Computational Models of Learning: Deepening Care and Carefulness in AI in Education

Daniel Weitekamp[1] and Kenneth Koedinger[1]

Carnegie Mellon University, Pittsburgh PA 15289, USA

Abstract. The field of Artificial Intelligence in Education (AIED) *cares* by supporting the needs of learners with technology, and does so *carefully* by leveraging a broad set of methodologies to understand learners and instruction. Recent trends in AIED do not always live up to these values, for instance, projects that simply fit data-driven models without quantifying their real world impact. This work discusses opportunities to deepen careful and caring AIED research by developing theories of instructional design using computational models of learning. A narrow set of advances have furthered this effort with simulations of inductive and abductive learning that explain how knowledge can be acquired from experience, initially produce mistakes, and become refined to mastery. In addition to being theoretically grounded, explainable, and empirically aligned with patterns in human data, these systems show practical interactive task learning capabilities that can be leveraged in tools that interactively learn from natural tutoring interactions. These efforts present a dramatically different perspective on machine-learning in AIED than the current trends of data-driven prediction.

Keywords: Student Models · Simulated Learners · Computational Models of Learning · Interactive Task Learning · Caring AI

1 Introduction

These proceedings celebrate the 30th anniversary of the Artificial Intelligence in Education society. At its founding in 1993 John Self was the editor of the JAIED journal and the AIED society's president. Self's belief was that AIED is set apart from other fields of computing because—as Kay and Mcella have paraphrased nicely—AIED "cares" and is "careful" [7] (or in Self's words "cares, precisely" [23]). Self argued that AIED *cares* by adapting to and supporting the needs of human learners. Unlike theoretical computer science, AIED engages with human participants and embraces the reality that educational technology has an impact on people in the real world. AIED cares by supporting learners as they are, not as we imagine them to be. AIED is *careful* because it uses a variety of research tools to deepen its understanding of its subjects. It draws from and contributes back to a broad set of disciplines, including artificial intelligence, cognitive psychology, education, and sociology. AIED researchers use interdisciplinary tools as needed to understand and model learners, build theories of learning and instruction, and effectively help learners become what they aim to be.

Self coined the term "computational mathetics" to refer to a broad set of disciplines that study matters of learning, including many forms of student modeling and diagnostic agents familiar to us today [22]. However, in recounting the legacy of computational

mathetics, he lamented that student modeling had failed to produce solid theoretical groundings to inform Intelligent Tutoring System (ITS) design, compared to for instance, other fields of engineering like aeronautics:

> "Aircraft design has progressed through many centuries of visions and a few decades of serious experimentation to largely depend on the theory of aeronautics and specialised test environments. Would ITSs ever be built by a blend of beautiful theory and empirical fine tuning?" (1998, pg. 354)

Self imagined that if student models were not just data-structures, but also executable programs then they could "be used not only descriptively but also predictively, to predict how a student would solve problems in the future (assuming the model were accurate)" (1998, pg. 351). In other words, if student models were simulations of learning they could be executed on instructional material to make relative predictions about instructional design choices, much like a wind tunnel guides aeronautic design. Self believed that the field was hesitant to approach the topic of student simulation because it was too "complex" (1995, pg. 92) for the theories of learning and AI of the time.

In 2023, it is worth reconsidering if Self's high fidelity computational models of learning are on the horizon. As we argue in the following pages, the answer is resoundingly yes!—but not necessarily because of the technologies driving the current AI hype. But, if we adopt an appropriate mindset of care and carefulness in choosing our foundations, there does seem to be an exciting path forward.

First, we should consider the state of *care* and *carefulness* in current AIED research. It's all too easy to claim those virtues by association without knowing how to, or caring to put them into practice, and there is a fine line between the ideal and debatable cases. Throughout its history AIED has thrived from the use of caring and careful interdisciplinary methods like user studies and A/B experiments that reach outside of our core computer science toolbelt, into the real world. But the consideration of care and carefulness is not characterized only by checking methodological boxes in the classroom. Care and carefulness are as much applicable in purely analytical or computational programs of research that begin and end behind a computer screen. The bare minimum requirement of *caring* in AIED research is that we quantify and build arguments for the value of our work in the real world. In advance of our consideration of how computational models of learning can help us deepen these values, let us digress to highlight a subfield of AIED in which this *caring* justification often falls short. In consideration of Self's sentiment that learning engineering had methodological holes [23], we will take an analytical estimation approach—an approach common among engineers but underutilized in AIED. With a simple back-of-the-napkin calculation we will estimate the potential real world impact of building better knowledge tracers and in doing so highlight some epistemic pitfalls to engaging in *caring* and *careful* student modeling.

2 Do better student models produce better knowledge tracing?

Knowledge tracers [5] estimates students' knowledge of individual skills, concepts, or facts as the probability of answering future practice items correctly [21], or of achieving a latent "mastery" state [5]. Students continue practicing problems associated with particular skills, concepts, or facts until the tracer's estimate of mastery exceeds some threshold. Knowledge tracers are typically fit to student performance data and utilize time, number of practice opportunities, or other features to estimate mastery.

Let us raise and attempt to answer the question analytically: If we built a 5% better statistical model for knowledge tracing how much time would we expect the new tracer to save students? Consider the following calculation (with code[1]). Our analytic approach, confronts us immediately with an often overlooked consideration: that the model's performance in the neighborhood of the mastery threshold is the only region where performance matters since this is where the tracer decides if it should stop selecting items of each kind. Between an old model and a new one a total RMSE (Root-Mean-Square-Error) improvement of 5% is a fairly large one—comparable for instance to the difference between the best model in [6] (0.329 RMSE) and normal Performance-Factors-Analysis (0.343 RMSE) [21][2]. But model improvements are rarely reported only around the mastery threshold. So we'll assume, perhaps incorrectly, that a total model improvement of 5% implies a 5% improvement in this critical neighborhood.

Consider a logistic regression model like Cen. et. al [3] where number of opportunities is our only feature. We'll assume that the performance threshold point for mastery is 95% and that we have a new model with 5% better RMSE at the theshold point than an older one. Independent of our choice of model, we can compute analytically that the old will report 87.84% student accuracy instead of 95% (assuming it was underestimating). Let us also assume that our new model is actually the ground-truth model and set a few constants of this ground truth: a first opportunity intercept of 35% accuracy, and an average of 12 opportunities to achieve mastery. Assuming equal proportions of change in intercept and slope, we can expect our new model to save students an average of 4.19 extraneous practice attempts. If every attempt takes 15 seconds, and there are 500 knowledge components learned over a year, then over the course of that year, the old 5% worse model would have students practice 8.75 hours more or about 35% of the total 25 hours expected in ground-truth. Compared to similar model-driven calculations reported by Yudelson et.al [32][31], our analytically derived 35% improvement is an overestimate, but well within the right order of magnitude.

Should we then conclude that the project of building incrementally better knowledge tracers is a valuable one? Yet more careful considerations notwithstanding[3], it would seem that the answer is yes!—at least insofar as we can build yet better student models. But, it would seem that this project is only accidentally aligned with a justifiable real-world impact. If our calculation had found that a 5% RMSE improvement produced only a 0.1% time saving, or if a review of the last two decades of student modeling showed no improvements in the neighborhood of the mastery threshold (only in the region preceding it), then we would be forced to accept those efforts as time wasted. The typical measures of overall fit statistics simply do not address these concerns. They do not quantify improvement in terms of the real-world experiences of students.

In this particular subfield, the *caring* consideration of connecting models to their real world impact is decidedly the exception and not the rule. This pattern should concern us. Neglecting to connect models to their potential real world impact is like building a car and never test driving it. The apparent source of this cognitive dissonance is

[1] https://github.com/DannyWeitekamp/Quantifying-Knowledge-Tracing-Time-Saved

[2] KDD Cup 2010 EDM Challenge: Algebra I 2005-2006 dataset

[3] Knowledge tracers are fit to formative assessment data not delayed summative assessments. An improved fit to formative data is no guarantee of improved supports for long-term retention.

an overemphasis on the logic of justification of big-data machine-learning which consider systematic benchmarks and improvements in fit statistics as valuable in their own right—hardly a caring or careful logic if it makes no direct connection to its impact on real human learners. Taking the final step of justifying the real world impact of models should be standard practice in AIED.

2.1 Examples of Fitting Models with Care

Today's data-driven machine-learning allows for an approach where problems are solved *just as prediction problems*. But we should consider what is lost in that simplicity. We can almost always better support learning by seeking to theoretically understand it. AIED researchers should always consider engaging with their research from a learning scientist's perspective—raising and attempting to answer questions regarding their subjects of inquiry: human learners and how to best support their learning with technology. Exemplary instances of this mindset include learning curve analysis and automatic domain model selection [3]. These tools also fit student performance models—the difference is how they are used. In these cases, fit statistics and patterns of performance are not ends in themselves, they are used diagnostically for finding real issues in learning materials. Negative signals like flat learning curves and poor relative domain model fits can help learning engineers identify issues in instructional materials [3]. Nevertheless, interpreting these signals and revising instruction in response is still more of an art than science, requiring the engineer to make educated guesses about how best to best modify instruction. While these tools deepen our understanding of our subjects of inquiry, there is a great deal more we can do to deepen our understanding of learners and instruction beyond what existing tools allow.

3 Going Deeper with Computational Models of Learning

A common mentality of AI researchers well into the 1980s was that AI research could be a means of building and testing theories of learning—a complementary approach to the experimental methods of cognitive psychology. AI research has shifted away from this perspective toward purely technical and practical concerns, but it is worth reflecting on whether we've lost elements of care and carefulness in that shift.

Unlike purely practical AI, computational models *care* by simulating elements of cognition *explicitly*, instead of treating them *implicitly* by reducing cognitive phenomena to numerical quantities or probabilities like in statistical models that fit parameters to data. A computational model embodies an executable theory which is often more detailed than an experimentally driven theory because the computational theory must be precise enough to be implemented in simulation, yet general enough to reproduce broad sets of human behaviors, and furthermore plausible enough to satisfy known constraints of human cognition, like for instance, having only particular prior knowledge or a finite working memory, to name just a few [12]. When computationally modeling *learning*, the modeler is confronted not only with these computational and cognitive constraints, but also with an empirical reality glimpsed through student data. The simulation must succeed at learning from the kinds of natural instruction that humans experience, [16] and reproduce patterns of correct performance and errors.

As a practical means of deepening our insights into human learning, computational models of learning can be used to make *a priori* predictions without need for prior collection of student data. For instance: Which of two forms of instruction will work best? What kinds of misinterpretations and mistakes will students make as they practice? How can we most optimally adapt to particular student behaviors or traits and produce the best learning outcomes? In answering any of these questions, a computational model has a key advantage over a statistical one: simulated cognitive reasons underlying its predictions. A computational model simulates the formation of knowledge in response to particular instructional events, and the execution of that acquired knowledge on new material. As a predictor, this goes well beyond simply fitting parameters to patterns in data, and holds the promise of enabling learning engineers to not only predict which instructional design decisions are optimal but also predict why. These explainable patterns of simulated cognition also allows researchers to debug and revise their theories intelligently when the computational model's predictions prove to be incorrect.

Developing AI under the many constraints of computational modeling is no doubt challenging. It is a cyclic project of refinement aimed at building general purpose executable simulations of learning that can be applied to a wide array of domains, learn from several forms of instruction, and match human learning behavior faithfully. It is a harder and more principled project than the development of purely practical or predictive AI where performance statistics are often the sole guiding objective. Nonetheless, it is likely the most direct and precise means of building a theory of instructional design.

3.1 The Baby in the Good Old-Fashion AI Bathwater

In setting foundations for computational models of learning, it is tempting to forgo early forms of AI and look to the trends of data-driven machine-learning. A dominant theme of the last decade of AI has been to demonstrate that new capacities can be acquired by fitting high-dimensional deep-learning [13] models to copious amounts of data. Deep-learning has been used to train many state-of-the-art AI like Alpha-Go and Chat-GPT, yet it it offers us very little toward the purpose of modeling human learning. Despite the complexity of its high-dimensional multi-layered structure, deep-learning relies on a single over-simplified and data-hungry learning mechanism: regression via gradient descent. By comparison, humans are remarkably data-efficient learners able to achieve mastery in academically relevant tasks from only a handful of instructional and practice opportunities. Humans, no doubt, owe their learning efficiency to a variety of forms of reasoning that can help them rapidly construct and verify knowledge. Since our objective with computational modeling is to explain and intelligently support these remarkable learning capacities, the data-inefficiency and inexplicability of deep-learning's *blackbox* knowledge structures make for poor foundations.

If deep-learning has dispensed with the capabilities we care about in computational modeling, perhaps we should return to the hard-coded expert-systems of the Good Old-Fashion AI (GOFAI) era? This too would be a poor choice. Expert-systems model the execution of human-like expertise using hard-coded rules, but do not model the acquisition of expertise from experience. One could be forgiven for thinking that we are necessarily stuck between two bad options: the choice between the flexibility of acquiring blackbox representations in a data-driven manner, and the rigidity of hard-coded but explainable symbolic knowledge structures. Even cognitive architectures like SOAR

[10] and ACT-R[2], that aim to model learning, align heavily with the hard-coded inflexibility of GOFAI. They rely heavily on predefined domain-specific rules—modeling learning only by tuning the activation patterns of hard-coded rules or by recombining them into new specialized structures. The gaping hole in these theories is how those initial rules came to be acquired through experience, and how that acquired knowledge can produce human errors without the sources of errors being explicitly programmed.

To achieve the best of both options, there is a very clear, but seriously underutilized approach to machine-learning—a neoclassical approach where expert-system-like rules can be efficiently acquired in a highly flexible yet mostly symbolic bottom-up fashion. One place to look for such advances is in the emerging field of interactive tasking learning (ITL) [11]. ITL systems seek to acquire performance capabilities from just a few instances of human provided instruction. Some ITL systems like Rosie [19], are built on cognitive architectures, and share many of their assumptions. Many more ITL innovations may deviate entirely from feasible human cognition, so our choice of adopting these innovations should be made carefully.

In the near-term the more directly useful neoclassical approaches to machine-learning can be found in past AIED research on simulated students, which have been touted as foundations of ITL in their own right [11]. Three such systems, Sierra [24], SimStudent [18] and the Apprentice Learner (AL) [17], are able to learn to perform academically relevant procedural tasks (including but not limited to, learning math and science procedures) from at least two kinds of instruction: demonstrations of correct performance, and positive and negative correctness feedback—core forms of instruction that students experience in a one-on-one tutoring setting or while working in an intelligent tutoring system. Since these systems can learn in a bottom-up fashion from learning materials like intelligent tutoring systems (ITSs), they are a compelling basis for Self's proposed executable student models. They provide an answer to the question of how knowledge can be efficiently induced from first experiences, initially produce errors, and then become refined into expertise through practice.

3.2 Setting our Foundations in Induction and Abduction

There are several broad categories of learning consistent with this neoclassical view of machine-learning that we might consider as foundations for a computational model of learning. I highlight Sierra, SimStudent and AL because they implement purely inductive and abductive learning mechanisms. Induction is the formation of knowledge by generalizing from examples. Abduction is the use of existing knowledge to find the most likely reason underlying an example. Both forms of reasoning can produce imperfect knowledge structures from instruction and reproduce human errors.

There are certainly alternative foundations. For instance, deduction: finding that which follows logically from what is already known, and—drawing from the suggestions of prior works [16][27]— learning from being told, learning by analogy [8], planning and debugging towards a goal [10], reinforcement learning [20], speed-up learning through practice [10][2], just to name a few. It is all too easy to fool ourselves into thinking that any one of these particular learning capabilities supersedes the rest, encompassing a vaguely characterized notion like *logical reasoning* or *general intelligence*. We should consider the scope and applicability of each of these proposed mechanism

individually and perhaps aim to unite these disparate methods under a unified toolset, and with it build a sort of model-human learner.

The reason to favor induction and abduction as foundations over other choices is they offer a means of generating potentially buggy knowledge from experiences. Many forms of learning require some initial prior domain-specific knowledge. Even learning from being told—which is the closest thing to directly programming a human, but not anywhere nearly as precise or unambiguous—requires prior language comprehension capabilities and vocabulary. If prior knowledge is always hard-coded as infallible expertise then we run the risk of creating toy-models that replicate the acquisition of capacities but fail to reproduce patterns of human error. Thus, the property of induction and abduction of generating buggy knowledge from experiences establishes a solid foundation for mechanistically modeling the real messiness of human learning. We shouldn't resign ourselves to the idea that learner errors are purely random, or that states of knowledge are simply scalars on a continuum from unknown to known. Principled computational modeling rooted in induction and abduction can simulate the reasons underlying human errors, and perhaps with this deeper theoretical understanding, allow us to make more intelligent instructional decisions.

3.3 Evidence of Executable Student Models

Results from prior work with Sierra, SimStudent and the Apprentice Learner (AL) show a compelling foundation for using inductive and abductive learning mechanisms in executable theories of learning. An algorithmic level discussion of these systems is beyond the scope of this work. Although, it is worth noting that they all rely on a combination of multiple, mostly symbolic machine-learning mechanisms, and share a common high-level structure—a testament to the fact that two computational modelers held to the same constraints tend to come to similar solutions.

Generators of Errors and Domain Models: VanLehn et. al. found that when taught multi-column subtraction, Sierra could reproduce two-thirds of the types of errors students produced on quizzes—more than twice the errors explained by ad-hoc expert-system based theories characterized by hard-coded "repairs" [25]. Sierra's general theory of inductive and abductive learning, was simultaneously more parsimonious than the hard-coded theories yet better at generating a broad set of observed human errors. A similar result, by Li et. al. showed that SimStudent could produce a better fitting domain model of a simple algebra ITS than those hand-built by learning engineers [14]. The domain model was constructed by using the execution of SimStudent's induced representations and production-rules to form knowledge-component to item mappings. These examples show evidence that general theories of bottom-up learning from experiences can generate empirically better predictions of overall student performance behaviors than ad hoc human-generated ones. An appropriate high-level theory can produce surprising and highly specific theoretical predictions.

Models of Individuals: To explain the behavior of a theoretical *average* student is one thing. It is another to explain the behavior of particular individuals. In this regard, simulated learner technologies are still very young, but the results to date offer a compelling picture of where they can take us. Maclellan et. al. showed that when Apprentice

Learner (AL) agents were taught on the same sequences of ITS problems as students, they could reproduce student error rates opportunity-by-opportunity in several different domains—producing population learning-curves that aligned well with student performance data without explicitly fitting to it [17]. As an interactive task learning system AL is remarkably data-efficient and can produce learning curves orders of magnitude closer to human curves than, for instance, deep reinforcement-learning.

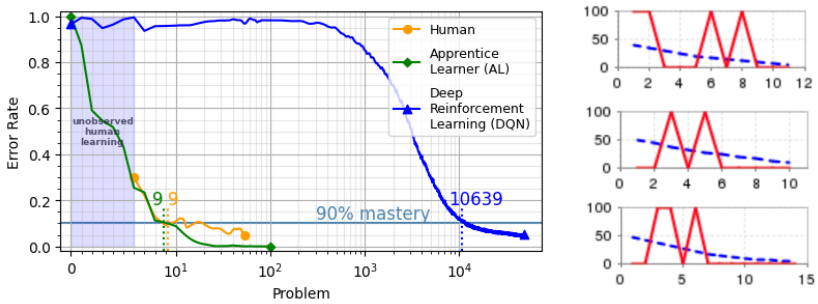

Fig. 1: (left) AL learning curves on fraction addition and multiplication. Learning-rate is far closer to humans than deep reinforcement-learning. (right) 1/0 pattern of correctness of single AL agents on denominator conversion steps.

Building learning curves by applying AL's theory of learning to an instructional environment (like an ITS) opens a profoundly different perspective on student modeling than the typical data-driven one. When AL works in an instructional environment, it learns to solve problems and produce actual step-by-step responses. By contrast, a typical statistical model reduces student attempts to a binary random variable where the probability of correct performance increases over time—an approach that seems appropriate because the patterns of correct (1) and incorrect (0) responses in student data often seem non-deterministic. For instance, a typical performance pattern might be "0010111", where the student answers correctly, but then incorrectly on later items before consistently answering correctly. AL can reproduce this seemingly random behavior (Figure 1, right) without explicitly injecting randomness. AL agents use evidence from individual instances of instruction to refine induced skills with every practice opportunity, especially on ones where its current knowledge structures produce mistakes that receive negative feedback. So an AL agent's pattern of performance can oscillate between correct and incorrect as it solves new problems and converges toward mastery. As a theory, this envisions the particular step-by-step differences in responses between individual learners as arising from differences in prior experiences, creating particular knowledge that when applied to particular new problems, produces particular responses.

This is not to say that this prior work has achieved a sort of Laplace's demon of human learning—an infallible predictor of future responses from a known starting point. A student's starting point, the state of their prior knowledge before using an ITS, is always a matter of uncertainty. Students can encounter instructional opportunities in the classroom or at home that an ITS can't be expected to know about. In these cases we have to make guesses. For instance Weitekamp et. al., found better fits in AL generated curves by pre-training agents with a number of random practice opportunities

commensurate with student's predicted prior knowledge [29]—the content of prior experience was guessed while the quantity was tailored to the individual. Other cognitive considerations, not included in these works like mechanisms of forgetting and attention are harder to model purely mechanistically, and may require some probabilistic treatment. Beyond these considerations there is a litany of individual biological, social, and environmental factors—did the student sleep well? eat breakfast? do they have ADHD?—that one could imagine injecting theories of into such a system.

In the near-term getting the general theory of learning broadly correct, is more important than adding precise theories of, typically unobservable, individual factors. For instance, Weitekamp et. al. have reported discrepancies between trends in the types of errors made by simulated and real students attempt-by-attempt [30]—so the cycle of theory refinement is well underway. Our larger takeaway should be that the path toward Self's proposed executable student model is very clear. These demonstrations of computational models of learning are surprisingly parsimonious, domain-general, and accurate *a priori* first-order approximations. And, since they are inspectable and debuggable there is a path forward of further incisive and deliberate model development constrained by computational, empirical, and cognitive constraints.

Testers of Instruction: So how closely do these simulated students need to align with human learning behavior before we begin to trust them to make decisions about instructional design? Maclellan et. al. showed that AL agents could replicate broad differences in patterns of student performance between blocked and interleaved instruction [17]. Beyond this, several unpublished results from participants in an AL-base workshop series capture the notion that if a simulated student fails to learn from an ITS, then it may not be adaptive enough to support low achieving human students. A simulated student failing to learn from an ITS can be a signal that the target knowledge taught by the ITS does not follow logically from the provided instruction, even by induction over several problems. A high-achieving, high prior knowledge student might succeed despite this lack of support. But just like a human student with low-prior knowledge a simulated student can typically only learn from instruction that begins from first principles and breaks down problems into fine-grained substeps.

In addtion to succeeding or failing outright a simulated learner can somteimes learn more efficiently from one form of instruction than another. For instance, AL agents can learn multi-column addition considerably faster when the ITS teaches a version of the procedure where 0's are explicitly carried for each partial sum instead of being omitted. This explicit-zero version of addition is easier for the agent because it provides a place in the problem interface for explicitly accounting for having calculated the carry value. So, the overall procedure follows a more consistent, easy to learn pattern. As a theory this result makes the novel prediction that the same would be true for students.

4 Additional Practical Uses of Simulated Learners

So far I've addressed how we can gain a deeper theoretical understanding of learning and instruction by building computational models of learning. Those of us accustomed to wielding AI as a sort of magic wand of prediction will surely see more direct data-driven or ready-made solutions (i.e like chat GPT) to a variety of practical problems in education. In the near-term I'm certain that a great deal will be achieved this way, but

in the long-term I'm less certain that this will be a consistently fruitful approach. It's hard to imagine deep-learning's fuzzy mimicry holding a monopoly over more cognitively principled approaches, especially when it comes to matters of supporting human learning.

If deep-learning is today's AI magic wand then simulated learners may well be tomorrow's—or at least find similarly wide applicability among the options in our AI toolbag. As purely practical AI, simulated learners are remarkably data-efficient and domain-general interactive tasks learners. They stand apart among ITL systems because of an emphasis on cheating less with hard-coded elements and specialized toy-environments, and not at all on mimicking patterns from large datasets. Instead they have sought domain-general mechanisms of learning that can efficiently construct knowledge bottom-up from natural tutoring interactions. For instance, Li [15] added a representation-learning mechanism to SimStudent to eliminate hard-coded representational prior knowledge. Maclellan [17] with the creation of AL loosened requirements of special supplementary instructional experiences accompanying demonstrations. Soon mechanisms for learning from natural language instruction will be incorporated into AL. All of these efforts have manifested, in simulation, a human-like ability to learn in diverse ways from diverse experiences. This flexibility presents an opportunity to build experiences where non-programmers teach simulated learners interactively.

Authoring ITSs: For instance, prior work has shown that simulated learners can be used to author ITSs [18] faster than existing tools [28]. Programming production-rule based ITSs require about 200-300 hours per hour of instruction authored. GUI-based tools cut this time down by about half, but add restrictions on what can be built [1]. By instead authoring via interactively tutoring a simulated learner, the production rules of an ITS can be induced from natural instruction instead of being programmed. This could reduce authoring times to about the time taken to tutor a student: one hour per hour of instruction. One does not need to do a particularly complex calculation to see the potential impact here—the multiplicative effect of a wide array of people building highly adaptive ITSs faster. With such a tool ITSs might be able to scale in the way that MOOCs have, transitioning learning at scale away from passive content delivery toward the highly adaptive deliberate practice based instruction characteristic of ITSs [9].

Teacher Training: There is a vast difference between knowing how to do something and knowing how to teach it. A core element of that difficulty is that humans are rather poor at reflecting on, and articulating the content of their tacit knowledge [4]. Another is that it is hard to make *a priori* predictions about the misconceptions that novices may acquire, and so it is difficult to tailor instruction to address or avoid them. If teachers practiced by teaching simulated learners they may acquire a deeper sense of why misconceptions arise since they can inspect the agent's knowledge and inspect instances of how buggy knowledge is constructed. They could also test various forms of instruction to see which methods are most effective. VanLehn et. al. have articulated this vision in much greater detail [26]. But, I might add to that vision the possibility of creating domain-general versions of such tools for teaching new or specialized material. Professional expertise is often acquired through experience, but not taught deliberately. Future simulated learners may aide everyday professional development by lending explicit cognitive support in workplace apprenticeship relationships.

5 Conclusion

This work has explored John Self's themes of care and carefulness in AIED, especially with respect to analytical and computational projects of research. We have argued that applications of data-driven machine-learning in AIED don't always live up to these virtues, as they often follow a misguided logic of justification based on improving performance statistics that rarely take the final step of quantifing real-world impact. We have proposed computational models of learning as means of engaging in more deeply theoretical, caring and careful computational AIED research. Computational models of learning care by seeking an explicit and detailed understanding of the mechanisms underlying learning, which are often treated only implicitly through statistical modeling.

In any applied field of research there is a balance between theoretical and practical advances. But we shouldn't forget that in many fields of engineering theoretical advances have streamlined the development of the greatest practical advances. Self's view was that instructional design could be improved significantly if we had high fidelity theories of learning that could inform instructional design. He imagined executing such theories on instructional material, much like a wind tunnel tests aeronautic designs. I've demonstrated here that prior simulated learner systems have begun this project in earnest. The results to date show a promising path toward realizing Self's vision of executable theories of learning. These systems additionally show several practical interactive task learning use cases like ITS authoring, and teacher training. Perhaps the most exciting element of these advances is their origins in AIED research. The unique caring and careful perspective of AIED can contribute as much to the field of machine-learning as it does to the classroom.

References

1. Aleven, V., McLaren, B., Sewall, J., Van Velsen, M., Popescu, O., Demi, S., Ringenberg, M., Koedinger, K.: Example-tracing tutors: Intelligent tutor development for non-programmers. International Journal of Artificial Intelligence in Education **26**(1), 224–269 (2016)
2. Anderson, J.R., Matessa, M., Lebiere, C.: ACT-R: A theory of higher level cognition and its relation to visual attention. Human–Computer Interaction **12**(4), 439–462 (1997)
3. Cen, H., Koedinger, K., Junker, B.: Learning factors analysis–a general method for cognitive model evaluation and improvement. In: International Conference on Intelligent Tutoring Systems. pp. 164–175. Springer (2006)
4. Clark, R.E., Feldon, D.F., Van Merrienboer, J.J., Yates, K.A., Early, S.: Cognitive task analysis. In: Handbook of research on educational communications and technology, pp. 577–593. Routledge (2008)
5. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction **4**(4), 253–278 (1994)
6. Gervet, T., Koedinger, K., Schneider, J., Mitchell, T., et al.: When is deep learning the best approach to knowledge tracing? Journal of Educational Data Mining **12**(3), 31–54 (2020)
7. Kay, J., McCalla, G.I.: The careful double vision of self. Int. J. Artif. Intell. Educ. **13**(1), 11–18 (2003)
8. Klenk, M., Forbus, K.: Analogical model formulation for transfer learning in ap physics. Artificial intelligence **173**(18), 1615–1638 (2009)
9. Koedinger, K.R., Kim, J., Jia, J.Z., McLaughlin, E.A., Bier, N.L.: Learning is not a spectator sport: Doing is better than watching for learning from a MOOC. In: Proceedings of the second (2015) ACM conference on learning@ scale. pp. 111–120 (2015)

10. Laird, J.E.: The Soar cognitive architecture. MIT press (2019)
11. Laird, J.E., Gluck, K., Anderson, J., Forbus, K.D., Jenkins, O.C., Lebiere, C., Salvucci, D., Scheutz, M., Thomaz, A., Trafton, G., Wray, R.E., Mohan, S., Kirk, J.R.: Interactive Task Learning. IEEE Intelligent Systems **32**(4), 6–21 (2017)
12. Langley, P.: The computational gauntlet of human-like learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 12268–12273 (2022)
13. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)
14. Li, N., Cohen, W.W., Koedinger, K.R., Matsuda, N.: A machine learning approach for automatic student model discovery. In: Edm. pp. 31–40. ERIC (2011)
15. Li, N., Matsuda, N., Cohen, W.W., Koedinger, K.R.: Integrating representation learning and skill learning in a human-like intelligent agent. Artificial Intelligence **219**, 67–91 (2015)
16. MacLellan, C.J., Harpstead, E., Marinier III, R.P., Koedinger, K.R.: A framework for natural cognitive system training interactions. Advances in Cognitive Systems **6**, 1–16 (2018)
17. Maclellan, C.J., Harpstead, E., Patel, R., Koedinger, K.R.: The Apprentice Learner Architecture: Closing the loop between learning theory and educational data. International Educational Data Mining Society (2016)
18. Matsuda, N., Cohen, W.W., Koedinger, K.R.: Teaching the teacher: Tutoring simstudent leads to more effective cognitive tutor authoring. International Journal of Artificial Intelligence in Education **25**(1), 1–34 (2015)
19. Mininger, A., Laird, J.: Interactively learning a blend of goal-based and procedural tasks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)
20. Nason, S., Laird, J.E.: Soar-RL: Integrating reinforcement learning with Soar. Cognitive Systems Research **6**(1), 51–59 (2005)
21. Pavlik Jr, P.I., Cen, H., Koedinger, K.R.: Performance factors analysis–a new alternative to knowledge tracing. Online Submission (2009)
22. Self, J.: Computational mathetics: towards a science of learning systems design (1995)
23. Self, J.: The defining characteristics of intelligent tutoring systems research: ITSs care, precisely. International Journal of Artificial Intelligence in Education (IJAIED) **10**, 350–364 (1998)
24. VanLehn, K.: Learning one subprocedure per lesson. Artificial Intelligence **31**(1), 1–40 (1987)
25. VanLehn, K.: Mind bugs: The origins of procedural misconceptions. MIT press (1990)
26. VanLehn, K., Ohlsson, S., Nason, R.: Applications of simulated students: An exploration. Journal of artificial intelligence in education **5**, 135–135 (1994)
27. VanLehn, K.A.: Arithmetic procedures are induced from examples. Tech. rep. (1985)
28. Weitekamp, D., Harpstead, E., Koedinger, K.: An interaction design for machine teaching to develop ai tutors. CHI (2020)
29. Weitekamp, D., Harpstead, E., MacLellan, C.J., Rachatasumrit, N., Koedinger, K.R.: Toward near zero-parameter prediction using a computational model of student learning. International Educational Data Mining Society (2019)
30. Weitekamp, D., Ye, Z., Rachatasumrit, N., Harpstead, E., Koedinger, K.: Investigating differential error types between human and simulated learners. In: International Conference on Artificial Intelligence in Education. pp. 586–597. Springer (2020)
31. Yudelson, M., Ritter, S.: Small improvement for the model accuracy–big difference for the students. In: Industry Track Proceedings of 17th International Conference on Artificial Intelligence in Education (AIED 2015), Madrid, Spain (2015)
32. Yudelson, M., Koedinger, K.: Estimating the benefits of student model improvements on a substantive scale. In: Educational Data Mining 2013 (2013)