Human-Machine Student Model Discovery and Improvement Using DataShop

John C. Stamper and Kenneth R. Koedinger

Human-Computer Interaction Institute, Carnegie Mellon University

Abstract. We show how data visualization and modeling tools can be used with human input to improve student models. We present strategies for discovering potential flaws in existing student models and use them to identify improvements in a Geometry model. A key discovery was that the student model should distinguish problem steps requiring problem decomposition planning and execution from problem steps requiring just execution of problem decomposition plans. This change to the student model better fits student data not only in the original data set, but also in two other data sets from different sets of students. We also show how such student model changes can be used to modify a tutoring system, not only in terms of the usual student model effects on the tutor's problem selection, but also in driving the creation of new problems and hint messages.

Keywords: data mining, machine learning, cognitive modeling.

1 Introduction

Student models drive many of the instructional decisions that automated tutoring systems make, whether it is what instructional messages to provide and when, how to sequence topics and problems in a curriculum, how to adapt pacing to student needs, and even what problems and instructional materials are needed. Better student models yield better instruction. And student models can be improved by mining student interaction data. A better student model is one that better matches student behavior patterns. A better student model, in this empirical sense, is one that better predicts task difficulty and transfer of learning between related problems (during and after tutoring). We present a method for guiding the application of data mining algorithms for discovery of better student models. We show how data analysis tools such as those in the PSLC DataShop [6] can be used to identify areas for improvement, and then discuss how to quantitatively evaluate the new models.

Student models have traditionally been developed by domain experts engaging in manual analysis of course content. Cognitive Task Analysis (CTA) is an approach to understanding domain learning that has resulted in the design of significantly better instruction [2][9]. But CTA methods, like structured interviews, think aloud protocols, and rational analysis, have limitations. They are highly subjective and different analysts may produce different results. They also demand substantial human effort at each stage in data collection, analysis, and modeling. Automated techniques applied to large sets of student data can provide both more objectivity and reduce human effort.

G. Biswas et al. (Eds.): AIED 2011, LNAI 6738, pp. 353–360, 2011.

Learning Factors Analysis (LFA) is an automated search technique for discovering student models [1]. Compared to some prior student model discovery approaches [10][12], LFA can apply to learning data not just performance data, and it is arguably more interpretable because of the use of human-provided labels (the "P matrix" in LFA). A key limitation, however, is that models can only be discovered within the space of the human-provided factors. If a better model exists but requires a factor that is not in the given set of input factors into LFA, it will not discover that model. How can such factors be discovered? We address this key question below. Applying automated methods is becoming more practical as the amount of student data continues to grow. Cognitive Tutors for mathematics are now in use for more than 500,000 students per year in the USA. While these systems have been quite successful, our analysis of log data suggests that the models behind them can be improved. Repositories to store large datasets from diverse educational domains can provide a central point to make these improvements. DataShop is such a repository that also includes a set of associated visualization and analysis tools (http://learlab.org/datashop). Student actions are coded as correct or incorrect and categorized in terms of the hypothesized competencies or "knowledge components" (KCs) needed to perform that action. A KC is a generalization of any element of a knowledge representation including a production rule, schema, or constraint. Each step the student performs that is related to a KC is recorded as an "opportunity" for the student to show mastery of that KC. (A step is a set of correct, incorrect, or help request actions related to the same problem subgoal [11].) Visualizations and analysis tools in DataShop are designed to help model builders find potential flaws in an existing student model and discover new KCs that yield a better fit to student data.

2 Human-Machine Student Model Discovery

Our Human-Machine Student Model Discovery method uses human input to identify model improvements from visualizations created from student log data and then evaluated by a statistical fit with the data. We use the Additive Factor Model (AFM), a statistical algorithm for modeling learning and performance that uses logistical regression performed over the "error rate" learning curve data [1]. AFM is a specific instance of logistic regression, with student-success (0 or 1) as the dependent variable and with independent variable terms for each student, each KC, and the KC by opportunity interaction. It is a generalization of the log-linear test model [13] produced by adding the KC by opportunity terms. Model discovery with AFM finds a set of KCs that best fits student data (without over-fitting). We chose AFM over other more complex alternatives (e.g., models with more terms in the logistic regression) because its simplicity enhances interpretation of the model parameters.

We describe three strategies for discovering opportunities for student model improvement that are supported by the visualization and analysis tools in DataShop:

1) *Smooth learning curves* - We expect that the learning curve for each KC will be reasonably smooth. When the learning curve of a purported KC is noisy, with upward or downward "blips", the student model is suspect.
2) *No apparent learning* - If the student model is accurate, we expect the error rate to decline over the number of opportunities a student has to learn and apply a KC. A flat learning curve is another indication of a potentially flawed student model.

3) *Problem steps with unexpected error rates* – A KC is suspect if the problem steps it labels have an error rate that is much higher or lower than the expected. In the ideal student model, the expected error rate for all steps labeled by the same KC should be about the same (albeit, the error rate should decline as students have more opportunities to practice). More precisely, the expected error rate is computed from the AFM statistical model that is built into the DataShop and the Performance Profiler tool provides a way to visualize whether any steps have error rates that are discrepant with this expectation.

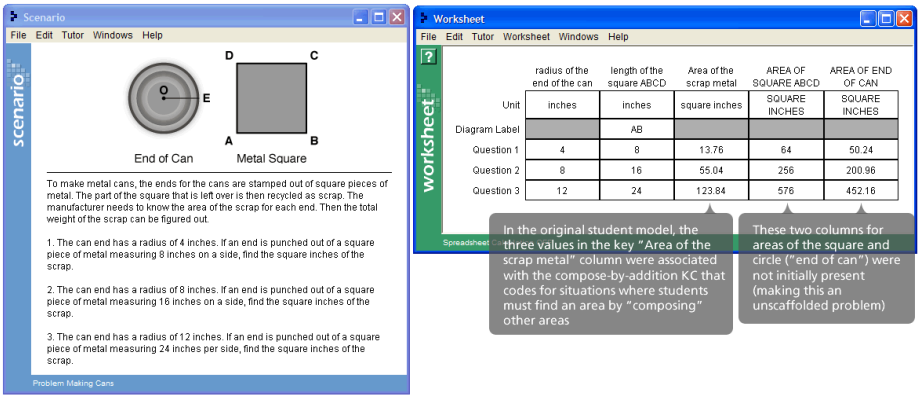

Fig. 1. A problem from the Geometry Cognitive Tutor. All cells values are filled by the student.

We focused on a publicly available data set from DataShop called "Geometry Area (1996-97)." This data was generated from student interactions with a cognitive tutor for learning Geometry, and a screen shot of from a newer version of the tutor can be seen in Figure 1. The data included 5,104 student steps completed by 59 students.

Smooth learning curves. The first discovery strategy was applied to this dataset by inspecting the learning curves of knowledge components (KCs) from the existing best student model, which was called Textbook-New. This model has 10 KCs. A subset of the learning curves for these KCs is shown in Figure 2. The lines represent the error rate (y-axis) averaged over all students for the first 20 practice opportunities for each KC (e.g., on the fifth opportunity on the trapezoid-area KC about 55% of students made an error). Defining exactly what constitutes a smooth learning curve is still an open research question, but most of the KCs have reasonably smooth learning curves, like compose-by-multiplication, parallelogram-area, and trapezoid-area. (Roughness in the learning curve can result from noise rather than a bad KC and particularly so when there a fewer observations being averaged is common at higher opportunity numbers.) The compose-by-addition curve is particularly jagged with upward blips in error rate. At opportunities 12 and 15-18, the curve jumps above from about 25% to about 50%. Assuming there are particular problem steps that are more likely to occur at these opportunities (which is the case in this data set), those steps appear to have some knowledge demand that the other steps do not. The compose-by-addition KC is involved in "composition problems", that is, problems where the area

of an irregular shape (e.g., what's left when a circle is cut from a square) must be found by combining (adding or subtracting) the areas of the regular shapes that make it up (e.g., a square and circle). Identifying what makes these steps harder, may improve the student model.

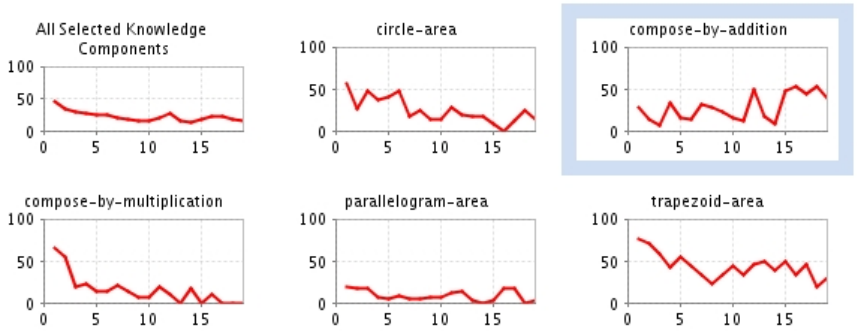

Fig. 2. Example learning curves from Textbook-New Student Model showing first 20 attempts for all students. Y-axis is the error rate and the X-axis is learning opportunities. Most of these curves are reasonably smooth and decreasing. "Compose-by-addition" is not smooth, with large jumps in the error rate at opportunities 12 and 15.

No apparent learning. The second discovery strategy is to identify KCs that do not indicate any student learning and are initially non-trivial. The parameter estimates for the KC terms in the AFM regression equation (introduced above) indicate the "intercept" or starting point of the learning curve and those for the KC by Opportunity interaction terms indicate the "slope" or rate of learning. Because AFM predicts success as the dependent variable (rather than error rate as shown in Fig 1), high values indicate a well-learned KC. To find KCs indicating little learning, DataShop's Learning Curve > AFM Values menu item provides parameter estimates. Looking at the slope column, we see little or no learning for compose-by-addition (0) and parallelogram-area (0.019). Parallelogram-area is of less concern because of the high intercept value, 89% correct (2.13 in log odds) at the first opportunity, indicating students mostly have this skill. That the compose-by-addition intercept is not high (74%) and the slope is absent indicates either that students are not learning or that the compose-by-addition is a poorly defined KC. We pursue the latter.

Problem steps with unexpected error rates. The third discovery strategy utilizes the Performance Profiler tool within DataShop (see Figure 3). Using this tool we confirmed that the error rates for problem steps coded by compose-by-addition are not well fit. As shown in Figure 3, there is a large discrepancy between easy steps at the top of the figure, with an error rate (indicated by the shaded bar) of about 5-10%, and hard steps at the bottom of the figure, with an error rate of about 40-60%. We also see that the predicted error rate from the Textbook-New model (the straighter line of connected points at about 30%) slices through the middle of these extremes and only rarely does an accurate job of predicting the error rates of any of these steps.

Not only does this strategy further implicate the compose-by-addition KC as a candidate for improvement, it also provides guidance for inspecting variations in the content of the problems to potentially identify new knowledge components. In particular, it suggests that we try to identify why some problem steps are harder than others and hypothesize what factor or knowledge-demand may make them harder? What we noticed is that some of the composition problems were "scaffolded" such that they included columns that cued students to find the component areas (square and circle) first [4]. Other problems were "unscaffolded" and did not start with such columns, thus students had to pose these subgoals themselves. Indeed the blips for compose-by-addition (seen in the learning curve in Figure 2) do correspond with a high frequency of these unscaffolded problems.

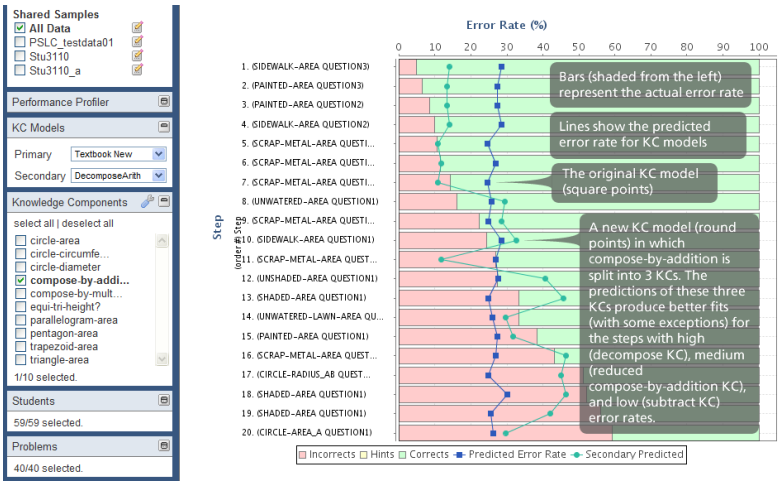

Fig. 3. DataShop's Performance Profiler shows the error rate on the steps for compose-by-addition KC

Based on the analysis, compose-by-addition was not at a fine enough level to accurately explain the student data, suggesting additional KCs may be present. To improve the model, compose-by-addition was split into 3 KCs, one representing the current compose-by-addition with scaffolding present, another where the student had to *decompose* the an irregular area without scaffolding, and a third where the student needs to *subtract* to execute the decomposition plan. For example, while all three cells in column 3 of Figure 1 (with values 13.76, 55.04, and 123.94) were originally coded as compose-by-addition, in the new model the first (where 13.76 was entered) is coded as *decompose* and the next two (55.04 and 123.94) are coded as *subtract*. After these KCs were hypothesized, the Textbook-New student model was exported from DataShop and modified in Excel. Of the 20 steps that were previously labeled with the compose-by-addition KC, 6 were labeled with the new decompose KC and 8 were labeled with the subtract KC. The model was imported back as "DecomposeArith".

For model evaluation we use Bayesian information criterion (BIC) and root mean-squared error (RMSE) from a 3 fold cross-validation where the folds are computed

with the constraint that each of the 3 training sets must have data points for each student and KC (See [1] for justification of use of the BIC metric). The fit metrics for this new model are lower (BIC 5,628 and cross validation RMSE of 0.4021), thus better, than those for the former model (5,677 and 0.4064) indicating that DecomposeArith student model improves on the previous model without over-fitting. The cross-validation results are from a 3 fold cross-validation

What did we learn toward an improved student model? Most importantly, we found that it is possible to distinguish, in geometry tutor data, the difference between the process of "planning" a problem decomposition (in an unscaffolded problem) and the "execution" of one (in a scaffolded problem).The planning process requires figuring out how the area of an irregular shape (e.g., Figure 1) can be found from the areas of regular shapes that make it up (e.g., the square and circle). In the context of this tutor, the execution involves seeing that available regular area values (present in provided columns) can be used to find the irregular area and performing the required arithmetic. The planning involved in Geometry decomposition may be reflective of more general student competence at problem decomposition and such problems are quite common in standardized mathematics tests. The intercept and slope estimates for the decompose KC in the new model indicate both that it is difficult (the initial success rate is only 40%) for students and that the tutor is helping. The slope parameter is 0.15 logits per opportunity, which is one of the higher learning rates in this unit. Most students, however, finished the tutor far from mastery of decomposition. Because the original student model confounded decomposition planning and execution, it over-estimated student progress on decomposition - in essence giving students credit for decompose when they correctly performed simpler scaffolded composition and subtraction.

To confirm the model discovered above, we performed a parallel analysis on a second Geometry Area data set also available in DataShop called "Geometry Area Hampton 2005-2006 Unit 34." The original Textbook student model associated with this data set has 13 KCs and the metric values: BIC 15,375.1 and cross validation RMSE of .4078. Based on the previous findings, the Textbook KC model was modified with the steps for compose-by-addition split into 3 KCs as suggested above. When the additional KCs were added, the new model (DecomposeArith) with 15 KCs showed improvement (BIC 15,176.7 and cross validation RMSE of .4042) further validating the existence of the new KCs. By demonstrating the success of the model changes on a new data set, not used to discover the model, we greatly reduce the chance that this model is not idiosyncratic or over-fit to the first data set.

3 Using a Discovered Student Model to Redesign a Tutor

Once an improved student model has been discovered, different kinds of changes in a student model can suggest redesign moves in the tutor. Potential changes include:

1) Resequencing – put problems requiring fewer KCs before ones needing more
2) Knowledge tracing – add/delete skill bars for better cognitive mastery
3) Creating new tasks – add problems to focus practice on new KCs
4) Changing instructional messages, feedback or hint messages

We applied the discovered student model to the Geometry area unit of a pre algebra course. The critical difference between the discovered and existing models is the new KCs for planning problem decomposition. As implied above, problem selection and knowledge tracing of a tutoring system is affected by changes in the model. We added 3 new skills to the tutor that make the distinction between unscaffolded decomposition, scaffolded, and simple subtraction. Students in this new version will not be able to get credit for the difficult decomposition step through success on simpler scaffolded or subtraction steps. New problems to target these newly identified skills were added. We identified 3 types of problem dimensions that would help isolate the new skills in area compositions problems. These are table scaffolded, area scaffolded, and problem statement scaffolded. Table scaffolded problems reflect the current setup in the tutor and include columns for intermediate areas. Area scaffolded problems go a step further and give the areas of the component shapes. Problem statement scaffolded problems provide less support in that the component area columns are not present, but there is an explicit hint in the problem statement directing the student to first find the areas of the individual shapes. Using these problem dimensions, four new problem types were created. In the new curriculum, there are more unscaffolded problems (and earlier in the curriculum), but and also problems that isolate just the decomposition step by giving students the component areas instead of requiring them to compute those areas. In general, changes in skills can lead to changes in the feedback and hint messages the tutor provides. The new problems give students focused instruction on these skills.

We performed a pilot *in vivo* experiment with the new student model and redesigned tutor in the Spring of 2010 with 5 classes working on the Carnegie Learning "Bridge to Algebra" cognitive tutor. 120 students were split into two conditions. The experimental condition received the new instruction driven by the new model while the control was given the original instruction with the old model.

Using the data collected, both the new student model with new skills and the old model without the new skills were fitted. The resulting model with 49 KCs had the following metrics: BIC 31,183.9 and cross validation RMSE of .3255. These were lower than the previous metric values of the model with 46 KCs (BIC 31,258.9 and cross validation RMSE of .3269) showing the model with the additional KCs labeled is better than the model without the new skills. The ultimate goal of the redesign was to show a better model leads to better learning. The experimental condition had better posttest means (M=.72, SD=.22) than the control (M=.64, SD=.20), but an ANCOVA analysis indicates only a marginal effect, $F(1,78) = 3.47$, $p = .07$, when accounting for pretest scores. Given the results are in the predicted direction, we would be justified to use a one-tailed test and reject the null hypothesis ($p<.05$). However, we intend future studies of this kind to get more solid evidence.

4 Conclusion and Future Work

We presented a human-machine discovery approach for improving student models by using DataShop tools. We demonstrated how this approach can produce non-trivial improvements in a student model, even in a domain (Geometry) where there has been considerable attention and prior cognitive analysis. We demonstrated how this new student model better fits student data, not only in the original data-set used to discover it, but also in two other data sets. We used the model to modify tutor instruction,

particularly to create new problems that help isolate the difficult skill of planning problem decompositions. While not confirming the instructional benefits of this new student model, our initial experiment showed promise. At least one study has demonstrated that data-driven student model improvements can yield better instruction [7], but this work is novel in showing how aspects of student model improvement can be increasingly systematized and automated.

The approach we describe still has a significant human component. Unsupervised methods for model discovery [12][3] have the potential to produce better fitting models with less effort. However, it is not clear how to interpret the results of these models and apply them in improving tutor design. Further investigation is needed. More generally, using data to optimize student models and, in turn, improve instructional systems is a tremendous opportunity. The achievement are likely to be greater to the extent that the discovered models involve deep or integrative KCs not directly apparent in surface task structure, like the problem decomposition skill we identified in Geometry. This work was supported by the Pittsburgh Science of Learning Center (NSF award 0836012).

References

1. Cen, H., Koedinger, K.R., Junker, B.: Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 164–175. Springer, Heidelberg (2006)
2. Clark, R.E., Feldon, D., van Merriënboer, J., Yates, K., Early, S.: Cognitive task analysis. In: Spector, J., Merrill, M., van Merriënboer, J., Driscoll, M. (eds.) Handbook of Research on Educational Communications and Technology, Mahwah, NJ, pp. 577–593 (2007)
3. Collins, M., Dasgupta, S., Schapire, R.: A generalization of PCA to the exponential family. In: Procs. of the 14th Conf. on Neural Info. Processing Systems, NIPS (2001)
4. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of Procedural knowledge. User Modeling and User-Adapted Interaction, 253–278 (1995)
5. Heffernan, N., Koedinger, K.: A developmental model for algebra symbolization: The results of a difficulty factors assessment. In: Gernsbacher, M.A., Derry, S.J. (eds.) Procs. of the 20th Annual Conf. of the Cognitive Science Society, Mahwah,NJ, pp. 484–489 (1998)
6. Koedinger, K.R., Baker, R.S.J.d., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J.: A Data Repository for the EDM commuity: The PSLC DataShop. In: Romero, V., Pechenizkiy, B. (eds.) Handbook of Educational Data Mining. CRC Press, Boca Raton (2010)
7. Koedinger, K., McLaughlin, E.: Seeing language learning inside the math: Cognitive analysis yields transfer. In: Procs. of the 32nd Ann. Conf. of the Cogitive Science Society (2010)
8. Koedinger, K.R., Nathan, M.J.: The real story behind story problems: Effects of representations on quantitative reasoning. The Jrnl of the Learning Sciences 13(2), 129–164 (2004)
9. Lee, R.L.: Cognitive task analysis: A meta-analysis of comparative studies. Unpublished doctoral dissertation, University of Southern California, Los Angeles, CA (2003)
10. Tatsuoka, K.K.: Rule space: An approach for dealing with misconceptions based on item response theory. Journal of Educational Measurement 20, 345–354 (1983)
11. VanLehn, K.: The behavior of tutoring systems. Intl Jrnl of AIED 16, 227–265 (2006)
12. Villano, M.: Probabilistic student models: Bayesian Belief Networks and Knowledge Space Theory. In: Procs. of the 2nd International Conference on ITS, pp. 491–498. Springer, Heidelberg (1992)
13. Wilson, M., de Boeck, P.: Descriptive and explanatory item response models. In: de Boeck, P., Wilson, M. (eds.) Explanatory Item Response Models, pp. 43–74. Springer, Heidelberg (2004)