The Sum is Greater than the Parts: Ensembling Models of Student Knowledge in Educational Software

Zachary A. Pardos
Department of Computer Science
Worcester Polytechnic Institute
Worcester, MA, USA
zpardos@wpi.edu

Ryan S.J.d. Baker
Department of Social Science and Policy Studies
Worcester Polytechnic Institute
Worcester, MA, USA
rsbaker@wpi.edu

Sujith M. Gowda
Department of Social Science and Policy Studies
Worcester Polytechnic Institute
Worcester, MA, USA
sujithmg@wpi.edu

Neil T. Heffernan
Department of Computer Science
Worcester Polytechnic Institute
Worcester, MA, USA
nth@wpi.edu

ABSTRACT

Many competing models have been proposed in the past decade for predicting student knowledge within educational software. Recent research attempted to combine these models in an effort to improve performance but have yielded inconsistent results. While work in the 2010 KDD Cup data set showed the benefits of ensemble methods, work in the Genetics Tutor failed to show similar benefits. We hypothesize that the key factor has been data set size. We explore the potential for improving student performance prediction with ensemble methods in a data set drawn from a different tutoring system, the ASSISTments Platform, which contains 15 times the number of responses of the Genetics Tutor data set. We evaluated the predictive performance of eight student models and eight methods of ensembling predictions. Within this data set, ensemble approaches were more effective than any single method with the best ensemble approach producing predictions of student performance 10% better than the best individual student knowledge model.

1. INTRODUCTION

In recent years, adaptive educational software has emerged out of the lab, and out of research classrooms, and into widespread usage, particularly in the United States [18]. This software typically adapts to individual differences in student knowledge, engagement, and motivation, attempting to select the curricular materials and methods of presentation best suited for the specific learner. This adaptation is in turn dependent on accurate assessment of the student – what the student knows, and other aspects and attributes of the student, an area termed *student modeling* [36].

Student modeling poses several challenges. The first is that student knowledge is inherently latent – in other words, the goal is to assess a quantity that is not directly measured. Instead, knowledge must be assessed from performance, which has a noisy relationship to knowledge: students often guess and get correct answers without knowledge, and students also often make simple errors ("slips") on material they know. However, performance can be used to validate models of knowledge – a successful knowledge model should be more successful at predicting future correctness than an unsuccessful knowledge model.

Over the last decades, there has been continual competition between different approaches towards student modeling, with particularly intense competition in terms of modeling student knowledge. One challenge is that within educational software, the system is attempting to teach the student, and therefore assessment cannot assume that student knowledge is static during the use of the system. As such, the decades of work on assessment within testing, using methods such as item-response theory [15] and computerized adaptive testing [35], are not immediately applicable and must be modified. Research on how to most effectively assess dynamic student knowledge in educational software began in the 1970s [13], with approaches based on Bayes Nets and Bayesian algorithms emerging in the 1990s [9; 21]. In recent years, models based on item-response theory, such as Performance Factors Analysis (PFA) [29], have also emerged and gained popularity.

Multiple variants within each of these paradigms have also been created – for instance, within the Bayesian Knowledge Tracing (BKT) framework proposed by Corbett and Anderson [9], models can be fit using curve-fitting [9], expectation maximization (EM) [8; 25], Dirichlet priors on EM [32], or grid search/brute force [2; 26]. BKT has also been extended with contextualization of guess and slip [1; 2] and student priors [25; 26].

This proliferation of models has led researchers to ask which model is best. Student models have been compared in several fashions, both within and across paradigms, including both theoretical comparisons [1; 6; 34] and empirical comparisons at predicting future student performance [1; 2; 14; 29], as a proxy for the models' ability to infer a student's knowledge. However, the results of these comparisons have been quite inconsistent, with models performing better in one evaluation and worse in another evaluation [1; 2; 14; 29].

An alternate approach to determining which model is best might be to combine models and leverage the different strengths of different frameworks, using ensemble methods [4; 10; 31]. Such an approach was seen in the 2010 KDD Cup competition, where teams competed to predict future data on students using Cognitive Tutors for Algebra [18], training on earlier data from the same students. Two successful entries in this competition used ensemble methods, including the winning entry [37], which ensembled across multiple classification algorithms, and the second-best student entry [27], which ensembled across multiple paradigms for student assessment. However, the structure of the training and test sets used in this competition was not representative of common conditions in student modeling, where it is necessary to train models on one cohort of students and apply the models on a new cohort of students (such as the next cohort of students to use the learning system).

To this end, Baker and colleagues [3] attempted to improve prediction by applying ensemble methods on a different tutor, Cognitive Tutor for Genetics, testing the models on data from entirely new students. Their results indicated that ensemble methods were as good at predicting future knowledge as the best individual model, but no better. However, this study had three potentially key limitations. First, this study used only simple methods for combining predictions, restricting ensemble techniques to linear and logistic regression methods. Second, these studies were conducted with a small data set, raising the possibility that the results were weak because ensemble methods may need larger data sets to succeed in this domain. Third, the classic student modeling methods which were ensembled produced similar predictions, suggesting that ensemble selection may have failed to perform better due to having relatively little difference between predictors to leverage.

In the study published here, we attempt to extend the work in Baker et al. [3], in order to discover what conditions are necessary for ensemble selection to perform better than the best individual student knowledge models. To that end, we replicate the same analyses using a student data set approximately one order of magnitude larger than the set used in [3], though still two orders of magnitude smaller than the set used in the KDD Cup [27; 37]. We also use a substantially broader set of ensemble selection algorithms. As in that work, we test models using data from new students, a step required for actual model use. Through this process, we can better understand the potential of ensemble selection to improve the precision of student models of knowledge and skills.

2. STUDENT MODELS USED

In this section, we present a set of student models that can be used to predict student knowledge as they learn from within educational software [18, 33]. Greater detail on the specific tutor environment is given in section 4, but in order to understand the student models that follow, a few aspects of the software must be discussed, and a few assumptions must be made. Students typically use tutoring software individually (though many exceptions exist); the data set we study has data from a large number of students each using the software. Students complete a set of problems/problem steps/items in the educational domain (each of these terms refers to a different granularity of educational materials, but these differences do not matter for the approaches discussed here).

Each model is used to predict the probability that a student can answer the next item involving the current skill correctly (where correctness on a specific item is treated as a binary variable – correct or incorrect). These models all assume a mapping where each item involves a single skill, a simplifying assumption frequently used in modeling student knowledge in educational software (although this assumption can lead to ineffective adaptation in rare cases [19]). Cross-validation was conducted at the student level, so that models were trained on one group of students and tested on another group of students, a scenario comparable to real-world training and usage of student knowledge models.

The performance of each student modeling approach can be assessed as follows. When the model is tested on a group of students (during cross-validation), the model is applied to every student's actions during problem-solving. The student's actions are run through the model in the same order as they occurred during learning. For each of these algorithms, the skill which the action involved is taken into account. As a new action is encountered, the algorithm is used to predict student latent knowledge, and the probability the student will give a correct answer. The actual correctness of that action is used to (indirectly) determine the goodness of the model at assessing student knowledge. Models are compared using A' (also called AUC, the Area under the Receiver-Operating Curve) [16] – greater detail on the computation of A' and model comparison is given in section 5.

2.1 Bayesian Knowledge-Tracing

Corbett & Anderson's [9] Bayesian Knowledge Tracing model is one of the most popular methods for estimating students' knowledge. It underlies the Cognitive Mastery Learning algorithm in Cognitive Tutors for Algebra, Geometry, Genetics, and other domains [18], used by millions of students each year.

The canonical Bayesian Knowledge Tracing (BKT) model, depicted in Figure 1, assumes a two-state model of knowledge: for each skill/knowledge component the student is either in the learned state or the unlearned state. A student who knows a skill can either *slip* and give an incorrect response with probability $P(S)$ or give a correct response with probability $1 - P(S)$. Similarly, a student who does not know a skill can either *guess* and give a correct response with probability $P(G)$ or give an incorrect response with probability $1 - P(G)$. The probability of a correct answer in a given situation is a function of the prior probability that the student knows the skill, $P(L_{n-1})$, and the probability of the student demonstrating that knowledge with their response, $P(G)$ and $P(S)$. This is expressed in Equation 1 (note that the subscript n denotes the knowledge state at time n, and the subscript n-1 denotes the knowledge state at the time immediately after the previous answer):

$$P(Correct_n) = P(L_{n-1})\big(1 - P(S)\big) + \big(1 - P(L_{n-1})\big) * \big(P(G)\big) \qquad (1)$$

The model has another parameter, $P(L_0)$, which is the prior probability that a student had learned the skill before answering any questions. On the very first question, this parameter takes the place of $P(L_{n-1})$. After each opportunity to answer a question relating to the skill, a posterior probability of the student's knowledge state, $P\big(L_{n-1}\big|correctness\big)$, is calculated, taking into account evidence from the current action's correctness. If the student answers correctly, the posterior probability of knowledge is found by using Equation 2:

$$P(L_{n-1}|Correct_n) = \frac{P(L_{n-1})*\big(1-P(S)\big)}{P(L_{n-1})*\big(1-P(S)\big)+\big(1-P(L_{n-1})\big)*\big(P(G)\big)} \qquad (2)$$

If the student answers incorrectly, the posterior probability of knowledge is calculated with Equation 3:

$$P(L_{n-1}|Incorrect_n) = \frac{P(L_{n-1})*P(S)}{P(L_{n-1})*P(S)+\big(1-P(L_{n-1})\big)*\big(1-P(G)\big)} \qquad (3)$$

Finally, at each opportunity to apply that skill, regardless of correctness, the student may make the transition from the unlearned to the learned state with learning probability P(T). The probability of a student going from the learned state to the unlearned state (i.e. forgetting a skill) is fixed at zero. Hence, the probability that the student knows the skill at time n, after taking the possibility that he learned into account, is calculated with Equation 4 and will serve as the new prior probability of knowledge at the next opportunity.

$$P(L_n) = P(L_{n-1}|Action_n) + \left((1 - P(L_{n-1}|Action_n)\right) * P(T)) \tag{4}$$

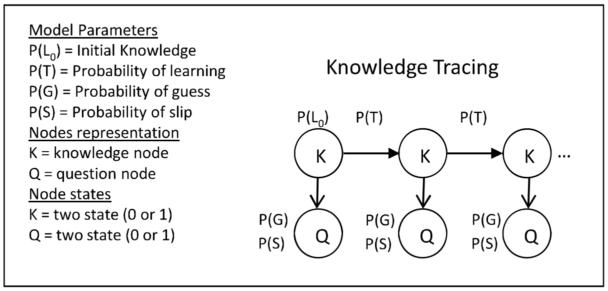


Figure 1: Bayesian Knowledge Tracing.

The four parameters of BKT, $P(L_0), P(T), P(G),$ and $P(S)$, are learned from existing data, originally using curve-fitting [9], but more recently using expectation maximization (BKT-EM) [8] or brute force/grid search (BKT-BF) [2; 25]. Within this paper we use BKT-EM and BKT-BF as two different models in this study. Within BKT-BF, for each of the 4 parameters all potential values at a grain-size of 0.01 are tried across all the students with an upper bound placed on $P(G)$ and $P(S)$ (e.g.: 0.01 0.01 0.01 0.01, 0.01 0.01 0.01 0.02, 0.01 0.01 0.01 0.03…… 0.99 0.99 0.3 0.1). The sum of squared residuals (SSR) is minimized in selecting the best set. For BKT-BF, the values for Guess and Slip are bounded in order to avoid the "model degeneracy" problems that arise when performance parameter estimates rise above 0.5 [1]. For BKT-EM the parameters were unbounded and initial parameters were set to a $P(G)$ of 0.14, $P(S)$ of 0.09, $P(L_0)$ of 0.50, and $P(T)$ of 0.14, a set of parameters previously found to be the average parameter values across all skills in modeling work conducted within a different tutoring system.

In addition, we include three other variants on BKT. The first variant uses a reduced data set in order to fit the model, under the hypothesis that not all historical data is relevant to predicting a student's current knowledge. BKT parameters are typically fit to all available students' performance data for a skill. It has been argued that if fitting is conducted using only the most recent student performance data, more accurate future performance prediction can be achieved than when fitting the model with all of the data [24; 27]. In this study, we included a BKT model trained only on a maximum of the 15 most recent student responses to the current skill, BKT-Less Data.

The second variant, the BKT-CGS (Contextual Guess and Slip) model, is an extension of BKT [1]. In this approach, Guess and Slip probabilities are no longer estimated for each skill; instead, they are computed each time a student attempts to answer a new problem step, based on machine-learned models of guess and slip response properties in context (for instance, longer responses and help requests are less likely to be slips). The same approach as in [1] is used to create the model, where 1) a four-parameter BKT model is obtained (in this case BKT-BF), 2) the four-parameter model is used to generate labels of the probability of slipping and guessing for each action within the data set, 3) machine learning is used to fit models predicting these labels, 4) the machine-learned models of guess and slip are substituted into Bayesian Knowledge Tracing in lieu of skill-by-skill labels for guess and slip, and finally 5) parameters for P(T) and $P(L_0)$ are fit.

An alternate approach to using the Contextual Slip estimates generated by the machine-learned model is to use them in the aggregate rather than applying them at each action, once the machine-learned models have been created [2]. For instance, if a student's values of Contextual Slip are averaged across all skills and actions, it can lead to better prediction of post-test performance. Combining a student's average Contextual Slip with standard BKT in linear regression improves prediction of post-test performance compared to BKT alone [2]. Hence, we include average Contextual Slip so far as an additional potential model.

The third BKT variant, the BKT-PPS (Prior Per Student) model [25], breaks from the standard BKT assumption that each student has the same incoming knowledge, $P(L_0)$. This individualization is accomplished by modifying the prior parameter for each student with the addition of a single node and arc to the standard BKT model. The model can be simplified to only model two different knowledge priors, a high and a low prior. No pre-test needs to be administered to determine which prior the student belongs to; instead their first response is used. If a student answers their first question of the skill incorrectly they are assumed to be in the low prior group. If they answer correctly, they assumed to be in the high prior group. The prior of each group can be learned or it can be set ad-hoc. The intuition behind the ad-hoc high prior, conditioned upon first response, is that it should be roughly 1 minus the probability of guess. Similarly, the low prior should be equivalent to the probability of slip. Using PPS with a low prior value of 0.10 and a high value of 0.85 has been shown to lead to improved accuracy at predicting student performance [27].

2.2 Performance Factors Analysis

Performance Factors Analysis (PFA) [28; 29] is a logistic regression model, an elaboration of the Rasch model from Item Response Theory. PFA predicts student correctness based on the student's number of prior failures F on that skill (weighted – typically negatively – by a parameter ρ fit for each skill using training data) and the student's number of prior successes S on that skill (weighted – typically positively – by a parameter γ fit for each skill using training data). An overall difficulty parameter β is also fit for each skill [29] or each item [28] – in this paper we use the variant of PFA that fits β for each skill using training data. The PFA equations use a two-step process to predict student correctness from the student's past history of successes S and failures F on each skill. First, a strength of student knowledge m is computed for the student i and the skill j:

$$m(i, j, S, F) = \beta_j + \sum(\gamma_j S_{ij} + \rho_j F_{ij}) \tag{5}$$

Then, this strength of student knowledge is converted to a probability of correctness, using a logistic function:

$$p(m) = \frac{1}{1 + e^{-m}}$$

Unlike Bayesian Knowledge Tracing, no running estimate of latent student knowledge is computed within PFA. Instead, the number of successes and failures are tracked, and these are used to predict future correctness.

2.3 CFAR

CFAR, which stands for "Correct First Attempt Rate", is an extremely simple algorithm for predicting student knowledge and future performance, utilized by the winners of the educational data KDD Cup in 2010 [37]. The prediction of student performance on a given skill is the student's average correctness on that skill, up until the current point.

3. ENSEMBLE METHODS

As discussed earlier, we use ensemble selection to integrate across the different student modeling methods discussed above, in order to see if a combination of student modeling methods can predict student knowledge more effectively than individual student modeling methods. We evaluated five ensemble methods for combining predictions, using multiple variants of four of the five methods. Two of the ensemble methods assign a weight for each model: straightforward averaging uses a uniform weight for each model, whereas regression methods fit a different weight for each model. The other three methods used can learn non-linear weightings, which allow for a different composition of models based on the values of the predictions of each model. Ensemble methods are subject to the same overfitting potential as any other model; thus, we used the same cross-validation process for ensemble evaluation as we did for evaluating the goodness of individual models.

3.1 Straightforward Averaging

In straightforward averaging, also known as uniform voting or bagging [11], each of the individual models' predictions of correctness (ranging from 0 to 1) is averaged for each student response.

3.2 Regression

We use linear, logistic and stepwise regression methods to ensemble the individual student model predictions. This method of linearly combining models is similar to that in [7]. Linear and logistic regression models were trained without feature selection (e.g. ensembles included all the student models). Another variant of regression we use is stepwise regression (based on a linear regression framework). In stepwise regression, predictions made by the best individual model (in terms of RMSE – root mean squared error) is chosen, and then the next best individual model that most improves the ensemble is added, until there no longer exists a model that significantly improves the fit.

3.3 AdaBoost

AdaBoost is an adaptive boosting algorithm [12]. This algorithm iterates over our ensemble data set of model predictions, focusing on improving prediction for incorrectly classified data. In this analysis we use two base learners: J48 [30] and Decision Stumps [20]. Parameters for these weak learners were left at the defaults found in v4.6 of RapidMiner [23]. Adaboost was run on the same ensemble data set that was used with regression and the other ensemble methods. The number of iterations for AdaBoost was set to 10.

3.4 Neural Network

Neural networks [17] are able to find complex non-linear relationships. A single feed forward hidden layer topology was chosen and the size of that hidden layer was varied with the following set of values {10, 25, 50, 100, and 125}. A learning rate of 0.3, momentum of 0.2 and epocs of 500 was used. We employed 5-fold cross-validation to evaluate the model, as with the other approaches but also used a sub cross-fold validation to select the size of the hidden layer. During the training phase of each fold, we split the training data into 2 sub-folds and then trained the neural nets with different sizes on one sub-fold and used the test sub-fold to select the best hidden layer size. After selecting the best size, we train the neural net on whole training set and apply the model on the test fold. We follow the same procedure during all the 5 folds.

3.5 Random Forest

Also referred to as bagged random decision trees, Random Forests [5] is an ensemble algorithm that trains many decision trees, each tree using a random resampling of the data (with replacement) and random sampling of the features of the data. In our case the features of the data comprise of predictions from the eight knowledge models. When making a prediction, each tree predicts the probability of a correct response and the average of the votes is taken as the final prediction of the Random Forest. We used 200 trees in the training of our Random Forests with a default feature sampling of 1/3rd and minimum data points per tree leaf of 5.

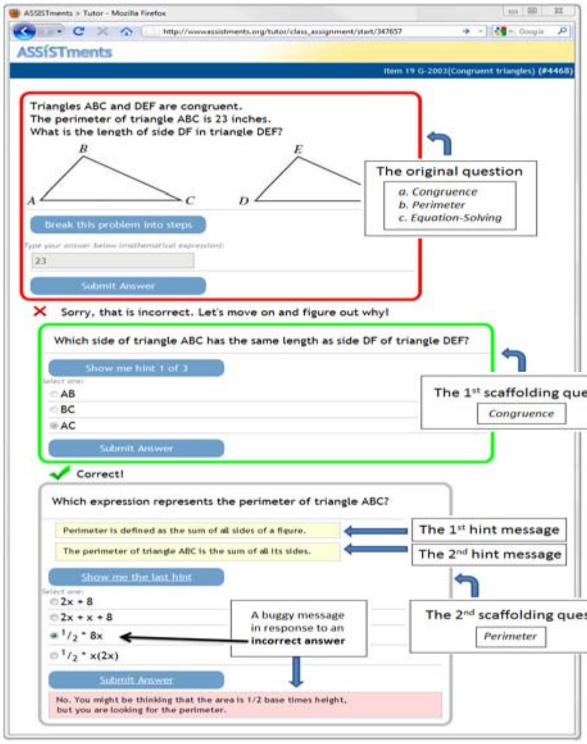
4. ASSISTMENT DATA SET

Our data set comes from student use of the ASSISTments Platform [33] during the 2005-2006 school year, an online learning environment used by approximately 20,000 students a year in the Northeastern United States. The ASSISTments Platform assesses as it assists, providing actionable data on student knowledge to teachers, while helping the students learn math. As a substitute for traditional homework, it has been found to lead to significantly improved learning outcomes [22]. The students in our data set were from 7th and 8th grade Geometry and Algebra classes with ages 12-14. Classes came from different schools and the teachers of the classes would take students to the computer lab to answer questions on ASSISTments about once every two weeks throughout the school year. There were 178,434 total student actions in this dataset produced by 5,422 students where each action is a student's first response to a question in the tutor. Students received a random selection of math problems from varying skills based on previously released state test items.

A picture of the ASSISTments interface is shown below (Figure 2). A student makes an attempt to answer a Pythagorean Theorem question but answers incorrectly. The tutor simplifies the question by providing scaffolding which breaks down the problem into sub-steps. The students answer these sub-steps and can request a final hint at the end which will provide the answer. Other pieces of content on ASSISTments provide only hint based help, which gives the student tips on solving the problem, rather than asking sub questions. Both forms of tutorial help (hints and scaffolding) are only provided if the student answers incorrectly or requests help. The immediate feedback of telling a student if they have answered correctly or not, as well as providing hints and scaffolding, has been shown to lead to student learning [22].

All the models used in this paper take into account which skill a specific item is associated with. The ASSISTments Platform uses a skill model [33] of 106 skills that provides this mapping between skills and items. Some problems were tagged with more than one skill. Since the knowledge models that were used assume each item is associated with a single skill, these problems were replicated in the dataset for each skill they were associated with.

For example, if a problem is associated with three skills, each student response to that problem is included in the data set three times, once for each skill.


ASSISTments > Tutor - Mozilla Firefox
http://www.assistments.org/tutor/class_assignment/start/347657
Google
ASSISTments
Item 19 G-2003(Congruent triangles) (#4468)
B
E
A
C
D
Break this problem into steps
Type your answer below (mathematical expression):
23
Submit Answer
The original question
a. Congruence
b. Perimeter
c. Equation-Solving
Show me hint 1 of 3
Select one:
Submit Answer
The 1st scaffolding question
Congruence
The 1st hint message
The 2nd hint message
Show me the last hint
Select one:
A buggy message in response to an incorrect answer
The 2nd scaffolding question
Perimeter
Submit Answer

Triangles ABC and DEF are congruent.
The perimeter of triangle ABC is 23 inches.
What is the length of side DF in triangle DEF?

Sorry, that is incorrect. Let's move on and figure out why!

Which side of triangle ABC has the same length as side DF of triangle DEF?

○ AB
○ BC
● AC

Correct!

Which expression represents the perimeter of triangle ABC?

Perimeter is defined as the sum of all sides of a figure.

The perimeter of triangle ABC is the sum of all its sides.

○ 2x + 8
○ 2x + x + 8
● $^1/_2$ * 8x
○ $^1/_2$ * x(2x)

No. You might be thinking that the area is 1/2 base times height, but you are looking for the perimeter.

Figure 2: An Example of an ASSISTments item.

5. EVALUATION OF MODELS

The eight student models used data from the tutoring system to make predictions of student performance using a 5 fold cross-validation. The predictions produced by these models served as the data used by the ensemble methods to explore if combining student model predictions could lead to improved accuracy. A 5 fold cross-validation was also used in the ensemble method evaluation.

5.1 In-tutor Performance of Models, at Student Level

We evaluate both student models and ensemble models using 5-fold cross-validation, at the student level. We balance the students in each of the folds to have equal number of actions and to have equal percent correct. By cross-validating at the student level rather than the action level, we can have greater confidence that the resultant models will generalize to new groups of students. We compute the A' (also called AUC, the Area under the Receiver-Operating Curve) [16] between the predictions obtained from the models and the correctness of each student first attempt on a problem step. We use A' as the goodness metric since it is a suitable metric to be used when the predicted variable is binary and the predictions are numerical (predictions of the probability of correctness at each item). To facilitate statistical comparison of A' without violating statistical independence, A' values were calculated for each student separately and then averaged across students (see [2] for more detail on this statistical method).

Table 1: A' values averaged across students for each of the models.

Model	Average A'
En-NeuralNet	0.7719
En-RandomForest	0.7662
En-AdaBoost-J48	0.7495
En-Logit	0.7162
En-LinReg	0.7129
En-StepWiseReg	0.7124
PFA	0.6994
En-AdaBoost-DecisionStumps	0.6840
BKT-EM	0.6817
BKT-LessData	0.6816
BKT-BF	0.6649
En-Average	0.6616
BKT-PPS	0.6548
CGS	0.6464
Cslip	0.5103
CFAR	0.5092

The average A' values are summarized in Table 1. The best ensemble model was Neural Net (A'=0.7719) and the best individual student model was PFA (A' =0.6994). Neural Net achieved statistically significantly higher performance than PFA, Z=27.21, $p<0.001$, indicating that the best ensembling method performed substantially better than the best individual model. Unlike the previous results seen with a smaller data set [3] the ensemble models generally appeared to outperform the individual student models, except for AdaBoost with Decision stumps (A'=0.6840) which performed comparably to PFA and the BKT variants, and Averaging (A'=0.6616) which was significantly outperformed by PFA and most of the BKT variants (the smallest difference in terms of statistical significance was between Averaging and BKT-BF, Z=2.18, p=0.03). The worst single model was CFAR (A'=0.5092), and the second-worst single model was Contextual Slip (A'=0.5103). All the other models achieved statistically significantly higher performance than CFAR and Contextual Slip at $p<0.001$.

5.2 In-tutor Performance of Models, at the Action Level

In this evaluation, the prediction ability of different models is compared based on how well each model predicts each first attempt at each problem step in the data set, instead of averaging within students and then across students. This is a more straightforward approach, which was used in the 2010 KDD Cup, although it has multiple limitations: it is less powerful for identifying individual students' learning, less usable in statistical analyses (analyses conducted at this level violate statistical independence assumptions [2]), and may bias in favor of predicting students who contribute more data.

Note that we do not re-fit the models in this section; we simply re-analyze the models with a different goodness metric. When we do so, we obtain the results shown in Table 2.

Table 2: A' computed at the action level for each of the models.

Model	A' (calculated for the whole dataset)
En-NeuralNet	0.7693
En-RandomForest	0.7651
En-AdaBoost-J48	0.7362
En-Logit	0.7183
En-StepWiseReg	0.7182
En-LinReg	0.7182
PFA	0.7053
BKT-LessData	0.7011
BKT-EM	0.7011
BKT-BF	0.6981
En-Average	0.6977
En-AdaBoost-DecisionStumps	0.6804
BKT-PPS	0.6716
Cslip	0.6148
CGS	0.6104
CFAR	0.6067

For this evaluation, the models follow the same pattern as the previous section. The ensemble models again outperform the individual models. The best ensemble model is again Neural Net (A'=0.7693), which is substantially better than the best individual model, which is again PFA (A'=0.7053). As in the previous section, the ensemble models again generally perform better than individual models.

6. DISCUSSION AND CONCLUSIONS

Within this paper, we have analyzed the effectiveness of a range of approaches for ensembling multiple student knowledge models within educational software data. We compared these ensembling approaches to the best individual student models of student knowledge in terms of their power to predict student behavior within the tutor (cross-validated) and evaluated them at the student and action level. We have found that with this data set, ensemble methods were unambiguously successful at predicting student performance with greater accuracy than single models, leading to as much as an 10% improvement in prediction accuracy. This improvement is much larger than differences typically seen between individual models [1; 2; 3; 14; 29]. The benefits of ensemble methods are seen even for relatively simple ensemble methods such as regression. This is contrary to the previous finding [3] with a smaller data set from a different tutor, where ensemble methods were not better than the best individual models.

Earlier, we hypothesized that there were three possible explanations for the observed lack of improvement using ensemble methods in previous work [3]: 1) the data set was too small for ensemble selection to be effective in this domain; 2) the ensemble selection methods used were too simple; 3) the knowledge models were too similar to each other. The knowledge models used in this paper remain similar to each other; since improved performance was achieved in this case, this hypothesis does not appear to be the correct explanation. In addition, the same regression methods for ensemble selection were used in this paper and that earlier work (along with additional methods). Hence, this also does not appear to be the key factor leading to success for ensemble methods (although neural networks *did* perform significantly better than regression methods). Instead, it appears that the size of the data set is the key difference leading to greater success in the current study than in that previous research.

Besides the results of ensembling, a primary contribution of this work is showing the relative predictive performance of the dominant knowledge models in the field on an additional data set. This paper gives further evidence that the relative performance of different individual student models is unstable between data sets. For example, BKT-PPS was the best model in [3], but was the worst-performing model among the BKT models in the analysis presented in this paper. PFA also performed worse than any BKT models in [3], the opposite pattern from the pattern of results seen here. It is not yet clear what features of a specific data set (and the tutor it comes from) are associated with better or worse performance for specific types of student models. This reinforces the motivation behind ensembling models to attain greater consistency across different tutors and student cohorts.

Overall this paper demonstrates that ensemble methods can be effective at substantially improving student performance prediction in a tutoring system, given sufficient amounts of data. It is not yet known exactly how much data is needed in this domain for ensemble methods to be effective – for future work, it may be valuable to generate samples of different sizes from a data set, and test the predictive performance of an ensemble trained on various sample sizes. An additional open research question is whether an ensemble trained on one year's cohort of students will generalize to the next year's cohort; at the current time, student models are often trained on one cohort's data and then used for the next year's cohort, making this a test that maps well to current practice. Through these steps, the field can utilize ensemble methods to increase the accuracy of predictions of latent student knowledge, and in turn improve the learning efficacy of educational software.

7. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation via the Pittsburgh Science of Learning Center, grant award #SBE-0836012, and by a "Graduates in K-12 Education" (GK-12) Fellowship, award number DGE0742503. We would like to thank the additional funders of the ASSISTments Platform found here: http://www.webcitation.org/5ym157Yfr

8. REFERENCES

[1] Baker, R.S.J.D., Corbett, A.T., Aleven, V., 2008. More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. In: Proc. of the 9th International Conference on Intelligent Tutoring Systems, 406-415.

[2] Baker, R.S.J.D., Corbett, A.T., Gowda, S.M., Wagner, A.Z., Maclaren, B.M., Kauffman, L.R., Mitchell, A.P., Giguere, S. 2010. Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. In: Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization, 52-63.

[3] Baker, R.S.J.D., Pardos, Z., Gowda, S., Nooraei, B., Heffernan, N., 2011. Ensembling Predictions of Student Knowledge within Intelligent Tutoring Systems. Proceedings of 19th International Conference on User Modeling, Adaptation, and Personalization, 13-24.

[4] Breiman, L., 1996. Bagging predictors. Machine Learning, 24, 123–14.

[5] Breiman. L., 2001. Statistical modeling: the two cultures. Stat Sci 2001;16:199–231

[6] Brusilovsky, P., Millán, E., 2007. User models for adaptive hypermedia and adaptive educational systems. In: P. Brusilovsky, A. Kobsa and W. Neidl (eds.): The Adaptive Web: Methods and Strategies of Web Personalization. Lecture Notes in Computer Science, Vol. 4321, Berlin Heidelberg New York: Springer-Verlag, pp. 3-53.

[7] Caruana, R., Niculescu-Mizil, A, 2004. Ensemble selection from libraries of models. In: Proceedings of the 21st International Conference on Machine Learning (ICML'04).

[8] Chang, K.-M., Beck, J., Mostow, J., Corbett, A, 2006. A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. In: Proceedings of the 8th International Conference on Intelligent Tutoring Systems, 104-113.

[9] Corbett, A.T., Anderson, J.R., 1995. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction, 4, 253-278.

[10] Dietterich, T. G., 2000. Ensemble Methods in Machine Learning, Proceedings of the First International Workshop on Multiple Classifier Systems, p.1-15, June 21-23

[11] Domingos, P., 1997. Why does bagging work? A Bayesian account and its implications. In D. Heckerman,H. Mannila, D. Pregibon, & R. Uthurusamy (Eds.), Proceedings of the Third International Conference onKnowledge Discovery and Data Mining (pp. 155–158). AAAI Press.

[12] Freund, Y. & Schapire, R. E., 1996. Experiments with a new boosting algorithm. In Proceedings of the 13th International Conference on Machine Learning, pp. 148-146. Morgan Kaufmann.

[13] Goldstein, I.J. (1979) The genetic graph: a representation for the evolution of procedural knowledge. International Journal of Man-Machine Studies, 11 (1), 51-77.

[14] Gong, Y., Beck, J.E., Heffernan, N.T., 2010. Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. In: Proceedings of the 10th International Conference on Intelligent Tutoring Systems, 35-44.

[15] Hambleton, R. K., Swaminathan, H., & Rogers, H. J. (1991). Fundamentals of Item Response Theory. Newbury Park, CA: Sage Press.

[16] Hanley, J. A., & Mcneil, B. J., 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology, 143(1), 29-36.

[17] Haykin, S., 1998. Neural Networks: A Comprehensive Foundation. New York, NY: Macmillan.

[18] Koedinger, K. R., Corbett, A. T., 2006. Cognitive tutors: Technology bringing learning science to the classroom. In K. Sawyer (Ed.), The Cambridge handbook of the learning sciences (pp. 61-78). New York: Cambridge University Press.

[19] Koedinger, K.R., Pavlik, P.I., Stamper, J., Nixon, T., Ritter, S., 2010. Avoiding Problem Selection Thrashing with Conjunctive Knowledge Tracing. Proceedings of the 3rd International Conference on Educational Data Mining, 91-100.

[20] Langley, P., Iba, W., 1992. An Analysis of Bayesian Classifiers. Proceedings of the 10th National Conference on Artificial Intelligence, 223-228.

[21] Martin, J., Vanlehn, K. (1995). Student Assessment Using Bayesian Nets. International Journal of Human-Computer Studies, 42, 575-591.

[22] Mendicino, M., Razzaq, L. & Heffernan, N. T. (2009). Comparison of Traditional Homework with Computer Supported Homework. Journal of Research on Technology in Education, 41(3), 331-359.

[23] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T. 2006. YALE: Rapid Prototyping for Complex Data Mining Tasks. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006), 935-940.

[24] Nooraei B., B., Pardos, A. Z., Heffernan, N. T., Baker, R.S.J.D., 2011. Less is More: Improving the Speed and Prediction Power of Knowledge Tracing by Using Less Data. Proceedings of the 4th International Conference on Educational Data Mining, 101-110.

[25] Pardos, Z. A., Heffernan, N. T., 2010a. Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. In P. De Bra, A. Kobsa, and D. Chin (Eds.): UMAP 2010, LNCS 6075, 225-266. Springer-Verlag: Berlin

[26] Pardos, Z. A., Heffernan, N. T., 2010b. Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm. In: Proc. of the 3rd International Conference on Educational Data Mining, 161-170.

[27] Pardos, Z.A., Heffernan, N. T., Using HMMs and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. To appear in Journal of Machine Learning Research W & CP.

[28] Pavlik, P.I., Cen, H., Koedinger, K.R., 2009a. Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. In: Proceedings of the 2nd International Conference on Educational Data Mining, 121-130.

[29] Pavlik, P.I., Cen, H., Koedinger, K.R., 2009b. Performance Factors Analysis – A New Alternative to Knowledge Tracing. In: Proceedings of the 14th International Conference on Artificial Intelligence in Education, 531-538. Version of paper used online at http://eric.ed.gov/PDFS/ED506305.pdf, retrieved 1/26/2011. This version has minor differences from the printed version of this paper.

[30] Quinlan, J.R., 1993. C4.5: Programs for Machine Learning. San Francisco, CA: Morgan Kaufmann.

[31] Quinlan, J.R., 1996. Bagging, boosting, and c4.5. Proceedings of the Thirteenth National Conference on Artificial Intelligence (pp. 725–730). AAAI Press and the MIT Press.

[32] Rai, D, Gong, Y, Beck, J. E, 2009. Using Dirichlet priors to improve model parameter plausibility. In: Proceedings of the 2nd International Conference on Educational Data Mining, Cordoba, Spain, 141-148.

[33] Razzaq, L., Heffernan, N.T., Feng, M., Pardos, Z.A., 2007. Developing Fine-Grained Transfer Models in the ASSISTment System. Journal of Technology, Instruction, Cognition, and Learning, Vol. 5. Number 3. Old City Publishing, Philadelphia, PA. 2007. pp. 289-304.

[34] Reye, J., 2004. Student modeling based on belief networks. International Journal of Artificial Intelligence in Education 14, 1-33.

[35] Thissen, D., Mislevy, R. 2000. Testing Algorithms. In H. Wainer, N.J. Dorans (Eds.) Computerized Adaptive Testing: A Primer (pp. 101-134). Hillsdale, NJ: Lawrence Erlbaum.

[36] Vanlehn, K., 1988. Student Modeling. In M.C. Polson & J.J. Richardson (Eds.) Foundations of Intelligent Tutoring Systems. London, UK: Psychology Press.

[37] Yu, H-F., Lo, H-Y., Hsieh, H-P., Lou, J-K., Mckenzie, T.G., Chou, J-W., et al., 2010 Feature Engineering and Classifier Ensemble for KDD Cup 2010. Proc. of the KDD Cup 2010 Workshop, 1-16.