Going Online: A simulated student approach for evaluating knowledge tracing in the context of mastery learning

Qiao Zhang
Drexel University
Philadelphia, PA, USA
qiao.zhang@drexel.edu

Christopher J. Maclellan
Drexel University
Philadelphia, PA, USA
christopher.maclellan@drexel.edu

ABSTRACT

Knowledge tracing algorithms are embedded in Intelligent Tutoring Systems (ITS) to keep track of students' learning process. While knowledge tracing models have been extensively studied in offline settings, very little work has explored their use in online settings. This is primarily because conducting experiments to evaluate and select knowledge tracing models in classroom settings is expensive. To fill this gap, we introduce a novel way of using machine-learning models to generate simulated students. We conduct experiments using agents generated by the Apprentice Learner Architecture to investigate the online use of different knowledge tracing models (Bayesian Knowledge Tracing, the Streak model, and Deep Knowledge Tracing). An analysis of our simulation results revealed an error in the initial implementation of our Bayesian knowledge tracing model that was not identified in our previous work. Our simulations also revealed a more fundamental limitation of Deep Knowledge Tracing that prevents the model from supporting mastery learning on multi-step problems. Together, these two findings suggest that Apprentice agents provide a practical means of evaluating knowledge tracing models prior to more costly classroom testing. Lastly, our analysis identifies a positive correlation between the Bayesian knowledge tracing parameters estimated from human data and the parameters estimated from simulated learners. This suggests that model parameters might be initialized using simulated data when no human-student data is yet available.

Keywords

Computational Models of Learning, Simulated Students, Knowledge Tracing

1. INTRODUCTION

Intelligent Tutoring Systems (ITS) are used within K-12 education to improve learning outcomes. In addition to providing students with scaffolding and feedback, tutors utilize an approach called *knowledge tracing* to estimate what students know and do not know [2]. When combined with a problem selection policy [16], knowledge tracing enable tutors to support mastery learning and to focus students practice where it is most needed (i.e., on the skills they do not yet know rather than the skills they already know). While many studies have explored knowledge tracing for offline evaluation (fitting knowledge tracing models to existing data sets), there is comparatively little work on evaluating these algorithms in online settings (evaluating how well these algorithms estimate students' mastery from just a few data points and decide when to stop giving them additional problems).

We aim to understand which knowledge tracing models yield the greatest mastery learning efficiency in online settings. Additionally, we want to find out how the parameters for knowledge tracing models can be selected before human data is collected. To meet our need for multiple experiments to investigate our knowledge tracing questions, we introduce a novel way of using computational models of learning, or simulated student models that learn from interactions with a tutor just like human students do, to simulate our knowledge tracing experiments. We use the Apprentice Learner architecture [10], a machine-learning framework that aims to model how humans learn from examples and feedback to generate simulated students and conduct experiments.

To explore the feasibility of this approach, we conducted experiments to compare Bayesian Knowledge Tracing (BKT)[2] to the Streak model [7] and Deep Knowledge Tracing (DKT) [15]. Our simulations show that both BKT and Streak stop before giving all the problems, but that BKT is slightly more aggressive than Streak and seems to assume students have mastered skills a bit earlier than expected. Upon further inspection, our analysis revealed a bug in our underlying implementation of BKT (which we fixed for this study). Further, we found that DKT exhibits strange behavior that makes it unusable in certain cases of mastery learning and problem selection. This limitation of DKT for mastery learning has not been identified in prior work. These findings demonstrate that simulation students might serve a valuable role in testing knowledge tracing models before more costly classroom deployments.

We also explore the use of simulated data from these experiments to estimate initial parameters for the BKT model. Prior to collecting human data, knowledge tracing parameters are often set to reasonable hand-picked defaults. A better approach is to run a pilot study with human students

Qiao Zhang and Christopher MacLellan "Going Online: A simulated student approach for evaluating knowledge tracing in the context of mastery learning". 2021. In: Proceedings of The 14th International Conference on Educational Data Mining (EDM21). International Educational Data Mining Society, 331-337. https://educationaldatamining.org/edm2021/
EDM '21 June 29 - July 02 2021, Paris, France

to collect data for model training, which requires additional time and labor. Our analysis identifies a positive correlation between the BKT parameters estimated from the simulated data and those estimated from human data, suggesting simulated data might be used to initialize parameters.

2. BACKGROUND

2.1 Knowledge Tracing

The main purpose of knowledge tracing is to track students' skill mastery and predict their future performance based on their past activity. Knowledge tracing uses labels of the skills needed for each step, which we call Knowledge Components (KCs), and students' first attempt correctness (correct or incorrect) to predict whether the students will correctly solve a new problem containing the same KC.

2.2 Bayesian Knowledge Tracing

BKT [2] is a well-known knowledge tracing algorithm that can estimates whether students have learned a particular skill given their past performance on opportunities to practice that skill. It models student knowledge using a Hidden Markov Model, where the hidden state is estimated from observations of students' correctness on each step. For each skill, a student can be in one of the two possible knowledge states: "unknown" or "known". A binary response (correct or incorrect) is generated at each opportunity a student practices a skill [5]. Although, BKT supports the ability to model forgetting, it is typically assumed that students never forget what they have mastered [16]. If a student reaches 95% probability of being in the known state for a skill, the skill is marked as being mastered by the student. With these assumptions, the BKT model has four parameters.

- P(L_0): initial probability of mastery ("known").
- P(T): probability of learning the skill ("learn").
- P(G): probability of guessing the answer ("guess").
- P(S): probability of making a mistake ("slip").

Researches have created variants of the BKT model. Yudelson et al. [20] introduced an individualized BKT model that can take student differences in initial mastery and skill learning probabilities into account. Nedungadi et al. [13] created PC-BKT (Personalized and Clustered), which has individual priors for each student and skill, and dynamically clusters students based on learning ability. This prior work aims to improve predictive performance over original BKT. Despite the quantity of research on BKT models, there is relatively little evaluation in online settings.

2.3 Streak Model

Another knowledge tracing approach that is popular for use within mastery learning is the Streak model. Also known as "three-in-a-row" [7], it is a relatively simple and intuitive model since it only has one parameter, how many correct answers in a row equates to mastery. It was first applied in ASSISTments and the key idea was to keep giving the student questions until some proficiency threshold was reached. The default setting was "three correct in a row" but this could be manipulated by teachers.

2.4 Deep Knowledge Tracing

DKT [15] is a knowledge tracing model that has been receiving increasing attention. This model leverages information about students sequence of steps and correctness on those steps to predict performance on subsequent steps. Additionally, DKT leverages information about performance on one skill to improve predictive performance on other skills. DKT uses a long short-term memory (LSTM) architecture, which is a kind of recurrent neural network that allows for the modeling of non-Markovian processes. Recent work evaluating DKT [6, 8] suggests that DKT often outperforms BKT in terms of predictive performance. Despite this promising finding, there has been very little work exploring the use of DKT within online knowledge tracing (e.g., for mastery learning within a tutor). Beyond the original DKT work [15], which explores the use of DKT for next step recommendation, we are unaware of any research programs that currently uses DKT in this way.

Finally, it is worth noting that knowledge tracing is not strictly necessary for tutoring systems. Many tutors either use a fixed problem sequence or present a fixed number of problems in a random order. However, we hypothesize that knowledge tracing in conjunction with mastery learning component is one of the main components of tutors that makes tutors effective.

2.5 Computational Model of Learning

The Apprentice Learner Architecture is a framework for modeling human learning from demonstrations and feedback in educational environments [10]. We use an Apprentice model previously developed in prior work; see [11] for a complete description of the model. Most work in the field of educational data mining focuses on building mathematical, predictive models of learning. In contrast, the Apprentice models actually perform the task (not just predict performance). They induce task-specific knowledge from the demonstrations and feedback they receive. Apprentice models are ideal for the current study because they do not require prior human data to operate. They can predict learning and behavior based solely on the task structure.

3. METHODOLOGY

We created 30 simulated students (Apprentice agents) to solve problems in a fraction arithmetic tutor (tutor presented in [14]). The tutor had three different types of problems: Add Different (AD), add fractions with different denominators; Add Same (AS), add fractions with same denominators; Multiplication (M), multiply two fractions.

3.1 Experiment Design

Our study had six conditions: Random, Streak, BKT_default, BKT_random, BKT_human and DKT_random. There were four types of conditions: Random, BKT, Streak, and DKT, which differ in the way they select the next problem to give to a simulated student. In the Random condition every problem was assigned only once in random order and the training ends when problems run out. Since Random gives the most training, it produces the highest correctness prediction by the end of practice. We use Random as a baseline for evaluating other models.

The other conditions use the respective knowledge tracing approaches for mastery learning and problem selection. During problem selection, each knowledge tracing model randomly chooses a problem with at least one unmastered skill

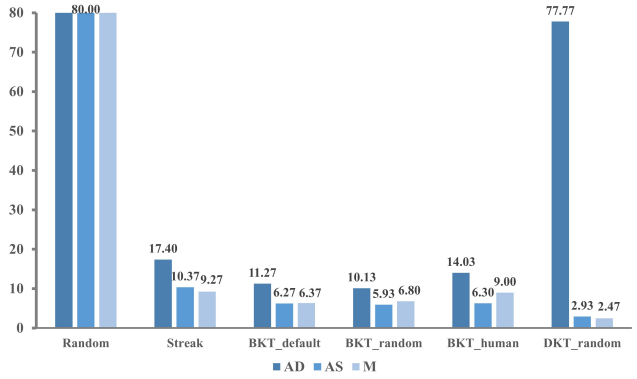

Figure 1: Num. problems given in each condition (by type).

and updates the student's mastery level based on the result. The training ends once the proficiency threshold is reached (95% in all cases but Streak, where it was 3 in a row).

The parameters in BKT_default were manually set based on our prior experience with BKT. We set P(L_0) to 0.1, P(T) to 0.05, P(G) to 0.05 and P(S) to 0.02. These parameters are identical for each KC. The BKT_random and BKT_human parameters were estimated using the BKT module on LearnSphere [9]. The human-based parameters were obtained from fitting BKT to the "Fraction Addition and Multiplication" dataset accessed via DataShop (Koedinger et al., 2010). The random-based parameters were obtained from fitting BKT to the log data generated from simulated students in the Random condition.

To support the use of DKT within online master learning, we created our own implementation using PyTorch's LSTM module.[1] Based on prior work [8], the model has 200 nodes in the hidden layer, uses a dropout of 0.4 during training, and uses a batch size of 5 (our sequences were longer than those in prior work, so a smaller batch size works well). This implementation supports the ability to fit DKT to data presented in standard DataShop [9] format. Trained models have a simple interface for use in online knowledge tracing settings. Similar to BKT, we fit DKT to the log data generated from simulated students in the Random condition to estimate model parameters.

3.2 Simulation Studies and Evaluation

During the experiment process, we created 30 simulated students for each of the six conditions and analyzed the data that they generated. For these experiments, we created a KC model that labels each step as a combination of "Problem Type" and "Selection". There are 14 unique KCs in our analysis, 8 for Add Different, 3 for Add Same and 3 for Multiplication. As the Additive Factors Model (AFM) is often used to examine learning curves from existing data [1], we used pyAFM [12] (a python implementation of AFM) to predict the probability that students will get a next step with the respective skill correct at the end of their practice. This provided an independent means for us to estimate how well each knowledge tracing approach did at appropriately recognizing when students had achieved mastery.

[1]Open-source code for the model is available here: https://gitlab.cci.drexel.edu/teachable-ai-lab/dkt_torch.

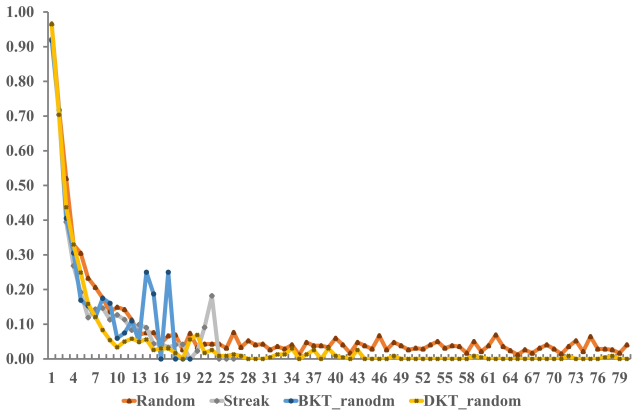

Figure 2: Learning curves for four conditions.

The AFM model assumes that performance monotonically converges to zero error in the tail. However, both humans and simulated students have non-zero error in the tail of their learning curve. This violation of the AFM model's assumption causes the model to estimate lower learning rates in order to accommodate non-zero error in the tail. We found that because the simulated students sometimes get a much larger number of practice opportunities than human students (e.g., 80 vs. 30 practice opportunities), the bias in AFM's learning rates was non-trivial. To address this challenge, we utilized the Additive Factors Model + Slip (AFM+S) approach [12], which explicitly models non-zero error rates in the tail using additional "slipping" parameters for each KC. The AFM+S model better fit the simulated student data from all six conditions than the AFM model (three fold cross-validated RMSE = 0.240 vs. 0.257). Qualitatively, we found that the AFM+ S learning curves seemed to better fit the data, particularly for slopes at the beginning of difficult to master skills.

4. SIMULATION RESULTS

4.1 Online Knowledge Tracing Results

Figure 1 shows the numbers of problems administered by the tutoring system in each condition. Random always gives all 80 problems each type. Streak gives around 17 problems for AD, 10 problems for AS and 9 problems for M. BKT_default gives around 11 problems for AD and 6 problems for AS and M. BKT_random has the similar statistics to the BKT_default, while BKT_human gives around 14 AD problems, 9 M problems and around 6 AS problems. DKT_random gives around 78 AD problems, almost as many as Random; however, it gives less than 3 problems in AS and M. The number of problems given by BKT_human is slightly higher than those given by BKT_random. We hypothesize that this is because the BKT_random parameters were fit specifically to the simulated students, so when used for knowledge tracing they provide better estimates of mastery than the the BKT_human parameters.

To get a better sense of the overall differences between Random, Streak, BKT (BKT_random), and DKT, we plotted the overall learning curves for the data from these conditions, see Figure 2. We can see from this figure that BKT

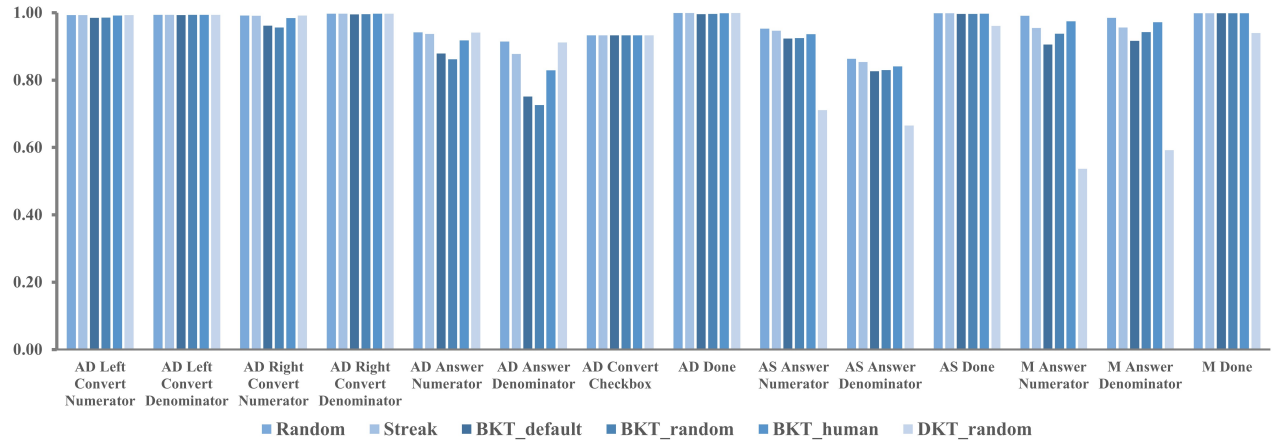

Figure 3: The AFM+S predicted probability at the end of training for each KC averaged over all students.

stops giving practice earlier than Streak, which subsequently stops giving practice earlier than Random and DKT. We observe higher variance in the tail of the learning curves for BKT and streak because the total number of students is decreasing as each student reaches mastery.

We applied the AFM+S model to predict performance on a hypothetical next opportunity for each KC and student. Figure 3 shows the average predicted correctness (across students) after the final practice opportunity for each skill. For most KCs, the prediction is higher than 95%, which suggests that mastery has been obtained in these KCs. Unfortunately the KC "AD Answer Denominator" has the lowest overall next-step correctness prediction in all six conditions. Figure 4 displays the learning curve for this skill across all six conditions and the number of students that have not yet mastered the skill at each point. This graph shows that there is a high slipping rate for this particular skill (see green line), indicating that there is a ceiling on the best possible AFM+S prediction that can be achieved.

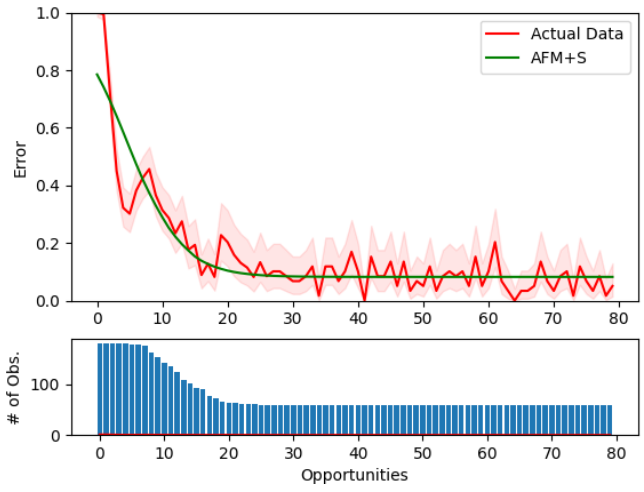

Figure 4: Learning curve for "AD Answer Denominator" and number of unmastered students at each opportunity.

Conditions	AD Average	AS Average	M Average
Random	0.01	0.01	0.01
Streak	0.06	0.11	0.12
BKT_default	0.09	0.16	0.16
BKT_random	0.10	0.17	0.15
BKT_human	0.08	0.16	0.12
DKT_random	0.01	0.30	0.31

Table 1: Model efficiency scores across six conditions.

To evaluate how well each approach handles the trade off between maximizing student's performance while minimizing the amount of practice, we computed a metric that we call "efficiency", see Table 1. To compute the score, we divided the AFM+S predictions at the end of training by the number of opportunities the student received for each student and KC. We then averaged over students to get a score for each KC. Finally, we averaged over KCs within each problem type. This produced, 3 model efficiency scores for each of the six models. Bigger value refers to a more efficient model. The efficiency score complements accuracy and provides more information for selecting the best model.

Although Random gives the highest prediction in Figure 3 among all KCs, it is the least efficient one as it gives all the problems during training. BKT_random has lower predictions than Streak, however the model efficiency suggests that it is more efficient. DKT_random yields the same efficiency as Random in AD problems. However, for AS and M problems, it appears to have the highest efficiency across all six models since it takes the least practice, but still achieves a moderate correctness prediction.

4.2 Simulated vs. Human BKT parameters

To validate the feasibility of generating BKT parameters using simulated student data, we did a correlation analysis of the BKT_random and BKT_human parameters. Figure 5 shows that there's a positive correlation of around 0.65 in the "Learn" parameter, which means the simulated students generated by Apprentice Learner have a similar learning rate as human students. We argue that this is one of the

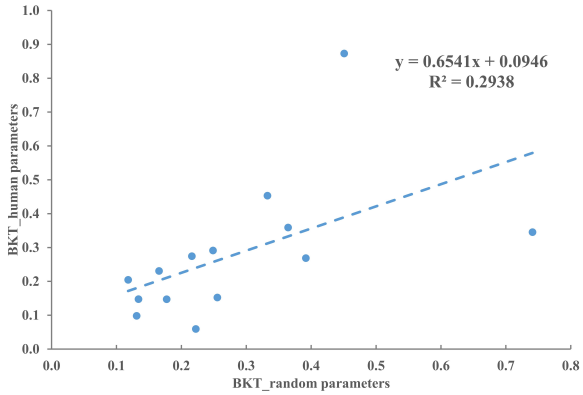

Figure 5: Positive correlation in the BKT "Learn" parameter values estimated from simulated and human data.

harder parameters to set. The "Known" parameter ($P(L_0)$) was near 0 for all skills in the simulated data because all agents start off without any prior knowledge. In human data, this parameter will vary based on the learning context. The "Guess" and "Slip" parameters based on the simulated data were reasonable (both greater than 0), but exhibited no notable correlation with the human guess and slip values. Taken together, we argue that this approach is a promising way to identify initial parameter values for BKT, but more research is needed to explore and generalize this idea.

4.3 Knowledge Tracing Sequence Analysis

Next, we take a closer look at the estimates of several knowledge tracing approaches for different sequences. Figure 6 shows the correctness sequence of a single student on "AD Answer Denominator" KC. This student was taken from the BKT_random condition. We added the mastery predictions generated by BKT, Streak and DKT given this sequence. For BKT and Streak, only the correctness on this skill was used. For DKT, the model was given the entire student sequence for all KCs, but only the predictions for this KC are shown. Predictions were taken at where the student just finished the problem that contained the target KC.

For BKT, the estimates trend towards increased mastery over the course of practice, but sometimes the probability decreases when it gets an item wrong. For Streak, each correct response yields a 33% increase in the mastery prediction, accumulating to 100% by the third correct response in a row. The DKT models predictions tend to jump around, but generally do not seem to be increasing despite getting the problem correct multiple times in a row. For example, the model's probability of correct jumps to 100% before returning to and staying close to 0%.

Correctness		✗	✗	✗	✓	✓	✓	✗	✓
BKT Mastery	0%	13%	16%	16%	72%	97%	100%	99%	100%
Streak Mastery	0%	0%	0%	0%	33%	67%	100%	0%	33%
DKT Mastery	36%	0%	0%	100%	0%	0%	0%	2%	0%

Figure 6: Different Model Predictions on "AD Answer Denominator" given student correctness sequence.

To figure out why the DKT model has such erratic behavior and why it gives so many AD problems, we fed a complete sequence from one student into the DKT model (student from BKT_random condition). Figure 7 shows the prediction of mastery for each KC after the student has completed each problem (problem type shown on the x-axis). It seems that the student never masters the "AD Answer Numerator" or "AD Answer Denominator", which explains why the DKT tutor is giving almost all the AD problems to the students.

Upon further investigation, we discovered that the DKT model has a fundamental issue that makes it difficult to use for mastery learning. The issue is caused by using the DKT predictions between problems when the mastery learning system is determining which KCs are mastered before picking another problem. Unfortunately, for multi-step problems some KCs cannot be correctly applied on the first step. DKT correctly predicts these KCs will have near 0% correctness (any attempts will be incorrect). However, this has the side effect of confusing the mastery learning system into thinking that the KC is unmastered. When the DKT model actually reaches a step where the KC can be correctly applied, then its predicted probability jumps to a more realistic estimate of the mastery. This problem was not identified in previous work on mastery learning with DKT (e.g., [15]) because the prior work only looked at problems with a single step, so this issue never occurred. However, most problems within tutoring systems are multi-step. Future work should explore how to correct this issue within DKT so it can be used for mastery learning.

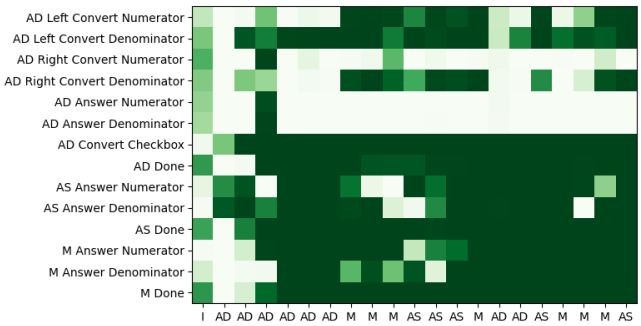

Figure 7: DKT predictions for each KC after each problem.

5. SUMMARY OF KEY FINDINGS

Our first key finding is that simulated students can successfully evaluate online knowledge tracing models. Our simulations indicate that Streak works well for mastery learning and has reasonable efficiency; although the model gives a bit more practice than strictly necessary (e.g. the average number of steps to master all AD KCs is around 17 in Streak and 10 in BKT_random). Still, Streak is very simple to operate, implement, and modify and it behaves reasonably well.

BKT also work well for mastery learning and generally seems to have the best efficiency of the approaches we compared (although DKT seems to be more efficient for AS and M problems). However, it seems to stop a little early in some cases, resulting in under practice. Figure 3 shows that the BKT_random model gets 86% correctness prediction for the KC "AD Answer Numerator", 73% for "AD Answer Denom-

inator" and 83% for "AS Answer Denominator". To some extent, it reveals that these KCs might be more difficult for both simulated students and human students to master, and more practice is required to obtain mastery. Our work suggests that it is important to look at multiple factors when evaluating knowledge tracing approaches in online settings. In particular, it is important to look at both student mastery and the amount of practice that is administered. To explore this trade off, we proposed the efficiency metric. We believe that this metric (or ones like it) might be useful for evaluating knowledge tracing approaches in online settings.

Multiple studies [18, 17] suggest that the DKT model has good predictive performance in offline setting. However, we found that it does not seem to work properly for online knowledge tracing, particularly in cases of multi-step problems. Yeung & Yeung [19] identified that DKT's predictions for KCs are not consistent across time-steps, which we believe is related to the issue we encountered. Even when a student performs well on a KC, DKT's predicted performance for that skill may drop. In general, DKT's predictions fluctuate drastically over time, so *when* its predictions are sampled has a big impact on its accuracy. The core issue is that DKT generates predictions for every KC at every step, but its loss function only constrains predictions that are likely to occur next. Yeung & Yeung suggested some possible modifications to the DKT objective function to mitigate this problem, but implementation and testing of these was beyond the scope of the current study.

In preliminary analysis, BKT estimated students reached mastery even though their error rates were still high. Further inspection revealed an error in our BKT model that was causing it to incorrectly estimate student mastery. Multiple researchers across multiple labs have used this open-source implementation. Despite wide use, we uncovered issues that had not been previously uncovered. Although we corrected these issues for the current study, we argue that this is a positive outcome for our simulated student approach. This finding reinforces the idea that simulated students can be used to test and improve knowledge tracing approaches before running more costly human studies.

Our second key finding is that researchers might use data generated by simulated students to initialize knowledge tracing parameters when human data is not available. To evaluate the feasibility of this idea for BKT, we conducted a correlation analysis between the random-based BKT parameters and human-based parameters. We found a strong correlation between the learning rate parameters suggesting that initializing BKT parameters using simulated student data might be an informed, but cost-effective approach.

6. RELATED WORKS

The closest work to ours is the simulation studies conducted by Doroudi et al. [4, 3], which investigates different knowledge tracing approaches using simulated students. They argue that it is important to evaluate knowledge tracing under various assumptions about how students learn. One of the major differences between their approach and ours is that they use statistical models that predict correctness to simulate students rather than computational models of learning that actually learn and perform the task, as we do with Apprentice agents. Apprentice agents are more complex than the knowledge tracing approaches that are being used to evaluate them. It would be interesting to explore the use of Apprentice agents as another kind of student model for the knowledge tracing evaluations proposed by Doroudi et al.

7. CONCLUSIONS AND FUTURE WORK

We were able to successfully apply simulated students to test different knowledge tracing models. When we compared the three knowledge tracing models (BKT, Streak, and DKT) to a no-knowledge-tracing baseline (Random), we found that BKT gave the fewest problems, Streak gave the second fewest, Random gave the most and DKT gave almost as many as Random in one problem type and the least in the other two. In general, we found that BKT seemed to be the most efficient approach, but streak gave reasonable results despite its simplicity. Through the use of simulated students, we also discovered a number of issues with our BKT implementation as well a fundamental issue with DKT. Despite widespread use of the BKT implementation and a lot of recent investigation into the DKT model, these issues had not been discovered in prior work. Together, these results support our primarily claim that simulated students are an effective tool for investigating and evaluating online knowledge tracing approaches.

Our analysis also found evidence to support the idea that simulated student data might be used to initialize BKT parameters when no human-student data is available. In particular, we found that BKT learning rates estimated from simulated data have a significant correlation to the learning rates estimated from human data. While these initial results are promising, more work is needed to further explore these ideas. In particular, we would like to try running human-subject experiments to compare BKT models initialized using simulated student data to those with default parameters. One surprising finding is how well BKT_default performs; despite somewhat arbitrary parameters, it was more efficient than Streak. Future work should explore how to manually pick robust default values for BKT.

We have a number of additional future directions we would like to explore. We intend to individualize the Apprentice models to make them better mimic the behaviors of different kinds of learners (e.g., high vs. low performing students), students with different motivation in learning, and those who suffer from learning disabilities. We should also explore variations of DKT that address concerns we have identified and enable its use in online mastery learning. Finally, we should move beyond simulation and explore how well our simulated students predict which knowledge tracing approaches will yield the best learning for human students.

8. ACKNOWLEDGMENTS

We would like to thank Danny Weitekamp and Erik Harpstead, who developed much of the framework for testing the Apprentice agents within the fractions tutors. We also thank Anna Raffery for creating the framework for applying knowledge tracing to simulated students and developing the initial version of Bayesian Knowledge Tracing that we used. We also thank Adit Gupta for reading earlier drafts and providing suggestions for improvement.

References

[1] H. Cen. *Generalized Learning Factors Analysis: Improving Cognitive Models with Machine Learning*. PhD thesis, Carnegie Mellon University Pittsburgh, 2009.

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

[3] S. Doroudi, V. Aleven, and E. Brunskill. Robust evaluation matrix: Towards a more principled offline exploration of instructional policies. In *Proceedings of the Fourth ACM Conference on Learning@Scale*, pages 3–12, 2017.

[4] S. Doroudi and E. Brunskill. Fairer but not fair enough on the equitability of knowledge tracing. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, pages 335–339, 2019.

[5] S. E. Fancsali, T. Nixon, and S. Ritter. Optimal and worst-case performance of mastery learning assessment with Bayesian knowledge tracing. *Proceedings of the 6th International Conference on Educational Data Mining, EDM 2013*, 2013.

[6] T. Gervet, K. Koedinger, J. Schneider, T. Mitchell, et al. When is deep learning the best approach to knowledge tracing? *Journal of Educational Data Mining*, 12(3):31–54, 2020.

[7] N. T. Heffernan and C. L. Heffernan. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497, 2014.

[8] M. Khajah, R. V. Lindsey, and M. C. Mozer. How deep is knowledge tracing? *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016*, pages 94–101, 2016.

[9] K. R. Koedinger, R. S. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the edm community: The pslc datashop. In *Handbook of Educational Data Mining*, volume 43, pages 43–56. CRC Press, 2010.

[10] C. J. MacLellan, E. Harpstead, R. Patel, and K. R. Koedinger. The apprentice learner architecture: Closing the loop between learning theory and educational data. *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016*, pages 151–158, 2016.

[11] C. J. MacLellan and K. R. Koedinger. Domain-general tutor authoring with apprentice learner models. *International Journal of Artificial Intelligence in Education*, pages 1–42, 2020.

[12] C. J. MacLellan, R. Liu, and K. R. Koedinger. Accounting for Slipping and Other False Negatives in Logistic Models of Student Learning. *Proceeding of the 8th International Conference on Educational Data Mining, EDM15*, pages 53–60, 2015.

[13] P. Nedungadi and M. S. Remya. Predicting students' performance on intelligent tutoring system - Personalized clustered BKT (PC-BKT) model. *Proceedings - Frontiers in Education Conference, FIE*, 2015-February(February), 2015.

[14] R. Patel, R. Liu, and K. R. Koedinger. When to block versus interleave practice? evidence against teaching fraction addition before fraction multiplication. In *Proceedings of Cognitive Science Conference*, 2016.

[15] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. Guibas, and J. Sohl-Dickstein. Deep knowledge tracing. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 505–513, Cambridge, MA, USA, 2015. MIT Press.

[16] J. Rollinson and E. Brunskill. From predictive models to instructional policies. *Proceedings of the 8th International Conference on Educational Data Mining, EDM 2015*, pages 179–186, 2015.

[17] L. Wang, A. Sy, L. Liu, and C. Piech. Deep knowledge tracing on programming exercises. In *Proceedings of the Fourth ACM Conference on Learning@Scale*, pages 201–204, 2017.

[18] X. Xiong, S. Zhao, E. G. Van Inwegen, and J. E. Beck. Going deeper with deep knowledge tracing. *Proceeding of the 9th International Conference on Educational Data Mining, EDM 2016*, pages 545–550, 2016.

[19] C.-K. Yeung and D.-Y. Yeung. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning@Scale*, pages 1–10, 2018.

[20] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models. In *Lecture Notes in Computer Science*, volume 7926 LNAI, pages 171–180, 2013.