The Apprentice Learner Architecture: Closing the loop between learning theory and educational data

Christopher J. MacLellan
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
cmaclell@cs.cmu.edu

Erik Harpstead
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
eharpste@cs.cmu.edu

Rony Patel
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
rbpatel@andrew.cmu.edu

Kenneth R. Koedinger
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
koedinger@cmu.edu

ABSTRACT

While Educational Data Mining research has traditionally emphasized the practical aspects of learner modeling, such as predictive modeling, estimating students knowledge, and informing adaptive instruction, in the current study, we argue that Educational Data Mining can also be used to test and improve our fundamental theories of human learning. Using the Apprentice Learner architecture, a computational theory of learning capable of simulating human behavior in interactive learning environments, we generate two models that embody alternative theories of human learning: (1) that humans perfectly recall previous training during learning and (2) that humans only recall a limited window of experience. We evaluate which of these models is better supported by data from two fractions tutoring systems. In general, we find that the model with a complete memory better fits the data than a model recalling only the previous training experience (data-driven theory development). Additionally, we demonstrate that both models are able to predict student performances, as well as, reproduce the main effects of an experimental paradigm without being trained on student data (theory-driven prediction). These results demonstrate how the Apprentice Learner architecture can be used to close the loop between learning theory and educational data.

1. INTRODUCTION

One branch of Educational Data Mining (EDM) research leverages data to improve our theoretical understanding of how people learn [28, 3]. Analogous to how data from the Large Hadron Collider can be used to gain insights into physical laws, educational data can be used to provide insights into the unobservable mechanisms underlying student learning. Surprisingly, little EDM research has explored this direction, rather, the main trends in research center on how statistical models can be used to perform latent knowledge estimation and domain-structure discovery (i.e., knowledge component discovery) [3]. While these research directions are important, we argue that the availability of educational data makes the EDM community well poised to contribute substantially towards our theoretical understanding of human learning.

Although many of the widely used predictive models of learning, e.g. Bayesian Knowledge Tracing [5], and Additive Factors Model [4], rely on existing theories of human learning, such as the power law of practice [22], researchers rarely apply these models to educational data with the aim of improving the underlying theory of learning. Further, there are a number of barriers to using educational data for this purpose. First, many EDM models are only loose approximations of the theories they are based on. For example, the Additive Factors Model predicts that improvements in human performance will follow a single logistic function, whereas the power law of practice states that the improvements should follow a power function [6]. Second, EDM models do not reflect the current state of learning theory. For example, recent studies of skill acquisition actually suggest that improvements should follows three distinct power functions, one for each phase of cognitive skill acquisition [30], rather than a single logistic function. This disconnect between theories and models makes it difficult to draw inferences about the underlying theories given the fit of models to data. By more tightly connecting our EDM models to theory, we can leverage educational data to improve our understanding of the mechanisms behind human learning and, in turn, use these theories to improve our abilities to predict student behavior.

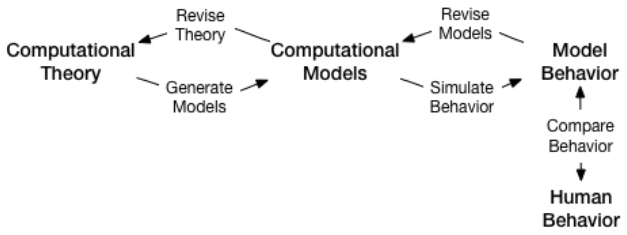

Figure 1: A depiction of how theories, models, and behavior relate. Theories are used to generate models, which can be used to simulate behaviors. Simulated behavior can be compared to human behavior and differences inform future models and theories.

To more tightly link a theory to models, researchers can develop a computational theory [17, 21]. Unlike a theory that only specifies the abstract relationships between constructs (e.g., that an increase in spatial skills leads to an increase in learning with graphical representations [26]), a computational theory represents a complete description of the mechanisms that produce observed phenomena. Within this paradigm, a model presents as a specific algorithmic implementation of these mechanisms that can be executed to simulate behavior, which then can be compared with observed behavior in order to test both the model and the underlying theory. A key component of this approach is not to explain or "fit" a relationship in observed data, but rather, to predict that a relationship will be present before any data is observed. Figure 1 shows the iterative relationship between theories, models, and behaviors. We argue that this approach complements existing approaches in the EDM literature.

In the current work, we present the Apprentice Learner architecture, a computational theory of learning in interactive learning environments, such as tutoring systems. Unlike prior models of student performance, such as Additive Factors Model and its variants, Apprentice Learner models seek to explain the mechanism students use to acquire new knowledge from instruction. This mechanical description allows us to simulate learner behavior within an instructional environment and use these simulations to predict human behavior. Rather than arriving at a general conclusions like students learn differently from positive and negative feedback this approach lets us explore possible explanations for the mechanisms driving these results. In presenting this computational theory we make two claims:

1. The Apprentice Learner architecture can be used to predict student behavior and experimental results before collecting any student data (purely theory-driven prediction).
2. The Apprentice Learner architecture can leverage data to improve learning theory through the creation and testing of different models of learning.

To support these claims, we explore different assumptions about memory and its effect on human learning in intelligent tutoring systems. We leverage the Apprentice Learner architecture to instantiate two models of human learning, one that hypothesizes perfect memory and another that assumes a more limited window of memory. We apply these models in two different fractions tutoring systems. In both cases, we generate datasets of simulated learner behavior that have high agreement with the patterns of behavior observed in human students. Additionally, we show that our models reproduce the main effects of a problem sequencing experiment without first being fit to student data. In general, we find that the model with perfect memory better fits the fractions data than the model with limited memory; these findings provide an initial demonstration of how our computational theory can be refined in response to data.

In the following sections, we first present the Apprentice Learner architecture and describe the theoretical commitments that it makes. Next, we describe our overarching simulation approach, the particular computational models that we investigate, and the results of our simulation studies in (1) fraction addition and (2) fraction arithmetic. Finally, we discuss the implications of our results and directions for future work.

2. THE PROPOSED ARCHITECTURE

In 2006, VanLehn published his seminal paper describing the step-level behavior of tutoring systems [31]. Although not commonly cited within the EDM literature, VanLehn's description of the general two-loop structure of tutoring systems (i.e., an inner loop for step-level feedback and an outer loop for problem selection) has direct relevance to many recent advances in EDM research. For example, researchers have used knowledge component discovery to create a better understanding of domain tasks [4, 13], so that the inner loop feedback can be improved. Other researchers have used latent knowledge estimation to improve outer loop instructional policies [27]. While VanLehn's theory promotes common ground between similar thrusts of work in EDM, it can only serve as half the picture of a computational theory of the tutoring process.

The Apprentice Learner architecture, shown in Figure 2, is a computational theory of human learning that aligns with the step-level interactions described by VanLehn. The theory embodied in the Apprentice Learner architecture states that students acquire skills by interactively solving problems in a tutored paradigm, receiving correctness feedback on their actions. In the event that the student does not know how to proceed, they can request a hint from the tutor, which provides the student with a demonstration of how to take the next problem-solving step.

The Apprentice Learner architecture uses a base of prior knowledge to induce new skills from its observed demonstrations and feedback. The first kind of knowledge consists of functions for manipulating data (e.g., adding two values, appending two strings together, etc.). The second kind of knowledge consists of features for recognizing different elements in the interface (e.g., recognizing numbers, mathematical symbols, etc.). Depending on the domain, different kinds of background knowledge may be appropriate. For example, Apprentice Learner models in equation solving might have features for recognizing polynomials, whereas models in stoichiometry might have different features for recognizing chemical symbols.

The Apprentice Learner architecture posits three learning mechanisms to induce new skills from prior knowledge and observed demonstrations and feedback. When given a demonstration, the *how* learning mechanism uses function knowledge to search for a sequence of functions that can explain the observed demonstration. After discovering a function sequence, the *where* learning mechanism acquires general perceptual patterns for recognizing the elements used in the discovered sequence. Finally, the *when* learning mechanism uses the tutor state, augmented with feature knowledge, to identify the conditions under which the discovered sequence should be executed. The combination of the components discovered by how, where, and when learning mechanisms constitutes a skill. Apprentice learners apply learned skills in subsequent problem solving.

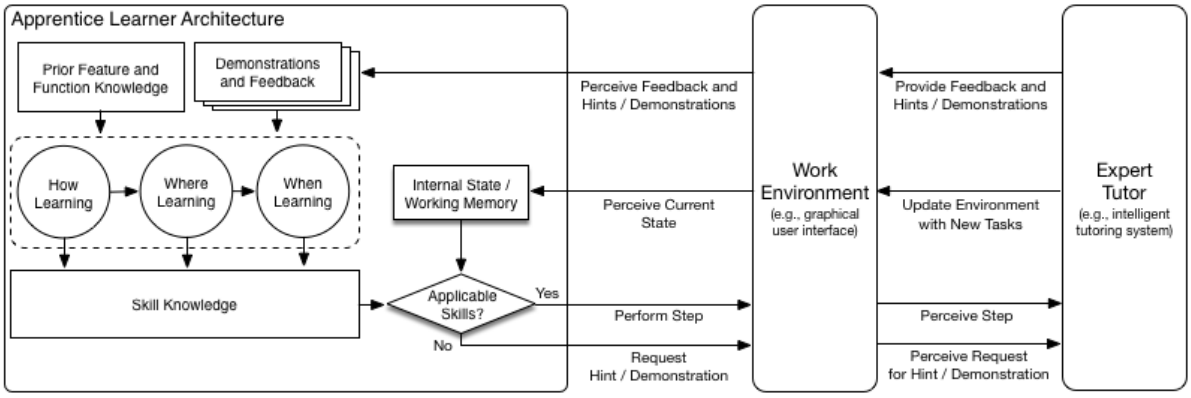

Figure 2: The Apprentice Learner architecture and its interactions between the work environment and expert tutor. The architecture possesses three learning mechanisms (how, where, and when) to generalize demonstrations and feedback into skill knowledge that can be used for problem solving.

In order to apply learned skills, the Apprentice Learner architecture posits that learners use a basic Recognize-Act cycle [32]. When presented with a problem, learners first query their skill knowledge to determine if any known skills are applicable. If an applicable skill is found, the learner executes it. The learner passes correctness feedback on the resulting action to its *when* learning mechanism, which uses the feedback to refine the conditions under which the skill can be executed. In the event that no skills are applicable, the learner requests a demonstration that is passed to the how, where, and when learners to produce a new skill.

Given the computational theory described by our architecture and our data-driven theory development approach (see Figure 1), our goal is to develop a theory that is consistent with available educational datasets, such as those found in DataShop [7] and other similar repositories. To pursue this goal, we propose a research program wherein different models of human learning are generated within the framework of the Apprentice Learner architecture, i.e., specific algorithms are implemented for each of the components of the architecture. These Apprentice Learner models can then be connected to the same intelligent tutoring systems that generated the data found on Datashop. Next, the behavior of these models can be compared to human behavior. Based on the differences between the models and humans, we can revise our theory (e.g., replacing a perfect memory of previous demonstrations and feedback with a memory that only recalls a window of experience), generate new models, and then simulate the revised models to determine if better agreement between models and human behavior can be demonstrated.

3. SIMULATION STUDIES

We make two key claims about the Apprentice Learner architecture: (1) it can be used to predict student behavior without data and (2) it can be used to improve theory by facilitating the exploration of different models. To demonstrate the potential of the architecture and to support our key claims, we conducted simulation studies with two tutoring systems in the domain of fractions [33, 14, 24].

For these simulations, we created an initial model of human learning by implementing each of the components of the Apprentice Learner architecture in computer code. This model was given two features, isPlusSign and isMultSign, which can be used to determine if a string is a plus or multiply sign (i.e., + or ×). It was also given six functions: Add(X,Y), Subtract(X,Y), Multiply(X,Y), Divide(X,Y), CopyPasteString(X), and GenerateCheckMark(). The Add, Subtract, Multiply, and Divide functions returned the result of applying their respective arithmetic operations to their arguments. The CopyPasteString function returns a copy of the string that is passed to it. Finally, the GenerateCheckMark takes no arguments and returns a check mark that can be used to fill checkboxes in the tutor interface. This prior feature and function knowledge represents the basic interface and arithmetic knowledge that students would be expected to know before using a fractions tutor.

Given this prior knowledge, we implemented three machine learning algorithms for the three learning mechanisms outlined in Figure 2. For how learning, we used a variation of Langley's BACON algorithm [9] to discover an explanation of expert demonstrations using the provided functions. For where learning, we used a variation of Mitchell's Version Space algorithm [20] to discover perceptual patterns for recognizing relevant interface elements. Finally, for when learning, we used Quinlan's FOIL algorithm [25] to learn the conditions under which the learned skills can be executed. More details of our algorithmic implementations can be found in previous work, which refers to this particular combination of learning algorithms as the SimStudent model [18, 11].

In this initial model, the skill knowledge acquired from the three learning mechanisms is stored in the form of production rules (i.e., IF-THEN rules). The perceptual patterns learned from the where learner and the conditions acquired by the when learner constitute the IF part of the rule. The function sequence discovered by the how learner constitutes the THEN part of the rule. An example of a human-readable version of a production rule discovered by one of our models might be: **IF** there are two fractions with denomina-

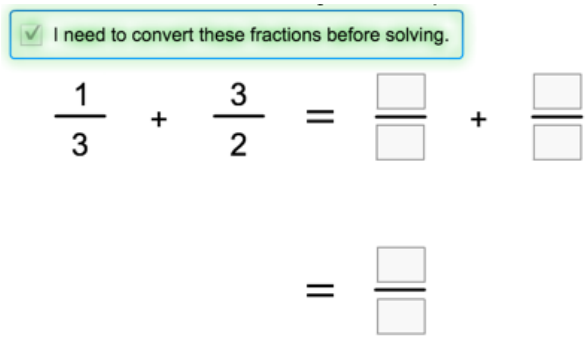

Figure 3: The Fraction Arithmetic Interface.

tors and a sign between them (i.e., the perceptual pattern is present) **AND** the sign is a plus sign and the denominators are equal (i.e., the conditions are satisfied) **THEN** copy one of the denominator values and put the result in the answer denominator box (i.e., perform the function sequence). During problem solving, the models check if they have any applicable production rules (i.e., skills) and if a match is found, then they take the prescribed action.

This initial model, which we refer to as the full-memory model (also the SimStudent model in previous work), has been used to model ordering effects [10], as a teachable agent [18], and for authoring cognitive models [16, 12]. In these previous studies, the full-memory model has been found to regularly outperform human students. One hypothesis is that this model outperforms human students because it revises its skill knowledge using a complete memory of all previous training examples [15]. To explore this hypothesis, we created a second model that duplicates our initial model, with the exception that it only recalls the previous training example during skill learning. This model, which we refer to as the one-back-memory model, instantiates an extreme version of the hypothesis that learners only recall a limited amount of their past experience during learning.

3.1 Data

To test the full-memory and one-back-memory models, we use data from two intelligent tutoring systems available on DataShop. In both tutors, students were asked to solve fraction arithmetic problems using a variation of the interface shown in Figure 3. The first dataset came from the control condition of a fraction addition study [33]. The dataset consisted of 24 students solving 20 fraction addition problems. The tutoring system used in this dataset omitted the "I need to convert these fractions before solving" checkbox and required students to convert fractions to common denominators, even if this meant copying fractions that already had the same denominators. Additionally, the tutor allowed students to use multiple approaches to find a common denominator; they could either multiply the denominators or compute the least common denominator. To allow our models to use this second approach, we added the LeastCommonMultiple(X,Y) function to the prior knowledge of both models, under the assumption that students utilizing this approach know how to compute the least common multiple.

The second dataset came from an experiment testing whether blocking or interleaving different types of fraction arithmetic problems was better for learning [24]. This dataset contains 79 students solving 24 fraction addition problems (10 with same denominators and 14 with different denominators) and 24 fraction multiplication problems. The tutor used in this study required students to check the "I need to convert these fractions before solving" box before making the fields necessary for converting visible. Additionally, on fraction addition problems with different denominators, students were only allowed to compute common denominators by multiplying denominators. Thus, the LeastCommonMultiple(X,Y) function was not included in the models for this dataset.

The experimental manipulation of the second datasets divided students into two conditions, blocked and interleaved. The students in the blocked condition received three blocks of problems: fraction addition problems with same denominators, then fraction addition problems with different denominators, and then fraction multiplication problems. The order of the problems within each block was randomized for each student. In contrast, the students in the interleaved condition received a random ordering of all problems. This experiment showed that students in the blocked condition have a lower overall error rate than students in the interleaved condition. Additionally, the error rates of students in the blocked condition increased when transitioning between different types of problems.

3.2 Method

For each dataset, we tested our full-memory and one-back-memory models of learning by creating instances of each model for each student and connecting these instances to the appropriate tutoring systems. The tutoring systems then tutored the instances through the same order of problems that the respective human students received. In each dataset, we compared the first attempt correctness on each step between the two models and their respective humans. For each model, we computed how often the first attempt correctness agreed with the respective human's first attempt correctness, i.e., accuracy, to quantitatively measure the agreement between model and educational data. We report the mean accuracy and its accompanying 95% confidence interval (95% CI) for each model. Next, for each dataset we plotted overall learning curves comparing the first-attempt performance of the humans to each of the two models. For these learning curves, we used a knowledge component model that labeled each step as exercising a skill corresponding to the field that was updated in the interface. These learning curve graphs demonstrate how the Apprentice Learner architecture can be used to generate theory-driven learning curve predictions. Because each model instance has the same prior knowledge, our simulation studies do not take into account the individual differences in students' prior knowledge. To determine if taking into account student-level effects impacts which model better fits the data, we fit a random-effects logistic regression model with a fixed effect for the model prediction and random effect for the student. We report the Akaike information criterion (AIC) and Bayesian information criterion (BIC) scores to determine which of our two models better fits the data in each case. Note, AIC and BIC values on one dataset are not comparable to the AIC and BIC values on another dataset, they can only be used to

Figure 4: The fraction addition learning curves for the human students, the full-memory model, and the one-back-memory model.

rank model fits on the same dataset. For a given dataset, lower values of AIC and BIC are better, and a difference of more than 3 in either measure is usually viewed as strong evidence to prefer one model over another.

3.3 Fraction Addition Results

After simulating the the 24 students in the fraction addition dataset, we found both models were significantly predictive of students' correctness on the 2,432 first attempts ($p < 0.01$ via a χ^2 test). The full-memory model correctly predicted 74.05% (95% CI : 72.26, 75.79) of first attempts, whereas the one-back-memory model correctly predicted only 70.93% (95% CI : 69.08, 72.73) of first attempts. This significant difference in accuracy ($p < 0.01$ via Mcnemar's test) suggests that the full-memory model more closely agrees with the fraction addition data than the one-back-memory model, when not taking into account differences in students prior knowledge.

Next, we plotted the learning curves comparing both models' performance to the human performance, see Figure 4. The opportunity counts for these learning curves were determined by how many times each student had practiced filling in the relevant interface field (each field is roughly analogous to the skill used to update that field). Both simulated models initially start off without any skills, so their error rate is 100% on the first step. However, the models quickly converge to human-level performance. Although the full-memory model achieves a lower overall error, the one-back memory model appears to have variation that is more equally distributed around the human performance.

To test which model best fits when taking the differences between students' prior knowledge into account, we fit two mixed-effect logistic regression models that had a single fixed effect for the respective simulation prediction (full-memory or one-back-memory) and a random effect for student. We found that the one-back-memory model better fit the student data (AIC=1727, BIC=1744) than that full-memory model (AIC=1754, BIC=1772), suggesting that students in the fraction addition dataset have differences in their overall performance that might correspond to differences in prior knowledge. Further, these results suggest that the one-back memory model better fits student performance when taking these differences into account.

3.4 Fraction Arithmetic Results

Similar to the previous dataset, we found both models were significantly predictive of the 79 students' 18,589 first attempts ($p < 0.01$ via a χ^2 test). We also found that the full-memory model (Accuracy : 84.04%, 95% CI : 83.5, 84.56) was more predictive of students' first attempts than the one-back-memory model (Accuracy : 80.24%, 95% CI : 79.66, 80.81). Similar to our previous fraction addition results, this significant difference in accuracy ($p < 0.01$ via Mcnemar's test) suggests that the full-memory model more closely agrees with the fraction arithmetic data than the one-back-memory model, when not taking into account differences in students prior knowledge.

Figure 5 shows the learning curves comparing the performance of the two models to the human data. Similar to the fraction addition dataset, the opportunity counts for these learning curves were determined by how many times each student had practice filling in the relevant interface field (again, fields are roughly analogous to the skills used to update them). However, in this dataset we plotted separate learning curves for students in the two experimental conditions, blocked and interleaved.

Similar to the fraction addition learning curves, the full-memory and one-back-memory models initially start off with an error rate of 100% on their first steps and quickly converge to human-level performance. However, in this dataset, we can see that both models seem to emulate key differences in the two conditions. First, the human students in the blocked condition have lower error than those in the interleaved condition ($z = -6.136$, $p < 0.01$ via a logistic regression). Both the full-memory ($z = -9.598$, $p < 0.01$) and the one-back-memory ($z = -4.626$, $p < 0.01$) models correctly predict this main effect of condition. Second, the human students in the interleaved condition slowly converge to asymptotic performance, whereas the human students in the blocked condition achieve lower initial error but then have drastic increases in error when transitioning between problem types (e.g., around opportunity 12). The simulated data from both models appears to mirror these effects. While both models experience a spike in error around opportunity 25 when transitioning to multiply problems, the human students, surprisingly, do not show a similar increase. This difference might be explained by the fact that the human students have prior experience multiplying numbers, and fraction multiplication is arguably easier than fraction addition with different denominators (i.e., students have to use multiplication to compute common denominators). In contrast, both the full-memory and one-back-memory models have no experience with multiplication prior to opportunity 25, so they have a 100% initial error on the first multiplication step. This suggests that future work is needed to explore how to populate models with initial training experiences (e.g., teaching the model to do whole-number multiplication before fraction multiplication).

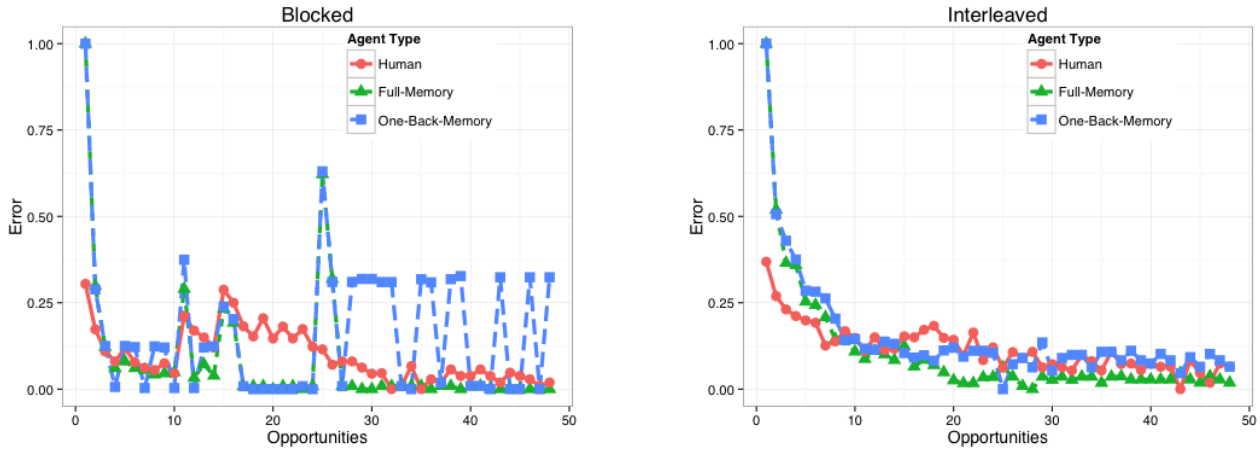

Figure 5: The fraction arithmetic learning curves for the human students, the full-memory model, and the one-back-memory model. The left graph shows the learning curves for the blocked condition and the right graph shows the learning curves for the interleaved condition. The spikes in error rate in the blocked condition occur when students transition from fractions with same denominators to fractions with different denominators (opportunity 12) and to fraction multiplication (opportunity 25).

Finally, we again fit two mixed-effects logistic regression models to determine if taking individual student differences into account would change which of the two models better fit the data. In contrast to the fraction addition results, we found that the full-memory model better fit the student data (AIC=10849, BIC=10872) than that one-back-memory model (AIC=11013, BIC=11036). These results show that, for fraction arithmetic, the full-memory model better fits the student data regardless of whether or not overall student differences are taken into account.

4. GENERAL DISCUSSION

We argue that our simulation studies in fraction addition and fraction arithmetic provide strong evidence in support of our two key claims about the Apprentice Learner architecture. First, our analysis shows that the behavior generated by both models agrees with the human behavior in both fractions datasets; i.e., the full-memory model, which fits best, achieves 75% agreement in the fraction addition dataset and 84% agreement in the fraction arithmetic dataset. Furthermore, we show that both of the models predict the main experimental effect for the fraction arithmetic dataset; i.e., both models correctly predict that the overall performance in the blocked condition will be better than the overall performance in interleaved condition. To our knowledge, these two results are the first example in the EDM literature of how student performance can be precisely predicted in a completely theory-driven way without having to fit the models to the student data first.

Although our models have a reasonably high agreement with the student data, there are still some key differences between the models and the humans. In particular, the models always have 100% first-attempt error on novel skills. While these exaggerated error rates might be useful for detecting transitions between skills (e.g., when using learning curve analysis to develop knowledge-component models [5]), they also suggest an opportunity to improve our underlying theory and models. In future studies we should explore approaches for initializing both prior knowledge (e.g., using students' pretests to choose prior features and functions) and skill knowledge (e.g., pretraining models in a whole-number arithmetic tutor).

Our second key claim was that the Apprentice Learner architecture can be used to improve our underlying theory of human learning using educational data. We argue that our simulation results provide strong evidence supporting this claim. In particular, we tested two different models that operationalize two alternative theories of human learning: the full-memory model, which posits that humans have perfect recall of prior demonstrations and feedback when learning skills, and the one-back-memory model, which is an extreme version of the theory that humans only recall a limited window of prior demonstrations and feedback during skill learning. In our analysis, we showed that the full-memory model better fits both fractions datasets, suggesting that it is a better model of human learning. Next, we used a mixed-effects logistic regression analysis to take into account student differences. Using this approach, we showed that the one-back-memory model better fit on the fraction addition dataset and the full-memory model better fit on the fraction arithmetic dataset.

In general, these results suggest that the full-memory model better fits the fractions datasets than the one-back-memory model (in three out of four cases). However, our results leave open the possibility that, when taking into account overall student differences, a hybrid model might be best (e.g., an n-back model). Further, the full-memory model best fits the educational data, but seems to have better asymptotic performance than the human students. The original inspiration for the one-back-memory model was to decrease this asymptotic performance to bring it into closer alignment with the human performance, but our results suggest that we should consider alternative approaches for decreasing performance.

One possibility would be to replace the when learner with an incremental machine learning algorithm, such as TRESTLE [15]. This approach would let apprentice learners leverage existing theories of interference effects [2] to improve their fit with educational data. In summary, our simulation studies provide strong evidence to support our claims that the Apprentice Learner architecture can be used perform theory-driven prediction and to improve theory based on differences between model and human behavior.

5. FUTURE WORK

The results of our studies have been encouraging, however, we do not wish to leave the impression that the Apprentice Learner architecture is a complete computational theory of learning. Instead, we present the theory as an initial framework that is flexible enough to support new hypotheses about learning. In future work, we plan to explore several variations of the current theoretical structure and invite the community to extend the theory to explain phenomena in their own work.

One affordance of the Apprentice Learner architecture is that it facilitates a search among alternative theories and models. Not unlike existing techniques for searching the space of domain models [4], a search among alternative Apprentice Learner models would let us explore several hypotheses of human learning. For example, it is questionable whether how, where, and when are the correct combination of internal learning mechanisms. It may be that the FOIL algorithm, currently used for when learning, could be used to model both the where and the when learning. This would suggest that the current distinction between where and when learning is artificial and that a single mechanism might produce more human-like simulated data. Alternatively, it could be argued that the architecture is biased by having features provided as prior knowledge, rather than learning features from experience. This argument implies that some mechanism for acquiring new features, effectively a *what* learner, could be included in the architecture [11]. Beyond adding or merging learning mechanisms each individual mechanism could be represented by several underlying algorithms. For example, our implementation of the Version Space algorithm conducts a specific-to-general search for perceptual patterns, but another possible variation would be to conduct a general-to-specific search. Exploring all of these possibilities could be framed as a search task over different parametrizations of the architecture for models that generate the most human-like simulation data.

In the current work, we compare model and human error rates, but the Apprentice Learner architecture allows for finer-grained evaluation. Rather than compare simulated and human learners on whether they performed a step correctly, we could compare learners in terms of their literal response on a step. This opens up the ability to evaluate theories of student misconceptions and how they might affect the particular responses students make [19]. Similarly, in this study we only compared performance on first step attempts, because this is a common convention in EDM, but the high-fidelity simulation data can be used to examine learner behavior beyond the first attempt. Ultimately a unified theory of apprentice learning should account for all of the behaviors learners exhibit on their path to mastery.

As we have stated previously, we view the current state of the Apprentice Learner architecture as incomplete. There are several aspects of learning that the model does not currently account for, such as the effects of delayed feedback [29], the impacts of metacognition [1], and the behavior of collaborative learners [23]. Crucially, however, the theory is not fundamentally incompatible with these ideas. For example, a reinforcement learning paradigm could be employed to back-propagate correctness from delayed feedback. The role of metacognition could be accounted for with a more nuanced variation on the recognize-act cycle that takes into account metacognitive decisions. Finally, instantiating multiple Apprentice Learner models within the same environment and allowing them to generate demonstrations for each other could serve as an initial computational model of collaborative learning. These are just a few examples of how the structure of the architecture can be augmented to incorporate and test additional learning theories.

Finally, in future work we would like to explore how the theoretical tenets of our architecture align with those made by other architectures, such as ACT-R or SOAR [8]. These architectures, which primarily focus on problem solving, have mechanisms for learning skill conditions and for compiling commonly executed sequences of skills into macro-skills. It would be interesting to investigate the extent to which these learning mechanisms align with the when (condition) and how (function sequence) learning mechanisms of the Apprentice Learner architecture. By investigating how these computational theories might be aligned, we hope to provide for learning science, and more generally cognitive science, the kinds of unified theories that have been so successful in physics and the other hard sciences.

6. CONCLUSIONS

In this paper, we have taken the first steps toward a complete computational theory of learning in interactive environments, such as tutoring systems. Not only do we believe that EDM is capable of improving our fundamental theories of learning, but that is uniquely positioned to do so. Using a computational theory approach, we can use every tutored learning dataset in the canon of EDM to test and advance learning theories. We hope that other EDM researchers will also see the potential of the Apprentice Learner architecture and the computational theory paradigm, and we look forward to working together to further develop our collective understanding of human learning.

7. ACKNOWLEDGEMENTS

We thank Peggy Tenison for her feedback on earlier versions of this work. We used the "Grounded Feedback Fraction Addition Tutor" and "Fraction Addition and Multiplication" datasets accessed via DataShop (pslcdatashop.org). This work was supported in part by the Department of Education (#R305B090023) and by the National Science Foundation (#SBE-0836012).

8. REFERENCES

[1] V. Aleven, B. M. McLaren, I. Roll, and K. R. Koedinger. Toward Meta-cognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education*, 16(2):101–130, 2006.

[2] J. R. Anderson and L. M. Reder. The fan effect: New results and new theories. *Journal of Experimental Psychology: General*, 128(2):186–197, June 1999.

[3] R. S. Baker and P. S. Inventado. Educational Data Mining and Learning Analytics. In *Learning Analytics*, pages 61–75. Springer, New York, 2014.

[4] H. Cen, K. R. Koedinger, and B. Junker. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. In *Intelligent Tutoring Systems*, pages 164–175, Berlin, 2006.

[5] 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, 1995.

[6] A. Heathcote, S. Brown, and D. J. K. Mewhort. The power law repealed: The case for an exponential law of practice. *Psychonomic Bulletin & Review*, 7(2):185–207, 2000.

[7] K. R. Koedinger, R. S. J. d. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A Data Repository for the EDM community: The PSLC DataShop. In C. Romero, S. Ventura, M. Pechenizkiy, and R. S. J. d. Baker, editors, *Handbook of Educational Data Mining*. CRC Press, 2010.

[8] P. Langley, J. E. Laird, and S. Rogers. Cognitive architectures: Research issues and challenges. *Cognitive Systems Research*, 2009.

[9] P. Langley, H. A. Simon, and G. L. Bradshaw. Heuristics for empirical discovery. In L. Bolc, editor, *Computational Models of Learning*. Springer, 1987.

[10] N. Li, W. W. Cohen, and K. R. Koedinger. Problem Order Implications for Learning Transfer. In *Proceedings of the Eleventh International Conference on Intelligent Tutoring Systems*, pages 185–194. Springer Berlin Heidelberg, 2012.

[11] N. Li, N. Matsuda, W. W. Cohen, and K. R. Koedinger. Integrating representation learning and skill learning in a human-like intelligent agent. *Artificial Intelligence*, 219:67–91, 2014.

[12] N. Li, E. Stampfer, W. W. Cohen, and K. R. Koedinger. General and Efficient Cognitive Model Discovery Using a Simulated Student. In *Proceedings of the 35th Annual Meeting of the Cognitive Science Society*, 2013.

[13] R. Liu, K. R. Koedinger, and E. A. McLaughlin. Interpreting Model Discovery and Testing Generalization to a New Dataset. In *Proceedings of the Sixth International Conference on Educational Data Mining*, pages 107–113, 2014.

[14] R. Liu, R. Patel, and K. R. Koedinger. Modeling Common Misconceptions in Learning Process Data. In *Proceedings of the 6th International Conference on Learning Analytics and Knowledge*, 2016.

[15] C. J. MacLellan, E. Harpstead, V. Aleven, and K. R. Koedinger. TRESTLE: Incremental Learning in Structured Domains using Partial Matching and Categorization. In *Proceedings of the Third Conference on Advances in Cognitive Systems*, pages 1–18, 2015.

[16] C. J. MacLellan, K. R. Koedinger, and N. Matsuda. Authoring Tutors with SimStudent: An Evaluation of Efficiency and Model Quality. In *Proceedings of the 12th International Conference on Intelligent Tutoring Systems*, 2014.

[17] D. Marr and T. Poggio. From understanding computation to understanding neural circuitry. 1976.

[18] N. Matsuda, W. W. Cohen, and K. R. Koedinger. Teaching the Teacher: Tutoring SimStudent Leads to More Effective Cognitive Tutor Authoring. *International Journal of Artificial Intelligence in Education*, 25(1):1–34, 2014.

[19] N. Matsuda, A. Lee, W. W. Cohen, and K. R. Koedinger. A Computational Model of How Learner Errors Arise from Weak Prior Knowledge. In *Annual Conference of the Cognitive Science Society*, pages 1288–1293, 2009.

[20] T. M. Mitchell. *Machine Learning*. McGraw-Hill, Boston, 1997.

[21] A. Newell. You can't play 20 questions with nature and win: Projective comments on the papers of this symposium. In *Visual Information Processing*, pages 283–308, 1973.

[22] A. Newell and P. S. Rosenbloom. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, 1981.

[23] J. K. Olsen, V. Aleven, and N. Rummel. Predicting Student Performance In a Collaborative Learning Environment. In *Proceedings of the 8th International Conference on Educational Data Mining*, 2015.

[24] R. Patel, R. Liu, and K. Koedinger. When to Block versus Interleave Practice? Evidence Against Teaching Fraction Addition before Fraction Multiplication. In *Proceedings of the 38th Annual Meeting of the Cognitive Science Society*, Philadelphia, PA, 2016.

[25] J. R. Quinlan and R. M. Cameron-Jones. Induction of logic programs: FOIL and related systems. *New Generation Computing*, 13(3-4):287–312, 1995.

[26] M. A. Rau. Why do the rich get richer? A structural equation model to test how spatial skills affect learning with representations. In *Proceedings of the Eighth International Conference on Educational Data Mining*, 2015.

[27] J. Rollinson and E. Brunskill. From Predictive Models to Instructional Policies. In *Proceedings of the Eighth International Conference on Educational Data Mining*, pages 1–8, 2015.

[28] C. Romero and S. Ventura. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 40(6):601–618, 2010.

[29] R. A. Schmidt and R. A. Bjork. New Conceptualizations of Practice: Common Principles in Three Paradigms Suggest New Concepts for Training. *Psychological Science*, 3(4):207–217, 1992.

[30] C. Tenison and J. R. Anderson. Modeling the Distinct Phases of Skill Acquisition. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 2015.

[31] K. Vanlehn. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education*, 16(3):227–265, 2006.

[32] D. A. Waterman and F. Hayes-Roth. An overview of pattern-directed inference systems. 1978.

[33] E. S. Wiese. *Toward Sense Making with Grounded Feedback*. PhD thesis, Pittsburgh, 2015.