Using Neural Network-Based Knowledge Tracing for a Learning System with Unreliable Skill Tags

Shamya Karumbaiah
Carnegie Mellon University
shamya@cmu.edu

Jiayi Zhang
University of Pennsylvania
jzhang7718@gmail.com

Ryan S. Baker
University of Pennsylvania
ryanshaunbaker@gmail.com

Richard Scruggs
Karolinska Institute
richardtscruggs@gmail.com

Whitney Cade
American Institutes for Research
wcade@air.org

Margaret Clements
American Institutes for Research
pclements@air.org

Shuqiong Lin
American Institutes for Research
slin@air.org

ABSTRACT

Considerable amount of research in educational data mining has focused on developing efficient algorithms for Knowledge Tracing (KT). However, in practice, many real-world learning systems used at scale struggle to implement KT capabilities, especially if they weren't originally designed for it. One key challenge is to accurately label existing items with skills, which often turns out to be a herculean task. In this paper, we investigate whether an increasingly popular approach to knowledge tracing, the use of neural network models, can be a partial solution to this problem. We conducted a case study within a commercial math blended learning system. Using the data collected from middle school students' use of the system over two years, we compare the performance of a neural network-based KT model (DKVMN) in three scenarios: 1) with the original (possibly unreliable) system-provided skill tags, 2) with coarser-grained domain tags based on state standards, and 3) without inputting any mappings between content and skills. Our results suggest that including the system-provided skills in the training of the model leads to the worst performance. The best performance is observed when the skills are entirely disregarded. This supports the possibility of bypassing the laborious step of item-skill tagging in real-world learning systems which were not originally designed to work with KT models, especially if the goal is only to predict the performance of a student on future items. We discuss the implications of our findings for practice and future research.

Keywords

Knowledge Tracing, Skill Tagging, Deep Learning, DKVMN, Adaptive Learning

1. INTRODUCTION

Knowledge tracing has become an essential part of modern adaptive learning systems and even many non-adaptive learning systems. The ability to infer a students' latent knowledge of a skill -- or at least predict their performance on future items within the skill -- has several applications. One of the key applications of knowledge tracing is mastery learning [7]. A system using knowledge tracing for mastery learning divides content into skills (sometimes also called concepts or knowledge components), and once a student starts a skill they cannot advance beyond it without demonstrating that they have mastered the skill. Knowledge tracing is also often used for creating reports for teachers [31] or open learner models for students [4]. A third use of knowledge tracing is as a component in automated detectors that recognize a range of student behaviors or states, including help-seeking strategies [1], and disengaged behaviors such as gaming the system [18].

While the first learning systems to use knowledge tracing were designed with that function in mind [2], many real-world learning systems used at scale today were not designed with knowledge tracing or adaptive learning originally in mind. Increasingly, these systems are being retrofit to use some form of these types of functionality [16]. One of the key ways that these systems often differ from learning systems originally designed for use with knowledge tracing is in how content is mapped to skills (often referred to as skill-item mappings or as KC mappings -- KC stands for knowledge component – [13]). Systems designed from the start to use KT first select what skills are to be included, and then develop items tailored to those skills [2]. Afterwards, there may even be a process of using data to refine the KC mapping, attempting to improve the correlation of items to each other while maintaining human comprehensibility of the overall mapping [12, 14, 15]. By contrast, when content is retrofit for use with knowledge tracing, items are created first, and then the items are labeled with skills.

Labeling an existing item with a skill is much harder than creating a new item for a skill. Often, items have been developed by multiple authors over time, or have come from different original sources such as different textbooks. Mapping this disparate content -- sometimes tens of thousands of items -- to a set of skills can be a herculean task. In many cases, items have been tagged in terms of governmental curricular standards, but these standards are typically much coarser-grained than the types of skills used within knowledge tracing models [6]. Therefore, there is a challenge to using many of the classic approaches to knowledge tracing, used at scale in past real-world systems, with content not initially designed with this use in mind.

In this paper, we investigate whether an increasingly popular approach to knowledge tracing, the use of neural network models [12, 17, 33], can be a partial solution to this problem. Unlike earlier approaches to knowledge tracing (i.e. [8, 19]), neural network models do not require a KC model in order to predict student performance on future items. As such, it may be possible to bypass the step of developing a KC model entirely, at least for goals such as choosing what problem to give the student next. In investigating this, we select the DKVMN (Dynamic Key-Value Memory Network – [33]) algorithm, a generally successful early approach to neural network-based knowledge tracing, as it has the ability to both use an existing KC model *and* to fit a new KC model, giving us the ability to compare between using a KC model known to have considerable limitations and using no KC model at all.

Towards this goal, we conduct a case study within a commercial math blended learning system. Using the data collected from middle school students' use of the system over two years, we compare the performance of KT models in three scenarios: 1) with the original (possibly unreliable) system-provided skill tags, 2) with coarser-grained domain tags based on state standards, and 3) without inputting any mappings between content and skills. Our results suggest that including the system-provided skills in the training of the model leads to the worst performance. The best performance is observed when the skills are entirely disregarded. This supports the possibility of bypassing KC modelling in real-world learning systems which were not originally designed to work with KT models, especially if the goal is only to predict the performance of a student on future items.

2. MOTIVATION

2.1 Knowledge Tracing

There have been a range of algorithms used for knowledge tracing over the last three decades. The first widely-used knowledge tracing algorithms relied on a KC model, as mentioned in the introduction. Perhaps the first widely-used algorithm for knowledge tracing was Bayesian Knowledge Tracing [8], based on a simple Markov Model (and also mathematically equivalent to a simple Bayesian Network – [27]). More recently, models based on logistic regression have become popular in the literature [5, 20] although they remain rare within real-world use (but see [16]). Algorithms related to item response theory (IRT) such as Elo [21] and temporal IRT [30] have also become more widely seen in the literature recently, and are used at scale in several learning systems [3, 22, 30]. While Elo and temporal IRT can be used without a KC model, typically a separate Elo model is used for each of several skills.

A contrasting set of approaches uses neural networks to avoid the need for a KC model. The first member of this family of algorithms, deep knowledge tracing [24], discovered complex item-to-item mappings as part of predicting future performance on items. Later approaches such as DKVMN introduced the ability to use or fit skill-item mappings (KC models) as well [33]. The last five years have seen a proliferation of knowledge models based on neural networks [11, 17, 29, 33], gaining increasing ability to predict future performance. However, due to concerns about unpredictable behavior [9, 32], and the challenge of using this type of models for mastery learning and reporting on student skill (as discussed above, the main applications of knowledge tracing in contemporary learning systems), neural networks have been much more popular in the published literature on knowledge tracing than in actual real-world use. One of the key limitations to neural network models in this regard has been that they predict correctness on specific problems but do not map that back to inferring proficiency on human-interpretable skills. Recent extensions have taken this step, suggesting the potential for broader real-world use and uptake of neural network knowledge tracing models. In particular, an extension in [28] can be applied to any neural network knowledge tracing model for which there is a KC model available (even at the coarse level of state standards).

2.2 Case Study: A Commercial Learning System with Imperfect Skill Tags

This study uses data from a commercial math blended learning platform used in schools across the United States by students from kindergarten through Algebra II. This platform allows teachers to assign problem sets to students by standards and topics, and assignments can be personalized based on the teacher's assessment of a student's progress. The system measures a student's mastery of underlying mathematical concepts through Bayesian Knowledge Tracing (BKT), and may suggest more foundational or advanced assignments based on a student's performance. In this way, the system contains several elements frequently seen in intelligent tutoring systems. Assignments may be worked on at home or at school, and guidance provided by the creators of the platform suggest that students should work in the platform for approximately 30 minutes per week.

The learning platform has two classes of problems, authored in different ways. In the "classic" problems, students answer multiple-choice, multi-step word problems, with a series of scaffolded, smaller word problems triggered when the student answers the initial problem incorrectly. The second "new" class of problems tends to be more interactive: for example, students plot points and lines on a graph, rotate shapes, and sort statements or numbers using drag-and-drop functionality. These problems tend to be open-response and typically lack the kind of scaffolding seen in the classic problems.

The difference in the content within this system can be attributed to the development of the content in two phases. In a first phase of development, a team of math content specialists created multiple-choice items with scaffolding for incorrect answers (based on the model in [26]). In this model, students that answer an item correctly will move onto the next problem, but if a problem is answered incorrectly initially, the problem will be broken into a series of additional, smaller questions that scaffold an effective set of problem-solving steps. The math content specialists tagged each item with a set of skills based on mathematics standards. After the acquisition of the system by a different company, and the subsequent departure of the original development team, a different approach was adopted to build out additional content for the same courses. In this new approach, existing content created by an outside company was added to the system. This additional content had more item types, as discussed above. However, this new content did not use the same scaffolding model as the original content. Items had been tagged by the outside company according to the same overall mathematics standards, but there were considerable inconsistencies and incompatibilities between the two tagging approaches. The result was an overall set of skill tags that were of uncertain usability.

While the case of a system being acquired by a different company and adopting a new process for content is perhaps somewhat unusual, the overall problem of content and skill tags being developed by different teams over time is not unusual. Many contemporary online learning systems integrate data from different textbooks, shift the membership of content authoring teams over the span of many years, and tag content according to multiple state standards and internal content schemas. While many of the systems which are most heavily published on at this conference (and related conferences) do not have these issues, it is likely that the majority of learning systems used at large scale suffer from these issues to at least some degree.

3. METHODS

3.1 Data Collection

The analyses presented here are based on data collected as part of a larger, exploratory study on the implementation of the blended learning platform discussed in section 2.2. Though the focus of the study was on 6th-8th grade students, we also collected data from sections of Algebra I and Geometry students when a teacher taught across several grade levels and wanted to use the platform across their classes. These data were collected in the 2018-19 and 2019-20 school years from students located in 19 schools across Texas. These schools tended to be located in less populated areas, with 11% of schools in the sample in cities (according to Federal classification), 5% of schools in the sample in suburbs, 26% in towns, and 58% in rural areas. 41% of the students in our sample schools identified as white, 9% as African-American/Black, 46% as Hispanic, and 1% as Asian. 60% of students in sample schools qualified for free/reduced price lunches.

Data collected occurred across two years, with 38 teachers and 2069 students participating in the first year, and 14 teachers and 743 students participating in the second year. Teachers were instructed to have students engage with the blended learning software for approximately 30 minutes each week, though use of the system diminished over the course of the school year (cf. [23]). The primary goal of the larger study was to examine the naturalistic relationships between learning, engagement, and classroom implementation, so researchers interfered very little in how teachers and students decided to use the platform. Teachers used the software differently: some teachers sat their students in front of the system, offering little help, while others used the system while teaching to the class, talking about the problems presented. Some teachers disliked the scaffolding problems and checked students' answers to top-level problems to avoid their students receiving scaffolding problems. In some classes, students asked each other for help, while in others they were expected to work alone.

3.2 Descriptive Statistics

This analysis involves data from 2564 students. Table 1 presents the number of problems, skills, and first attempts for classic and new content. The overall correctness of students' first attempts on problems is 32% across all the problems. However, when the content is categorized based on the new and classic content (see description in Section 2.2), we see that the percentage correctness is higher for classic content (59%) with the new content having a much lower correctness (15%). There are several possible explanations for this finding. Students may be more likely to get scaffolding problems correct than initial problems, as scaffolding problems are designed to have lower difficulty. In addition, the more interactive problems often require the student to take several actions or input multiple steps in order to arrive at the correct final answer, which creates more opportunities for mistakes. Due to the clear differences in these two types of learning content, we will look at these separately in our final experiment.

Table 1. Total number of problems, skills, first attempts, and percentage correctness for all content, classic content, and new content in the data used in this analysis

Content Type	**#problems**	**#skills**	**#first attempts**	**%correct**
All content	7252	2,237	110,214	32%
Classic content only	1764	532	42,353	59%
New content only	5488	1,705	67,861	15%

3.3 DKVMN

In this analysis, we use Dynamic Key-Value Memory Networks (DKVMN; [33]) for knowledge tracing. DKVMN is a KT model developed based on neural networks and has demonstrated improved performance compared to traditional KT models, such as Bayesian Knowledge Tracing or Performance Factors Analysis [25, 28, 33]. In DKVMN, the model employs two matrices to predict student performance on items and estimate mastery on a set of automatically-derived skills. By learning from the relationships between the two matrices (i.e, a static matrix that stores skill relationships and a dynamic matrix that stores and updates mastery level), the algorithm generates underlying skills associated with items and identifies connections between them, creating a skill map. DKVMN utilizes the map to make predictions on student performance and estimates mastery learning, as opposed to the human-generated skill-item mapping that the traditional KT models rely on. Given DKVMN's capability of automatically generating underlying skills and exploiting the relationships between them, we are interested in investigating how the algorithm can be applied to handle data with unreliable skill tagging.

4. ANALYSIS

The goal of our analysis in this paper is to investigate the use of a neural network-based KT model (DKVMN) that doesn't require skill tags, for a system where the skill tags may be unreliable or inconsistent. Accordingly, we compare the performance of KT models with and without mappings between content and skills. Since we don't have out-of-system performance data on students to validate knowledge estimates, we infer performance within the system rather than outside the system (note that DKVMN can also be used to predict out-of-system performance – [28]). Thus, we validate the model based on its ability to predict a student's correctness on the next problem. We use [33]'s implementation of the DKVMN model with a set of hyperparameters that have been reported to produce optimal outcome for a previously collected dataset (the ASSISTments2009 data set in [33]), including a state dimension of 50 and a memory size of 20. We use this previously established set of hyperparameters instead of tuning them, to avoid overfitting. In addition, since the goal of the paper is to study whether the skill tags are useful and whether DKVMN can compensate for a lack of good skill tags in a data set of this nature, tuning the hyperparameters to find the best performing KT model is not related to the goals of the analysis. We evaluate our model using AUC ROC. We perform a student-level 10-fold cross validation, and the reported results are the average across the CV folds.

Using this approach, we conduct the following three kinds of analysis:

1) Experiment #1 (Default-Skills): In this experiment, we chose to include the initial skill mapping provided by the

learning system during the training of the DKVMN model. The system provides a total of 2237 skills.

2) Experiment #2 (Domains): In this experiment, instead of using the default skills, we group the skill tags provided by the learning system into coarser-grained domains defined by the Common Core Standards [6]. We derive domain information from the nomenclature used within the standards (e.g., exponential functions, quadratic equations, and two-dimensional shapes). This reduced the 2237 skills to 24 domains.

3) Experiment #3 (No-Skill): In this experiment, we disregard the skill tags entirely and instead treat all problems as a single skill. As discussed above, DKVMN will then find a latent mapping of skills to the problems on its own.

We repeat these three experiments for three data setups: 1) with all content, 2) classic content only, and 3) new content only. In the last two settings, we separate the training and testing data to only include new content or classic content since we have some evidence that data for these two types of content is qualitatively different (see discussion in Section 2.2).

5. RESULTS

Table 2 presents the results of our experiments. Here we summarize our observations:

1) Experiment #1 (Default-Skills): We observe that including the system-provided skills in the training of the DKVMN model leads to the worst performance in all three cases (all content, classic content, new content). For all content and classic content models, the model performance as measured by cross-validated AUC ROC is between 0.5 and 0.6 (0.564 and 0.532), relatively modest improvements on chance and much lower than is typically seen for DKVMN (e.g. [25, 33]).

2) Experiment #2 (Domains): When the system-provided skills are grouped into coarser-grained domain mappings, we observe a big improvement in DKVMN's performance. For example, there is a 0.206 increase in AUC when the model trained on all content uses the domain tags instead of skill tags (0.770 vs. 0.564). Similarly, there is a big increase in AUC of 0.233 for the model trained on new content (0.900 vs. 667). However, there is a relatively smaller increase in AUC of 0.084 for the model trained on classic content (0.616 vs. 0.532).

3) Experiment #3 (No-Skill): Finally, disregarding the skills entirely leads to the best performance in all three cases: 0.821 for all content, a still relatively unimpressive 0.634 for classic content, and 0.920 for new content.

4) If we compare the results of the DKVMN models separately for the new and classic content, we observe that the model trained only on new content perform much better than models trained only on classic content (e.g, 0.667 vs. 0.532 with default skills and 0.920 vs. 0.634 with no skills)

Table 2. Average AUC of DKVMN models across 10-fold cross-validation for the four experiments conducted using default skills (#1), domains (#2), no skills (#3), and for new and classic content separately.

Content Type	**Default-Skills** (exp #1)	**Domains** (exp #2)	**No-Skill** (exp #3)
All content	0.564	0.770	0.821
Classic content only	0.532	0.616	0.634
New content only	0.667	0.900	0.920

Overall, we also see that the DKVMN model trained without the skill information on all content (0.821) or new content (0.920) has a comparable or better performance than the best-performing models reported on some benchmark datasets (0.827 for Synthetic-5, 0.816 for ASSISTments2009, 0.727 for ASSISTments2015, 0.828 for Statics2011; [33]). With the exception of ASSISTments2015 dataset, the best performing models for the other three datasets have achieved AUCs between 0.80 and 0.83. Our best-performing model for all content combined (0.821) is comparable to the performance of the best models on these benchmark datasets. The best model trained only on new content has higher performance (0.920) than the best models on benchmark datasets. However, the best model for the classic content has much lower performance (0.634) than previous uses of DKVMN.

6. DISCUSSION

Knowledge tracing models are often used in learning systems to estimate students' knowledge of a skill. In some cases, this is operationally defined as simply predicting whether or not a student will get the next problem (or a specific problem) right. Accordingly, considerable amount of research has focused on developing efficient algorithms for knowledge tracing. However, in practice, many real-world learning systems used at scale are difficult to implement knowledge tracing for, especially if the system or content was not originally designed for use with a skill model. The key challenge is to accurately label existing items with skills, which often turns out to be a herculean task (Section 1). Little research has explored ways to address this practical constraint limiting the use of knowledge tracing models at scale. In this paper, we investigate whether an increasingly popular approach to knowledge tracing, the use of neural network models, can be a partial solution to this problem.

Our analysis investigates the use of a neural network-based KT model (DKVMN) that doesn't require skill tags (and can even automatically assign its own skill tags) to bypass the step of retrofitting content with skills. We conduct a case study of a commercial math blended learning system which has a potentially unreliable and/or inconsistent skill tagging, due to the content and skill tags being developed by different teams over time (Section 2.2). We collected data from 6th-8th grade students' system usage over two years within 19 schools across Texas. We compare the performance of DKVMN in three scenarios: 1) with the potentially unreliable default system-provided skill tags, 2) with coarser-grained domain tags based on state standards, and 3) without inputting any mappings between content and skills. We also investigate differences in predictive performance for two disparate content types in the system: classic content (multiple choice, with scaffolds) and new content (more interactive, open-response, with no scaffolds).

Our results suggest that including the system-provided default skills in the training of the DKVMN model leads to the worst performance at predicting future student performance within the learning system. The AUC ROC for this case is much lower than what is typically seen for DKVMN. Big improvements in performance are observed when the system-provided skills are grouped into coarser-grained domain mappings. However, the best performance is obtained when the skills are entirely disregarded. To our surprise, there is a noticeable difference in the improvement between the classic and new content, with relatively smaller increase in AUC for the model trained on classic content.

6.1 Implications

The objective of our investigations was to explore the possibility of bypassing the herculean task of item-skill tagging (KC model) for real-world learning systems which were not originally designed to work with KT models. Our results provide some support to this possibility, especially if the goal is only to predict the performance of a student on future items. For example, if a KT model is being developed for a system for optimal next problem selection, using a neural network-based model like DKVMN and ignoring the skill tags may be more effective than using unreliable system-provided skills.

However, our results were not consistent across all content, suggesting that this approach may not be sufficient for all legacy content. Despite using DKVMN, the best model for classic content still achieves an AUC that is much lower than is typically seen for neural network-based KT models. Therefore, careful consideration is needed before making the decision on deploying such a model to make real-time decisions on what content the student will see next. In comparison, the DKVMN model trained on new content without any skill information is at par with the best models on benchmark KT datasets. It is difficult at this point to explain why DKVMN was so much more successful for new content than for classic content. There is no obvious reason why either of the two clear attributes of the older content – scaffolding problems and the use of multiple-choice – would lead to poorer prediction by DKVMN, given the past successful use of DKVMN and related algorithms on content with these attributes.

Though these findings suggest that DKVMN may be useful for legacy systems with unreliable or inconsistent skill tags, it is important to keep a few things in mind before assuming that these findings will apply to other contexts. First, this is a case study of one commercial learning system. These results need to be replicated in other content types, system design, subject matters, student demographics, etc. We have made the code public at (redacted for review) to aid replication. Second, a better performing model may still not necessarily serve all student subgroups equitably. Before actually applying an algorithm, it is necessary to investigate it for potential biases that could lead to discriminatory behaviors. In this case, a biased KT model could deliver content below a student's actual skill level more often for students from certain subgroups, leading to missed learning opportunities. Since neural network models are prone to overfitting due to their complex decision boundaries, it is particularly important to investigate whether a model like DKVMN generalizes less well for students who are unrepresented in the training data. For instance, it may be that a neural network model captures complex interrelationships between skills that only occur in students with specific past curricular experiences.

6.2 Limitations and Future Work

There are a few limitations to the analysis conducted in this paper. First, since the learning system itself made pedagogical decisions based on a knowledge tracing model using unreliable skill tags, the problems it gave students were probably not always appropriately chosen. As such, the data may not represent students participating in mastery learning (although this is also true of the data sets used to initially train KT for mastery learning in other commercial systems). Second, we chose to explore DKVMN because it has generally been reported to be successful, and generates its own skill mapping. A more comprehensive comparison between other neural network-based models could be helpful in understanding whether other algorithms can perform better for the old content. Lastly, before disregarding non-neural network-based KT models for the case discussed in this paper, it may be worth comparing these results with newer extensions proposed for classic KT models. For example, future work could explore BKT variants that include skill refitting (cf. [12]).

Future work also may be able to shed more light on the contexts where this approach does and does not work. For instance, why is there a noticeable difference in the improvement between the classic and new content with relatively smaller increase in AUC for the model trained on classic content? Exploring answers to this question may help identify cases where this approach may be more efficient than others and identify ways to improve it.

Finally, like most neural network models, DKVMN lacks easy interpretation. This could make it harder for instructors to trust its recommendations and hard to troubleshoot. A potential solution is to interpret the output of the DKVMN model. Considering that DKVMN creates its own internal tag-concept mapping, this study's results suggest that it may be worth studying those mappings in greater detail. Since the no-skill model outperformed the default-skill model to such a degree in this dataset, it is likely that DKVMN's distilled concepts represent accurate relationships between the problems in the data. Zhang and colleagues [33] mention the possibility of using DKVMN for concept discovery, but most subsequent work on the algorithm has instead focused on its predictive accuracy (but see [10], who discuss interpretability). Additional research on DKVMN's process of concept discovery could not only improve its interpretability but potentially also allow for better automated skill tagging and more accurate student skill level estimation.

7. ACKNOWLEDGMENTS

The research reported here was supported by the IES grant R305A170167. The opinions expressed are those of the authors and do not represent views of IES.

8. REFERENCES

[1] Aleven, V., McLaren, B., Roll, I., & Koedinger, K. (2006). Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education*, *16*(2), 101-128.

[2] Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The journal of the learning sciences*, *4*(2), 167-207.

[3] Brinkhuis, M., Cordes, W., & Hofman, A. (2020). Governing games Adaptive game selection in the Math Garden. In *ITM Web of Conferences* (Vol. 33, p. 03003).

[4] Bull, S., & Mabbott, A. (2006, June). 20000 inspections of a domain-independent open learner model with individual and comparison views. In *International conference on intelligent tutoring systems* (pp. 422-432). Springer, Berlin, Heidelberg.

[5] Choffin, B., Popineau, F., Bourda, Y., & Vie, J. J. (2019, July). DAS3H: Modeling Student Learning and Forgetting for Optimally Scheduling Distributed Practice of Skills. In *International Conference on Educational Data Mining (EDM 2019).*

[6] Common Core (2022) Mathematics Standards. Accessed March 8, 2022 from http://www.corestandards.org/Math/.

[7] Corbett, A. (2001). Cognitive computer tutors: Solving the two-sigma problem. In *International Conference on User Modeling* (pp. 137-147).

[8] Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, *4*(4), 253-278.

[9] Ding, X., & Larson, E. C. (2019). Why Deep Knowledge Tracing Has Less Depth than Anticipated. *Proc. International Conference on Educational Data Mining.*

[10] Ding, X., & Larson, E. C. (2021). On the Interpretability of Deep Learning Based Models for Knowledge Tracing. ArXiv:2101.11335 [Cs].

[11] Ghosh, A., Heffernan, N., & Lan, A. S. (2020, August). Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2330-2339).

[12] Khajah, M., Lindsey, R. V., & Mozer, M. C. (2016). How Deep is Knowledge Tracing?. *Proc. International Conf Educational Data Mining*.

[13] Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The Knowledge-Learning-Instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, *36*(5), 757-798.

[14] Koedinger, K. R., Stamper, J. C., McLaughlin, E. A., & Nixon, T. (2013, July). Using data-driven discovery of better student models to improve student learning. In *International conference on artificial intelligence in education* (pp. 421-430).

[15] Liu, R., & Koedinger, K. R. (2017). Closing the Loop: Automated Data-Driven Cognitive Model Discoveries Lead to Improved Instruction and Learning Gains. *Journal of Educational Data Mining*, *9*(1), 25-41.

[16] Maier, C., Baker, R.S., Stalzer, S. (2021) Challenges to Applying Performance Factor Analysis to Existing Learning Systems. *Proceedings of the 29th International Conference on Computers in Education*.

[17] Pandey, S., & Karypis, G. (2019). A self-attentive model for knowledge tracing. *arXiv preprint arXiv:1907.06837*.

[18] Paquette, L., Baker, R.S. (2019) Comparing machine learning to knowledge engineering for student behavior modelling: A case study in gaming the system. *Interactive Learning Environments*, 585-597.

[19] Pavlik, P. I., Cen, H., & Koedinger, K. R. (2009). Performance Factors Analysis--A New Alternative to Knowledge Tracing. In *Proceedings of the 2009 conference on Artificial Intelligence in Education.*

[20] Pavlik, P. I., Eglington, L. G., & Harrell-Williams, L. M. (2021). Logistic Knowledge Tracing: A Constrained Framework for Learner Modeling. IEEE Transactions on Learning Technologies, 14(5), 624-639.

[21] Pelánek, R. (2016). Applications of the Elo rating system in adaptive educational systems. *Computers & Education*, *98*, 169-179.

[22] Pelánek, R., Papoušek, J., Řihák, J., Stanislav, V., & Nižnan, J. (2017). Elo-based learner modeling for the adaptive practice of facts. *User Modeling and User-Adapted Interaction*, *27*(1), 89-118.

[23] Phillips, A., Pane, J. F., Reumann-Moore, R., & Shenbanjo, O. (2020). Implementing an adaptive intelligent tutoring system as an instructional supplement. *Educational Technology Research and Development*, *68*(3), 1409-1437.

[24] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J., & Sohl-Dickstein, J. (2015). Deep knowledge tracing. *Advances in neural information processing systems*, *28*.

[25] Pu, S., Converse, G., & Huang, Y. (2021, June). Deep Performance Factors Analysis for Knowledge Tracing. In *International Conference on Artificial Intelligence in Education* (pp. 331-341).

[26] Razzaq, L., & Heffernan, N. T. (2006). Scaffolding vs. hints in the Assistment System. In *International Conference on Intelligent Tutoring Systems* (pp. 635-644).

[27] Reye, J. (2004). Student modelling based on belief networks. International Journal of Artificial Intelligence in Education, 14, 63–96.

[28] Scruggs, R., Baker, R.S., McLaren, B.M. (2020) Extending Deep Knowledge Tracing: Inferring Interpretable Knowledge and Predicting Post System Performance. *Proceedings of the 28th International Conference on Computers in Education*.

[29] Shin, D., Shim, Y., Yu, H., Lee, S., Kim, B., & Choi, Y. (2021, April). Saint+: Integrating temporal features for ednet correctness prediction. In *LAK21: 11th International Learning Analytics and Knowledge Conference* (pp. 490-496).

[30] Wilson, K. H., Karklin, Y., Han, B., & Ekanadham, C. (2016). Back to the Basics: Bayesian Extensions of IRT Outperform Neural Networks for Proficiency Estimation. *Proc. International Conference on Educational Data Mining.*

[31] Xhakaj, F., Aleven, V., & McLaren, B. M. (2017, September). Effects of a teacher dashboard for an intelligent tutoring system on teacher knowledge, lesson planning, lessons and student learning. In *European conference on technology enhanced learning* (pp. 315-329).

[32] Yeung, C. K., & Yeung, D. Y. (2018, June). Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale* (pp. 1-10).

[33] Zhang, J., Shi, X., King, I., & Yeung, D. Y. (2017, April). Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web* (pp. 765-774).