Analytic Comparison of Three Methods to Evaluate Tutorial Behaviors

Jack Mostow and Xiaonan Zhang
mostow@cs.cmu.edu, xiaonanz@cs.cmu.edu
Project LISTEN, School of Computer Science, Carnegie Mellon University

Abstract. We compare the purposes, inputs, representations, and assumptions of three methods to evaluate the fine-grained interactions of intelligent tutors with their students. One method is conventional analysis of randomized controlled trials (RCTs). The second method is learning decomposition, which estimates the impact of each practice type as a parameter in an exponential learning curve. The third method is knowledge tracing, which estimates the impact of practice as a probability in a dynamic Bayes net. The comparison leads to a generalization of learning decomposition to account for slips and guesses.

1 Introduction

Educational data mining researchers have used various methods to analyze fine-grained tutor data in order to evaluate the effects of tutorial actions on student behavior, including randomized controlled trials (RCTs), learning decomposition, and knowledge tracing. Other papers [1, 2] compare these three methods empirically. In this paper we compare their fundamental characteristics descriptively and analytically. We now briefly describe these methods, and then the dimensions along which the remainder of the paper compares them. We describe the three methods in terms of the high-level strategic ideas they embody, the particular instantiations that we compare, and specific implementations of these instantiations.

1.1 Randomized controlled trials analysis

The high-level strategic idea of RCTs is that randomizing assignment to treatments allows the strong causal inference that significant differences in outcome are due to differences in treatment. In RCTs, participants are randomly assigned to receive one of several different interventions. RCT analysis aggregates and compares the outcomes under each condition, to assess the effectiveness of different interventions. For example, Project LISTEN's Reading Tutor [3] chooses randomly among multiple ways to help the student learn a given word. To compare their effectiveness using RCT analysis, the Reading Tutor randomly chooses for each student how to teach or practice a given word. For each such randomized trial, we measure its outcome, e.g., how well the student reads the word at the next encounter of it. By comparing the aggregated outcomes for each intervention, we can statistically estimate how effective they are relative to one another.

RCT analysis is instantiated by a statistical test or model to estimate the effects of treatment and possibly other variables. Common examples include t-test and ANOVA.

The test or model is implemented as the computation of a standard statistical formula, typically in a statistical package such as SPSS.

1.2 Learning decomposition

The strategic idea of learning decomposition is to distinguish among different types of exposure in fitting a model of learning to students' performance data, in order to obtain empirical estimates of the relative impact of different types of practice or instruction. Prior work [4-6] instantiates this idea in terms of a particular model form, namely an exponential learning curve, by expressing the amount of exposure as a function of the amount of exposure of each type, namely a linear combination of them.

Alternative model forms would produce alternative instantiations. One plausible alternative to an exponential learning curve is a power law. Power laws are often used to fit performance data averaged over multiple participants, but exponential curves tend to fit individual performance data better than power law curves do [7]. A more complex alternative [8] is based on a richer theoretical model of learning and forgetting over time, but must be expressed recursively instead of in closed form, and has more parameters to fit.

The instantiation of learning decomposition used in this paper generalizes the classic exponential learning curve to distinguish the effectiveness of *m* different types of practice, as shown in Equation 1:

$$performance = A \times e^{-b \times (\beta_1 t_1 + \ldots + \beta_m t_m)} \quad \textbf{(1)}$$

In the equation, *performance* measures a learning outcome, such as error rate, help requests, or response time: the lower the value, the better the student learned. The variables t_i $(i=1\ldots m)$ represent the amount of type *i* practice that the student has had, as measured by the number of practice opportunities of that type. The free parameters *A*, *b*, and β_i's are estimated from observations of student performance. Here *A* represents the student's initial performance without any prior practice, and *b* is the learning rate. Finally, the free parameters β_i $(i=2\ldots m)$ represent the impact of type *i* practice relative to type 1 practice, whose impact β_1 we define to be 1 as a baseline for comparison.

For example, consider how learning decomposition can apply to our previous example of comparing different tutorial interventions to teach student a word. The *performance* measure can be word reading time. The variable t_i counts the number of times the student received intervention of type *i*. The estimate of each β_i measures the effectiveness of intervention type *i* compared to the baseline. For instance, if $\beta_2 = 2$, it means that on average a type 2 intervention is twice as helpful as a type 1 intervention.

Implementation of learning decomposition refers to how the model is fit to the data. For example, our implementation uses SPSS to perform unconstrained non-linear regression using the Levenberg-Marquardt procedure.

1.3 Knowledge tracing

The general idea of knowledge tracing is to model a student's performance and changing knowledge state during skill acquisition, and to use the model to update an estimate of

this state based on successive observations of student performance. Atkinson [9] proposed a Markov model relating knowledge to performance. Corbett and Anderson [10] used a simple 2-state model – the state of not knowing a given skill, and the state of knowing the skill.

An instantiation of knowledge tracing specifies a particular model of learning. We represent the two knowledge states in Corbett and Anderson's model as a hidden node in a dynamic Bayes network (DBN). The value of this node is true if the student knows the skill at a given step, and false if not. The node is hidden, so its value cannot be observed directly. Instead, knowledge tracing maintains the probability $\Pr(K_i)$ that the student has learned that skill at time step i. It uses observations of the student's performance to update this probability each time the student encounters an opportunity to use the skill. We extend this basic knowledge tracing model to incorporate and compare the efficacy of different types of tutorial interventions. Figure 1 shows the graphical representation of the extended model.

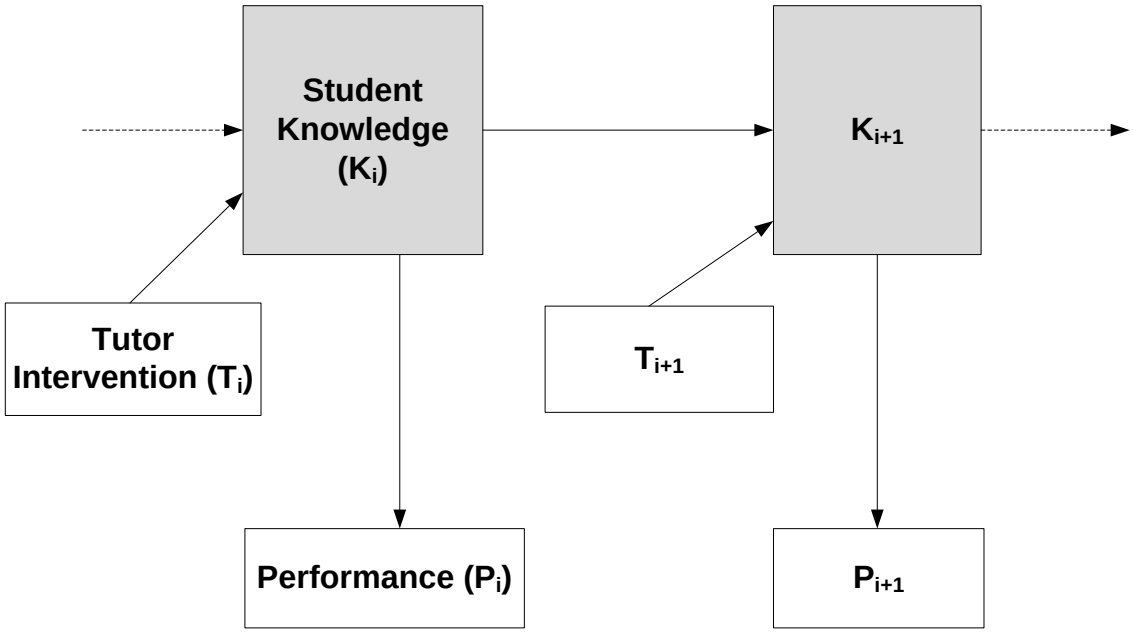

Figure 1. Knowledge tracing model extended with "Tutor Intervention" node

From the graph we can see that the model is a DBN with 3 nodes at each time step i. The binary valued hidden node K_i represents whether the student knows the skill at time i. $\Pr(K_i)$ is a probabilistic estimate of this binary student knowledge variable. P_i represents the student's observed performance (e.g. correct or incorrect) on that skill at time i. Using a DBN for knowledge tracing lets us generalize its original formulation [10] by including additional nodes. For instance, Chang et al. [11] added a node to represent help requests. To evaluate interventions, we include tutor intervention T_i as a discrete variable to represent which tutorial intervention is made at time i. Links in the model encode conditional dependencies between variables.

A model is described by the following parameters:

- *knew*: Probability that the student already knew the skill prior to any instruction
- *prob_Ti*: Probability that the intervention is of type *i*.
- *learn_Ti*: Probability of acquiring the skill from an intervention of type T_i, i.e. P(K_{i+1}=true | K_i= false, T_i)
- *forget_Ti*: Probability of losing a known skill conditioned on an intervention of type T_i, i.e. P(K_{i+1}=false | K_i= true, T_i)
- *guess*: Probability of answering correctly without knowing the skill, i.e. P(P_i=true| K_i= false)
- *slip*: Probability of answering incorrectly despite knowing the skill, i.e., P(P_i=false| K_i= true).

The problem of comparing the effects of different types of tutorial behaviors then reduces to comparing the values of the *learn_Ti* and *forget_Ti* parameters for the different T_i's corresponding to those types of behaviors.

An implementation of this model includes an off-line training procedure to learn the model parameters from a corpus of performance data, and a runtime update procedure to revise the estimate of the student's knowledge based on observed performance at step *n*. The BNT-SM implementation of knowledge tracing [12] uses BNT's EM procedure to fit the model. The update procedure can be implemented by a general Bayes net inference procedure, or by a procedure to compute specific formulas derived from the model.

1.4 Dimensions of comparison

Basic questions about RCTs, learning decomposition, and knowledge tracing include:

- What outputs are these methods designed to produce?
- What inputs do they require?
- What models are they based on?
- What are their underlying assumptions?

Sections 2-5, respectively, compare the three methods with respect to these questions.

2 Intended output

The three methods differ in the purpose they are designed to achieve, that is, what outputs they compute.

RCT analysis compares trial outcomes to identify factors that predict or affect outcomes, primarily treatment condition. Thus it is a natural way to compare interventions. It can

estimate student ability that affects outcomes by including student identity as a factor in a statistical test, but such assessment does not exploit the randomized treatment assignment. Thus RCT analysis evaluates interventions more than students. Moreover, this evaluation analyzes an entire set of trials simultaneously, so even if it estimates student ability as a factor, it is not designed to update this estimate incrementally based on sequential observations of student performance.

Learning decomposition can also fit student parameters such as initial performance and learning rate, but its primary purpose is to compare the relative impact of different types of practice. Its performance prediction for how a student will do at the N^{th} encounter of a skill depends only on how much prior practice of each type he or she got on that skill – not on how well he or she performed on it previously – unless encounters are treated as different types based on their immediate outcomes.

Learning decomposition could incorporate information about prior performance by treating successful and unsuccessful encounters as two different types of practice. This possibility might in fact be interesting to explore. However, it does not escape the fact that simply counting the number of encounters of different types ignores order and recency effects. Thus the resulting model would predict the same performance after 5 correct responses followed by 5 incorrect responses as it would predict after 5 incorrect responses followed by 5 correct responses. However, learning decomposition can model some recency effects by treating spaced and massed practice as different types [4].

Knowledge tracing is specifically designed to update, at each practice opportunity, a tutor's estimate of an individual student's mastery of the skill(s) it exercises. Conditioning this update on practice type makes it possible to evaluate how each type of practice affects learning. Based on the latest knowledge estimate (plus the slip and guess parameters), the tutor can predict how well the student is about to perform, and plan accordingly – for example by picking an easier problem to prevent a likely imminent failure.

Knowledge tracing makes a more informed prediction of performance than learning decomposition does, because the prediction reflects performance during prior practice, not just the amount of practice. For example, suppose the data set includes sequences of N-1 practice opportunities by two different students on the same skill, with the same sequence of practice types, but that one sequence consists of successes and the other sequence consists of failures. Knowledge tracing will predict higher performance after the successes than after the failures, because its prediction is conditioned on observed performance. In contrast, learning decomposition – unless it includes any student-specific parameters or distinguishes past encounters by performance -- will predict the same performance in both cases, because its prediction is based solely on the number and type of practice opportunities.

Knowledge tracing updates its estimate of student knowledge based on sequential observations of student performance, and is sensitive to their order – but not their timing. Learning decomposition can address this limitation to some extent by treating massed and spaced trials as different types of practice. Analogously, knowledge tracing can

condition the parameters *learn* and *forget* based on the time elapsed between successive encounters of a skill. For both methods, this *ad hoc* refinement models temporal effects as discrete, for example by classifying a student's successive encounters of a skill as occurring on the same or different days [4]. A more complex method [8] models continuous temporal effects based on a deeper theory of memory.

3 Information input

The three methods differ in what data they input, that is, which observations they consider.

RCT analysis compares only observed outcomes, so it simply ignores trials whose outcomes were unobserved, undefined, or masked. For example, a student's performance in reading a word is unobserved when the Reading Tutor reads it, undefined when the Reading Tutor gives unrequested help on the word before accepting it, and masked by recency effects if the student has seen the word earlier the same day.

Learning decomposition analyzes trials both as outcomes and as practice. It excludes trials as outcomes where performance is unobserved, undefined, or masked, but in each case the encounter counts as practice.

Knowledge tracing represents outcomes and practice as the same performance nodes in a dynamic Bayes net, and tolerates missing and partial observations. For example, credit for a word is undefined if masked by tutor-initiated help, but the word encounter still updates the estimated probability that the student has learned the word.

4 Model form: representation, computation, and extension

The three methods differ in how they are represented, applied, and extended.

RCT analysis has no hidden variables to estimate, no initialization conditions to be sensitive to, and no local minima to get stuck on. An appropriate statistical test to compare outcomes between different subsets of trials is practically instantaneous in a standard statistical package such as SPSS, even with many thousands of trials. Extending the analysis involves adding outcome variables to test, and features to disaggregate by or include as factors in statistical tests such as ANOVA.

Learning decomposition estimates parameters that represent the impact of each mode of practice in an exponential model of performance over time. Non-linear regression in SPSS 11.0 took only about one minute to fit hundreds of models (one for each word) to thousands of word encounters in our empirical comparison of evaluation methods [1]. Extending models means adding parameters to further distinguish types of practice or other influence on performance, e.g. effect of length on word reading.

Knowledge tracing, generalized to distinguish different practice types, estimates parameters that represent the impact of each type of practice on a hidden knowledge state in a dynamic Bayes net, and the effect of this knowledge on observed student

performance. BNT-SM [12] takes about 10 hours to train and evaluate models on a data set that learning decomposition takes only a minute to fit. Extending the model means adding nodes to represent observations, and links to represent hypothesized causal influences.

5 Underlying assumptions

RCT analysis treats trials as separate and does not attempt to model the causal effects of one trial on the outcome of another, such as diminishing effects of successive trials on the same skill, or transfer effects from practice on one skill to performance on another [13]. At most it accounts for statistical dependencies among related trials by using tests with the appropriate number of degrees of freedom. For example, instead of treating all trials on a story word as independent, we can average outcomes for each type of practice over all the students who received that type of practice on the word.

In contrast, both learning decomposition and knowledge tracing assume particular models of how performance improves with practice. On the surface, they look very different. Knowledge tracing can be expressed as a dynamic Bayes net that incrementally updates estimates of a hidden skill, while learning decomposition is expressed as a closed-form exponential function that predicts performance directly.

Although their surface form differs markedly, the mathematics of these two methods is closely related at a deeper level in a way that is useful to elucidate because it not only reveals their underlying similarities, but shows how to bridge some of their differences. In particular, starting from a Bayesian model of knowledge tracing we can derive learning and performance curves that we can relate to learning decomposition.

5.1 Derivation of learning decomposition from knowledge tracing

Knowledge tracing estimates the probability $\Pr(K_n)$ that the student knows a given skill at time step n, according to a dynamic Bayes net model. For brevity we will abbreviate $\Pr(K_n)$ as K_n. The parameters of this model include the *knew* probability K_0 that the student already knew the skill prior to instruction, the *learn* probability L of acquiring the skill from a step, the *forget* probability of losing a skill, the *guess* probability g of answering correctly without knowing the skill, and the *slip* probability s of answering incorrectly despite knowing the skill.

We are interested in how K_n changes over time according to this model. The probability of knowing the skill at step n given that the student did not know it at step $n-1$ is L. If we assume zero probability of forgetting, the probability $(1 - K_n)$ of *not* knowing the skill at step n is the probability of not knowing the skill at the previous step, times the probability of not learning it, or $(1 - K_{n-1}) \times (1 - L)$. Consequently the ratio $(1 - K_n) / (1 - K_{n-1})$ is the constant $(1 - L)$, and the probability $(1 - K_n)$ can be expressed in closed form as $(1 - K_0) \times (1 - L)^n$. That is, ignorance at step n requires not knowing the skill in advance, followed by not learning it.

This formula expresses the *prior* probability of (not) knowing the skill at time *n*; it is not conditioned on any observed student performance. Observed performance affects our estimate of student skills, but this effect is *evidentiary*, not *causal*: according to the model, student performance does not influence student knowledge. In short, *even without any observation of student performance*, we can still use the model to predict student knowledge.

We are also interested in the performance predicted by the model, specifically in the error rate expected at step *n*. The probability W_n of a wrong response at step *n* can be split into two cases – either not knowing and not guessing, or knowing but slipping. Consequently $W_n = (1 - K_n) \times (1- g) + K_n \times s$. Plugging in our closed form expression for $(1 - K_n)$ gives

$$W_n = (1 - K_0) \times (1 - L)^n \times (1- g) + (1 - (1 - K_0) \times (1 - L)^n) \times s$$

$$= [(1 - K_0) \times (1- g) - (1 - K_0) \times s] \times (1 - L)^n + s$$

$$= s + (1 - g - s) \times (1 - K_0) \times (1 - L)^n.$$

What if the learning probability varies with the type of step? We can generalize $(1 - L)^n$ to the product $(1 - L_1) \times \ldots \times (1 - L_n)$. If there are *m* types of steps, with t_i steps of type *i*, each with learning probability L_i, we rewrite this product as $(1-L_1)^{t_1} \times \ldots \times (1-L_m)^{t_m}$, and generalize the predicted error rate to:

$$s + (1 - g - s) \times (1 - K_0) \times (1 - L_1)^{t_1} \times \ldots \times (1 - L_m)^{t_m}$$

If $s = g = 0$, this formula reduces to $(1 - K_0) \times (1 - L_1)^{t_1} \times \ldots \times (1 - L_m)^{t_m}$, which relates to the learning decomposition formula $A \times e^{-b \times (\beta_1 t_1 + \ldots + \beta_m t_m)}$ as follows. The factors $(1 - L_1) \ldots (1 - L_m)$ correspond to the coefficients $\beta_1, \ldots, \beta_m$ that represent the relative value of different types of practice. More precisely, since $(1 - L_i)^{t_i} = \exp(\log(1-L_i) \times t_i)$, the expression $\log(1 - L_i)$ corresponds to β_1. The factor $(1 - K_0)$ corresponds to $A \times e^{-b}$.

This derivation reveals that we can extend learning decomposition to include slip rate *s* and guess rate *g* in the generalized formula $s + (1 - g - s) \times A \times e^{-b \times (\beta_1 t_1 + \ldots + \beta_m t_m)}$. The additive term *s* represents the asymptotic performance beyond which the error rate will not decrease even with unlimited practice.

However, it is important to point out that the generalized formula, at least in the form above, applies to the rate of errors or help requests, but not to response time. The reason is that the parameter *s* appears both as an additive term and in the factor $(1 - g - s)$. If the formula represents a time quantity, *s* must be expressed in some unit of time. But the *s* in $(1 - g - s)$ must be unit-free to be comparable with the constant *1* and the probability *g*. Since *s* cannot be both a temporal quantity and unit-free, the overall formula cannot represent a time quantity, and in fact must itself be unit-free.

Extending this generalized formula to predict time or other continuous measures of performance requires modeling how they are affected by slips and guesses. For example, if guessing takes negligible time, the guess parameter *g* already models the effect of

guesses on performance time: if $s = 0$, then $g = 0.5$ cuts predicted performance time in half. But if guesses take non-negligible time, modeling them requires extending the formula by interpolating it with guessing time. Likewise, modeling the effect of slips on time requires replacing the additive term s with slip time, e.g. the time to misread a known word. The extended form of the generalized formula therefore looks like this, where the expressions *slip time* and *guess time* depend on how they are modeled:

$$s \times (\text{slip time}) + g \times (\text{guess time}) + (1 - g - s) \times A \times e^{-b \times (\beta_1 t_1 + \ldots + \beta_m t_m)}$$

6 Contributions

This paper provides a descriptive and analytical comparison of three methods to evaluate tutorial behaviors: RCT analysis, learning decomposition, and knowledge tracing using DBNs. We compare their inputs, outputs, models, and assumptions. In particular, we elucidate the underlying mathematical relationship between knowledge tracing and learning decomposition, thereby showing how to generalize learning decomposition to incorporate slip and guess parameters. We hope that other researchers will find this paper useful in making informed choices of which method(s) to use to model and evaluate learning in tutors.

Acknowledgements

The research reported here was supported by the National Science Foundation under ITR/IERI Grant No. REC-0326153, by the Institute of Education Sciences, U.S. Department of Education, through Grant R305B070458 to Carnegie Mellon University, and by the Heinz Endowments. The opinions expressed are those of the authors and do not necessarily represent the views of the National Science Foundation, the Institute, the U.S. Department of Education, or the Heinz Endowments.

References (LISTEN publications are at www.cs.cmu.edu/~listen)

[1] Zhang, X., J. Mostow, and J.E. Beck. A Case Study Empirical Comparison of Three Methods to Evaluate Tutorial Behaviors. *9th International Conference on Intelligent Tutoring Systems*, 122-131. 2008. Montreal: Springer-Verlag.

[2] Beck, J.E., K.-m. Chang, J. Mostow, and A. Corbett. Does help help? Introducing the Bayesian Evaluation and Assessment methodology. *9th International Conference on Intelligent Tutoring Systems* 2008. Montreal.

[3] Mostow, J. and G. Aist. Evaluating tutors that listen: An overview of Project LISTEN. In K. Forbus and P. Feltovich, Editors, *Smart Machines in Education*, 169-234. MIT/AAAI Press: Menlo Park, CA, 2001.

[4] Beck, J.E. Using learning decomposition to analyze student fluency development. *ITS2006 Educational Data Mining Workshop*, 21-28. 2006. Jhongli, Taiwan.

[5] Beck, J.E. Does learner control affect learning? *Proceedings of the 13th International Conference on Artificial Intelligence in Education*, 135-142. 2007. Los Angeles, CA: IOS Press.

[6] Beck, J.E. and J. Mostow. How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of students. *9th International Conference on Intelligent Tutoring Systems* 2008. Montreal.

[7] Heathcote, A., S. Brown, and D.J.K. Mewhort. The Power Law Repealed:The Case for an Exponential Law of Practice. *Psychonomics Bulletin Review*, 2000: p. 185-207.

[8] Pavlik Jr., P.I. and J.R. Anderson. Practice and Forgetting Effects on Vocabulary Memory: An Activation-Based Model of the Spacing Effect. *Cognitive Science*, 2005. *29*(4): p. 559-586.

[9] Atkinson, R.C. Ingredients for a theory of instruction. *American Psychologist*, 1972. *27*(10): p. 921-931.

[10] Corbett, A. and J. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 1995. *4*: p. 253-278.

[11] Chang, K.-m., J.E. Beck, J. Mostow, and A. Corbett. Does Help Help? A Bayes Net Approach to Modeling Tutor Interventions. *AAAI2006 Workshop on Educational Data Mining* 2006. Boston, MA.

[12] Chang, K.-m., J. Beck, J. Mostow, and A. Corbett. A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 104-113. 2006. Jhongli, Taiwan.

[13] Zhang, X., J. Mostow, and J.E. Beck. All in the (word) family: Using learning decomposition to estimate transfer between skills in a Reading Tutor that listens. *AIED2007 Educational Data Mining Workshop* 2007. Marina del Rey, CA.