

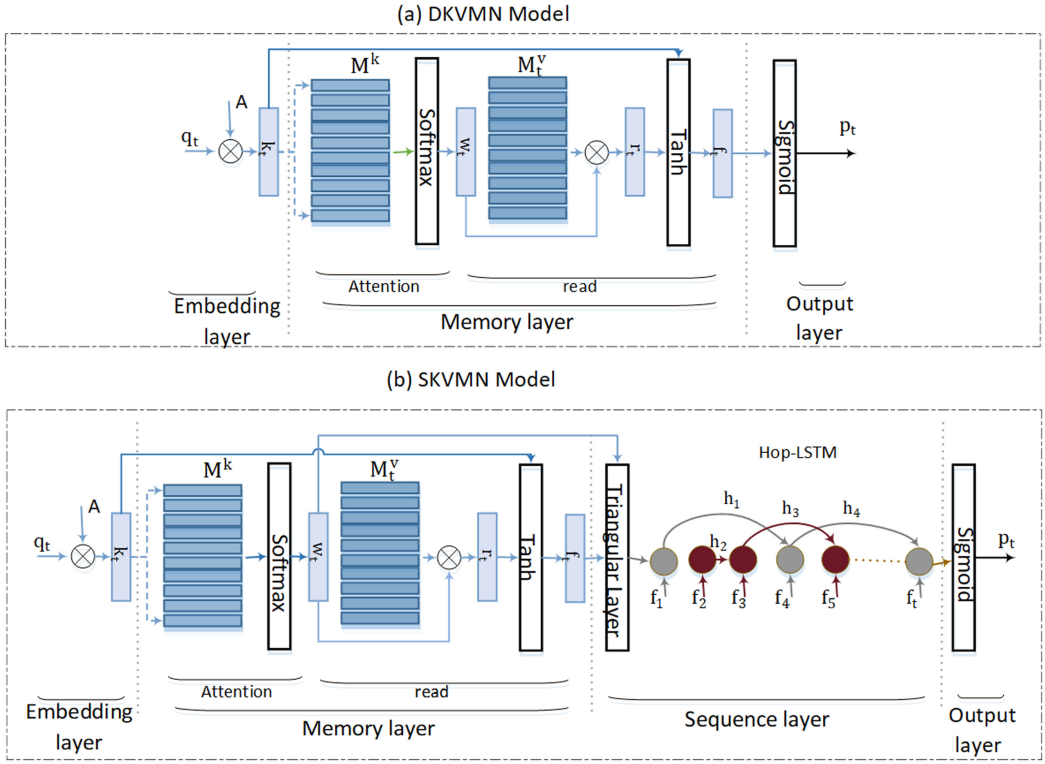

Fig. 6. A comparison of the model architecture design between (a) DKVMN and (b) SKVMN.

student profiles into groups based on their performance over the KCs and dynamically update the current cluster information over time while the performance changes.

2.2.2 Memory-Augmented Knowledge Tracing Models. To trace complex KCs learned by students, several works have extended DKT by augmenting an external memory structure, inspired by memory-augmented neural networks [39]. In particular, following ***Key-Value Memory Network* (KVMN)** [74], a key-value memory has been employed to represent knowledge state, which has more representational power than a hidden variable used in DKT. Such a key-value memory consists of two matrices: *key* and *value*. The *key* matrix stores the representations of KCs and the *value* matrix stores the student's mastery level of each KC. Below, we discuss two popular key-value memory networks for knowledge tracing.

- **Dynamic Key-Value Memory Network (DKVMN)**
  ***Dynamic Key-Value Memory Network* (DKVMN)** [131] has augmented DKT with two memory matrices: *key* and *value*. To trace how the knowledge state of a student evolves over time, unlike KVMN in which both *key* and *value* matrices are static [74], DKVMN designs the *value* matrix to be dynamic while keeping the *key* matrix to be static.

  Figure 6(a) shows the model architecture of DKVMN, where $\mathbf{M}^k \in \mathbb{R}^{N \times d_k}$ is the *key* matrix and $\mathbf{M}_t^v \in \mathbb{R}^{N \times d_v}$ is the *value* matrix at the time step t. It is assumed that there are N latent KCs underlying all questions in a learning task. For a question q_t at the time step t, a correlation weight w_t is computed, which represents the correlation between the question q_t and the underlying latent KCs stored in the *key* matrix $\mathbf{M}^k$. The model first retrieves a student's

Table 3. A Comparison of Attentive Knowledge Tracing Models

KT Model	Forgetting-Aware	Text-Aware	Attention Mechanism
SAKT [81]	×	×	Multi-head self-attention
AKT [35]	✓	×	Monotonic attention
SAINT [22]	×	×	Multi-head self-attention
SAINT+ [102]	✓	×	Multi-head self-attention
RKT [82]	✓	✓	Relational multi-head self-attention
CKT [101]	×	×	Dot-product attention
CoKT [68]	×	×	Collaborative multi-head self-attention

knowledge state with regard to the question q_t from the *value* matrix $\mathbf{M}_t^v$, calculated as:

$$r_t = \sum_{i=1}^{N} w_t(i)\mathbf{M}_t^v(i). \tag{11}$$

Then, the student's response for the question q_t is predicted based on the retrieved knowledge state. After the student answers the question q_t, the *value* matrix is updated to reflect the knowledge growth of the student after working on q_t.

- **Sequential Key-Value Memory Network (SKVMN)**
  ***Sequential Key-Value Memory Network* (SKVMN)** [1] aimed to address a limitation in DKT and DKVMN that KCs required by answering the past questions in a sequence are not necessarily relevant to KCs required by answering the current question. Thus, SKVMN employs a modified LSTM for sequential modeling, called *Hop-LSTM*, while remaining the same key-value memory structure and loss function as in DKVMN. Different from the standard LSTM, Hop-LSTM can explicitly capture sequential dependencies among questions in a sequence of interactions, and update the knowledge state of a student based on their responses to relevant questions.

  Figure 6(b) shows the model architecture of SKVMN. More precisely, two LSTM cells in Hop-LSTM are connected only if the input question of one LSTM cell is sequentially dependent on the input question of the other LSTM cell. This means that Hop-LSTM has the capability of hopping across the LSTM cells when their input questions are irrelevant to the current question q_t. Thus, it enables to capture long-term dependencies among questions that require similar KCs.

2.2.3 Attentive Knowledge Tracing Models. Following the *Transformer* architecture [115], several works [22, 35, 81, 82, 102] have attempted to incorporate an attention mechanism into KT models. Although the attention mechanisms introduced by these works vary, their key ideas are similar, i.e., to learn the attention weights of questions in a sequence of interactions in a way that can reflect the relative importance of these questions for predicting the probability of correctly answering the next question. This mitigates one limitation of DKT that treats all questions in a sequence of interactions equally important. In what follows, we discuss the main attentive knowledge tracing models. Table 3 briefly summarizes these models.

- **Self-Attentive Knowledge Tracing (SAKT)**
  ***Self-Attentive Knowledge Tracing* (SAKT)** [81] was the first to add an attention mechanism into the KT models. It uses the scaled dot-product attention mechanism proposed by Vaswani et al. [115] to learn attention matrices using multiple attention heads. Specifically, each attention matrix contains relative weights from a representative subspace, which

indicate the importance of questions in the past interactions for predicting a student's answer to the current question. Then, attention matrices from different representative subspaces are sent to a feed forward network for predicating student performance.

- **Attentive Knowledge Tracing (AKT)**
  ***Attentive Knowledge Tracing* (AKT)** was proposed by Ghosh, Heffernan, and Lan [35]. AKT differs from SAKT in its attention mechanism called *monotonic attention* (i.e., a modified, monotonic version of the scaled dot-product attention mechanism [115]) that can reduce attention weights for questions in a sequence of interactions proportional to their time distance in an exponential decay rate. The exponential weight decay is meant to consider the forgetting effect in a student's memory over time. In addition, an embedding representation was proposed to take into account a parameter for controlling how far a question deviates from a knowledge component it involves by following the *Rasch* model [92].

- **Separated Self-AttentIve Neural Knowledge Tracing (SAINT)**
  ***Separated Self-AttentIve Neural Knowledge Tracing* (SAINT)**, proposed by Choi et al. [22], differs from AKT and SAKT in the way that it has further applied a encoder-decoder model along with the scaled dot-product attention mechanism as in the original architecture of Transformer [115]. Specifically, SAINT separates a sequence of interactions by a student into a *question embedding sequence* and a *response embedding sequence*, which are then sent to the encoder and the decoder as input, respectively. The encoder and decoder are combinations of multi-head attention networks with the scaled dot-product attention mechanism [115].

  Recently, SAINT was extended by adding two time-related features into a response embedding sequence: *elapsed time* for the time taken by a student to answer each question, and *lag time* for the time interval between two consecutive learning interactions. This variant was named as the SAINT+ model [102].

- **Relation-Aware Self-Attention for Knowledge Tracing (RKT)**
  ***Relation-Aware Self-Attention for Knowledge Tracing* (RKT)** was proposed by Pandey and Srivastava [82]. Similar to SAKT and SAINT, RKT employs the scaled dot-product attention mechanism proposed by Vaswani et al. [115] to learn attention weights using multiple attention heads. However, it differs from the other attention-based KT models; RKT combines attention weights with *relation coefficients*, which are obtained from *exercise relation modeling* and *forgetting behavior modeling*. For the exercise relation modeling, it leverages the text information of questions (e.g., a question's textual information) to represent questions and estimate the relation between questions in a sequence of past interactions. For the forgetting behavior modeling, similar to AKT, RKT considers an exponential decay to count for a student's forgetting behavior over time.

- **Convolutional Knowledge Tracing (CKT)**
  Shen et al. [101] proposed the CKT model that combines attention with 1-D convolutional networks [59] for predicting correct answers. In addition to the past question-answering sequence, the CKT model considers a student's individualized skills represented as the historical relevant performance and the overall concept performance. The former uses a masked-attention between the latest question and each previous question in the answer sequence to extract a mastery score, while the latter extracts the mastery level on the learning concepts level by getting the percentage of the correctly answered questions for each defined concept. During answer prediction, the CKT combines individualized features

Table 4. A Comparison of Graph-based Knowledge Tracing Models

KT Model	Graph		Assumption
	Node	Edge	
GKT [80]	KC	KC-KC relation	One KC per question
GIKT [126]	Question or KC	Question-KC relation	Many KC per question
SKT [112]	KC	KC-KC multiple relations	One KC per question
PEBG [67]	Question or KC	Question-KC relation	Many KC per question

with embeddings from the answer sequence using a linear gating mechanism, and applies 1-D convolution operations to extract the final input representation for answer prediction.

- **Collaborative Knowledge Tracing (CoKT)**
  ***Collaborative Knowledge Tracing* (CoKT)** [68] borrowed the idea of collaborative filtering from the recommendation literature in order to consider knowledge states of peer students in predicting the answer probability of a given student. The authors empirically showed that fusing the intra-state (extracted from the previous answer history) and inter-state (extracted from knowledge states from peer students) features could enhance the answer prediction performance for students with a short answer history. The model estimates a peer's similarity score by deploying the *BM25* [93] string similarity function between the string-encoded question answering sequences and projects embedding vectors representing the inter-state from similar peers using a multi-head self-attention mechanism [115]. Similarly, the previous answer history of a student is embedded using a multi-head self-attention to represent the intra-state and is combined with the intra-state to get the final answer prediction.

2.2.4 Graph-Based Knowledge Tracing Models. In KT tasks, various relational structures often exist, for example, similarity of KCs, dependency between KCs, and correspondence between questions and their KCs. To capture such relational structures for better addressing the KT problem, a recent trend is to explore the power of graph representation learning techniques such as **Graph Neural Network (GNN)**. Table 4 briefly summarizes several main graph-based knowledge tracing models and we discuss each of them separately.

- **Graph-based Knowledge Tracing (GKT)**
  ***Graph-based Knowledge Tracing* (GKT)**, proposed by Nakagawa, Iwasawa, and Matsuo [80], attempted to incorporate a graph where nodes represent KCs and edges represent the dependency relation between KCs for a relational inductive bias. They reformulated the KT problem as a time series node-level classification problem and solved it using standard graph learning techniques such as message-passing GNNs [95]. Since such a graph is not explicitly given in KT tasks, the authors proposed two approaches to construct such graphs from a sequence of interactions by a student: (1) Statistics-based approach to construct a graph based on statistics such as how many times one KC was answered after another KC was answered; and (2) Learning-based approach to learn a graph with the performance optimization in an end-to-end manner.

- **Graph-based Interaction Knowledge Tracing (GIKT)**
  ***Graph-based Interaction Knowledge Tracing* (GIKT)**, proposed by Yang et al. [126], leverages the relation between questions and KCs, represented as a graph to learn useful embedding for answer prediction. Different from GKT which implicitly assumes that each question corresponds to one KC, GIKT assumes that one KC may be related to many questions and one question may correspond to more than one KC. Thus, GIKT can use

a GNN to aggregate the embeddings of questions and KCs based on their relation in the graph, and sends the embedding of each question in a sequence of interactions to an RNN model to predict a student's answer for the next question.

- **Structure-based Knowledge Tracing (SKT)**
  ***Structure-based Knowledge Tracing* (SKT)** was proposed by Tong et al. [112] which aimed to capture multiple relations among KCs such as similarity relation and prerequisites relation. Similar to GKT, SKT also assumes that each question corresponds to one KC. However, instead of a single relation between KCs as captured in GKT, SKT exploits multiple relations between KCs. Further, SKT supports information propagation to jointly model the temporal and spatial effects when summarizing graph data. These two kinds of graph embeddings are combined at each time step and fed to a recurrent model to predict the correct answer by a student.

- **Pre-training Embeddings via Bipartite Graph (PEBG)**
  ***Pre-training Embeddings via Bipartite Graph* (PEBG)** [67] presented a method for obtaining pre-trained exercise embeddings so as to improve the accuracy of knowledge tracing. In KT tasks, explicit exercise-KC relations and implicit exercise similarity as well as KC similarity often exist simultaneously. To capture all these relations in exercise embeddings, the authors represent them together with exercise difficulties as a bipartite graph. Then, they fuse these features in the defined bipartite graph to obtain the pre-trained exercise embeddings. Their experiments have indicated that the obtained exercise embeddings can significantly improve the performance of some KT models, such as DKT [89].

2.2.5 Text-Aware Knowledge Tracing Models. Until now, the deep KT models that we have discussed mainly focused on a student's interactions with questions in a sequence to predict the probability of correctly answering the latest question by the student. Yet, they did not consider much about the textual features of questions themselves. Text-aware KT models are motivated by leveraging the textual features of questions to enhance the performance in tackling the KT tasks.

- **Exercise-Enhanced Recurrent Neural Network (EERNN)**
  ***Exercise-Enhanced Recurrent Neural Network* (EERNN)** was proposed by Su et al. [104], which is a text-aware KT model to predict the probability of correctly answering a given question. The model uses a bi-directional LSTM module to extract the representation (i.e., a vector) of each question from the question's text and then trace a student's knowledge states by combining it with the representations of the previously answered questions using another LSTM module. Two variants of EERNN were developed: EERNNM and EERNNA. The EERNNM variant assumes that a sequence of interactions satisfies the Markov property, i.e., the answer prediction for the next question only depends on the latest observed knowledge state; thus, it only considers the last hidden state. The EERNNA variant considers all the previous knowledge states and combines them through an attention mechanism.

  Later, Yin et al. [128] has further extended the work by Su et al. [104] through leveraging a pre-training task to learn question representations. The authors followed a **masked language model (MLM)** objective [73] and showed that this pre-training step could further enhance the model's performance compared to the original model.

- **Exercise-Aware Knowledge Tracing (EKT)**
  ***Exercise-Aware Knowledge Tracing* (EKT)** [64] extends EERNN to incorporate the information of multiple KCs during answer prediction, where a student's knowledge state is represented by a knowledge state matrix, rather than a knowledge state vector. Specifically,

the model uses a memory network for quantifying how much each question can affect the mastery of a student on multiple KCs during a sequence of interactions by the student.

- **Adaptable Knowledge Tracing (AdaptKT)**
  Cheng et al. [21] addressed the problem of transfer learning across KT domains aiming to transfer a trained KT model on the source domain to operate on a target domain while preserving its performance. Their proposed method works on the textual data of questions through learning a question embedding using a deep auto-encoder architecture that is trained on questions from both domains, and then training a KT model on the source domain while regularizing it to minimize the mean discrepancy [110] between the knowledge states in both domains. The final layer is randomly reinitialized while keeping the weights of the earlier layers frozen to train on the target domain.

In addition to the above text-aware KT models, other types of KT models such as ***Relation-Aware Self-Attention for Knowledge Tracing*** **(RKT)** [82] and ***Hierarchical Graph Knowledge Tracing*** **(HGKT)** [111] also extract features from the textual information of questions for learning question representations in their models.

2.2.6 Forgetting-Aware Knowledge Tracing Models. Learning psychological studies [52, 88, 98] showed that forgetting is an important aspect to consider for an accurate estimation of a student's knowledge state. This is because the knowledge mastery level of a student tends to decline with an exponential rate over time since the last practice of the relevant questions. From an experimental psychology perspective, Hermann Ebbinghaus [29, 78] studied forces that affect memory retention, leading to formulate what is currently known as the *learning curve theory* [77]. Two effects that are reflecting these forces on memory retention are the *forgetting* effect and the *learning* effect.

Modeling the forgetting effect is one of the major challenges that the KT literature has aimed at tackling. Traditional KT models have attempted to incorporate forgetting behavior by adding features such as the number of past trials or the lag time from the previous interaction [55, 88, 91, 99]. In recent years, several deep learning KT models have been developed to take a student's forgetting behavior into consideration during tracing knowledge states.

- **Deep Knowledge Tracing (DKT) + Forgetting**
  Nagatani et al. [79] proposed to extend the **Deep Knowledge Tracing (DKT)** model [89] by adding sequence-related forgetting features. These features include: (1) the number of times a student answers questions with the same KC till the current point of time; (2) the time lapse since the last interaction on a question with the same KC; and (3) the time lapse since the last interaction on a question regardless of its relating KC. The first feature reflects the learning effect while the other two features reflect the forgetting effect. Different from the previous traditional KT models that use forgetting features only with regard to questions with the same KC [55, 88, 91, 99], this work considers a student's interactions in the whole sequence so as to model more complex forgetting behavior.

- **Knowledge Proficiency Tracing (KPT)**
  Knowledge Proficiency Tracing **(KPT)** was proposed by Chen et al. [20], which is a probabilistic matrix factorization model that leverages prior information for knowledge tracing. Specifically, two kinds of priors have been considered in this model: (1) *question priors*: the model uses a Q-matrix which was marked by experts to depict the relationship between questions and KCs for generating question representations; and (2) *student priors*: the model captures the changes in a student's knowledge state over time by jointly applying both

learning curve and forgetting curve theories. The learning and forgetting factors are designed based on the assumption that a student's current knowledge state is mainly influenced by two underlying reasons: (a) the more exercises she does, the higher level of related knowledge state she will get; and (b) the more the time passes, the more knowledge she will forget.

To further improve the predictive performance, an improved version of KPT, called ***Exercise-correlated Knowledge Proficiency Tracing*** **(EKPT)** [48] was later developed, which incorporates the connectivity among questions over knowledge concepts into the probabilistic modeling.

- **HawkesKT**
  Inspired by the Hawkes process [43], Wang et al. [118] proposed *HawkesKT*, a model that uses point process to adaptively model temporal cross-effects in KT. It assumes that the mastery of a KC by a student is not only affected by previous interactions on questions of the same KC, but also interactions on the other questions (*cross-effects*). Further, the model assumes that cross effects caused by different previous interactions may also have different temporal evolutions on the mastery of different KCs. Although cross effects all decay with time, their decay rates differ from each other because some KCs may be easier to forget than the others.

- **Deep Graph Memory Network (DGMN)**
  Deep Graph Memory Network **(DGMN)** [3] is a hybrid KT model that combines graph neural networks with memory for forgetting aware knowledge tracing. The model aims at modeling the forgetting behavior over a KC space, which has the advantage to capture indirect relationships between questions. DGMN builds a dynamic graph from a knowledge state memory to capture relationships across KCs. Given a sequence of interactions, DGMN uses an attention mechanism to associate questions to their relevant KCs. Then, it calculates forgetting features over the sequence, and fuses question embedding, KC graph embedding, and forgetting features using a gating mechanism. The gating output is used to predict the probability of answering the next question correctly.

- **Learning Process-consistent Knowledge Tracing (LPKT)**
  The LPKT model [100] considers the learning gain of a student during the answer prediction. The learning gain is defined as the change on a knowledge state across a time interval since the last answered question and integrate the time interval lapse between answering questions into the embedding representation of the exercise sequence. The LPKT model consists of three sequential memory cells: (1) a learning progress gate that projects an embedding for the latest exercise in the sequence considering its tag, time to answer and time lapse before answering it, (2) a forgetting gate that gets the latest two learning progress embeddings and projects an output representing the forgetting effect, and (3) a prediction cell that considers the forgetting output and the latest question tag embedding to predict the correct answer.

2.2.7 Discussion. Deep learning KT models have demonstrated their great potential in solving the KT problem. Below, we discuss several key aspects that are crucial for being considered in their designs.

- **Knowledge state:** One fundamental assumption underlying each deep learning KT model is whether a knowledge state is considered over a single KC or multiple KCs. Accordingly, modeling a student's knowledge states based on the mastery level of KCs by the student is an important task in designing the deep learning KT models. Generally, from early works

such as DKT which uses a hidden state to model knowledge states over a single KC, to later works by memory-augmented KT models (e.g., DKVMN and SKVMN) and by text-aware KT models (e.g., EKT) which use matrices to model knowledge states over multiple KCs, a trend in deep learning KT models is to develop a mechanism that is expressive for dynamically capturing knowledge state representations over complex KCs.

- **KC dependencies:** In a KT task, each question is assumed to associate with a single KC or multiple KCs, which is often provided as a prior knowledge such as Q-matrix. One main challenge faced by deep learning KT models is to discover the dependencies among different KCs, for example, one KC requires several other KCs as the prerequisite skills. To address this challenge, two lines of research have been explored in the KT literature, including: (1) using an attention mechanism to learn how questions are related to each other in terms of their required KCs; and (2) using a graph-based learning model such as graph neural networks to learn the relationships between KCs or between questions according to their required KCs.
- **Feature augmentation:** To improve the model performance on KT tasks, additional features such as temporal features relating to forgetting behavior and textual features relating to question texts have been leveraged by a number of deep learning KT models in recent years. On one hand, augmenting additional features can usually lead to more accurate prediction on student learning performance; on the other hand, the augmentation of such additional features depends on their availability in databases, thus limiting their applicability within specific KT applications.

3 KNOWLEDGE TRACING DATASETS

This section presents an overview of the benchmark datasets used in the literature to support the evaluation of KT models. All publicly available datasets were downloaded, inspected, and relevant information reported. Table 6 lists the datasets and provides the general information such as student interactions, the number of questions, and data availability. More details about the datasets are presented below.

3.1 ASSISTments Datasets

The ASSISTments datasets [32, 84] contain longitudinal data collected from the free online tutoring ASSISTment platform.[1] Table 7 shows that the ASSISTments datasets are the most popular datasets used to benchmark KT models and the ones containing the most questions in total.

These datasets are composed of grade school math exercises sampled from the ***Massachusetts Comprehensive Assessment System*** **(MCAS)**[2] containing different types of questions, such as multiple choice, text, and open-ended questions. There are different versions of the ASSISTments datasets with data collected in different periods. Details about each version is presented below.

- **ASSISTments2009:** This dataset was collected during the school year 2009−2010 and, when first released, contained a total of 525,535 interactions (i.e., student responses to questions in the dataset), including duplicates, as discussed in [125]. The latest updated version of this dataset contains 346,860 interactions given by 4,217 students to a total of 26,688 distinct questions and 123 KCs. We note that only two thirds of the questions (17,751) are annotated with KCs. Questions without assigned KCs are annotated with **'NA' (not available)** or have no assigned value ('null') whereas all other questions are annotated with up to four KCs.

[1] https://www.assistments.org/.
[2] https://www.doe.mass.edu/mcas/testitems.html.

Table 5. A Summary of Descriptive Characteristics of Main Knowledge Tracing Models

KT Model	Learning Model	Knowledge Component		Knowledge State	Forgetting
		Single	Multiple		
BKT [25]	HMM/BN	✓	-	Binary scalar	×
DBN [54]	DBN	-	✓	Binary vector	×
IRT [40]	LR	✓	-	Real-valued vector	×
HIRT [122]	LR	✓	-	Real-valued vector	×
TIRT [122]	LR	✓	-	Real-valued vector	×
LFA [16]	LR	✓	-	Real-valued vector	×
AFM [17]	LR	✓	-	Real-valued vector	×
PFA [87]	LR	✓	-	Real-valued vector	×
KTM [116]	FM	-	✓	Real-valued vector	×
KTM-DLF [34]	FM	-	✓	Real-valued vector	✓
DKT [89]	RNN/LSTM	✓	-	Hidden state (vector)	×
DKT+ [127]	RNN/LSTM	✓	-	Hidden state (vector)	✓
DKT-DSC [75]	RNN/LSTM	✓	-	Hidden state (vector)	×
DKVMN [131]	KVMN	-	✓	Key-value memory (matrix)	×
SKVMN [1]	LSTM+KVMN	-	✓	Key-value memory (matrix)	×
SAKT [81]	FFN+MSA	-	✓	Attentive embedding (matrix)	×
AKT [35]	FFN+MSA	-	✓	Attentive embedding (vector)	✓
SAINT [22]	FFN+ED+MSA	-	✓	Attentive embedding (matrix)	×
SAINT+ [102]	FFN+ED+MSA	-	✓	Attentive embedding (matrix)	✓
RKT [82]	FFN+MSA	-	✓	Attentive embedding (vector)	✓
CKT [101]	1-D(CNN)	-	✓	Attentive embedding (vector)	×
CoKT [68]	FFN+MSA	-	✓	Attentive embedding (matrix)	×
GKT [80]	GNN	-	✓	Vector	×
GIKT [126]	GNN/RNN	-	✓	Vector	×
SKT [112]	GNN	-	✓	Vector	×
PEBG [67]	RNN/LSTM	-	✓	Hidden state (vector)	×
EERNN [104]	RNN/LSTM	✓	-	Hidden state (vector)	×
EKT [64]	AM/LSTM	-	✓	Attentive embedding (matrix)	×
AdaptKT [21]	ED/LSTM	-	✓	Hidden state (vector)	×
DKT+forget [79]	RNN/LSTM	✓	-	Hidden state (vector)	✓
KPT [20, 48]	FM	-	✓	Real-valued vector	✓
HawkesKT [118]	FM	-	✓	Real-valued vector	✓
DGMN [3]	GCN/KVMN	-	✓	Key-value memory (matrix)	✓
LPKT [100]	RNN/LSTM	-	✓	Hidden state (vector)	✓

Despite the popularity of this dataset, its original version is not reliable as discussed in [131] and the updated 'skill-builder'[3] version is preferred as it fixes data modeling issues and removes duplicated records. It is also noteworthy to mention that results obtained with the different versions of this dataset (or with duplicated records) are often reported in the literature but they should not be directly compared to other approaches [122].

- **ASSISTments2012:**[4] This is the largest version of the ASSISTments datasets consisting of data collected for one year (from Sept. 2012 to Oct. 2013). Despite the ASSISTments team reporting that the dataset contains approximately 10 million 'exercises', the available dataset consists of 179,999 distinct questions answered by 46,674 students resulting in 6,123,270 interactions. We note that the vast majority (126,908) of the questions do not have any of the 265 KCs associated with them. The lack of questions annotated with KCs may explain the overall lower performance of the KT models when applied to this dataset (see Table 6).

[3] https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data.
[4] https://sites.google.com/site/assistmentsdata/home/2012-13-school-data-with-affect.

Table 6. Dataset Statistics

Dataset	#Questions	#Students	#Interactions	#KCs	Public available
ASSISTments2009	26, 688	4, 217	346, 860	123	Yes
ASSISTments2012	179, 999	46, 674	6, 123, 270	265	Yes
ASSISTments2015	100	19, 917	708, 631	-	Yes
ASSISTChall	3, 162	1, 709	942, 816	102	Yes
STATICS2011	1, 224	335	361, 092	80	No
Junyi Academy	722	247, 606	25, 925, 992	41	Yes
Simulated-5 (Synthetic)	50	4, 000	200, 000	5	Yes
Algebra 2005-2006	1, 084	575	813, 661	112	Yes
Algebra 2006-2007	90, 831	1, 840	2, 289, 726	523	Yes
Bridge to Algebra	19, 258	1, 146	3, 686, 871	493	Yes
EdNet-KT1	13, 169	784, 309	95, 293, 926	188	Yes
EdNet-KT2	13, 169	297, 444	56, 360, 602	188	Yes
EdNet-KT3	13, 169	297, 915	89, 270, 654	293	Yes
EdNet-KT4	13, 169	297, 915	131, 441, 538	293	Yes

- **ASSISTments2015:**[5] This dataset contains 708, 631 student interactions with the ASSISTments platform in the year 2015 produced by 19, 917 students answering 100 distinct questions. The dataset contains only four attributes: (i) the questions' identifiers; (ii) the identifier of the student who answered the questions; (iii) an attribute indicating the correctness of the answer given by each student; and (iv) a log attribute where a temporal sequence of the answers given by the students can be inferred.
  Unlike previous versions of the ASSISTments datasets, no metadata and no KC are provided. Another difference between this and previous datasets is related to the average number of responses given to each question. This dataset has an average of 7, 086.31 answers per question whereas the 2009 and 2012 versions have 12.99 and 34.01, respectively.
- **ASSISTment Challenge (ASSISTChall):**[6] Released in 2017, the full ASSISTment Challenge dataset contains data from 2004–2005 and 2005–2006 academic years. It contains 3, 162 distinct questions answered by 1, 709 students over 102 KCs resulting in 942, 816 interactions. On average, this dataset has 298.17 answers per question, placing it second in terms of the answer per question ratio (only lower than the ASSISTments2015 dataset). As part of a data mining competition, this dataset contains the most descriptive information among the ASSISTments datasets.

3.2 STATICS2011 Dataset

This dataset is available upon request and contains students' interactions with questions related to the Engineering Statics course[7] taught at the Carnegie Mellon University during Fall 2011 [60].

The original dataset contains 361, 092 interactions, 335 students, and 1, 224 questions. In the KT literature, this dataset is often preprocessed [131], resulting in 1, 223 distinct questions answered by 333 students over 80 KCs. After preprocessing, the number of students' interactions is almost

[5] https://sites.google.com/site/assistmentsdata/home/2015-assistments-skill-builder-data.
[6] https://sites.google.com/view/assistmentsdatamining.
[7] https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=507.

halved to 189, 297. This preprocessing was justified due to the large number of interactions without information on whether the questions have been correctly answered. The preprocessing considers the concatenation of the attributes 'problem name' and 'step name' and only the interactions with a valid first attempt. On average, this dataset has 568.45 answers per question.

3.3 Junyi Academy Dataset

This dataset[8] was collected between November 2010 and March 2015 from the Junyi Academy [19], an e-learning platform in Taiwan. The original dataset contains 25, 925, 992 interactions, 247, 606 students, 722 distinct questions, and 41 KCs covering a number of topics in math. However, considering the same preprocessing step made in the previous datasets (i.e., students interactions with no hints given to help solve the questions and only students who have attempted each question once), the number of interactions drops to 21, 571, 469 (∼17%), 220, 441 (∼11%) students, and 716 (<1%) distinct questions. Finally, on average, this dataset has 97.85 answers per question.

Although this dataset has been commonly used in the KT literature, the performance reported in some of the works cannot be directly compared. This is because these works use different subsets or preprocessing techniques [1, 82, 112]. Further, note that an updated version of the Junyi dataset is available in Kaggle[9] with data collected from August 2018 to August 2019. This dataset contains 11, 468, 379 interactions, 25, 649 students, and 1, 701 distinct questions where no hints were given and students have attempted each question only once.

3.4 Simulated-5 (Synthetic) Dataset

This synthetic dataset was proposed by Piech et al. [89], which simulates virtual students to answer the same sequence of questions over a set of KCs in a controlled environment.[10] The dataset is divided into two subsets, including training and testing. Each subset contains 50 distinct questions associated with a single KC and a difficulty level. In total, each question is answered by 4, 000 virtual students resulting in 200, 000 interactions. The classic Item Response Theory [33] was used to create the interactions and simulate students learning over time [89, 127]. This dataset provides a standard format and does not require preprocessing steps such as removing duplicates or steps to infer a sequence of questions answered by a student, potentially allowing direct comparison between different KT models.

3.5 KDDcup Dataset

This dataset was presented at the KDDcup 2010 Educational Data Mining challenge[11] [103] and contains 13–14 year old students' responses to Algebra questions from 2005 to 2007 extracted from the intelligent tutoring system called "The Cognitive Tutors" developed by Carnegie Learning Inc. in the US. The dataset is split into three subsets described in what follows.

- **Algebra 2005-2006:** Considering both training and test data, collected between August 2005 and June 2006, this dataset contains 1, 084 distinct questions answered by 575 students resulting in 813, 661 interactions. Unlike other datasets, each question is divided into sub-questions totaling 210, 136 sub-items. On average, there are 750.60 answers per question. The total number of distinct KCs is 112 and each sub-question is associated with one or more KCs. As in previous datasets, the total number of interactions drops to 57.8%, the number of students roughly remains the same (574) and the number of sub-items is reduced to 130, 256 if the

[8] https://pslcdatashop.web.cmu.edu/DatasetInfo?datasetId=1198.
[9] https://www.kaggle.com/junyiacademy/learning-activity-public-dataset-by-junyi-academy/tasks.
[10] https://github.com/chrispiech/DeepKnowledgeTracing/tree/master/data/synthetic.
[11] https://pslcdatashop.web.cmu.edu/KDDCup.

data is preprocessed removing sub-items with no KCs assigned, interactions where students used any hints and questions with two or more attempts.

- **Algebra 2006-2007:** This subset contains 2,289,726 interactions generated by 1,840 students answering 90,831 distinct questions with a total of 577,119 sub-items classified by one or more of the 523 KCs. On average, there are 49.36 answers per question. Similarly to the Algebra 2005-2006 dataset, the number of interactions drops to 1,567,072, the number of students falls to 1,813 and the number of questions and sub-items reduce to 89,877 and 470,187, respectively, after the data preprocessing step. After inspecting this subset, we found that the timestamps provided are incorrect. This may affect the prediction of KT approaches and may explain the low adoption of this subset in comparison to the others.
- **Bridge to Algebra:** This subset contains 3,686,871 interactions collected between November 2006 and June 2007 generated by 1,146 students, 19,258 distinct questions divided into 207,790 sub-items classified by one or more of the 493 KCs. After preprocessing, the total number of interactions is almost halved (1,721,987), the number of questions drops (18,715) slightly whereas the number of items is reduced to 123,778 (~40%) and the number of students remains almost the same (1,145).

3.6 EdNet Dataset

EdNet[12] is a hierarchical dataset composed of four subsets identified by the ids: 'KT1', 'KT2', 'KT3', and 'KT4', each containing different types of student activities. 'KT1', for example, contains question-response pairs similar to other datasets. The main difference is that some questions in this dataset are organized in bundles (a set of questions that must be completed altogether). This dataset contains over 95,293,926 interactions, 13,169 questions, 784,309 students, and 188 KCs.

Unlike the other datasets, 'KT2' contains the actions of the users during question-solving activities. For example, it records the final submission and student decision-making (alternating choices) before submitting the final answer. This subset has 56,360,602 interactions and 297,444 students. Besides the actions in 'KT1' and 'KT2', 'KT3' subset includes information about how students interact with learning activities to answer a question (e.g., watch a lecture). This subset contains more KCs (293), interactions (89,270,654) and students (297,915). Finally, 'KT4' is the most complete subset containing every action recorded by the EdNet system, including students' purchases (e.g., course purchases). This subset contains 131,441,538 interactions in total.

Overall, the EdNet dataset series incrementally provides information into student activities and behaviors. The dataset was collected over two years from the intelligent online tutoring platform named *Riid TUTOR*[13] dedicated to practicing English for international communication (TOEIC) assessment [23] in South Korea. The variety of recorded behaviors and the large size of data points are unique aspects of this dataset.

3.7 Considerations

Table 7 presents the results obtained by several KT models using the aforementioned datasets. The results are reported using the AUC-ROC curve—a traditional performance measurement for binary classification models. The probabilistic **receiver operating characteristic (ROC)** curve plots the **true positive rate (TPR)** against the **false positive rate (FPR)** while the **area under the curve (AUC)** reports how well a KT model can distinguish between correct and incorrect answers. The

[12] https://github.com/riiid/ednet.
[13] https://company.riiid.co/en/product.

Table 7. AUC Results Reported by Different KT Models

Model	Data Split Ratio (Train-Test)	Dataset									
		ASSISTments				STATICS	Junyi	Synthetic	KDDcup		EdNet
		2009	2012	2015	Chall				Algebra 2005-2006	Bridge 2006-2007	
BKT [25]	80%-20%	0.651 [75] 0.67 [89] 0.6571 [126]	0.623 [75] 0.6204 [126]	0.611 [75] 0.678 [112]	-	-	0.68 [89] 0.831 [112]	0.54 [89]	0.642 [75]	-	0.6027 [126]
IRT [40]	80%-20%	0.7651 [122] 0.6908 [116] 0.751 [75] 0.5869 [118]	0.702 [34] 0.743 [75] 0.6340 [118]	0.672 [75]	-	-	-	-	0.771 [34] 0.806 [75]	0.747 [34] 0.8542 [122]	-
HIRT [122]	80%-20%	0.774 [122]	-	-	-	-	-	-	-	0.8597 [122]	-
TIRT [122]	80%-20%	0.7653 [122]	-	-	-	-	-	-	-	0.8542 [122]	-
AFM [17]	80%-20%	0.6163 [116]	0.610 [34]	-	-	-	-	-	0.707 [34]	0.706 [34]	-
PFA [87]	80%-20%	0.6849 [116] 0.703 [75]	0.670 [75] 0.669 [34]	0.689 [75]	-	-	-	-	0.760 [75] 0.744 [34]	0.746 [34]	-
KTM [116]	80%-20%	0.8186 [116] 0.7169 [126] 0.7425 [118]	0.7535 [118] 0.6788 [126]	-	-	-	-	-	-	-	0.6888 [126]
KTM-DLF [34]	80%-20%		0.756 [34]	-	-	-	-	-	0.837 [34]	0.812 [34]	-
DKT [89]	80%-20%	0.7429 [122] 0.86 [89] 0.8212 [127] 0.721 [75] 0.820 [81] 0.817 [35] 0.709 [80] 0.7561 [126] 0.7515 [118]	0.713 [75] 0.712 [82] 0.7286 [126] 0.7235 [79] 0.7308 [118]	0.707 [75] 0.731 [35] 0.736 [81] 0.7365 [127] 0.727 [112]	0.7263 [35] 0.734 [81] 0.7343 [127]	0.8233 [35] 0.815 [81] 0.8159 [127]	0.814 [82] 0.85 [89] 0.847 [112]	0.8255 [127] 0.823 [81] 0.75 [89]	0.784 [75]	0.751 [80] 0.8901 [122]	0.6822 [126] 0.7638 [102]
DKT+ [127]	80%-20%	0.8227 [127] 0.8024 [35] 0.822 [81]	-	0.7371 [127] 0.728 [112] 0.737 [81] 0.7313 [35]	0.7343 [127] 0.728 [81] 0.7124 [35]	0.8349 [127] 0.835 [81] 0.8301 [35]	0.889 [112]	0.8264 [127] 0.824 [81]	-	-	-
DKT-DSC [75]	80%-20%	0.735 [75]	0.721 [75]	0.716 [75]	0.430 [75]	-	-	-	0.792 [75]	-	-
DKVMN [131]	70%-30%	0.8157 [1] [131]	-	0.7268 [1] [131]	-	0.8284 [1] [131]	0.8027 [1]	0.8273 [1] [131]	-	-	-
SKVMN [1]	70%-30%	0.8363 [1]	-	0.7484 [1]	-	0.8485 [1]	0.8267 [1]	0.840 [1]	-	-	-
SAKT [81]	80%-20%	0.848 [81] 0.752 [35] 0.6860 [118]	0.735 [82] 0.6906 [118]	0.854 [81] 0.7212 [35]	0.734 [81] 0.6569 [35]	0.853 [81] 0.8029 [35]	0.834 [82]	0.832 [81]	-	-	0.7671 [22] 0.7663 [102]
AKT [35]	80%-20%	0.8346 [35] 0.7474 [118]	0.7555 [118]	0.7828 [35]	0.7702 [35]	-	-	-	-	-	-
SAINT [22]	80%-20%	-	-	-	-	-	-	-	-	-	0.7811 [22] 0.7816 [102]
SAINT+ [102]	80%-20%	-	-	-	-	-	-	-	-	-	0.7914 [102]
RKT [82]	80%-20%	-	0.793 [82]	-	-	-	0.860 [82]	-	-	-	-
GKT [80]	80%-20%	0.723 [80]	0.735 [112]	-	-	-	0.893 [112]	-	-	0.769 [80]	-
GIKT [126]	80%-20%	0.7896 [126]	0.7754 [126]	-	-	-	-	-	-	-	0.7523 [126]
SKT [112]	80%-20%	-	-	0.746 [112]	-	-	0.908 [112]	-	-	-	-
DKT+forget [79]	80%-20%	0.754 [118]	0.722 [82] 0.7309 [79] 0.7462 [118]	-	-	-	0.840 [82]	-	-	-	-
HawkesKT [118]	80%-20%	0.763	0.768	-	-	-	-	-	-	-	-
DGMN [3]	70%-30%	86.1 [3]	-	-	-	86.4 [3]	-	85.9 [3]	83.4 [3]	-	-

AUC-ROC ranges from 0 to 1, where 0.5 indicates an uninformative classifier (random guesses) and 1 a perfect classifier.

It is important to note that the results cannot be directly compared if experimental settings are not standardized. As discussed in [122], the results obtained without removing duplicate interactions (preprocessing steps) may be inaccurate and appear elevated. The existence of duplicates is also acknowledged in the KDDcup dataset.[14] We reported in the previous sections the raw number of interactions, students, KCs, and so on, and whenever different, the total after removing duplicates and discarding noise data.

The lack of accurate and descriptive information about each of the attributes in the datasets also hinders experiments. An example is in the ASSISTments datasets where terminology used is confusing[15] or lacking.[16] This may explain the different AUC values reported to the same approach and dataset pairs presented in Table 7.

The sequence of students' interactions is also an important component of a KT task that is affected by the quality of the datasets. Due to noise in the data (e.g., incorrect timestamps, null values, etc.), the sequence of interactions may not be correctly extracted from the datasets which may accordingly impact the performance of the proposed KT models. For example, after inspecting the datasets, we identified timestamp errors in the Algebra 2006 – 2007, which may justify its low adoption in comparison to the other two Algebra datasets.

The fact that the benchmark datasets are often used for multiple tasks other than the KT problem hinders the correct use and interpretation of the datasets in the KT context. The works in [35, 75, 81, 118, 126, 127, 131], for example, report different numbers of knowledge components for the same datasets and the work in [35] discusses how different experimental settings (association between KC and questions) may impact the reported results. The data model and the file format chosen to represent the datasets may also contribute to misinterpretation of the data. For example, the data is often extracted from relational databases and stored in **comma-separated values (CSV)**, compressing information in a single attribute using non-standard characters, e.g., the KDDcup dataset uses double tilde characters to assign multiple KCs to a question. Given the hierarchical and relational nature of the data, XML and JSON are more suitable and explicit formats.

Another critical aspect is the lack of more diverse datasets. The existing public datasets are often from a specific domain (mainly math) and a specific region (e.g., the ASSISTment datasets from the US, and EdNet, the most recent publicly available dataset, from South Korea). Most of the datasets do not provide demographic information, and therefore gender-based, or other similar predictions cannot be performed.

Finally, the benchmark datasets have been updated over the years, and the version used by the KT models is not readily identifiable, which can compromise their direct comparison. A version control mechanism for the publicly available datasets would help keep track of every change made to a dataset over time and allows for consistent and comparable results.

4 KNOWLEDGE TRACING APPLICATIONS

This section explores possible application areas that can benefit from KT models. We broadly divide the KT application areas into the following four categories: (i) Educational Recommender Systems; (ii) Learning Provision and Quality Assurance; (iii) Student Engagement via Interactive Learning; and (iv) Learning to Teach. We first introduce each application area. Then we discuss the conditions required by methods in different KT categories and their suitable application scenarios.

[14] https://pslcdatashop.web.cmu.edu/KDDCup/FAQ/.
[15] https://sites.google.com/site/assistmentsdata/home/2012-13-school-data-with-affect.
[16] https://sites.google.com/site/assistmentsdata/home/2015-assistments-skill-builder-data.

4.1 Educational Recommender Systems

An application area of KT that comes directly to mind is online education systems where the main objective is to provide effective learning experience for their students. Tracing the knowledge state of a student would facilitate to tailor students' learning experience according to their capabilities and skills. This can be achieved through recommending learning materials (e.g., lectures, labs, and/or exercises) based on the learnt knowledge state of a student. Thus, the aim of such online education systems is twofold: (1) to estimate the knowledge state of a student using a KT model; and (2) to recommend learning materials conditioned on the knowledge state using a recommendation model [14]. Below are some examples of applications that have been recently studied. A graph-based recommendation method has been proposed by Chanaa and Faddouli [18] to aid an educational instructor to segment students into groups based on their knowledge states and recommend the most appropriate kinds of exercises for each group. More specifically, the instructor first selects a specific **knowledge component (KC)**, and then, the model constructs a dynamic knowledge graph based on historical practice information that includes student knowledge vectors as nodes and uses edges for representing their mastery level similarities around the selected KC. Such a graph is clustered into node groups and for each group a shared embedding is constructed using GNNs to get the final recommendations. Cai et al. [15] followed an interactive recommendation approach in which a reinforcement learning recommender agent selects learning materials to recommend based on a reward signal calculated from the progress in a knowledge state estimated by a KT model. Huang et al. [49] proposed an interactive educational video recommender model that follows a multi-objective setup of rewards. The authors designed three reward functions to reflect three main aspects in online education systems including a reviewing reward function for recommending videos about KCs that a student did not perform well previously, a smoothing reward function for recommending videos with a gradual difficulty to understand, and an engagement reward function for recommending videos about KCs a student started to master recently. The recommender agent followed a reinforcement learning design with a state estimated by a DKT variant model, an action for the video id to recommend, and a combined reward by using a weighted sum of the three reward functions.

4.2 Learning Provision and Quality Assurance

Another potential application area is to provision a learning curriculum (e.g., an ordered list of topics to study) for a specific subject based on the knowledge state of a student. One direction [25, 97] in this application area follows a semi-automated approach in which an instructor would conduct simulations using a KT model trained on the historical exercise records of students to identify a suitable curriculum of learning materials for a course to maximize the knowledge gain of students. Another direction [83] uses KT models to assess the effectiveness of a course structure in achieving its targeted objectives by assessing the impact of each module (i.e., a collection of learning materials) on the knowledge growth of students. Recently, deep KT models have been adopted in this direction to provide quality assurance for the course design [66]. The authors used a DKT model [89] to trace the progress in the knowledge state of a student after taking a specific course module, and then, an actor-critic reinforcement learning agent [41] considers the knowledge state across different course modules and their predefined relationships (i.e., prerequisites) and takes an action to select the next module to work on for the student to maximize their knowledge gain in achieving course objectives. Following this approach, the structure of a course could adapt dynamically to a student's needs and skills instead of having one fixed structure that does not fit all students.

4.3 Student Engagement via Interactive Learning

Interactive education aims at making the learning process more exciting and engaging through delivering knowledge components into a gaming shell. Cognitive studies [44] showed that students can easily gain new knowledge components and be able to spend more time on learning if the learning materials were delivered in an engaging manner such as gaming. This is due to the nature of the human mind that was mainly designed for learning by real-world practices that usually come as a form of an interactive experience (i.e., state, action, and rewards). Thus, an educational game can provide similar experiences that better align with our natural learning capabilities than the conventional ways (e.g., textbooks and lectures). Long and Aleven [70] evaluated the effect of educational games in comparison to conventional online tutoring systems through a study that involved two groups of students: one group was learning on an educational game for math equation solving and the other group was learning using a non-interactive system that presents math concepts in a conventional manner through demonstrative examples and exercises. The study found that the group using the educational game was more engaged to continue the learning process in comparison to the other group. Another study [4] focused on the effect of mobile educational games on the learning progress for elementary school kids. The authors divided the student into two groups: one used mobile educational games to review and practice math concepts taught at school and another group practiced the math concepts using conventional text exercises. The study concluded that students with access to the mobile educational games were performing better in retaining the math concepts in comparison to the other group. These findings demonstrate the great potential of education games as a promising application area for KT models.

Central to the effectiveness of an educational game is to assess the knowledge progress of a player and adjust the gaming experience accordingly. This might include adjusting the difficulty of challenges, opening new part(s) in a game, or the competency of a computer opponent. Kantharaju et al. [53] proposed a KT model that could detect when a player attempts a specific skill in an educational game and quantify their knowledge state across different states, so the game experience could be focused on challenging skills for each player. Cui et al. [26] used the BKT [117] model to trace the knowledge state of fifth-grade elementary school students in Canada during a science gaming assessment. The authors were not only able to effectively predict the final score of a student based on their partial observations from the game assessment, but also identify pitfalls in the game design and assumptions that tend not to work as expected by the designer in terms of assessing dedicated skills. Hooshyar et al. [46] proposed a method that tackles the problem of predicting a player's skill proficiency in education games over multiple tasks with overlapping demonstrations between skills. The authors investigated deep knowledge tracing using different sequence models including LSTMs, RNNs, and CNNs while decomposing temporal dynamics using a moving fixed-size time window. Their results showed that the CNN variant outperformed others with classification accuracy of 85% on average. Recently, several studies have shown that psycho-physiological signals could provide vital cues on the current engagement state of a student, especially in virtual learning sessions [12, 113]. These studies use recent advancements in computer vision to detect facial expressions and body poses with a web camera attached to a student's computer for estimating an engagement score. We note that such signals could work as an effective auxiliary source of information for knowledge tracing models besides the question answering history to better estimate the knowledge state of a student.

4.4 Learning to Teach

Going beyond the conventional assumption of a human student in a KT setting opens the door to a wide range of application areas. Virtual students, such as intelligent agents which adopt a

reinforcement learning setup or machine learning models, can be treated in a similar way as real-life students who are in need to learn a set of skills in different machine learning tasks. For example, this can be a deep neural network model that needs to master a skill of classifying different class labels (e.g., cats, dogs, furniture, etc.) in an image classification task or a reinforcement learning agent aiming at mastering different skills in an Atari game. **Curriculum learning (CL)** [11] aims at learning a curriculum of tasks to enable a student agent from mastering a set of skills. A CL policy would imply a statistical distribution on learning tasks that gradually drive the student agent towards convergence. Another relevant paradigm is **machine teaching (MT)** [133] which aims at minimizing the teaching cost represented by the size of training samples drawn from training data in a machine learning scenario. In MT, there are two models being included: the teacher model and the student model. The former targets to sample training data for the latter to learn an optimal parameter set θ^* the minimizes the loss function in the task. **Learning to teach (L2T)** [31, 124] targeted customizing the learning process for a student agent/model through optimizing three main aspects including training data sampling, neural architecture design, and loss function design. In L2T, a teacher agent follows a reinforcement learning approach to optimize a teaching policy that handles one or more of the three aspects previously mentioned. It can be observed that a shared characteristic across these different attempts to enhance the conventional machine learning procedure is the need to trace the knowledge state of a student model. Thus, there is a significant potential for KT models to contribute in this application area by tracing the knowledge state of a student model during training procedure. The output of the KT model would form the input/state of a teacher model that aims at customizing the training procedure to speedup the student model's convergence. **Knowledge Augmented Data Teaching (KADT)** [2] aims at improving a data teaching strategy of a student ML model by tracking its knowledge progress across multiple knowledge components in a learning task. The KADT method includes a knowledge tracking model to dynamically capture the knowledge progression of the student model in terms of latent knowledge components. The authors develop an attention-pooling mechanism to extract knowledge representations of the student model with respect to class labels, which enables the development of a data-teaching strategy on significant training samples. The authors evaluated the performance of the KADT method on four different machine learning tasks including knowledge tracing, sentiment analysis, movie recommendation, and image classification. Results compared to the state-of-the-art machine teaching methods have been empirically proven that KADT consistently outperforms the others in all tasks.

4.5 Discussion

We have introduced different categories for KT methods in Section 2, yet it is essential to understand the assumptions underlying each of them when choosing KT methods for applications. Thus, in the following, we discuss the impact of design assumptions for each KT category on applications such training data and computational resources.

Starting with traditional KT methods such as BKT and its variants [117], we note that these traditional KT methods have a simplified view of the KT problem including assuming the knowledge state to be a binary random variable or having a single KC for each question, which is mainly to keep the posterior computation tractable. IRT and factor analysis methods [30] assume prior learning ability information for each student to be available in addition to an exercise answering history, and this limits their usage in scenarios where such information is missing such as cold-start ones. Accordingly, these methods are more suitable for simple application scenarios that deal with primitive skills (e.g., early learning problems) and do not include multiple KCs in the learning process. Nonetheless, these methods require less computational overhead in comparison to advanced

categories (e.g., deep KT methods) and are self-explainable through the learned state space transition probabilities.

The early work of deep KT methods [89] primarily uses conventional neural sequence models such as RNN or LSTM cells, which limits their ability to model multiple KC relationships and long-term dependencies in a exercise answering sequence. Thus, these deep sequence KT methods are suitable for application scenarios were the number of involved KCs is limited and exercise answering sequences are relatively short. Memory-augmented deep KT methods overcome the limitations of these deep sequence KT category by facilitating to model multiple KC relationships and longer sequence dependencies via a memory mechanism, yet this comes with an additional memory cost to store memory structures. As a result, this category is suitable where memory and computational resources are available. Attention KT methods provide another alternative for modelling long-term dependencies in an exercise answering sequence with a significant computational overhead to execute self-attention computation in quadratic time. This overhead is further magnified when using multiple attention heads. The added performance value of this category is justifiable given the need to deal with long-term dependencies with multiple KC relationships and the availability of extended computing resources such as multiple GPU units that could distribute attention head computations in parallel. Graph-based KT methods have the capacity to capture complicated question and KC relationships, yet, the majority of these methods assumes that graph data exists in prior, which however cannot be guaranteed in many real-world application scenarios. There are automated techniques [80] being studied to learn graph structure directly from exercise answering sequences with an additional computational cost. Finally, text-aware deep KT methods assume the existence of text data for questions and KC tags in order to learn an effective embeddings. Nevertheless, some KT datasets do not provide text tags for questions and/or KCs, which will limit the usage of these methods in such application scenarios.

5 KNOWLEDGE TRACING FUTURE RESEARCH DIRECTIONS

Despite the promising results achieved by state-of-the-art KT models, limitations and gaps of current approaches and available datasets open up several opportunities for future research.

Multimodal and informative representation learning & datasets. The choice of data representation directly impacts the performance of any machine learning model [10]. KT models tend to learn embedding representations for questions and KCs from abstract formats such as one-hot encoding; however, some data in the description of a question such as images and mathematical equations that can lead to more informative embedding representations, are overlooked, either by the proposed models or by the available datasets. This opens up research directions through the following questions: (1) What information data can be used to improve the performance of KT models? (2) How to represent such data for the KT tasks? (3) How to create a dataset for the KT tasks that enables a more informative embedding representation learning? The **Exercise-aware Knowledge Tracing (EKT)** approach proposed by Liu et al. [64] is a recent attempt to learn richer embedding representations, taking into account textual context and relationships between questions. However, despite previous efforts, the representations of multimodal and domain-specific data such as mathematical equations and code snippets remain mostly unexplored in the literature resulting in low-informative representation learning for KT models. Fusing signals from multiple feature spaces may enable better representation learning in addition to mitigating noise in data [8].

Self-supervised learning in knowledge tracing. Although supervised learning has led to advances in different areas, it still has a major drawback: the need for large, high-quality labeled data for training. **Self-supervised learning (SSL)** [76, 130], on the other hand, has proven to be effective in several areas (e.g., natural language processing [28] and computer vision [36]) learning from

unlabeled data. SSL often adopts similarity ranking loss functions (e.g., contrastive loss [119]) in a process called *pre-training* or *pretext task* [76] to automatically generate labels. Such pre-trained models can thus be transferred to a downstream task to train in a supervised learning manner with a limited amount of labeled data. Along with SSL, therefore, further contributions to the KT field can be made, for example, by creating pre-trained models (e.g., using existing pre-trained language and computer vision models) to generate informative representations for KT; and investigating how it can mitigate limited training of students' activities in cases of a cold-start scenario or skewed participation data.

Interactive knowledge tracing. Most KT models adopt a passive approach of observing question answering response history to estimate students' knowledge states; however, interactive methods, driven by question answering response behavior, are still unexplored. Interactive methods are particularly useful in cold start scenarios where an interactive approach can reveal students' knowledge states by directly asking questions related to different KCs. Thus, another potential future work is to develop optimized question sampling policies to enhance the performance of KT models in, but not limited to, cold start scenarios. **Reinforcement Learning (RL)** [106] approaches are a possible natural choice given its maximal rewarding scheme.

Last but not least, considering that KT involves human knowledge and learning, transparency in the logic and results obtained by KT models would benefit educational stakeholders and processes. This leads us to investigating **eXplainable Artificial Intelligence (XAI)**, as seen in other research fields such as [50, 51]. Potential research avenues include the development of techniques to understand and explain the prediction process in KT models; and how algorithmic decisions make impacts on learning processes, course design, instructor performance, the quality of learning materials, and student engagement. Promising research to explain deep learning models has been carried out using knowledge distillation [42] to understand and explain predictions in other models.

6 CONCLUSION

In this work, we presented a comprehensive survey for knowledge tracing. We identified four research questions to guide our survey agenda. To answer the first question, we proposed a categorization that divides knowledge tracing methods into two broad categories including traditional methods covering Bayesian and factor model approaches, and deep learning methods covering recent state-of-the-art techniques including recurrent, memory, attention, and graph approaches. For each category, we highlighted the main characteristics including assumptions, governing factors for applications, and their strengths and weaknesses. The second question targeted the identification of existent knowledge tracing datasets, their characteristics, and the latest performance results on each of them. In answering this question, we proposed a comprehensive review for the key datasets utilized by the knowledge tracing literature along with the main characteristics of each dataset. Moreover, we summarized all the reported performance results on each dataset in one table to provide a holistic view of the recent state. The third question aimed at identifying the possible application areas for knowledge tracing methods and how could they benefit other related fields. To approach this question, we identified four application areas and thoroughly discussed each of them to elaborate the potential of knowledge tracing methods. Our fourth research question was about the possible future research directions in the knowledge tracing field. We answered this question by exploring three directions for future research including: (1) the use of multimodal information to enhance the situation awareness of knowledge tracing models, (2) self-supervised learning techniques to harness the potential of unlabeled data and deal with limited training data

scenarios, and (3) interactive and explainable techniques for knowledge tracing that could provide a better visibility of predictions and recommendations coming from knowledge tracing models.

REFERENCES

[1] Ghodai Abdelrahman and Qing Wang. 2019. Knowledge tracing with sequential key-value memory networks. In *SIGIR*. 175–184.

[2] Ghodai Abdelrahman and Qing Wang. 2021. Learning data teaching strategies via knowledge tracing. *arXiv preprint arXiv:2111.07083* (2021).

[3] Ghodai Abdelrahman and Qing Wang. 2022. Deep graph memory networks for forgetting-robust knowledge tracing. *TKDE* (2022).

[4] Mohammad Ahmad Al Khateeb. 2019. Effect of mobile gaming on mathematical achievement among 4th graders. *IJET* 14 (2019).

[5] John R. Anderson et al. 1986. Cognitive modelling and intelligent tutoring. (1986).

[6] John R. Anderson, C. Franklin Boyle, Albert T. Corbett, and Matthew W. Lewis. 1990. Cognitive modeling and intelligent tutoring. *Artificial Intelligence* 42 (1990), 7–49.

[7] David Andrich. 1981. Book review: Probabilistic models for some intelligence and attainment tests (expanded edition: Georg Rasch Chicago: The University of Chicago Press, 1980. *Applied Psychological Measurement* 5 (1981), 545–550.

[8] Tadas Baltrusaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2019. Multimodal machine learning: A survey and taxonomy. *IEEE Trans. Pattern Anal. Mach. Intell.* 41, 2 (2019), 423–443.

[9] Mark A. Barton and Frederic M. Lord. 1981. An upper asymptote for the three-parameter logistic item-response model. *ETS Research Report Series* 1981 (1981), i–8.

[10] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (2013), 1798–1828.

[11] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*. 41–48.

[12] Prakhar Bhardwaj, P. K. Gupta, Harsh Panwar, Mohammad Khubeb Siddiqui, Ruben Morales-Menendez, and Anubha Bhaik. 2021. Application of deep learning on student engagement in e-learning environments. *Computers & Electrical Engineering* 93 (2021), 107277.

[13] Allan Birnbaum. 1969. Statistical theory for logistic mental test models with a prior distribution of ability. *Journal of Mathematical Psychology* 6 (1969), 258–276.

[14] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. 2013. Recommender systems survey. *Knowledge-Based Systems* 46 (2013), 109–132.

[15] Dejun Cai, Yuan Zhang, and Bintao Dai. 2019. Learning path recommendation based on knowledge tracing model and reinforcement learning. In *ICCC*. 1881–1885.

[16] Hao Cen, Kenneth R. Koedinger, and Brian Junker. 2006. Learning factors analysis - A general method for cognitive model evaluation and improvement. In *ITS*, Vol. 4053. 164–175.

[17] Hao Cen, Kenneth R. Koedinger, and Brian Junker. 2008. Comparing two IRT models for conjunctive skills. In *ITS*, Vol. 5091. 796–798.

[18] Abdessamad Chanaa, El Faddouli, et al. 2020. Predicting learners need for recommendation using dynamic graph-based knowledge tracing. In *AIED*. 49–53.

[19] Haw-Shiuan Chang, Hwai-Jung Hsu, and Kuan-Ta Chen. 2015. Modeling exercise relationships in e-learning: A unified approach. In *EDM*. 532–535.

[20] Yuying Chen, Qi Liu, Zhenya Huang, Le Wu, Enhong Chen, Run-ze Wu, Yu Su, and Guoping Hu. 2017. Tracking knowledge proficiency of students with educational priors. In *CIKM*. 989–998.

[21] Song Cheng, Qi Liu, Enhong Chen, Kai Zhang, Zhenya Huang, Yu Yin, Xiaoqing Huang, and Yu Su. 2022. AdaptKT: A domain adaptable method for knowledge tracing. In *WSDM*. 123–131.

[22] Youngduck Choi, Youngnam Lee, Junghyun Cho, Jineon Baek, Byungsoo Kim, Yeongmin Cha, Dongmin Shin, Chan Bae, and Jaewe Heo. 2020. Towards an appropriate query, key, and value computation for knowledge tracing. In *L@S*. 341–344.

[23] Youngduck Choi, Youngnam Lee, Dongmin Shin, Junghyun Cho, Seoyon Park, Seewoo Lee, Jineon Baek, Chan Bae, Byungsoo Kim, and Jaewe Heo. 2020. EdNet: A large-scale hierarchical dataset in education. In *AIED*, Vol. 12164. 69–73.

[24] Albert Corbett. 2000. Cognitive mastery learning in the act programming tutor. In *Adaptive User Interfaces AAAI*.

[25] Albert T. Corbett and John R. Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *UMUAI* 4 (1994), 253–278.

[26] Yang Cui, Man-Wai Chu, and Fu Chen. 2019. Analyzing student process data in game-based assessments with Bayesian knowledge tracing and dynamic Bayesian networks. *JEDM* 11 (2019), 80–100.

[27] Ryan Shaun Joazeiro de Baker, Albert T. Corbett, and Vincent Aleven. 2008. More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian knowledge tracing. In *ITS*, Vol. 5091. 406–415.
[28] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*. 4171–4186.
[29] Hermann Ebbinghaus. 2013. Memory: A contribution to experimental psychology. *Annals of Neurosciences* 20 (2013), 155.
[30] Susan E. Embretson and Steven P. Reise. 2013. *Item Response Theory*. Psychology Press.
[31] Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. 2018. Learning to teach. In *ICLR*.
[32] Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *UMUAI* 19 (2009), 243–266.
[33] Garett C. Foster, Hanyi Min, and Michael J. Zickar. 2017. Review of item response theory practices in organizational research: Lessons learned and paths forward. *Organizational Research Methods* 20 (2017), 465–486.
[34] Wenbin Gan, Yuan Sun, Xian Peng, and Yi Sun. 2020. Modeling learner's dynamic knowledge construction procedure and cognitive item difficulty for knowledge tracing. *Applied Intelligence* 50 (2020), 3894–3912.
[35] Aritra Ghosh, Neil Heffernan, and Andrew S. Lan. 2020. Context-aware attentive knowledge tracing. In *SIGKDD*. 2330–2339.
[36] Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulin, and Piotr Bojanowski. 2021. Self-supervised pretraining of visual features in the wild. *arXiv preprint arXiv:2103.01988* (2021).
[37] Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850* (2013).
[38] A. Graves, A. Mohamed, and G. Hinton. 2013. Speech recognition with deep recurrent neural networks. In *ICASSP*. 6645–6649.
[39] Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401* (2014).
[40] Bert F. Green. 1951. A general solution for the latent class model of latent structure analysis. *Psychometrika* 16 (1951), 151–166.
[41] Ivo Grondman, Lucian Busoniu, Gabriel A. D. Lopes, and Robert Babuska. 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42 (2012), 1291–1307.
[42] Anselm Haselhoff, Jan Kronenberger, Fabian Kuppers, and Jonas Schneider. 2021. Towards black-box explainability with Gaussian discriminant knowledge distillation. In *CVPR*. 21–28.
[43] Alan G. Hawkes. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58, 1 (1971), 83–90.
[44] Anja Hawlitschek and Sven Joeckel. 2017. Increasing the effectiveness of digital educational games: The effects of a learning instruction on students' learning, motivation and cognitive load. *Computers in Human Behavior* 72 (2017), 79–86.
[45] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9 (1997), 1735–1780.
[46] Danial Hooshyar, Yueh-Min Huang, and Yeongwook Yang. 2022. GameDKT: Deep knowledge tracing in educational games. *Expert Systems with Applications* 196 (2022), 116670.
[47] Yun Huang, José P. González-Brenes, and Peter Brusilovsky. 2014. General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge. In *EDM*. 84–91.
[48] Zhenya Huang, Qi Liu, Yuying Chen, Le Wu, Keli Xiao, Enhong Chen, Haiping Ma, and Guoping Hu. 2020. Learning or forgetting? A dynamic approach for tracking the knowledge proficiency of students. *TOIS* 38 (2020), 1–33.
[49] Zhenya Huang, Qi Liu, Chengxiang Zhai, Yu Yin, Enhong Chen, Weibo Gao, and Guoping Hu. 2019. Exploring multi-objective exercise recommendations in online education systems. In *CIKM*. 1261–1270.
[50] Alexey Ignatiev. 2020. Towards trustable explainable AI. In *IJCAI*. 5154–5158.
[51] Sheikh Rabiul Islam, William Eberle, and Sheikh K. Ghafoor. 2020. Towards quantification of explainability in explainable artificial intelligence methods. In *AAAI*. 75–81.
[52] Philip I. Pavlik Jr. and John R. Anderson. 2005. Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. *Cognitive Science* 29 (2005), 559–586.
[53] Pavan Kantharaju, Katelyn Bright Alderfer, Jichen Zhu, Bruce Char, Brian K. Smith, and Santiago Ontañón. 2018. Tracing player knowledge in a parallel programming educational game. In *AAAI*. 173–179.
[54] Tanja Käser, Severin Klingler, Alexander G. Schwing, and Markus H. Gross. 2017. Dynamic Bayesian networks for student modeling. *IEEE Trans. Learn. Technol.* 10 (2017), 450–462.
[55] Mohammad Khajah, Robert V. Lindsey, and Michael Mozer. 2016. How deep is knowledge tracing? In *EDM*.
[56] Mohammad Khajah, Rowan Wing, Robert Lindsey, and Michael Mozer. 2014. Integrating latent-factor and knowledge-tracing models to predict individual differences in learning. In *EDM*. 99–106.

[57] Mohammad M. Khajah, Yun Huang, José P. González-Brenes, Michael C. Mozer, and Peter Brusilovsky. 2014. Integrating knowledge tracing and item response theory: A tale of two frameworks. In *UMAP*, Vol. 1181.
[58] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*. OpenReview.net.
[59] Serkan Kiranyaz, Turker Ince, Osama Abdeljaber, Onur Avci, and Moncef Gabbouj. 2019. 1-D convolutional neural networks for signal processing applications. In *ICASSP*. 8360–8364.
[60] Kenneth R. Koedinger, Ryan S. J. d. 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.
[61] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521 (2015), 436.
[62] Jung In Lee and Emma Brunskill. 2012. The impact on individualizing student models on necessary practice opportunities. In *EDM*. 118–125.
[63] Zachary C. Lipton, John Berkowitz, and Charles Elkan. 2015. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019* (2015).
[64] Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, and Guoping Hu. 2021. EKT: Exercise-aware knowledge tracing for student performance prediction. *TKDE* 33 (2021), 100–115.
[65] Qi Liu, Shuanghong Shen, Zhenya Huang, Enhong Chen, and Yonghe Zheng. 2021. A survey of knowledge tracing. *arXiv preprint arXiv:2105.15106* (2021).
[66] Qi Liu, Shiwei Tong, Chuanren Liu, Hongke Zhao, Enhong Chen, Haiping Ma, and Shijin Wang. 2019. Exploiting cognitive structure for adaptive learning. In *SIGKDD*. 627–635.
[67] Yunfei Liu, Yang Yang, Xianyu Chen, Jian Shen, Haifeng Zhang, and Yong Yu. 2021. Improving knowledge tracing via pre-training question embeddings. In *IJCAI*.
[68] Ting Long, Jiarui Qin, Jian Shen, Weinan Zhang, Wei Xia, Ruiming Tang, Xiuqiang He, and Yong Yu. 2022. Improving knowledge tracing with collaborative information. In *WSDM*. 599–607.
[69] Yanjin Long and Vincent Aleven. 2017. Educational game and intelligent tutoring system: A classroom study and comparative design analysis. *TOCHI* 24 (2017), 1–27.
[70] Yanjin Long and Vincent Aleven. 2017. Educational game and intelligent tutoring system: A classroom study and comparative design analysis. *ACM Trans. Comput.-Hum. Interact.* 24 (2017), 27 pages.
[71] F. M. Lord, M. R. Novick, and Allan Birnbaum. 1968. *Statistical Theories of Mental Test Scores*. Addison-Wesley.
[72] Frederic M. Lord. 1951. A theory of test scores and their relation to the trait measured. *ETS Research Bulletin Series* 1951 (1951), i–126.
[73] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.
[74] Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *EMNLP*. 1400–1409.
[75] Sein Minn, Yi Yu, Michel C. Desmarais, Feida Zhu, and Jill-Jênn Vie. 2018. Deep knowledge tracing and dynamic student classification for knowledge tracing. In *ICDM*. 1182–1187.
[76] Ishan Misra and Laurens van der Maaten. 2020. Self-supervised learning of pretext-invariant representations. In *CVPR*.
[77] R. Charles Murray, Steven Ritter, Tristan Nixon, Ryan Schwiebert, Robert G. M. Hausmann, Brendon Towle, Stephen E. Fancsali, and Annalies Vuong. 2013. Revealing the learning in learning curves. In *AIED*. 473–482.
[78] Jaap M. J. Murre and Joeri Dros. 2015. Replication and analysis of Ebbinghaus' forgetting curve. *PLOS ONE* 10 (2015), 1–23.
[79] Koki Nagatani, Qian Zhang, Masahiro Sato, Yan-Ying Chen, Francine Chen, and Tomoko Ohkuma. 2019. Augmenting knowledge tracing by considering forgetting behavior. In *WWW*. 3101–3107.
[80] Hiromi Nakagawa, Yusuke Iwasawa, and Yutaka Matsuo. 2019. Graph-based knowledge tracing: Modeling student proficiency using graph neural network. In *WIC*. Association for Computing Machinery, New York, NY, USA, 156–163.
[81] Shalini Pandey and George Karypis. 2019. A self attentive model for knowledge tracing. In *EDM*.
[82] Shalini Pandey and Jaideep Srivastava. 2020. RKT: Relation-aware self-attention for knowledge tracing. In *CIKM*. 1205–1214.
[83] Zachary A. Pardos, Yoav Bergner, Daniel T. Seaton, and David E. Pritchard. 2013. Adapting Bayesian knowledge tracing to a massive open online course in edX. In *EDM*.
[84] Zachary A. Pardos, Ryan S. J. D. Baker, Maria O. C. Z. San Pedro, Sujith M. Gowda, and Supreeth M. Gowda. 2014. Affective states and state tests: Investigating how affect and engagement during the school year predict end-of-year learning outcomes. *Journal of Learning Analytics* 1 (2014), 107–128.

[85] Zachary A. Pardos and Neil T. Heffernan. 2010. Modeling individualization in a Bayesian networks implementation of knowledge tracing. In *UMAP*. 255–266.
[86] Zachary A. Pardos and Neil T. Heffernan. 2011. KT-IDEM: Introducing item difficulty to the knowledge tracing model. In *UMAP*. 243–254.
[87] Philip I. Pavlik, Hao Cen, and Kenneth R. Koedinger. 2009. Performance factors analysis - A new alternative to knowledge tracing. In *AIED*, Vol. 200. 531–538.
[88] Radek Pelánek. 2015. Modeling students' memory for application in adaptive educational systems. In *EDM*. 480–483.
[89] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J. Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. In *NeurIPS*. 505–513.
[90] Joseph Psotka, Leonard Daniel Massey, and Sharon A. Mutter. 1988. *Intelligent Tutoring Systems: Lessons Learned*. Psychology Press.
[91] Yumeng Qiu, Yingmei Qi, Hanyuan Lu, Zachary A. Pardos, and Neil T. Heffernan. 2011. Does time matter? Modeling the effect of time with Bayesian knowledge tracing. In *EDM*. 139–148.
[92] Georg Rasch. 1993. *Probabilistic Models for Some Intelligence and Attainment Tests*. ERIC.
[93] Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.* 3 (2009), 333–389.
[94] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks* 20 (2009), 61–80.
[95] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks* 20 (2009), 61–80.
[96] Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural Networks* 61 (2015), 85–117.
[97] Thorsten Schodde, Kirsten Bergmann, and Stefan Kopp. 2017. Adaptive robot language tutoring based on Bayesian knowledge tracing and predictive decision-making. In *HRI*. 128–136.
[98] Florian Sense, Friederike Behrens, Rob R. Meijer, and Hedderik van Rijn. 2016. An individual's rate of forgetting is stable over time but differs across materials. *Topics in Cognitive Science* 8 (2016), 305–321.
[99] Burr Settles and Brendan Meeder. 2016. A trainable spaced repetition model for language learning. In *ACL*. 1848–1858.
[100] Shuanghong Shen, Qi Liu, Enhong Chen, Zhenya Huang, Wei Huang, Yu Yin, Yu Su, and Shijin Wang. 2021. Learning process-consistent knowledge tracing. In *SIGKDD*. 1452–1460.
[101] Shuanghong Shen, Qi Liu, Enhong Chen, Han Wu, Zhenya Huang, Weihao Zhao, Yu Su, Haiping Ma, and Shijin Wang. 2020. Convolutional knowledge tracing: Modeling individualization in student learning process. In *SIGIR*. 1857–1860.
[102] Dongmin Shin, Yugeun Shim, Hangyeol Yu, Seewoo Lee, Byungsoo Kim, and Youngduck Choi. 2021. SAINT+: Integrating temporal features for EdNet correctness prediction. In *LAK*. 490–496.
[103] J. Stamper, A. Niculescu-Mizil, S. Ritter, G. J. Gordon, and K. R. Koedinger. 2010. Algebra i 2008–2009. Challenge data set from KDD Cup 2010 educational data mining challenge. *Retrieved April* 25 (2010), 2015.
[104] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris H. Q. Ding, Si Wei, and Guoping Hu. 2018. Exercise-enhanced sequential modeling for student performance prediction. In *AAAI*. 2435–2443.
[105] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NeurIPS*. 3104–3112.
[106] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. MIT Press.
[107] Mack Sweeney, Jaime Lester, Huzefa Rangwala, and Aditya Johri. 2016. Next-term student performance prediction: A recommender systems approach. In *EDM*. 7.
[108] Tamás Szabados. 2010. An elementary introduction to the Wiener process and stochastic integrals. *arXiv preprint arXiv:1008.1510* (2010).
[109] Nguyen Thai-Nghe, Lucas Drumond, Tomás Horváth, and Lars Schmidt-Thieme. 2012. Using factorization machines for student modeling. In *UMAP*, Vol. 872.
[110] Ilya O. Tolstikhin, Bharath K. Sriperumbudur, and Bernhard Schölkopf. 2016. Minimax estimation of maximum mean discrepancy with radial kernels. In *NeurIPS*. 1930–1938.
[111] Hanshuang Tong, Zhen Wang, Yun Zhou, Shiwei Tong, Wenyuan Han, and Qi Liu. 2022. Introducing problem schema with hierarchical exercise graph for knowledge tracing. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'22)*. (2022), 405–415.
[112] Shiwei Tong, Qi Liu, Wei Huang, Zhenya Huang, Enhong Chen, Chuanren Liu, Haiping Ma, and Shijin Wang. 2020. Structure-based knowledge tracing: An influence propagation view. In *ICDM*. 541–550.
[113] Pieter Vanneste, José Oramas, Thomas Verelst, Tinne Tuytelaars, Annelies Raes, Fien Depaepe, and Wim Van den Noortgate. 2021. Computer vision and human behaviour, emotion and cognition detection: A use case on student engagement. *Mathematics* 9 (2021).

[114] Moshe Y. Vardi. 2012. Will MOOCs destroy academia? *Commun. ACM* 55 (2012), 5–5.
[115] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*. 6000–6010.
[116] Jill-Jênn Vie and Hisashi Kashima. 2019. Knowledge tracing machines: Factorization machines for knowledge tracing. In *AAAI*. 750–757.
[117] Michael Villano. 1992. Probabilistic student models: Bayesian belief networks and knowledge space theory. In *ITS*, Vol. 608. 491–498.
[118] Chenyang Wang, Weizhi Ma, Min Zhang, Chuancheng Lv, Fengyuan Wan, Huijie Lin, Taoran Tang, Yiqun Liu, and Shaoping Ma. 2021. Temporal cross-effects in knowledge tracing. In *WSDM*. 517–525.
[119] Feng Wang and Huaping Liu. 2021. Understanding the behaviour of contrastive loss. In *CVPR*. 2495–2504.
[120] 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 *AAAI*. 6153–6161.
[121] Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-learning. *Machine Learning* 8 (1992), 279–292.
[122] Kevin H. Wilson, Yan Karklin, Bojian Han, and Chaitanya Ekanadham. 2016. Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. In *EDM*. 539–544.
[123] Raymond E. Wright. 1995. Logistic regression. In *Reading and Understanding Multivariate Statistics*. 217–244.
[124] Lijun Wu, Fei Tian, Yingce Xia, Yang Fan, Tao Qin, Jian-Huang Lai, and Tie-Yan Liu. 2018. Learning to teach with dynamic loss functions. In *NeurIPS*. 6467–6478.
[125] Xiaolu Xiong, Siyuan Zhao, Eric Van Inwegen, and Joseph Beck. 2016. Going deeper with deep knowledge tracing. In *EDM*. 545–550.
[126] Yang Yang, Jian Shen, Yanru Qu, Yunfei Liu, Kerong Wang, Yaoming Zhu, Weinan Zhang, and Yong Yu. 2020. GIKT: A graph-based interaction model for knowledge tracing. 12457 (2020), 299–315.
[127] Chun-Kit Yeung and Dit-Yan Yeung. 2018. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *L@S*. 1–10.
[128] Yu Yin, Qi Liu, Zhenya Huang, Enhong Chen, Wei Tong, Shijin Wang, and Yu Su. 2019. QuesNet: A unified representation for heterogeneous test questions. In *SIGKDD*. 1328–1336.
[129] Michael V. Yudelson, Kenneth R. Koedinger, and Geoffrey J. Gordon. 2013. Individualized Bayesian knowledge tracing models. In *AIED*. 171–180.
[130] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. 2019. S4L: Self-supervised semi-supervised learning. In *ICCV*.
[131] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic key-value memory networks for knowledge tracing. In *WWW*. 765–774.
[132] Yuqiang Zhou, Qi Liu, Jinze Wu, Fei Wang, Zhenya Huang, Wei Tong, Hui Xiong, Enhong Chen, and Jianhui Ma. 2021. Modeling context-aware features for cognitive diagnosis in student learning. In *SIGKDD*. 2420–2428.
[133] Xiaojin Zhu. 2015. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *AAAI*. 4083–4087.
[134] Yan Zhuang, Qi Liu, Zhenya Huang, Zhi Li, Shuanghong Shen, and Haiping Ma. 2022. Fully adaptive framework: Neural computerized adaptive testing for online education. In *Proceedings of the AAAI Conference on Artificial Intelligence*. (2022).

Received 8 January 2022; revised 11 October 2022; accepted 17 October 2022