When is Deep Learning the Best Approach to Knowledge Tracing?

Theophile Gervet
Carnegie Mellon University
tgervet@andrew.cmu.edu

Ken Koedinger
Carnegie Mellon University
kk1u@andrew.cmu.edu

Jeff Schneider
Carnegie Mellon University
jeff4@andrew.cmu.edu

Tom Mitchell
Carnegie Mellon University
tom.mitchell@cs.cmu.edu

Intelligent tutoring systems (ITSs) teach skills using learning-by-doing principles and provide learners with individualized feedback and materials adapted to their level of understanding. Given a learner's history of past interactions with an ITS, a learner performance model estimates the current state of a learner's knowledge and predicts her future performance. The advent of increasingly large scale datasets has turned deep learning models for learner performance prediction into competitive alternatives to classical Markov process and logistic regression models. In an extensive empirical comparison on nine real-world datasets, we ask which approach makes the most accurate predictions and in what conditions. Logistic regression – with the right set of features – leads on datasets of moderate size or containing or containing a very large number of interactions per student, whereas Deep Knowledge Tracing leads on datasets of large size or where precise temporal information matters most. Markov process methods, like Bayesian Knowledge Tracing, lag behind other approaches. We follow this analysis with ablation studies to determine what components of leading algorithms explain their performance and a discussion of model calibration (reliability), which is crucial for downstream applications of learner performance prediction models.

Keywords: empirical comparison, knowledge tracing, deep learning

1. Introduction

Educational technology promises to expand access to high-quality education, personalize and accelerate learning, and cut teaching and training costs. Intelligent tutoring systems (ITSs; Anderson et al. 1985) successfully embody this promise: they teach skills (like algebra, computer programming, or medical diagnosis) using learning-by-doing principles, and provide learners with individualized feedback and materials adapted to their level of understanding. Studies have demonstrated that ITSs can teach some subjects nearly as effectively as personal human tutors, at close to zero marginal cost (VanLehn, 2011). Although ITSs have not yet reached the mainstream in the United States – they are used by *only* a few hundred thousand students a year – they have exploded in popularity in China over the last five years, due to government incentives, a competitive academic environment, and improvements in learner modeling enabled by large datasets (Hao, 2019). Modeling the performance of learners as they interact with an ITS is the

workhorse of most systems deployed in production today, and a core problem in educational data mining (EDM). Given a learner's history of past interactions with an ITS, a learner performance model estimates the current state of a learner's knowledge and predicts her future performance.

Modeling the performance of learners as they interact with an ITS has three major purposes: enabling adaptive behavior of the instructional policy, displaying the learner's estimated knowledge as a means of support for learning, and generating interpretable and actionable insights (Pelánek, 2017). First, most adaptive instructional policies used in practice today rely on an estimate of a learner's performance. They either require learners to become proficient in one topic before proceeding to the next, i.e., mastery learning (Ritter et al., 2016), or sequence items based on some notion of optimal difficulty, i.e., the *goldilocks* principle (Koedinger et al., 2013). Secondly, displaying to the student a representation of their estimated performance, typically in the form of progress bars for different skills – also called open learner modeling (Bull and Kay, 2010) – promotes metacognitive abilities of learners, facilitates the discussion between learners and educators, and fosters learner trust in the system. Lastly, outputs and parameters of a fitted learner performance model may also provide interpretable and actionable insights to learning engineers, educators, and educational researchers (Rosé et al., 2019). Such insights can help to develop the ITS further – for example, by enabling more accurate modeling of the educational domain concerned (Koedinger et al., 2012) – or contribute to learning science.

These purposes of learner performance models are often at odds with each other. For adaptive behavior, accurate predictions matter most, while for actionable insights, the interpretability and the stability of parameter estimates supersede accuracy; open learner modeling requires both. Some authors argue for putting more weight on interpretability in the accuracy-interpretability tradeoff and avoiding black-box learner performance models, i.e., models where we do not seek to interpret parameters (Rosé et al., 2019). While we agree that actionable insights can be paramount for specific purposes, we see no compelling reason to use a single model for all purposes. One can, and should, use a model optimized for accuracy – black-box if need be – as part of the instructional policy and another model explicitly designed to refine the model of the educational domain. In this paper, we evaluate models only in terms of the accuracy of their predictions.

The advent of increasingly large scale datasets, collected through ITSs and MOOCs, has turned deep learning models into competitive alternatives to classical statistical models for learner performance prediction, like Bayesian Knowledge Tracing (BKT; Corbett and Anderson 1994) or Performance Factors Analysis (PFA; Pavlik et al. 2009). The authors of Deep Knowledge Tracing (DKT; Piech et al. 2015), the first such deep learning approach, reported a massive 25% gain in AUC (a measure of prediction quality) over BKT on two real-world datasets. Since then, some authors have shown that the performance gain of DKT over BKT is not as substantial as initially reported (Xiong et al., 2016). Others have shown classical methods, when given more flexibility, can match or even outperform DKT (Khajah et al., 2016; Wilson et al., 2016). These contradictory results raise the question: which approach makes the most accurate predictions, in what conditions? In this paper, we shed new light on this question by evaluating three families of learner performance models – Markov processes, logistic regression, and deep learning models – on nine real-world datasets exhibiting a broad range of properties. We seek to (1) disentangle what characteristics of a dataset lead to a method outperforming others, and (2) determine what components of an algorithm explain its performance.

Which methods make the most accurate predictions? In Section 5, we will see that logistic regression – with the right set of features – leads on 4 datasets out of 9, DKT leads on the

remaining 5, and Markov process methods lag behind other approaches. What properties of a dataset make a particular algorithm suitable? In Section 6, we will see that logistic regression leads on datasets of moderate size or containing or containing a very large number of interactions per student, whereas Deep Knowledge Tracing leads on datasets of large size or where precise temporal information matters most. What components of leading algorithms affect their performance the most? In Section 7, we will see that – surprisingly – the time-window features introduced in DAS3H (Choffin et al., 2019) add no predictive power to logistic regression models but can be exploited by a non-linear model; that on most datasets, the expert-designed domain model adds little predictive power; and that the input/output representation of DKT significantly affects its performance. Are learner performance models calibrated (reliable) for downstream applications? In Section 8, we will see that the current best models are severely biased on some datasets – hindering their applicability in adaptive policies and open learner models.

Before presenting our results, we review related work in Section 2, formalize the problem and introduce approaches we compare in Section 3, and introduce the datasets we selected and our evaluation methodology in Section 4.

2. Related Work

In this section, we discuss literature reviews of learner modeling at large, before we dive into reviews and empirical comparisons of learner performance prediction in tutoring systems.

Learner modeling has a myriad of applications in e-learning, mobile learning, educational games, and tutoring systems; Chrysafiadi and Virvou (2013) review what to model in each setting, how, and why. Narrowing our discussion to tutoring systems, Desmarais and Baker (2012) survey approaches modeling the learner's knowledge state, motivation, attention, metacognition, and affect. The modeling of the learner is inherently linked with the modeling of the educational domain concerned; Koedinger et al. (2013) discuss data-driven techniques to improve tutoring systems and show how to use a learner model to refine the model of an educational domain. Narrowing our discussion to the core problem of estimating the learner's knowledge state, Pelánek (2017) addresses practical issues concerning data collection, evaluation metrics, and cross-validation methodologies.

Turning to accuracy considerations, the community compared classical methods with each other (Pavlik et al., 2009; Gong et al., 2010), and deep learning methods with each other (Zhang et al., 2017; Yeung and Yeung, 2018; Pandey and Karypis, 2019). But most of these comparisons were conducted on different datasets, and direct comparisons between classical and deep learning methods remain sparse. Xiong et al. (2016) show DKT does not outperform classical algorithms (BKT and PFA) by as large a margin as initially reported (because a publicly available dataset had duplicate rows and the authors of DKT misunderstood the format of the data). Khajah et al. (2016) investigate what statistical regularities DKT can exploit that BKT cannot, and show BKT with relaxed assumptions can outperform DKT. Wilson et al. (2016) show that Bayesian extensions of Item Response Theory (IRT; van der Linden and Hambleton 2013) – arguably the simplest learner performance model – can also outperform DKT.

These works raise an important question: is deep learning the best approach to knowledge tracing? But the answer they provide is not entirely satisfactory. Xiong et al. (2016) compare DKT to weak baselines: the original BKT is far from the best Markov process model, and PFA is far from the best logistic regression model. Khajah et al. (2016) and Wilson et al. (2016) compare DKT to state-of-the-art approaches, but each of these papers investigates a single com-

petitor on distinct datasets. The lack of overlap between the datasets used in different comparisons leaves us in the dark as to what algorithm performs best overall. Moreover, none of these works investigates *in what conditions* deep learning is the best approach to knowledge tracing. In contrast, we explicitly try to identify the characteristics of a dataset that make a particular approach suitable.

3. APPROACHES

3.1. THE PROBLEM

We can formalize the learner performance prediction problem as a supervised sequence learning task. Given a learner's history of past interactions with a learning system $\mathbf{x}_{1:t} = (x_1, ..., x_t)$, predict some aspect of her next interaction x_{t+1}. In a tutoring system, we represent the t^{th} interaction as a tuple $x_t = (q_t, a_t)$, where q_t is the tag for the question that the learner attempts, and a_t is the binary correctness of the learner's answer. We predict the probability that the learner will be able to answer the next question correctly $p(a_{t+1} = 1|q_{t+1}, \mathbf{x}_{1:t})$. When necessary, we make the dependence on the learner explicit with a subscript s (for student), as illustrated in Table 1. In the datasets we consider, each interaction also includes a set of knowledge component (KC) tags involved in the question, which we denote $KC(q_t)$, and a timestamp TS_t, the time elapsed since the first interaction of the learner with the system. The timestamps reflect the inherent time-continuity of the problem: we observe discrete tokens (learner interactions with the system) at irregular time intervals. The length of the time intervals matters because learners could be practicing or forgetting outside of the system.

Table 1: Notation we use in this paper.

Notation	Description
$[n]$	Set $\{1, ..., n\}$
$s \in [S]$	Learner index
$i \in [I]$	Item (question) index
$k \in [K]$	KC index
$TS_{s,t} \in \mathbb{R}^+$	Timestamp for learner s at time step t
$KC(i)$	Set of KC indices involved in item i
$q_{s,t}$	Question of learner s at time step t
$a_{s,t} \in \{0, 1\}$	Correctness of learner s at time step t
$u_{s,t}$	Encoding of interaction of learner s at time step t
$x_{s,t}$	All data of learner s at time step t
$\mathbf{x}_{s,1:t}$	All data of learner s up to time step t
$\sigma(.)$	Logistic function $\sigma(x) = 1/(1 + e^{-x})$
$\tanh(.)$	Hyperbolic tangent function $\tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$
$\delta(.)$	One hot encoding of categorical feature

A knowledge component model is a fine-grained decomposition of the knowledge targeted in the instruction, together with a mapping that specifies which items (questions) involve which knowledge components. For example, we might tag an exercise $3 + 4$ with the KC *single digit*

addition. One item can be tagged with multiple KCs. We assume that the probability of answering an item i that involves KC k correctly depends on the learner's mastery of KC k. We may think of the item-KC mapping as an overlapping clustering of items that provides a correlation structure between items. For a more detailed discussion of knowledge components and their relation to cognitive processes and instructional principles, we refer the reader to the knowledge-learning-instruction (KLI) framework (Koedinger et al., 2012).

It is useful to distinguish between the input representation (what features to extract from the raw data) and the algorithm operating on this representation. This distinction allows us to analyze what type of information is available to an algorithm separately from the capacity of the algorithm to exploit this information. One caveat: the input representation is entangled with the choice of the algorithm to some extent.

3.2. Input Representation

Two approaches are commonly used for supervised sequence learning tasks: either summarize historical information in a feature vector and use any generic supervised learning algorithm (like logistic regression) – the success of this approach lies in the quality of the historical feature engineering – or directly process the variable-length sequence with an algorithm designed specifically for this purpose (like a Markov process method or a recurrent neural network).

3.2.1. Feature Vector Input

Generic supervised learning algorithms operate on a feature vector. Predictions take the form

$$p(a_{t+1} = 1|q_{t+1}, \mathbf{x}_{1:t}) = f(\Phi(q_{t+1}, \mathbf{x}_{1:t}))$$

where $\Phi = (\phi_1, ..., \phi_d)$ is a vector of d features of the learner, question q_{t+1} and history $\mathbf{x}_{1:t}$ and the form of the prediction function f depends on the algorithm. We could consider three types of historical features: cumulative counts of correct answers, wrong answers, and attempts on KCs involved in the question to be predicted $KC(q_{t+1})$ and on all questions confounded (these features completely discard time); time-window counts based on discrete time steps (discard the continuous timestamps); and time-window counts based on continuous timestamps.

3.2.2. Variable-length Input

Algorithms designed to process sequences operate on variable length inputs and rely on a computational inductive bias to deal with the time dimension. Predictions often take the form

$$p(a_{t+1} = 1|q_{t+1}, \mathbf{x}_{1:t}) = f(q_{t+1}, \phi(x_1), ..., \phi(x_t))$$

where the prediction function f takes a variable number of arguments and processes token features $\phi(x_1), ..., \phi(x_t)$ sequentially.

Sequence learning algorithms we consider were developed to deal with discrete sequences of tokens (text and speech); their inductive biases ignore the continuity of time. We could encode this information as features of tokens: $\phi(x_t)$ could include the time interval since the last interaction $TS_t - TS_{t-1}$. A more principled approach would be to model the learner knowledge state as a dynamical system in continuous time. To the best of our knowledge, this approach has not been pursued.

3.3. ALGORITHMS

3.3.1. Logistic Regression

Logistic regression predictions take the form

$$p(a_{s,t+1} = 1|q_{s,t+1}, \mathbf{x}_{s,1:t}) = \sigma\left(\mathbf{w}^T \Phi(q_{s,t+1}, \mathbf{x}_{s,1:t})\right)$$

where $\mathbf{w} \in \mathbb{R}^d$ is a trainable weight vector, $\sigma(x) = 1/(1 + e^{-x})$ is the logistic function, and we stress the dependence of variables on the learner (student) s. We fit logistic regression models by maximizing the likelihood of the training set.

Logistic regression models have a long history in the EDM community. Different feature vectors Φ yield different models in the literature; in this article we consider Item Response Theory (IRT; van der Linden and Hambleton 2013), Performance Factors Analysis (PFA; Pavlik et al. 2009), and DAS3H (Choffin et al., 2019). IRT predictions take the form

$$p_{\text{IRT}}(a_{s,t+1} = 1|q_{s,t+1}, \mathbf{x}_{s,1:t}) = \sigma(\alpha_s - \delta_{q_{s,t+1}})$$

where α_s is the ability of learner s, and $\delta_{q_{s,t+1}}$ is the difficulty of question $q_{s,t+1}$. PFA predictions take the form

$$p_{\text{PFA}}(a_{s,t+1} = 1|q_{s,t+1}, \mathbf{x}_{s,1:t}) = \sigma\left(\sum_{k \in KC(q_{s,t+1})} \beta_k + \gamma_k c_{s,k} + \rho_k f_{s,k}\right)$$

where β_k is the easiness of KC k, $c_{s,k}$ is the number of correct answers of learner s on KC k prior to this attempt, and $f_{s,k}$ is the number of wrong answers of learner s on KC k prior to this attempt. DAS3H combines IRT and PFA features and introduces continuous time-window features, its predictions take the form

$$p_{\text{DAS3H}}(a_{s,t+1} = 1|q_{s,t+1}, \mathbf{x}_{s,1:t}) = \sigma\left(\alpha_s - \delta_{q_{s,t+1}} + \sum_{k \in KC(q_{s,t+1})} \beta_k + \sum_{k \in KC(q_{s,t+1})} \sum_{w=0}^{W-1} \theta_{k,2w+1}\phi(c_{s,k,w}) - \theta_{k,2w+2}\phi(a_{s,k,w})\right)$$

where w indexes a set of expanding time windows, $c_{s,k,w}$ is the number of correct answers of learner s on KC k in time window w, $a_{s,k,w}$ is the number of attempts of learner s on KC k in time window w, and $\phi(x) = \log(1 + x)$ rescales the counts. Following Choffin et al. 2019, we used five time windows in our experiments: $\{1/24, 1, 7, 30, +\infty\}$ (in days).

The predictions of our logistic regression model with the best performing set of features, which we refer to as Best-LR, take the form

$$p_{\text{Best-LR}}(a_{s,t+1} = 1|q_{s,t+1}, \mathbf{x}_{s,1:t}) = \sigma\left(\alpha_s - \delta_{q_{s,t+1}} + \phi(c_s) + \phi(f_s) + \sum_{k \in KC(q_{s,t+1})} \beta_k + \gamma_k \phi(c_{s,k}) + \rho_k \phi(f_{s,k})\right)$$

where c_s and f_s are the total number of correct and wrong answers of learner s prior to this attempt. Best-LR is DAS3H without time-window features but augmented with total count features, or equivalently PFA with rescaled count features, augmented with total counts features and IRT student ability and question difficulty parameters.

3.3.2. Markov Processes

Markov processes – stochastic processes in which the conditional distribution over future states of a system is independent of its history, given its present state – also have a long history in the EDM community. Bayesian Knowledge Tracing (BKT; Corbett and Anderson 1994) is the original and still most widespread learner performance model. BKT tracks the state of the learner's knowledge independently for each KC. For a certain KC, BKT treats the learner as being in one of two possible hidden states: *Mastery* and *Non-Mastery*. The model assumes the learner never forgets a mastered KC, and every new question has a fixed probability of helping the learner master the KC. Under these assumptions, BKT requires four trainable parameters per KC, as illustrated in Figure 1. BKT is usually fit with expectation-maximization or brute-force search among a predefined set of plausible parameters, which is possible because it has only $4K$ trainable parameters in total. At test time, the probability of mastery $P(M_t)$ is updated with Bayesian inference after every response.

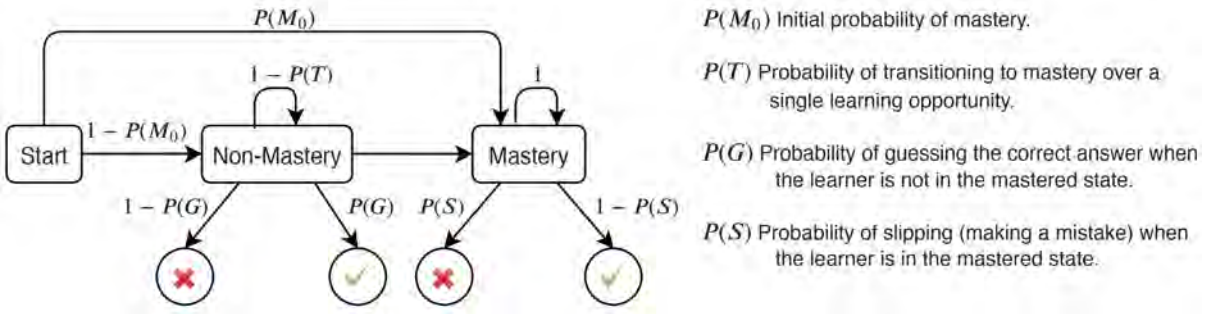

Figure 1: Markov process representation of Bayesian Knowledge Tracing (BKT). *Mastery* and *Non-Mastery* are latent states and arrow values are the probabilities of transition and observation.

The restrictive assumptions of the original BKT model prevent it from fully leveraging the large scale of the datasets we consider. Many extensions of BKT increase its flexibility, including item difficulty (Pardos and Heffernan, 2011), individualization (Pardos and Heffernan, 2010; Yudelson et al., 2013), time between attempts (Qiu et al., 2011), forgetting (Khajah et al., 2016), and discovering the KC model with Bayesian nonparametric models (Lindsey et al., 2014; González-Brenes, 2015).

We do not include BKT in our experiments, but we include BKT+ (Khajah et al., 2016) – its best performing extension to date that combines individualization, forgetting, and discovering the KC model. BKT+ performs much better than BKT, but fitting the model requires Markov chain Monte-Carlo methods. This requirement makes BKT+ significantly trickier to implement and orders of magnitude slower to fit than other approaches we consider: on large datasets, training BKT+ takes days, whereas training logistic regression or Deep Knowledge Tracing takes under two minutes.

3.3.3. Deep Learning Methods

Deep learning made its apparition in the EDM community more recently with Deep Knowledge Tracing (DKT; Piech et al. 2015). Deep learning algorithms discard hand-crafted features in favor of flexible function approximation mapping the input to the output through a feature hierarchy learned directly from data.

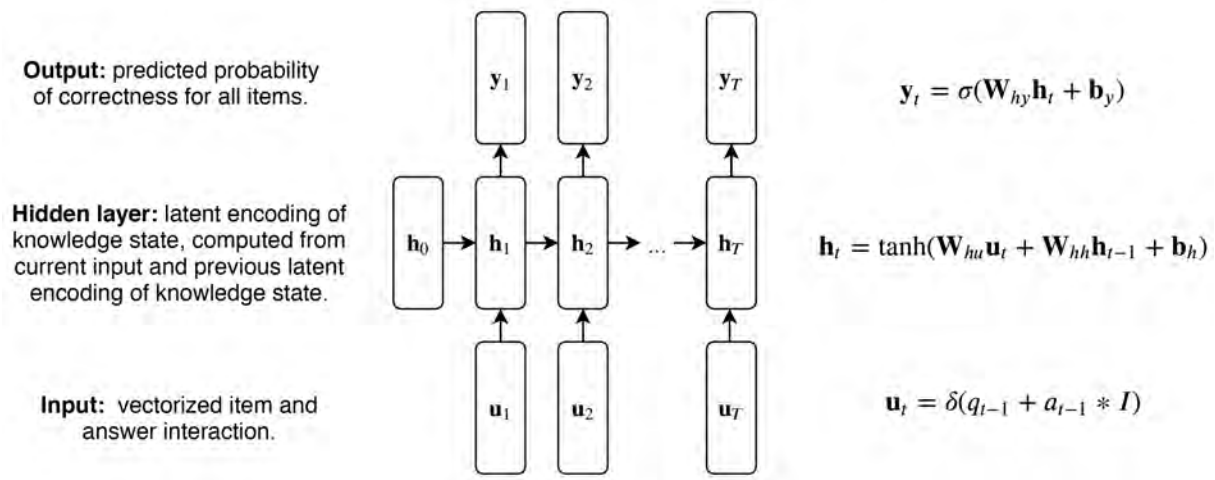

Figure 2: Enrolled recurrent neural network representation of Deep Knowledge Tracing (DKT). The model is parameterized by an input weight matrix $\mathbf{W}_{hu}$, recurrent weight matrix $\mathbf{W}_{hh}$, output weight matrix $\mathbf{W}_{hy}$, latent bias vector $\mathbf{b}_h$, and output bias vector $\mathbf{b}_y$. The initial latent state $\mathbf{h}_0$ is a vector of zeros and the nonlinearities $\tanh(.)$ and $\sigma(.)$ are applied elementwise.

DKT is a recurrent neural network (RNN). At time step t, we represent the previous interaction $x_{t-1} = (q_{t-1}, a_{t-1})$ uniquely as a one-hot encoding $\mathbf{u}_t = \phi(x_{t-1}) = \delta(q_{t-1}+a_{t-1}*I)$, where I is the total number of items. Figure 2 illustrates the sequential computation of a RNN: we map the sequence of interaction encodings $(\mathbf{u}_1, ..., \mathbf{u}_T)$ to a sequence of hidden states $(\mathbf{h}_1, ..., \mathbf{h}_T)$ – which we can view as successive summaries of information from past observations relevant for future predictions – and then to a sequence of vectors of predicted probabilities of answering each question correctly $(\mathbf{y}_1, ..., \mathbf{y}_T)$. We can read the predicted probability of answering question q_t correctly from the corresponding entry of vector $\mathbf{y}_t$. DKT uses an RNN variant with long short-term memory (LSTM) cells, which can better exploit long-term dependencies in the data. We fit the model by maximizing the likelihood of the training data.

The authors of most deep learning methods developed since DKT claim improvements in interpretability, but report minor performance gains relative to DKT (Zhang et al., 2017; Yeung and Yeung, 2018; Lee and Yeung, 2019). One notable exception reporting substantial performance gains is self-attentive knowledge tracing (SAKT; Pandey and Karypis 2019). SAKT is of particular interest as it reflects a paradigm shift – from recurrent neural networks to attention mechanisms – in how the deep learning and natural language processing communities process sequences. When we present it a question for which we want a prediction (the query), SAKT first identifies relevant past interactions – it *attends* to them – and then predicts future performance from these interactions. Operationally, this is done in three steps. First, we compute the similarity between the query (question embedding) and each past data point (interaction encoding), often with a dot product. Second, we form a convex combination of past data points weighted by normalized similarity scores. Finally, we transform this convex combination linearly and pass it through a sigmoid nonlinearity to obtain the predicted probability of correctness.

4. Experimental Setting

In this section, we introduce the datasets we selected and the broad range of characteristics they cover and discuss our evaluation methodology.

4.1. Datasets

We selected nine real-world datasets commonly used as benchmarks in prior work. Four datasets from the ASSISTment intelligent tutoring system: ASSISTment 2009-2010 (`assist09`), ASSISTment 2012-2013 (`assist12`), ASSISTment 2015 (`assist15`), and ASSISTment Challenge 2017 (`assist17`; Feng et al. 2009); two datasets from the KDD Cup 2010 EDM Challenge: Algebra I 2005-2006 (`algebra05`) and Bridge to Algebra 2006-2007 (`bridge06`; Stamper et al. 2010); a dataset from middle-school students practicing spanish exercises (`spanish`; Lindsey et al. 2014); a dataset from a college-level engineering `statics` course from the PSLC DataShop (`statics`; Koedinger et al. 2010); and a dataset from middle-school students practicing math on the Squirrel AI tutoring system (`squirrel`).

To be consistent with data preprocessing steps taken in prior work, we discarded learners that had fewer than ten interactions with the system and removed interactions with NaN KCs. Items can be tagged with one or multiple KCs; we converted each unique combination of KCs to a new KC when necessary (for DKT and SAKT). Algorithms that require timestamps (like DAS3H) were only applicable to the subset of the datasets containing this information.

Table 2 illustrates how datasets vary widely in the number of items and KCs they cover, the number of learners and total interactions they contain, and temporal characteristics like the number of interactions per learner and the period over which these interactions occur. We found the size of a dataset most directly influences the relative performance of learner performance prediction models, as each model needs a different amount of data to generalize. In all models we consider, the number of fitted parameters scales linearly with the number of items and KCs. Thus, for generalization purposes, the size of a dataset depends not only on the number of learners and total interactions it contains but, most importantly, on the number of learners that attempt each item and KC. In Table 2, we ordered datasets by the number of learners per item, as a rough measure of dataset size. The `algebra05`, `bridge06`, and `assistments09` datasets are the smallest for generalization purposes: they have the least learners per item and few learners per KC. In Section 6.1, we will see that logistic regression leads on these datasets because deep learning methods, like DKT, severely overfit.

Table 2: Properties of the datasets we consider.

	Dataset								
	`algebra05`	`bridge06`	`assist09`	`assist12`	`assist17`	`statics`	`squirrel`	`spanish`	`assist15`
Interactions	607,025	1,817,476	278,868	2,711,602	934,638	189,297	6,003,641	578,726	658,887
Learners	574	1,146	3,241	29,018	1,708	282	24,500	182	14,657
Items	173,113	129,263	17,709	53,086	3,162	1,223	20,201	409	-
KCs	112	493	124	265	102	98	742	221	100
Mean KCs per item	1.36	1.01	1.20	1.00	1.23	1.00	1.00	1.00	-
Median items per KC	85	101	120	35	18	4	28	1	-
Median learners per item	1	4	10	22	90	136	137	178	-
Median learners per KC	86	90	243	398	312	202	1,825	178	1,143
Timestamps	✓	✓	✗	✓	✓	✗	✓	✗	✗
Median days per learner	84	162	-	52	184	-	30	-	-
Median interactions per learner	581	1,373	32	59	489	635	154	2,924	31

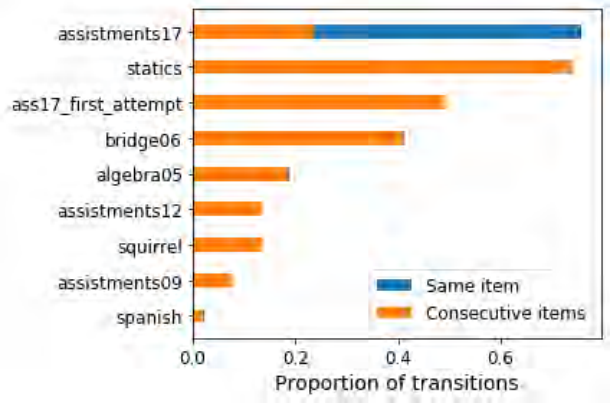

Figure 3: Proportion of consecutive interactions involving the same or consecutive items. The `assistments17` (and `ass17_first_attempt`, its restriction to first attempts) and the `statics` datasets contain the most transitions involving consecutive items. Students progress through the material the most sequentially on these datasets.

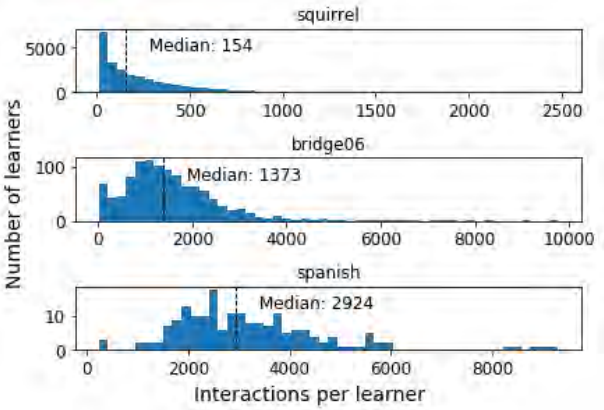

Figure 4: Distribution of the number of interactions per learner. Most datasets exhibit a power-law distribution like the `squirrel` dataset. But the `bridge06` and `spanish` datasets are outliers: they do not display a power-law distribution and contain many more interactions per learner (refer to median interactions/learner in Table 2).

The second characteristic of datasets we have found to correlate with performance differences across models is the degree to which learners progress sequentially through the material, as opposed to switching between different topics every question. In all tutoring system datasets we consider (except the `spanish` dataset), the material is ordered sequentially by topic: questions that provide practice for the same topic have consecutive item numbers. Thus the proportion of consecutive interactions involving consecutive item numbers is a measure of the degree to which learners progress sequentially through the material. Figure 3 shows the proportion of consecutive interactions involving consecutive item numbers. Students progress through the material the most sequentially on the `assistments17` and `statics` datasets[1]. In Section 6.2, we will see that DKT leads on these sequential datasets because it can better exploit precise temporal information than competing approaches.

The last dataset characteristic with a significant impact on performance is the number of interactions per learner. Figure 4 shows the distribution of the number of interactions per learner for different datasets. Most datasets exhibit a power-law distribution with up to a few hundreds of interactions for the typical learner. But the `bridge06` and `spanish` datasets are outliers: they do not exhibit a power-law distribution and contain thousands of interactions per learner. In Section 6.3, we will see that DKT lags behind other approaches on these datasets as it struggles to keep track of long-term information, whereas approaches operating on a feature vector of historical counts (like logistic regression) sidestep this issue.

Datasets also vary significantly in terms of the spread of interactions over time (intervals between practice sessions and the period over which learning occurs). In the `squirrel` dataset, students typically use the system for a month, whereas in the `bridge06` and `assistments17` datasets, students usually practice over six months. However, we have not found any significant

[1]The `assistments17` dataset comprises all consecutive attempts of a student on the same item, whereas other datasets record only the first attempt. In Section 6.2, we will discuss implications of this fact, and check how results vary on a version of the `assistments17` dataset restricted to first attempts.

performance differences between models along these lines.

4.2. Evaluation Methodology and Metrics

We implemented deep learning algorithms in PyTorch (Paszke et al., 2017). We used the logistic regression implementation from scikit-learn (Pedregosa et al., 2011) and the BKT+ implementation from Khajah et al. 2016 in C++. Our code and links to public datasets we used are freely available on GitHub[2].

We performed 5-fold nested cross-validation at the learner level. This means we split the learner population into 5 disjoint groups and performed cross-validation on this basis. Each group acts as the test set once, while we use the other 4 groups to train the model and select the best hyper-parameters (and perform early-stopping for deep learning models). We report metrics across the entire dataset when each data point was in the test set once, and standard deviations across folds. We chose to cross-validate across learners only, as opposed to learners and time, for our results to be comparable to most prior work (Piech et al., 2015; Choffin et al., 2019; Pandey and Karypis, 2019) and to reduce training time.

We report results for the best hyper-parameters selected by grid search on a portion of the training set reserved for this purpose. For logistic regression, we used the L-BFGS solver with L2 regularization with strength hyper-parameter C in $\{0.1, 1, 10\}$. Results were insensitive to the regularization strength. For DKT, SAKT, and the feedforward neural network, we used the Adam (Kingma and Ba, 2015) optimizer with a learning rate of 0.001, batch size 100, dropout probability in $\{0, 0.25, 0.5\}$, and early-stopping. For DKT, we selected the embedding and the hidden state dimension in $\{50, 100, 200\}$, and the number of recurrent layers in $\{1, 2\}$. We also tried different combinations of inputs and outputs for DKT, which we will explore in Section 7.2. On most datasets, we obtained the best results for a single recurrent layer, dropout probability 0.5, and embedding and hidden state dimension 200. Results were sensitive to the dropout probability and input/output representation but not the embedding and hidden state dimensions or the number of layers. For SAKT, we set the maximum length of the sequence to the median number of interactions per learner (as did the authors), selected the embedding dimension in $\{50, 100, 200\}$, the number of attention layers in $\{1, 2\}$, and the number of attention heads per layer in $\{1, 5\}$. On most datasets, we obtained the best results for a single attention layer with 5 heads, dropout probability 0.25, and embedding dimension 200. For the two-layer feedforward neural network in ablation studies, we selected the hidden layer dimension in $\{100, 200, 400\}$, and obtained the best results for 200 on most datasets. For BKT+, we used the default parameters of the authors' code.

We compare all approaches in terms of AUC (area under the curve that plots the true-positive rate against the false-positive rate at all decision thresholds) and root mean squared error (RMSE). The AUC is the most widely used evaluation metric for learner performance models; we report it for our results to be comparable to prior work. But it has a significant disadvantage compared to the RMSE: it considers only the relative ordering of predictions, which makes it invariant to a rescaling of predictions. In our case, we care about model calibration, and this characteristic is undesirable (Pelánek, 2015).

Tables 3-11: Dataset performance for logistic regression models: Item Response Theory (IRT), Performance Factors Analysis (PFA), DAS3H, and our best features (Best-LR); Markov processes: Bayesian Knowledge Tracing (BKT) and its best variant (BKT+); and deep learning methods: Deep Knowledge Tracing (DKT), Self-Attentive Knowledge Tracing (SAKT), and a feedforward network with our best features (Best-FFW).

Table 3: `algebra05` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
Best-LR		**0.831 ± 0.003**	**0.329 ± 0.002**
Best-FFW		**0.829 ± 0.004**	**0.330 ± 0.003**
DciteNAS3H	**0.826 ± 0.003**[4]	**0.827 ± 0.003**	**0.331 ± 0.003**
DKT		0.821 ± 0.005	0.334 ± 0.004
SAKT		0.801 ± 0.005	0.337 ± 0.004
PFA	0.744 ± 0.004[4]	0.769 ± 0.006	0.343 ± 0.005
IRT	0.771 ± 0.007[4]	0.768 ± 0.007	0.347 ± 0.005
BKT	0.62[1]		
BKT+		Too slow	Too slow

Table 4: `bridge06` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
Best-LR		**0.803 ± 0.003**	**0.366 ± 0.002**
Best-FFW		**0.802 ± 0.004**	**0.367 ± 0.002**
DAS3H	**0.790 ± 0.004**[4]	0.791 ± 0.004	0.373 ± 0.003
DKT		0.790 ± 0.003	0.368 ± 0.002
SAKT		0.784 ± 0.004	0.371 ± 0.003
IRT	0.747 ± 0.002[4]	0.749 ± 0.003	0.375 ± 0.003
PFA	0.739 ± 0.003[4]	0.741 ± 0.004	0.392 ± 0.003
BKT+		Too slow	Too slow

Table 5: `assistments09` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
Best-LR		**0.772 ± 0.004**	**0.423 ± 0.003**
BKT+		0.759 ± 0.004	0.425 ± 0.003
DKT	**0.75**[1]	0.757 ± 0.004	0.428 ± 0.003
SAKT		0.756 ± 0.005	0.428 ± 0.004
PFA	0.73[1]	0.724 ± 0.005	0.437 ± 0.004
IRT		0.692 ± 0.006	0.456 ± 0.005
BKT	0.63[1]		
DAS3H		No timestamps	No timestamps
Best-FFW		No timestamps	No timestamps

Table 6: `assistments12` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
DKT		**0.771 ± 0.001**	**0.414 ± 0.001**
Best-FFW		0.767 ± 0.001	0.415 ± 0.001
Best-LR		0.751 ± 0.001	0.416 ± 0.001
DAS3H	**0.739 ± 0.001**[4]	0.740 ± 0.001	0.417 ± 0.002
SAKT		0.732 ± 0.002	0.419 ± 0.002
IRT	0.702 ± 0.001[4]	0.713 ± 0.001	0.431 ± 0.001
PFA	0.668 ± 0.002[4]	0.669 ± 0.002	0.439 ± 0.002
BKT+		Too slow	Too slow

Table 7: `assistments17` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
DKT	**0.734 ± 0.001**[3]	**0.770 ± 0.002**	**0.434 ± 0.002**
Best-FFW		0.761 ± 0.002	0.437 ± 0.002
SAKT	**0.734**[5]	0.722 ± 0.003	0.447 ± 0.003
Best-LR		0.714 ± 0.003	0.450 ± 0.003
BKT+		0.710 ± 0.003	0.451 ± 0.003
DAS3H		0.693 ± 0.003	0.453 ± 0.003
IRT		0.681 ± 0.003	0.459 ± 0.004
PFA		0.619 ± 0.005	0.471 ± 0.004

Table 8: `statics` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
DKT	0.815[5]	**0.829 ± 0.004**	**0.367** ± 0.001
Best-LR		0.819 ± 0.005	0.368 ± 0.002
SAKT	**0.853**[5]	0.813 ± 0.005	0.370 ± 0.003
BKT+	0.76[2]	0.811 ± 0.004	0.371 ± 0.003
IRT		0.789 ± 0.006	0.374 ± 0.003
PFA		0.691 ± 0.007	0.407 ± 0.005
BKT	0.73[2]		
DAS3H		No timestamps	No timestamps
Best-FFW		No timestamps	No timestamps

Table 9: `squirrel` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
DKT		**0.772 ± 0.001**	**0.421 ± 0.001**
SAKT		**0.771 ± 0.001**	**0.422 ± 0.001**
Best-FFW		0.768 ± 0.001	0.423 ± 0.001
Best-LR		0.765 ± 0.001	0.423 ± 0.001
DAS3H		0.746 ± 0.002	0.431 ± 0.001
IRT		0.733 ± 0.002	0.433 ± 0.002
PFA		0.614 ± 0.003	0.462 ± 0.002
BKT+		Too slow	Too slow

Table 10: `spanish` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
Best-LR		**0.863 ± 0.002**	**0.334 ± 0.003**
BKT+	**0.85**[2]	0.851 ± 0.002	0.342 ± 0.004
PFA		0.847 ± 0.003	0.346 ± 0.005
DKT	0.83[2]	0.832 ± 0.004	0.364 ± 0.006
SAKT		0.831 ± 0.004	0.369 ± 0.007
BKT	0.83[2]		
IRT		0.679 ± 0.005	0.407 ± 0.008
DAS3H		No timestamps	No timestamps
Best-FFW		No timestamps	No timestamps

Table 11: `assistments15` dataset

Model	Best previous AUC ↑	Our AUC ↑	Our RMSE ↓
DKT	0.737 ± 0.001[3]	**0.731 ± 0.005**	**0.416 ± 0.003**
SAKT	**0.854**[5]	**0.730 ± 0.006**	**0.417 ± 0.003**
Best-LR		0.702 ± 0.004	0.419 ± 0.003
BKT+		0.701 ± 0.004	0.420 ± 0.004
PFA		0.690 ± 0.005	0.422 ± 0.005
IRT		0.638 ± 0.006	0.431 ± 0.005
DAS3H		No timestamps	No timestamps
Best-FFW		No timestamps	No timestamps

Sources: [1] is from Xiong et al. (2016), [2] from Khajah et al. (2016), [3] from Yeung and Yeung (2018), [4] from Choffin et al. (2019), and [5] from Pandey and Karypis (2019).

5. Results

In this section, we present performance results for all approaches on all datasets in Tables 3 to 11. For the reader's convenience, we also reproduce previously reported results. They are too sparse to compare different approaches properly, and we fill this gap in this paper.

The logistic regression model with the best feature vector (Best-LR) outperforms all other approaches – including deep learning methods (DKT and SAKT) and the best logistic regression model in prior literature (DAS3H) – on 4 datasets out of 9 (`algebra05`, `bridge06`, `assistments09`, and `spanish`). DKT leads on the remaining 5 datasets (`assistments12`, `assistments17`, `statics`, `squirrel`, and `assistments15`). Previously reported results show that the original BKT model cannot compete on datasets of the scale we consider. BKT+ is competitive with Best-LR, but because it is orders of magnitude slower to train than alternatives, we were not able to run experiments to completion on large datasets.

In our experiments, SAKT underperforms DKT on all datasets. This observation contradicts results from Pandey and Karypis (2019). The most substantial gap is on the ASSISTment 2015 dataset: the authors reported an AUC of 0.85, while we observed an AUC of 0.73 (no improvement over DKT). In their ablation study, the authors reported an AUC of 0.82 on the same dataset for a simple baseline (a feedforward neural network using only the previous interaction encoding as input). This figure – which we could not reproduce and seems impossible – leads us to believe our results reflect the actual performance of SAKT. We hypothesize the datasets we consider are too small for self-attention to perform to its full potential; even our largest dataset is small by the standards of the natural language processing community that developed these models.

In Section 6, we investigate the effect of dataset characteristics on the relative performance of leading approaches. In Section 7 we examine the impact of algorithmic choices through extensive ablation studies. In particular, we will see how we obtained new state-of-the-art results on 5 datasets out of 8 (`bridge06`, `assistments09`, `assistments12`, `assistments17`, and `spanish`) through minor feature engineering and algorithmic improvements.

6. Effect of Dataset Characteristics

In this section, we investigate what properties of a dataset correlate with the relative performance of the best logistic regression model (Best-LR) and DKT, propose explanations, and test our hypotheses with further experiments.

6.1. DKT Overfits Small Datasets but LR Underfits Large Datasets

Logistic regression dominates on the smallest datasets (`algebra05`, `bridge06`, and `assistments09`) — as measured by the number of learners per item and per KC — but DKT takes over on the largest datasets (`assistments12`, `assistments17`, and `squirrel`). Logistic regression is less susceptible to overfitting, but given enough data, DKT can model more complex functions. In particular, we will see in the next section that DKT can fit a relationship between the history and the correctness log-odds that logistic regression cannot capture.

To confirm the hypothesis that DKT overfits small datasets but better exploits large datasets, we ran an additional experiment varying the amount of training data on the `squirrel` dataset.

[2] https://github.com/theophilee/learner-performance-prediction

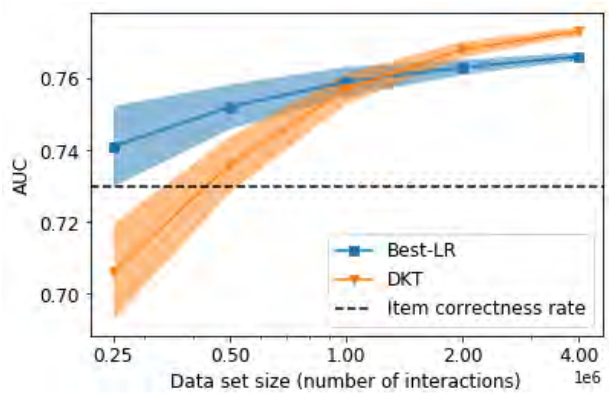

Figure 5: AUC as a function of the amount of training data on the `squirrel` data set (mean and two SD confidence intervals). Deep Knowledge Tracing (DKT) outperforms our best logistic regression model (Best-LR) in the high data regime (more than one million interactions).

	Algorithm		
Dataset	Regular DKT	KC-specific DKT	Best-LR
`assist12`	0.771	0.701	**0.751**
`assist17`	0.770	**0.744**	0.714
`statics`	0.829	**0.827**	0.819
`squirrel`	0.772	0.745	**0.765**

Table 12: Separate DKT model per KC on the datasets where DKT leads, AUC (↑). DKT restricted to information from a single KC still outperforms logistic regression (Best-LR) on the `assistments17` and `statics` datasets. DKT better exploits temporal information on these datasets.

Figure 5 compares the performance of Best-LR and DKT as a function of the amount of training data. For each of the 5 folds, we subsampled the training set to varying degrees and trained DKT and Best-LR; we obtained confidence intervals based on the standard deviation across folds. Best-LR dominates in the low and medium data regimes (fewer than one million interactions), and DKT takes over in the high data regime.

While this observation is not surprising — deep learning is known to shine in the big data regime — we note that in our context, the effective size of a dataset depends on the number of learners relative to the amount of material covered. In other words, for generalization purposes, the number of learners per item and per KC of a dataset better capture its size than the number of interactions it contains. For instance, the `statics` dataset, which is the smallest dataset by total number of interactions (only 189,297 when `bridge06` has 1,817,476), has enough learners per item and KC to fit DKT without too much overfitting (whereas `bridge06` does not).

6.2. DKT Better Exploits Temporal Information

DKT could exploit two types of statistical regularities that logistic regression (and all methods operating on a feature vector) have limited access to: the precise temporal order of interactions and information from KCs not involved in the question. Feature vectors encode the temporal order through imprecise time-windows, whereas DKT has access to the exact temporal order of the interaction sequence. Feature vectors encode information from KCs not involved in the question only through the total counts of correct answers and attempts on all questions confounded, whereas DKT has access to the exact order of interactions.

DKT outperforms the best logistic regression model (Best-LR) on the `assistments17` and `statics` datasets. In Section 4.1, we observed these datasets are the ones where learners progress the most sequentially through the material. We hypothesize the precise temporal order of recent interactions is more predictive of future performance on such datasets (Galyardt and

Goldin, 2015). For instance, a mistake on the first of a block of questions that always appear in the same order could be a strong signal of subsequent mistakes on specific questions. We further conjecture DKT can better exploit such temporal patterns than approaches that operate on historical count features, as it has access to the exact order of interactions and nonlinearities allow modeling more complex functions.

To verify this hypothesis, we isolate the benefits of better exploiting the temporal information by extending an experiment from Montero et al. (2018): we train one DKT model per KC – in the same way as Bayesian Knowledge Tracing maintains one Markov process per KC – on the datasets where DKT leads. If such a set of KC-specific DKT models outperforms logistic regression, it must be because it is better able to exploit the temporal order of interactions, not because it has access to information about other KCs. Table 12 shows that, even under such restrictions, DKT still outperforms Best-LR on the `assistments17` and `statics` datasets. This finding confirms that DKT makes better use of the temporal order of the sequence – which carries significant predictive power in sequential datasets – than logistic regression.

The `assistments17` dataset comprises all consecutive attempts of a student on the same item, whereas other datasets record only the first attempt (as we saw in Figure 3). The gap between DKT and Best-LR is the largest on this dataset (+0.056 AUC for DKT). This result can be partly explained by the fact that DKT can better exploit previous attempts on a question than the logistic regression models we consider, which were designed primarily for first attempts. Restricting `assistments17` to first attempts, the gap between DKT and Best-LR closes to +0.016 AUC, which is more in line with the gaps observed on other datasets where DKT leads (+0.029 on `assistments15`, +0.020 on `assistments12`, +0.010 on `statics`, and +0.007 on `squirrel`). The superior performance of our best feedforward neural network model (Best-FFW) on the `assistments17` dataset (0.761 AUC) compared to our best linear model (0.714 AUC) is further evidence for the necessity of nonlinearities to model complex temporal phenomena.

6.3. DKT Fails to Retain Long-term Information but Reaches Peak Performance Faster

DKT lags behind logistic regression on datasets containing a very large number of interactions per learner (`spanish` and `bridge06`). These datasets, with thousands of interactions per learner, are outliers for tutoring systems, which often exhibit power-law distributions of sequence lengths (see Table 2 and Figure 4). Figure 4 shows the performance of DKT and Best-LR as a function of the amount of training data available on a student on the `spanish` dataset. DKT plateaus after 1000 student interactions, whereas Best-LR keeps improving with more data. We observed a similar pattern on the `bridge06` dataset. This is evidence that DKT is unable to keep track of long-term information – a well-documented problem of recurrent neural networks (Hochreiter et al., 2001). Approaches operating on a feature vector of historical counts (like logistic regression) sidestep this issue. This problem arises only when typical learners have thousands of interactions with the system, an unusual characteristic for tutoring systems.

On the other hand, DKT reaches peak performance faster than logistic regression models. Figure 7 shows that DKT needs 6 times fewer interactions than Best-LR to reach close to peak performance on a new student on the `squirrel` dataset. We observed a similar pattern to varying degrees on most datasets. This finding could have significant practical benefits as it significantly reduces the ”burn-in” period of a learner performance prediction model.

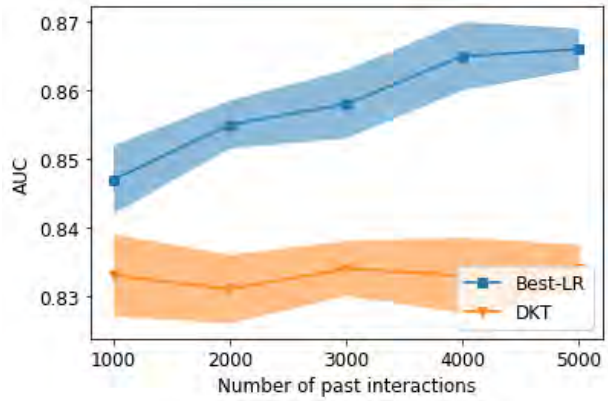

Figure 6: AUC as a function of the amount of data available on a student on the `spanish` dataset (mean and 2 SD confidence intervals). DKT plateaus after 1000 student interactions, whereas Best-LR keeps improving with more data.

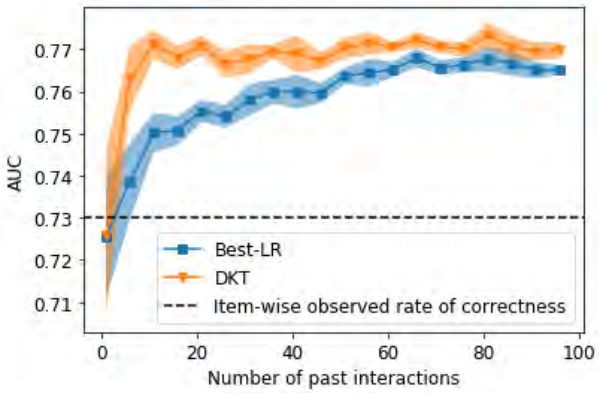

Figure 7: AUC as a function of the amount of data available on a student on the `squirrel` dataset (mean and 2 SD confidence intervals). In only 10 interactions, DKT reaches close to peak performance, whereas Best-LR needs 60.

7. Effect of Algorithmic Choices: Ablation Studies

In this section, we investigate what components of leading algorithms explain their performance through extensive ablation studies of approaches operating on a feature vector and DKT.

7.1. Ablation Study of Approaches Operating on a Feature Vector

Table 13: Ablation study of approaches operating on a feature vector. AUC (↑) averaged over 5 folds. We report results for a log-odds linear model (logistic regression) and a non-linear model (two-layer feedforward neural network) with different sets of features.

		Features					Dataset								
Algorithm	Linear	Item	KCs	Total counts	KCs counts	Time windows	as09	as12	a15	as17	alg05	bri06	span	stat	squi
IRT	✓	✓					0.692	0.713	0.638	0.681	0.768	0.749	0.679	0.789	0.733
PFA	✓		✓		✓		0.724	0.669	0.690	0.619	0.769	0.761	0.847	0.691	0.614
DAS3H	✓	✓	✓		✓	✓	-	0.740	-	0.693	**0.827**	0.791	-	-	0.746
NoKC-LR	✓	✓		✓			0.763	0.739	**0.699**	0.701	0.802	0.769	**0.861**	0.810	0.761
Best-LR	✓	✓	✓	✓	✓		**0.772**	0.751	**0.702**	0.714	**0.831**	**0.803**	**0.863**	**0.819**	0.765
TW-LR	✓	✓	✓	✓	✓	✓	-	0.750	-	0.717	**0.829**	**0.799**	-	-	0.766
NoTW-FFW	✗	✓	✓	✓	✓		**0.769**	0.753	**0.701**	0.713	**0.831**	**0.801**	**0.862**	**0.818**	0.766
Best-FFW	✗	✓	✓	✓	✓	✓	-	**0.767**	-	**0.761**	**0.829**	**0.802**	-	-	**0.768**

In this section, we present an ablation study of approaches operating on a feature vector with two purposes: identify features with the most predictive power, and determine whether a log-odds linear model is expressive enough to exploit information present in the features. We report results for a log-odds linear logistic regression model and a non-linear two-layer feed-forward neural network model for different sets of features in Table 13. We consider a set of features absent from prior work: the total number of prior correct answers and attempts (total counts). These features substantially boost performance on all datasets.

Surprisingly, the time-window features introduced in DAS3H (best paper at EDM 2019) do not add any predictive power to our best logistic regression model (see Best-LR vs. TW-LR in Table 13). This finding suggests that the performance boost of DAS3H over PFA is simply due to the addition of an item difficulty parameter inspired by IRT, rather than time-windows. On the other hand, time-window features boost the performance of a feedforward neural network on the `assistments17` dataset by +0.048 AUC and on the `assistments12` dataset by +0.014 AUC (see NoTW-LR vs. Best-FFW in Table 13). This finding suggests a non-linear relation between the history and the correctness log-odds, which makes sense given the better performance of DKT on these datasets (covered in Section 6.2).

The expert-designed KC model adds little predictive power on most datasets. On seven out of nine datasets, our best logistic regression model using KC features (Best-LR) provides a boost of +0.01 AUC or less relative to a baseline not using KC features (NoKC-LR). Although this observation surprised us and might surprise the reader, it is consistent with results from Lindsey et al. (2014). The only two datasets where the expert-designed KC model adds significant predictive power (+0.03 AUC) are the KDD Cup 2010 Challenge datasets (`algebra05` and `bridge06`). When a KC model is suitable, each item is tagged with one KC per difficulty factor it presents. A model relying only on KC features (like PFA) should beat a baseline using only item difficulty biases (IRT). Out of 9 datasets, 4 do not meet this condition, suggesting low-quality KC models.

7.2. Ablation Study of DKT

Table 14: Items/KCs as inputs/outputs for Deep Knowledge Tracing (DKT), AUC (↑).

Input	Output	assist09	assist12	assist15	assist17	algebra05	bridge06	spanish	statics	squirrel
KCs	KCs	**0.757**	**0.752**	**0.731**	0.731	0.821	0.790	0.831	0.774	0.704
KCs	Items	0.751	**0.771**	-	**0.770**	-	-	**0.832**	**0.829**	**0.772**
Items	Items	0.702	0.718	-	0.762	-	-	0.829	0.819	0.763

DKT can process either items or KCs as inputs, and output one probability per item or per KC. This choice significantly impacts performance. If DKT processes item inputs, at time step t, we represent the previous interaction $x_{t-1} = (q_{t-1}, a_{t-1})$ – where q_{t-1} is the item number and a_{t-1} the binary student correctness – uniquely as a one-hot encoding $\mathbf{u}_t = \delta(q_{t-1} + a_{t-1} * I)$, where I is the total number of items. Alternatively, if DKT processes KC inputs, we convert the combination of KCs involved in the question $KC(q_{t-1})$ to a unique identifier k_{t-1}, and represent the previous interaction as a one-hot encoding $\mathbf{u}_t = \delta(k_{t-1} + a_{t-1} * K)$, where K is the total number of KC identifiers. If DKT outputs one probability per item, the output vector $\mathbf{y}_t$ has I entries; conversely, if it outputs one probability per unique KC identifier, $\mathbf{y}_t$ has K entries.

KC inputs reduce the amount of information available but help generalization because datasets contain many more learners per KC than per item (refer to Table 2). KC outputs (predicting a single probability of success common to all questions in a KC) should also favor generalization but might be too restrictive. Most prior works used KC inputs and KC outputs. Table 14 shows this choice considerably impacts performance, and the KC inputs and item outputs combination works best on most datasets (when the number of items is small enough for this combination to be tractable). This finding means that DKT – although initially advertised as independent from any expert-designed structure (Piech et al., 2015) – relies on the KC model to perform optimally.

8. Model Calibration (Reliability Diagrams)

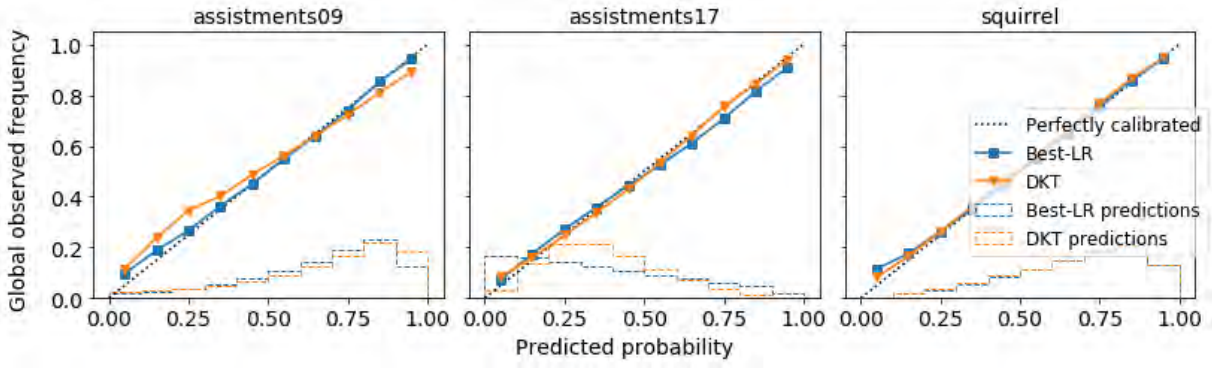

Figure 8: Calibration binned by predictions. We compare our best logistic regression model (Best-LR) to Deep Knowledge Tracing (DKT). A model can achieve perfect calibration – a line on the diagonal – by predicting the base rate of events; a useful model should also predict a wide range of probabilities.

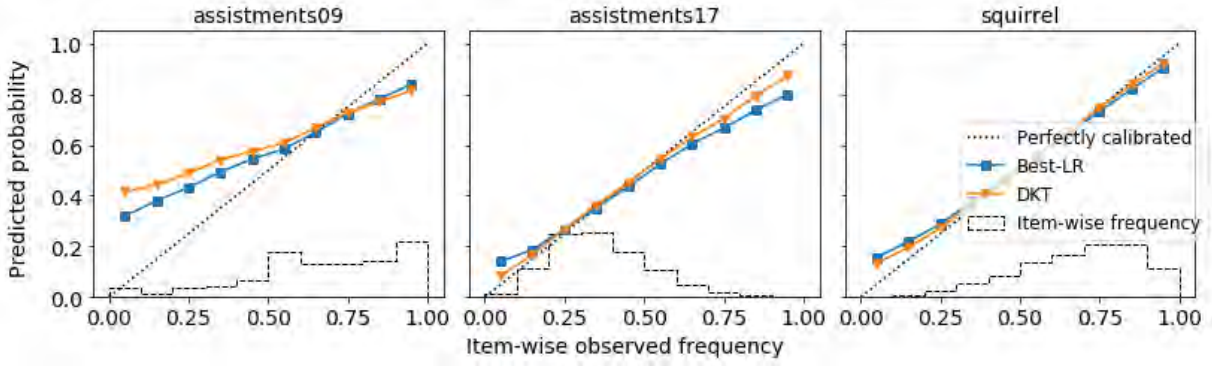

Figure 9: Calibration binned by item-wise observed frequency of correctness. Predictions exhibit systematic biases outside of the interval containing most of the data. DKT is better calibrated on the `assistments17` dataset, and Best-LR on the `assistments09` dataset; this is consistent with our AUC and RMSE results in Section 5.

In this section, we investigate the calibration (also called reliability, depending on the research community) of our best logistic regression model (Best-LR) and Deep Knowledge Tracing (DKT). Calibration – which measures the difference between predicted probabilities and observed frequencies – is crucial for downstream applications of learner performance models (like an adaptive instructional policy, or an open learner model). An instructional policy that bases its adaptive behavior on a severely biased learner performance model is of limited value, if not downright harmful. Visualizing the calibration of learner performance models allows us to detect and possibly correct for systematic biases before they propagate into downstream applications.

We report two sets of calibration plots: Figure 8 shows calibration plots binned by predicted probabilities, and Figure 9 shows calibration plots binned by observed probabilities (item-wise observed frequencies of correctness). Calibration plots binned by predicted probabilities would allow us to correct for systematic biases with recalibration techniques, but in our case they

do not reveal any biases. Conversely, calibration plots binned by observed probabilities reveal systematic biases, but no standard technique can correct for them. For example, on the `assistments09` dataset, both models severely overestimate learners when the probability of correct answer is low, and underestimate learners when the probability of correct answer is high. In general, models are calibrated only in the interval containing most of the data. On large datasets, like the `squirrel` dataset, both models are close to perfectly calibrated over the unit interval.

9. Limitations, Conclusion, and Future Research

9.1. Limitations

The main limitation of our work is the lack of cross-validation in the item space. Our cross-validation methodology demonstrates models generalize to new students, but it does not tell us how well models generalize to items out of the training set. In the problem formulation we consider (no information on items besides their tag and KCs), the KC model enables generalization to new items. Generalization in the item space – which we could capture with cross-validation across time – is a measure of the quality of the KC model, and the capacity of an approach to exploit it. Our cross-validation methodology across students does not do justice to Item Response Theory (IRT), as the student ability parameter is never fit. Wilson et al. (2016) refit the student ability parameter after every question, which makes IRT significantly more competitive.

Another limitation of our work is the absence of satisfying explanations for some properties of DKT we observed. Exactly what temporal information can DKT – and a feedforward neural network with time-window features to a lesser degree – exploit that a log-odds linear model (logistic regression with time-window features) cannot? For instance, can DKT better pick up on local patterns of student behavior, such as gaming the system? Also, why does DKT reach peak performance faster than logistic regression? We believe answering these questions could help us better understand what factors DKT relies on to make predictions and guide further algorithmic improvements.

9.2. Conclusion

In this paper, we investigated which approach to knowledge tracing makes the most accurate predictions, in what conditions. We evaluated three families of learner performance models – Markov processes, logistic regression, and deep learning models – on nine real-world tutoring system datasets exhibiting a broad range of properties. We attempted to (1) disentangle what characteristics of a dataset lead to a method outperforming others, and (2) determine what components of an algorithm explain its performance through ablation studies.

We discovered that Markov process methods, like Bayesian Knowledge Tracing, lag behind other approaches. Logistic regression is less susceptible to overfitting than Deep Knowledge Tracing (DKT) on smaller datasets. But given enough data, DKT takes the lead as it can model more complex relationships between the history and the correctness log-odds. In particular, DKT makes better use of the temporal order of the sequence – which carries particularly high predictive power in datasets where students progress sequentially through the material. However, DKT is unable to keep track of long-term information on datasets containing thousands of interactions per learner.

In our ablation studies, we discovered that time-window features add no predictive power to our best logistic regression models but can be exploited by a feedforward neural network. This finding further emphasizes the necessity to use non-linear models to leverage historical data optimally. On most datasets, the expert-designed domain model (knowledge component model) is of low quality and adds little predictive power to logistic regression models. The input and output representation of DKT – using questions or their associated knowledge components – significantly affects its performance.

Finally, we saw that the current best models are not calibrated (reliable) on some tutoring system datasets – i.e., predicted probabilities do not match observed frequencies – hindering their applicability in downstream applications like an adaptive policy or an open learner model.

9.3. Future research

The efficacy of intelligent tutoring systems (ITSs) is primarily determined by their weakest link (Pelánek, 2017). We believe the weakest links today are often the adaptive instructional policy and the KC model, not the performance prediction model.

Most instructional policies deployed in ITSs today are heuristics lacking theoretical or empirical justification. For example, the *goldilocks* principle (Koedinger et al., 2013) recommends an item estimated to be not too hard nor too easy. But is this even a good idea? If so, what is the optimal item difficulty? When employed as part of an adaptive policy, learner performance models with similar predictive accuracies can make vastly different recommendations (Rollinson and Brunskill, 2015). Even worse, heuristics ignore the feedback loop inherent to the problem: data collected by the instructional policy will be used to train the model further. By ignoring this feedback loop, we introduce biases in predictions (Pelánek et al., 2016), and we forgo the benefits of active learning – in the machine learning sense: an algorithm can perform better with less training data if it is allowed to choose the data from which it learns (Settles, 2009). A more principled approach is to directly optimize for the instructional policy with machine learning techniques: bandit optimization (Clement et al., 2015; Lan and Baraniuk, 2016) and reinforcement learning (Chi et al., 2011; Zhou et al., 2019) take into account the feedback loop by trading off *exploration* (recommend items that inform future recommendations) and *exploitation* (recommend items that maximize student learning).

One direction of future work is assessing whether the different models can not only predict performance but estimate student learning gains. This is straightforward in the logistic regression models as we can inspect whether the parameter estimates associated with the number of attempts are positive and significant (preliminary inspections indicate that most are). Estimating the learning gains is harder in the DKT model, as there are no interpretable parameter estimates to inspect. However, we could test whether the DKT predictions indicate higher success at higher attempt values.

Another future research topic is to use such prediction models to infer optimal instructional policies with reinforcement learning. Ideally, we would use learning gain measures from pre- and post-assessments as the reward for reinforcement learning training. But these assessments are not always available. Alternatively, within-practice measures of learning, like those discussed, could provide another form of reward. These within-practice measures of learning may be better if they could more accurately reflect long-term, robust learning outcomes, not just the immediate effects of learning. Creating such robust learning measures is an important future research area.

The low quality of the KC models of half the datasets we considered suggests new methods to automatically refine the KC model would be particularly valuable. We believe clustering items and KCs in a continuous embedding space with deep learning methods could be a promising direction.

10. ACKNOWLEDGMENTS

This research was supported in part by generous funding from Yixue through the CMU - squirrel AI Research Lab on Personalized Education at Scale. We are grateful to Dr. Dan Bindman for enlightening discussions about how to evaluate and visualize the performance of the alternative models considered here.

REFERENCES

ANDERSON, J. R., BOYLE, C. F., AND REISER, B. J. 1985. Intelligent tutoring systems. *Science 228,* 4698, 456–462.

BULL, S. AND KAY, J. 2010. Open learner models. In *Advances in Intelligent Tutoring Systems*, R. Nkambou, J. Bourdeau, and R. Mizoguchi, Eds. Springer, 301–322.

CHI, M., VANLEHN, K., LITMAN, D., AND JORDAN, P. 2011. Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. *User Modeling and User-Adapted Interaction 21,* 1-2, 137–180.

CHOFFIN, B., POPINEAU, F., BOURDA, Y., AND VIE, J.-J. 2019. DAS3H: Modeling student learning and forgetting for optimally scheduling distributed practice of skills. In *Proceedings of the 12th International Conference on Educational Data Mining*, C. F. Lynch, A. Merceron, M. Desmarais, and R. Nkambou, Eds. 29–39.

CHRYSAFIADI, K. AND VIRVOU, M. 2013. Student modeling approaches: A literature review for the last decade. *Expert Systems with Applications 40,* 11, 4715–4729.

CLEMENT, B., ROY, D., OUDEYER, P.-Y., AND LOPES, M. 2015. Multi-armed bandits for intelligent tutoring systems. *Journal of Educational Data Mining 7,* 2, 20–48.

CORBETT, A. T. AND ANDERSON, J. R. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and User-Adapted Interaction 4,* 4, 253–278.

DESMARAIS, M. C. AND BAKER, R. S. 2012. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction 22,* 1-2, 9–38.

FENG, M., HEFFERNAN, N., AND KOEDINGER, K. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction 19,* 3, 243–266.

GALYARDT, A. AND GOLDIN, I. 2015. Move your lamp post: Recent data reflects learner knowledge better than older data. *Journal of Educational Data Mining 7,* 2, 83–108.

GONG, Y., BECK, J. E., AND HEFFERNAN, N. T. 2010. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *International Conference on Intelligent Tutoring Systems*. Springer, 35–44.

GONZÁLEZ-BRENES, J. 2015. Modeling skill acquisition over time with sequence and topic modeling. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*. 296–305.

HAO, K. 2019. MIT Technology Review. "China has started a grand experiment in AI education. It could reshape how the world learns.". https://www.technologyreview.com/s/614057/ch

ina-squirrel-has-started-a-grand-experiment-in-ai-education-it-could-reshape-how-the.

HOCHREITER, S., BENGIO, Y., FRASCONI, P., AND SCHMIDHUBER, J. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In *A field guide to dynamical recurrent neural networks*, J. F. Kolen and S. C. Kremer, Eds. IEEE Press, 237–243.

KHAJAH, M., LINDSEY, R. V., AND MOZER, M. C. 2016. How deep is knowledge tracing? In *Proceedings of the 9th International Conference on Educational Data Mining*, T. Barnes, M. Chi, and M. Feng, Eds. 94–101.

KINGMA, D. P. AND BA, J. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds.

KOEDINGER, K. R., BAKER, R. S., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., AND STAMPER, J. 2010. A data repository for the EDM community: the PSLC DataShop. In *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy, and R. Baker, Eds. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. Taylor & Francis, 43–56.

KOEDINGER, K. R., BOOTH, J. L., AND KLAHR, D. 2013. Instructional complexity and the science to constrain it. *Science 342,* 6161, 935–937.

KOEDINGER, K. R., BRUNSKILL, E., BAKER, R. S., MCLAUGHLIN, E. A., AND STAMPER, J. 2013. New potentials for data-driven intelligent tutoring system development and optimization. *AI Magazine 34,* 3, 27–41.

KOEDINGER, K. R., CORBETT, A. T., AND PERFETTI, C. 2012. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science 36,* 5, 757–798.

KOEDINGER, K. R., MCLAUGHLIN, E. A., AND STAMPER, J. C. 2012. Automated student model improvement. In *Proceedings of the 5th International Conference on Educational Data Mining*, K. Yacef, O. Zaïane, A. Hershkovitz, M. Yudelson, and J. Stamper, Eds. 17–24.

LAN, A. S. AND BARANIUK, R. G. 2016. A contextual bandits framework for personalized learning action selection. In *Proceedings of the 9th International Conference on Educational Data Mining*, T. Barnes, M. Chi, and M. Feng, Eds. 424–429.

LEE, J. AND YEUNG, D.-Y. 2019. Knowledge query network for knowledge tracing: How knowledge interacts with skills. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*. ACM, 491–500.

LINDSEY, R. V., KHAJAH, M., AND MOZER, M. C. 2014. Automatic discovery of cognitive skills to improve the prediction of student learning. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'14. MIT Press, Cambridge, MA, USA, 1386–1394.

MONTERO, S., ARORA, A., KELLY, S., MILNE, B., AND MOZER, M. 2018. Does deep knowledge tracing model interactions among skills? In *Proceedings of the 11th International Conference on Educational Data Mining*, K. E. Boyer and M. Yudelson, Eds. 462–467.

PANDEY, S. AND KARYPIS, G. 2019. A self-attentive model for knowledge tracing. In *Proceedings of the 12th International Conference on Educational Data Mining*, C. F. Lynch, A. Merceron, M. Desmarais, and R. Nkambou, Eds. 384–389.

PARDOS, Z. A. AND HEFFERNAN, N. T. 2010. Modeling individualization in a Bayesian networks implementation of knowledge tracing. In *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 255–266.

PARDOS, Z. A. AND HEFFERNAN, N. T. 2011. KT-IDEM: introducing item difficulty to the knowledge tracing model. In *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 243–254.

PASZKE, A., GROSS, S., CHINTALA, S., CHANAN, G., YANG, E., DEVITO, Z., LIN, Z., DESMAISON, A., ANTIGA, L., AND LERER, A. 2017. Automatic differentiation in PyTorch. In *Autodiff Workshop at the 31st Conference on Neural Information Processing Systems (NIPS 2017)*. Long Beach, CA, USA.

PAVLIK, P. I., CEN, H., AND KOEDINGER, K. R. 2009. Performance factors analysis –a new alternative to knowledge tracing. In *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling*. IOS Press, 531–538.

PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M., AND DUCHESNAY, E. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12*, 2825–2830.

PELÁNEK, R. 2015. Metrics for evaluation of student models. *Journal of Educational Data Mining 7,* 2, 1–19.

PELÁNEK, R. 2017. Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques. *User Modeling and User-Adapted Interaction 27,* 3-5, 313–350.

PELÁNEK, R., RIHÁK, J., AND PAPOUŠEK, J. 2016. Impact of data collection on interpretation and evaluation of student models. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge*. LAK '16. Association for Computing Machinery, New York, NY, USA, 40–47.

PIECH, C., BASSEN, J., HUANG, J., GANGULI, S., SAHAMI, M., GUIBAS, L. J., AND SOHL-DICKSTEIN, J. 2015. Deep knowledge tracing. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 505–513.

QIU, Y., QI, Y., LU, H., PARDOS, Z. A., AND HEFFERNAN, N. T. 2011. Does time matter? modeling the effect of time with Bayesian knowledge tracing. In *Proceedings of the 4th International Conference on Educational Data Mining*, M. Pechenizkiy, T. Calders, C. Conati, S. Ventura, C. Romero, and J. Stamper, Eds. 139–148.

RITTER, S., YUDELSON, M., FANCSALI, S. E., AND BERMAN, S. R. 2016. How mastery learning works at scale. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale*. L@S '16. Association for Computing Machinery, New York, NY, USA, 71–79.

ROLLINSON, J. AND BRUNSKILL, E. 2015. From predictive models to instructional policies. In *Proceedings of the 8th International Conference on Educational Data Mining*, O. C. Santos, J. G. Boticario, C. Romero, M. Pechenizkiy, A. Merceron, P. Mitros, J. M. Luna, C. Mihaescu, P. Moreno, A. Hershkovitz, S. Ventura, , and M. Desmarais, Eds. 179–196.

ROSÉ, C. P., MCLAUGHLIN, E. A., LIU, R., AND KOEDINGER, K. R. 2019. Explanatory learner models: Why machine learning (alone) is not the answer. *British Journal of Educational Technology 50,* 6, 2943–2958.

SETTLES, B. 2009. Active learning literature survey. Tech. rep., University of Wisconsin-Madison Department of Computer Sciences.

STAMPER, J., NICULESCU-MIZIL, A., RITTER, S., GORDON, G., AND KOEDINGER, K. 2010. Algebra I 2005-2006 and Bridge to Algebra 2006-2007. Development data sets from KDD Cup 2010 Educa-

tional Data Mining Challenge. http://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp.

VAN DER LINDEN, W. J. AND HAMBLETON, R. K. 2013. *Handbook of modern item response theory*. Springer Science & Business Media, New York.

VANLEHN, K. 2011. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist 46,* 4, 197–221.

WILSON, K. H., KARKLIN, Y., HAN, B., AND EKANADHAM, C. 2016. Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation. In *Proceedings of the 9th International Conference on Educational Data Mining*, T. Barnes, M. Chi, and M. Feng, Eds. 539–544.

WILSON, K. H., XIONG, X., KHAJAH, M., LINDSEY, R. V., ZHAO, S., KARKLIN, Y., VAN INWEGEN, E. G., HAN, B., EKANADHAM, C., BECK, J. E., HEFFERNAN, N., AND MOZER, M. C. 2016. Estimating student proficiency: Deep learning is not the panacea. In *Workshop on Machine Learning for Education at the 30th Conference on Neural Information Processing Systems (NIPS 2016)*.

XIONG, X., ZHAO, S., VAN INWEGEN, E. G., AND BECK, J. E. 2016. Going deeper with deep knowledge tracing. In *Proceedings of the 9th International Conference on Educational Data Mining*, T. Barnes, M. Chi, and M. Feng, Eds. 545–550.

YEUNG, C.-K. AND YEUNG, D.-Y. 2018. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. L@S '18. Association for Computing Machinery, New York, NY, USA.

YUDELSON, M. V., KOEDINGER, K. R., AND GORDON, G. J. 2013. Individualized bayesian knowledge tracing models. In *Artificial Intelligence in Education*, H. C. Lane, K. Yacef, J. Mostow, and P. Pavlik, Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 171–180.

ZHANG, J., SHI, X., KING, I., AND YEUNG, D.-Y. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*. WWW '17. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, 765–774.

ZHOU, G., AZIZSOLTANI, H., AUSIN, M. S., BARNES, T., AND CHI, M. 2019. Hierarchical reinforcement learning for pedagogical policy induction. In *Artificial Intelligence in Education*, S. Isotani, E. Millán, A. Ogan, P. Hastings, B. McLaren, and R. Luckin, Eds. Springer International Publishing, Cham, 544–556.