Improving Stealth Assessment in Game-based Learning with LSTM-based Analytics

Bita Akram
North Carolina State University
Raleigh, NC 27695
bakram@ncsu.edu

Wookhee Min
North Carolina State University
Raleigh, NC 27695
wmin@ncsu.edu

Eric Wiebe
North Carolina State University
Raleigh, NC 27695
wiebe@ncsu.edu

Bradford Mott
North Carolina State University
Raleigh, NC 27695
bwmott@ncsu.edu

Kristy Elizabeth Boyer
University of Florida
Gainesville, FL 32611
keboyer@ufl.edu

James Lester
North Carolina State University
Raleigh, NC 27695
lester@ncsu.edu

ABSTRACT

A key affordance of game-based learning environments is their potential to unobtrusively assess student learning without interfering with gameplay. In this paper, we introduce a temporal analytics framework for stealth assessment that analyzes students' problem-solving strategies. The strategy-based temporal analytic framework uses long short-term memory network-based evidence models and clusters sequences of students' problem-solving behaviors across consecutive tasks. We investigate this strategy-based temporal analytics framework on a dataset of problem-solving behaviors collected from student interactions with a game-based learning environment for middle school computational thinking. The results of an evaluation indicate that the strategy-based temporal analytics framework significantly outperforms competitive baseline models with respect to stealth assessment predictive accuracy.

Keywords

Game-based Learning, Stealth Assessment, Temporal Analytics, LSTM, Strategy Use

1. INTRODUCTION

Recent years have seen significant growth in investigations of game-based learning. Game-based learning environments utilize the motivational elements of games to foster students' learning and engagement [7, 34, 36]. Studies have shown that learners who engage in game-based learning experience higher motivation compared to those who learn with conventional methods [8, 39]. Intelligent game-based learning environments integrate the adaptive learning support of intelligent tutoring systems and the motivational elements of games [15]. Like intelligent tutoring systems, they utilize students' interactions with the learning environment to infer student models of cognitive, affective, and metacognitive states [20, 26, 40]. The resulting student models can then guide tailored problem-solving scenarios, cognitive feedback, affective support [1,25].

In contrast to traditional assessment, stealth assessment of student learning can rely solely on student interaction trace data from the game-based learning environment without disrupting the natural flow of learning [38]. Stealth assessment infers students' competency with respect to knowledge, skills, and performance using evidence derived from students' game-based learning activities often based on evidence-centered design (ECD) [27]. ECD utilizes task, evidence, and competency models to assess students' relevant competency and proficiency [35]. In game-based learning environments, stealth assessment can monitor granular game-based behaviors across multiple tasks in the game to generate evidence, which can then be used to dynamically infer a competency model of the student. Operating in this fashion, stealth assessment has been examined to unobtrusively perform assessments of a wide range of constructs [40], and provide formative feedback to students and teachers to inform instruction and enhance learning [39, 5, 18].

Although an abundance of data can be readily captured from student interactions within game-based learning environments, a key challenge posed by stealth assessment is translating the raw data into meaningful representations to model students' competencies and performance [39]. This problem is exacerbated by the fact that student behavior unfolds over time in a manner dependent on prior actions. In this work, we present an approach to stealth assessment that leverages temporal analytics based on students' problem-solving strategies. Building on findings that problem-solving strategies significantly influence learning outcomes [11, 33], we introduce a strategy-based temporal analytics method using *n*-gram features and investigate whether problem-solving strategies identified from clustering students' interaction patterns can improve the predictive accuracy of evidence models for stealth assessment.

After clustering students based on their problem-solving behaviors, we predict their post-test performance using their cluster assignments as predictive features for a suite of classifiers. This approach is based on the intuition that as students' progress through a series of learning tasks, their choice of strategy affects their learning outcomes. For example, if a student first pursues a trial-and-error strategy for initial tasks and later in the learning session begins to adopt a more effective strategy, her strategy shift may lead to higher post-test scores. We hypothesize that drawing

inferences about strategy shifts may serve as the basis for accurate predictions of learning performance.

Because strategies and strategy shifts are inherently time-based phenomena, we propose a strategy-based temporal analytics approach to stealth assessment based on long short-term memory networks (LSTMs). In this approach, we develop predictive models that capture the temporal dependencies between students' dynamically changing problem-solving behaviors. We find that the strategy-based temporal analytics framework outperforms baseline models that do not capture strategic temporal dependencies on predictive accuracy. Further, we find that the strategy-based temporal analytics framework utilizing both student problem-solving behavior traces and pre-test performance outperforms a model that uses only pre-test data. The results suggest that strategy-based temporal analytics can serve as the foundation for effective stealth assessment in game-based learning.

2. RELATED WORK

Game-based learning leverages game design elements to foster engagement in learning [7]. Because of its potential to create motivating learning experiences, game-based learning has been explored for a broad range of subjects including science [29], mathematics [17], computer science [3, 24], and public policy [36]. A notable family of game-based learning environments, intelligent game-based learning environments, integrate intelligent tutoring system functionalities and game-based learning [15, 20]. Intelligent game-based learning environments can embed stealth assessments, which have emerged as a promising approach to assessing game-based learning [37, 31, 39]. In stealth assessment, student competencies are assessed unobtrusively by drawing inferences from observations of students' learning interactions.

In one approach to stealth assessment, a directed graphical model was built based on relevant competencies, and related variables were extracted from the observed data to be used as evidence for the targeted competencies [19]. In another approach, Falakmasir and colleagues investigated two hidden Markov models (HMMs) that were trained for high-performing and low-performing students [12]. Subsequently, for observed sequences of events, log-likelihoods were calculated for each HMM. Finally, the difference between the two log-likelihoods was used in a linear regression model to predict post-test scores. This approach reduces the need for labor-intensive domain knowledge engineering.

Work on deep learning-based stealth assessment, DeepStealth, offers an alternate approach that uses artificial neural networks to perform stealth assessment [24]. DeepStealth used a deep feedforward neural network (FFNN) to learn multi-level, hierarchical representations of the input data for evidence modeling. In subsequent work, structural limitations in the FFNNs were addressed with a long short-term memory network-based stealth assessment framework that directly uses students' raw interaction data as input [25]. The strategy-based temporal analytics framework we propose in this paper builds on this prior work, but while the previous work focused primarily on computational methods to model evidence within ECD, the approach introduced in this paper derives temporal evidence from students' dynamic in-game strategy use throughout their problem solving. We cluster students to categorize them based on in-game strategy utilization per task, and then use sequences of in-game strategy features over multiple tasks to predict post-test performance.

Previous work has also explored approaches to detect students' problem-solving strategies using trace data. For example, one effort focused on building a probabilistic model that jointly represent students' knowledge and strategies [16], which was effective at predicting learning outcomes. Another approach focused on selecting features for classifying students' efficiency in solving challenges [22]. The temporal analytics framework we introduce in the paper uses problem-solving strategies that are automatically discovered through clustering based on *n*-grams of players' sequences of interactions with a game-based learning environment, thus obviating the need for labeling or expert knowledge.

3. EXPERIMENTAL SETUP

We investigate the strategy-based temporal analytics approach for stealth assessment with data collected from middle school students' interactions with a game-based learning environment for computational thinking. We describe the learning environment, its in-game problem-solving challenges, and the dataset generated from students' interactions with the game-based learning environment.

3.1 ENGAGE Game-based Learning Environment

ENGAGE is a game-based learning environment designed to introduce computational thinking to middle school students (ages 11-13) (Figure 1). The game was developed with the Unity multi-platform game engine and features a rich, immersive 3D storyworld for learning computing concepts [3, 24]. The game-based learning environment aims to promote computational thinking skills including abstraction and algorithmic thinking through problem solving and programming. The computational challenges within the game were designed to prepare middle school students for computer science work in high school, and to promote positive attitudes toward computer science.

A diverse set of over 300 middle school students participated in focus group activities, pilot tests, and classroom studies with the game. Of the students who provided demographic information, 47% were female; 24% were African American or Black,16% were Hispanic or Latino/a, 17% were Asian, 38% were White, and 5% of the students were Multiracial. The research team worked closely with a similarly diverse group of teachers throughout the project. A subset of teachers helped to co-design the game-based learning activities, providing iterative feedback throughout development. Each of the teachers implementing the game in their classrooms attended either one or two summer professional development workshops that introduced computational thinking concepts and the ENGAGE game-based learning environment.

In the game, students play the role of the protagonist who is sent to investigate an underwater research facility that has lost communications with the outside world. As students progress through the game, they discover that a nefarious villain has taken control of the computing devices within the facility. Students navigate through a series of interconnected rooms and solve a set of computational challenges. Each of the challenges can be solved either by programming devices or interacting with devices in reference to their pre-written programs. Students use a visual block-based programming language to program the devices [25]. They are supported throughout the game by a cast of non-player characters who help them progress through the narrative, offer clues, and provide feedback while they navigate the game and solve computational challenges [24].

The game consists of three major levels: the Introductory Level, in which students learn the basics of the game and simple programming; the Digital World Level, in which students learn

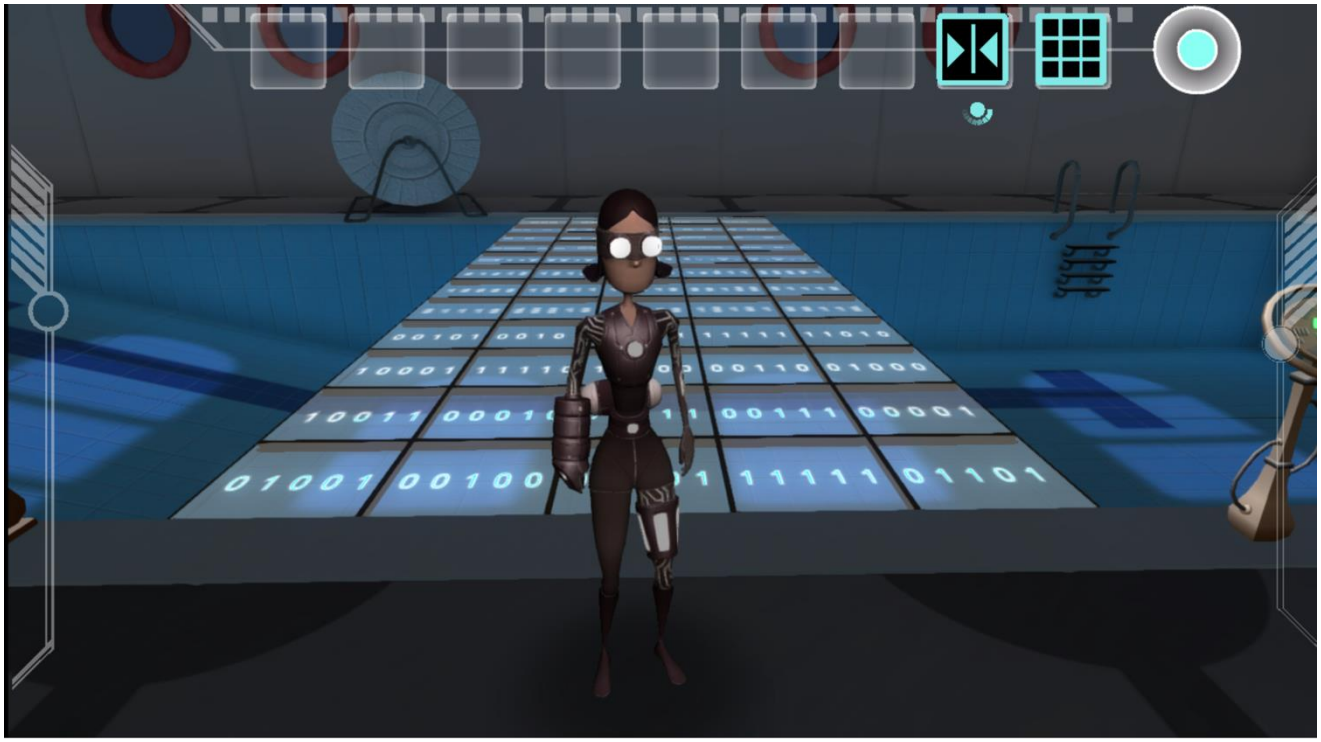

Figure 1. ENGAGE game-based learning environment: students write a program that loops over a binary grid.

how digital data is represented with binary sequences; and the Big Data Level, in which students have the opportunity to work with various datasets and retrieve hidden information by cycling through data and filtering it based on different conditions.

The work presented in this paper focuses on students' problem-solving activities within the Digital World level. The first set of tasks in this level consists of binary locks that are programmed to open if the binary representation of a specific base-ten number is generated by the students. Similarly, the second set of tasks (Figure 2) in the Digital World level features lift devices that are activated when students generate the target base-ten value by flipping binary tiles and execute the program associated with the lift. For example, students can find the target number by reviewing an existing program (Figure 2, right) associated with the binary lift device. Each lift provides students with five consecutive flip tiles representing bits of a five-digit binary number. Players can toggle each bit between 0 and 1 by flipping the corresponding tile on the tile panel. The decimal representation of their generated binary number will be presented on a small screen above the panel. To teach variations of binary representations, the game enables students to flip tiles between '0' and '1', 'black' and 'white', and 'F' (False) and 'T' (True), as in (Figure 2, left).

To advance to the next task, students must flip binary tiles on the binary lift device to generate the target decimal number (Figure 2, left) execute its program, and lift up the binary device. Through these tasks, students learn about the concept of bits in binary numbers and the weight assigned to each bit. In the analyses reported here, we used behavior trace data from students' interactions with 11 binary tasks from the Digital World level, where students learn the weight associated with each of the five bits through the first five tasks and then learn how to combine multiple bits to make more complex numbers with binary representations.

3.2 Dataset

We analyzed 244 students' behavior trace data obtained from a teacher-led study in four public middle school classrooms in the urban area in the United States. The four schools reported an average percentage of free or reduced lunch as 34.75%, 41.07%, 31.65%, and 63.17% during the years of data collection, respectively. Furthermore, three of the schools were magnets for gifted and talented students and the fourth was a magnet for leadership and innovation. To support collaborative learning, which is prominent in computer science education [3], we collected student behavior trace interaction data from pairs of students in which they took turns serving as navigator (traversing the game) and driver (action planning). Pre- and post-test assessments measuring content knowledge (e.g., binary representation) were completed individually by students before starting the Digital World level (pre-test) and immediately after finishing it (post-test). Both pre-test and post-test are on a scale of 0 to 1. Out of 244 students, 168 students completed the pre-test and post-test for content knowledge as well as all 11 binary representation tasks for this level. The results of conducting a paired t-test on students' content knowledge pre-test (M=0.44, SD=0.20) and post-test (M=0.59, SD=0.24) revealed a significant improvement from pre-test to post-test scores ($t(167) = 11.24$, $p<0.001$).

4. MODELING STUDENTS' PROBLEM-SOLVING STRATEGIES

Students exhibited a broad spectrum of problem-solving strategies while solving the binary challenges in the Digital World level. For example, some students pursued random trial-and-error strategies to find solutions, while at the other end of the spectrum, some students pursued thoughtful systematic approaches to solve the challenge. As would be expected, some students fell in the middle of this spectrum by utilizing more thoughtful trial-and-error.

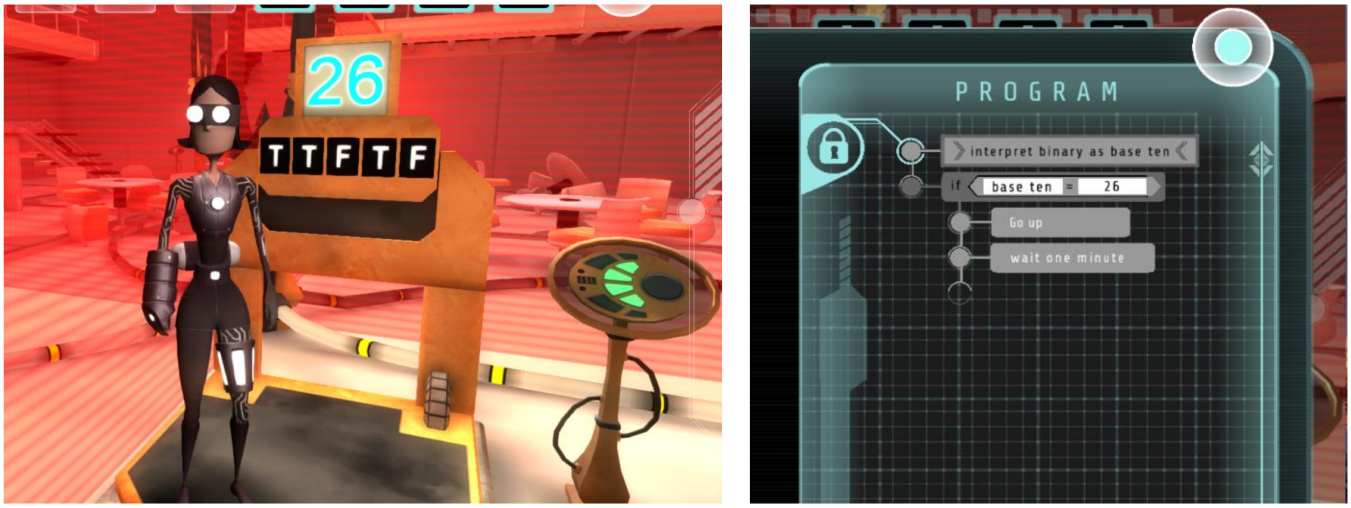

Figure 2. (Left) A binary lock device that students must unlock. The T (true) tiles indicate the bits are 1, whereas F (false) tiles denote 0. The current binary number is 11010 and the corresponding base-ten number, 26, is displayed on the device as immediate feedback. (Right) The visual programming interface displaying the binary lock's program.

For each of the 11 consecutive binary challenges, we used students' tile-flip sequences to cluster them into distinct groups. Subsequently, we interpreted these clusters in terms of the problem-solving strategy exhibited by members of each cluster. Below we first describe the process of clustering students' task-level strategies based on their binary tile flip sequences and then describe the representative problem-solving strategy in each cluster.

4.1 Methodology

In order to group students based on their problem-solving strategies, we first derived features from students' binary tile flip sequences. We encoded the flip sequences as *n*-grams, commonly used as a representation for sequential data such as text and speech [41], as well as for sequential trace data [12]. The *n*-gram representation extracts sequences of *n* adjacent elements from the original string. We consider each unique *n*-gram as a feature in our *n*-gram based feature vector, while we use the frequency of each *n*-gram occurrence in a flip string as a value in this work.

For each of the 11 binary challenges, students' behaviors (i.e., the flip sequence generated for that specific task) were clustered based on the extracted *n*-gram features, resulting in 11 sequential cluster-memberships per student. Since each task differed slightly from the other tasks, we analyzed students problem-solving behavior separately for each task. In the following sections, we describe how we identified different problem-solving strategies using the proposed clustering method.

4.1.1 Feature Engineering

We extracted students' interactions with binary flips in the format of a string containing students' consecutive flips of the binary tiles for each task. Each task is associated with a decimal number to operate the device (e.g., 26 in Figure 2), where the binary number displayed on the five tiles is set to 00000 by default. For example, considering tiles' indices starting at one from the right most tile, if a student has flipped tile number four (i.e., 01000 with the decimal representation of 8), followed by flipping tile number five (i.e., 11000 with the decimal representation of 24), their tile flip string would become {4, 5}.

In order to capture the most fine-grained information present in the series of flips, we used *n*-grams with varying lengths of *n*. Preliminary explorations showed including sequences of lengths larger than four exponentially increases the sparsity of the dataset. To eliminate the sparsity issue, we capped the *n*-gram size at 4. Our final feature set ranges from sequences of length one (i.e., unigram features) to sequences of length four (i.e., 4-grams) that are repeated at least three times throughout our dataset. We used the natural language processing toolkit (NLTK) library for Python to extract *n*-grams and their associated frequency from each flip string. For example, for one of the tasks, a total of 2,495 unique *n*-grams with at least three occurrences were generated from the student flip strings for that task. These *n*-gram feature vectors were then used to cluster students' in-game strategy use per task, where an *n*-gram feature vector per student was generated separately for each of the 11 tasks.

Flip strings provide a fine-grained representation of students' problem-solving behaviors in solving binary representation challenges, and these features offer a method to identify students' adopted strategies. As an example, consecutive flips of the same tile by a student can be an indicator of the student's intention to learn the weight assigned to that binary digit. Further, the overall number of flips conducted to generate the target base-ten value can be used to gauge the students' overall efficiency in solving the problem.

4.1.2 Clustering

Next, we applied the expectation-maximization (EM) clustering technique to students' flip behaviors represented using an *n*-gram feature vectors to identify students' problem-solving strategies. Because each of the 11 tasks in the Digital World level targets a different base-ten number, the binary code needed to solve the task is different. Consequently, flip sequences obtained from students' interactions with a binary device reveal information specific to the target value designed for the task. Thus, clustering was performed separately for each of the 11 tasks. We used the MClust package in R to cluster the feature vectors. EM clustering can explore a range of cluster numbers and return the (local) optimal number of clusters based on the maximum likelihood estimation. A different optimal number of clusters was identified for each task. Three, four, and nine clusters emerged most frequently when we explored the number of clusters between two to ten. A preliminary investigation on these different number of clusters found that three clusters

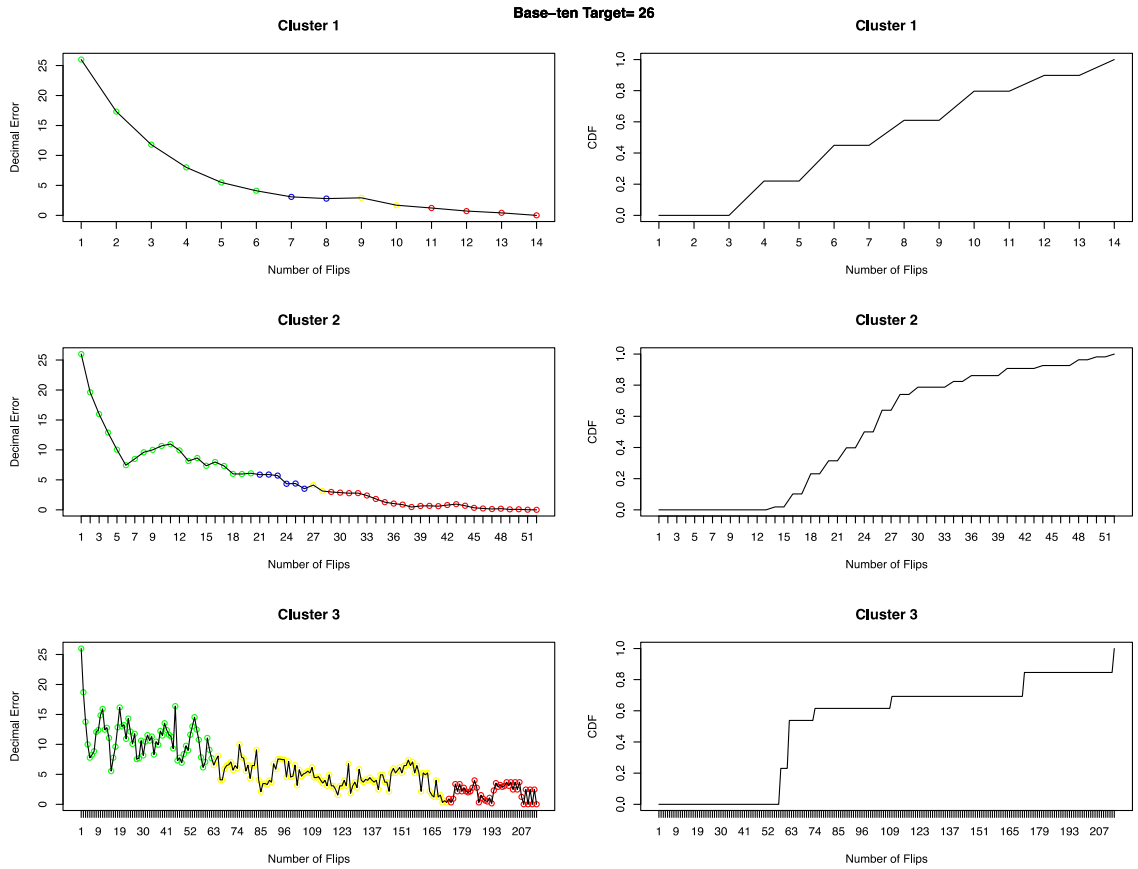

Figure 3. Students' average decimal error and the CDF of the present population at each flip.

showed coherent patterns for problem-solving strategies across all tasks, and we thus chose three as the number of clusters for all tasks in this work.

4.2 Interpreting Clusters

To interpret student problem-solving strategies using the clusters as identified above, we present two novel methods that measure the error between the target value and the student-generated value. To analyze students' problem-solving patterns for each cluster of each task, we calculated the average error at each flip relative to the solution target of students who belong to the same cluster. All students start from the difference between the default value zero and the target decimal value. Since we analyzed students who completed a task, the average error for each cluster decays toward zero, and we expected to observe distinct error-decay patterns across the three clusters. We introduce two error calculation metrics that measure students' error based on the distance from the current value to the targeted value: the decimal error and binary error. These two approaches are described below.

4.2.1 Decimal Error

The decimal error is the absolute difference between the target base-ten value and the base-ten representation of the student-generated binary string. Each student starts with an error equal to the target value and ends with an error equal to zero. We calculate the decimal error after every new flip. As a result, a sequence of decimal errors is generated for each student per flip action when completing each of the 11 tasks. We then plot the average decimal error where the y-axis shows students in the same cluster (separately for each task), and the x-axis shows the maximum number of flips observed in the cluster as in Figure 3 (left). Because the total number of flips is different for each student in a cluster, we use the decimal error value of zero for students who already completed the task and calculate the average decimal error over all students in the cluster.

For example, suppose there are two students in a cluster, where student A's decimal error sequence is {2, 1, 2, 0} and student B's error sequence is {2, 0} in the task of making the value two in base-ten. We use the maximum length of sequence, four, obtained from student A, and reformulate student B's sequence to {2, 0, 0, 0}. In this case, the average decimal error sequence becomes {2, 0.5, 1, 0}. The average error at each flip for the eighth binary challenge where the student is asked to find the binary number for the target 26 is shown in Figure 3 (left). For this task 118 students were grouped in the first cluster, 108 students were grouped in the second cluster, and 19 students were grouped in the third cluster. In Figure 3, because the target value for this task is 26, the average error for students is 26 in the beginning, which becomes zero at the end, while decay patterns differ across clusters. Students in each cluster solved the problem within a varying number of flips. The error for students who finished earlier is represented with zero. We show the percentage of students still working on the challenge at each flip using a color coding scheme. In Figure 3, green points mark flips where between 70% to 100% of the population is present, blue points indicate the presence of 50% to 70% of the population, yellow points mark 30% to 50% of the population, and red points indicate flips were less than 30% of the population of that cluster are still working on the problem. These percentages are derived from the cumulative density functions (CDFs) of clusters'

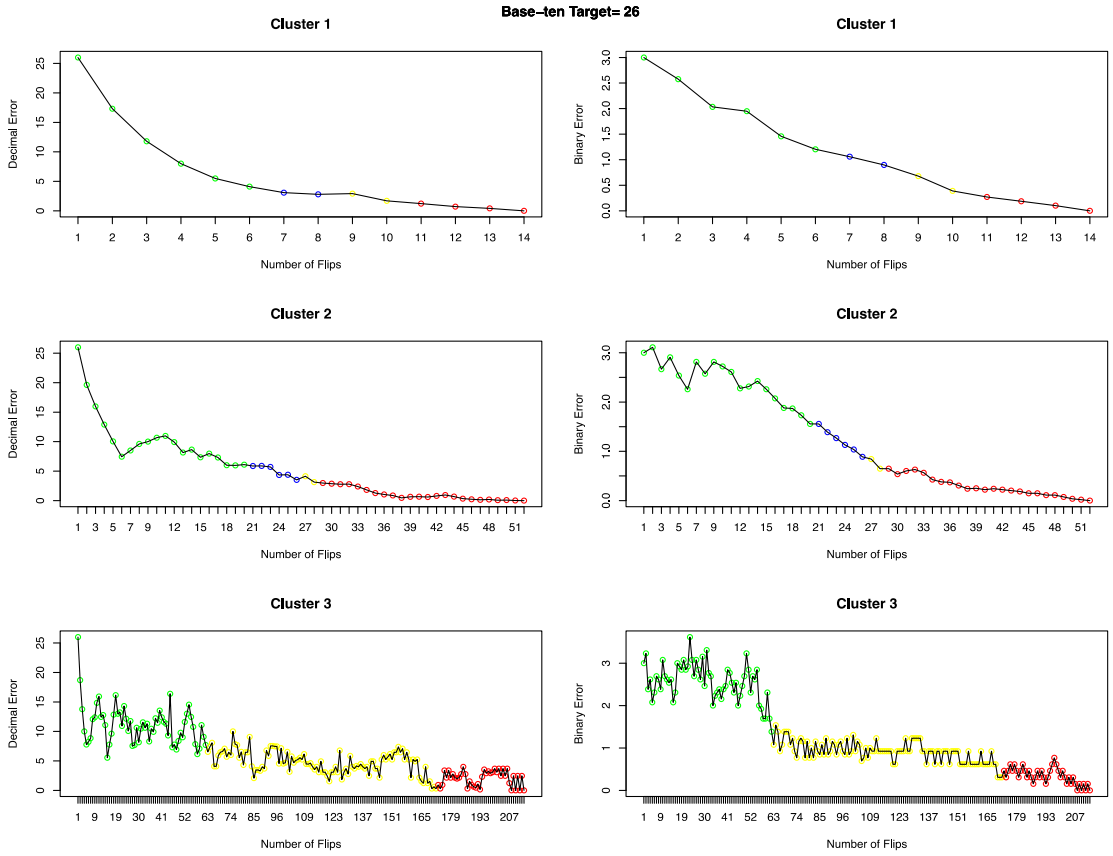

Figure 4. Students' average binary and decimal error at each flip.

present population at each flip that are plotted in Figure 3 (right). The clusters for the other ten binary representation tasks in the Digital World level follow similar error-decay patterns.

4.2.2 Binary Error

Binary error is the Hamming distance, the number of different elements in two strings with the same size, between the current state of the student-generated binary string and the binary representation of the target base-ten value. The approach for plotting the binary error is similar to the approach for plotting the decimal error. Figure 4 shows binary (right) and decimal (left) error plots for each cluster of the challenge, finding the binary representation of the number, 26. As the binary error plots are generated from the same cluster-based population used for generating decimal error plots, the same CDFs as in (Figure 3, right) hold for binary error plots.

4.2.3 Resulting Strategies

The same general patterns apply to other challenges analyzed in this study. As seen in Figure 4, there is a coherence in the error decay pattern between the decimal and the binary error. The decimal error captures students' strategies to make the base-ten errors between the target number and current binary representations as small as possible, while the binary error places more emphasis on the representational difference between binary sequences focusing on students' understanding on each bit and its associated weight. The analyses reveal a clear distinction in students' problem-solving strategies in solving the in-game challenges. After clustering, the following distinct groups emerge for all analyzed tasks:

- Students who completed the task more quickly than the other groups and with fewer trial-and-error attempts (Cluster 1).
- Students who had a moderate number of flips and demonstrated error decaying continuously toward zero with some trial-and-error attempts (Cluster 2).
- Students who completed the activity with many more flips compared to students in the other clusters, which may be an indicator of less thoughtful trial-and-error attempts (Cluster 3).

The binary and the decimal error decay patterns paralleled each other for every cluster of every task. The analyses reveal that the two error metrics similarly capture *n*-gram-encoded student behaviors, while students' per-task behaviors naturally fall into one of the three groups. We used these identified clusters as game strategy features for our evidence model for stealth assessment.

5. STEALTH ASSESSMENT

Modeling students' strategies can contribute to improving their learning outcomes [11, 33]. In this work, we aim to evaluate the predictive power of models of students' in-game problem-solving strategies over time to predict their post-test performance. We seek to determine if the in-game strategies observed in students' interactions with the game environment can be used as evidence for stealth assessment.

The feature set implicitly represents rich temporal dependencies among student behaviors over the course of interactions with the

ENGAGE game-based learning environment. To effectively model temporal dependencies in the feature set we investigate an evidence model based on long short-term memory network (LSTM) [14] to infer students' post-test performance based on their in-game strategy use over time. We also examine two baseline classification techniques, random forest (RF) [1] and support vector machines (SVM) [9], to predict students' post-test performance. It is important to note that, in contrast to the LSTM approach, neither the random forest nor the support vector machine approaches explicitly capture the temporal relationships in students' strategies. Thus, they treat in-game strategy features as independent features in their predictions.

We devise two LSTM-based evidence models, one model utilizing a feature set that contains pre-test performance only and another model utilizing both pre-test performance and in-game strategy features, to isolate the effects of incorporating the temporal dependencies captured by the LSTM-based model.

5.1 Data Preparation

For the classification task, we use data from 168 students who finished both pre- and post-tests and also completed all 11 binary challenge tasks within the game. We excluded data for students who did not complete all 11 tasks as we intended to perform a temporal analysis across these tasks. Initially, the dataset included pre- and post-test scores along with students' flip sequences, which were then transformed into *n*-gram features for each of the 11 tasks. We divide the data into training and held-out test sets. We first perform clustering using students' *n*-gram feature vectors in the training set. After identifying distinct clusters for each challenge in the training set, we use the Gaussian finite mixture models estimated by the MClust package to cluster students' data in the test set. This maintains the independence of the training and test sets. Students' data in both the training and test sets are represented with sequences of in-game strategies across the 11 binary-representation tasks along with their pre-test performance (i.e., high, medium, low), a categorical representation of the pre-test score based on a tertile split obtained from the distributions of the pre-test scores. We use these input features to predict post-test performance also using the three labels, which are obtained based on a tertile split of students' post-test scores. We chose tertiles to create a balanced distribution among all classes. The initial pre- and post-test scores are continuous variables, ranging between 0 to 1. For the pre-test, scores between ($0 \leq$ score ≤ 0.36) are categorized as low, scores between ($0.36 <$ score ≤ 0.54) as medium, and scores between ($0.54 <$ score ≤ 1.00) as high. Similarly, for the post-test, scores between ($0 \leq$ score ≤ 0.45) are categorized as low, scores between ($0.45 <$ score ≤ 0.72) as medium, and scores between ($0.72 <$ score ≤ 1.00) are categorized as high. Table 1 presents the distribution of students (n=168) with respect to students' pre- and post-test performance.

Table 1. Distribution of students (n=168) in relation to their pre- and post-test performance

Test	Low	Medium	High
Pre-test	74	53	41
Post-test	60	59	49

To transform the data into a trainable representation, we use one-hot encoding on the categorical variables (i.e., pre-test performance and the 11 in-game strategy changes) in preparation for the classification task. One-hot encoding is a feature representation method for a categorical variable, where a feature vector whose length is the size of the possible values is created, and only the associated feature bit is on (i.e., 1) while all other feature bits are off (i.e., 0). We also prepare two distinct feature sets to evaluate the predictive power of the in-game strategy features:

- Full feature set: For RF and SVM, 36 features including 33 one-hot encoded features representing the cluster membership among the three clusters for each of the 11 binary tasks and three one-hot encoding-based features (i.e., low, medium, and high) representing students' pre-test performance and 3 features to represent students' pre-test performance. For LSTMs, since they take as input the pre-test performance (3 features) and a task-specific in-game strategy (3 features) per time step, it utilizes six features.
- Pre-test performance feature set: Three one-hot encoding-based features (i.e., low, medium, and high) representing students' pre-test performance.

5.2 Classification Methods

We use 'randomForest' [21] and 'e1071' [23] packages in R to train random forest and SVM classifiers, respectively. For LSTM-based evidence models, we use the Keras [6] and scikit-learn [30] libraries in Python.

We use 5-fold cross-validation within the training data to tune the hyperparameters of the classification techniques based on the full feature set. After optimizing the hyperparameters, we train each of the classifiers using the full training set and evaluate them on the held-out test set. After comparing classifiers, we take the best performing classification technique and train an additional model based on the other feature set, pre-test performance feature-set, using the same test/train data split used for the full feature set-level analysis. The classification process for each classifier and their results are described below.

5.2.1 Baseline Method

The majority class-based method assigns the most frequent label in the dataset as the predicted label for all data instances. Since the most common label is the grade 'low', all labels will simply be predicted as the first class (i.e., low post-performance). The result of applying the baseline method on the full feature set achieves an accuracy of 35.71%. The macro average for recall is 33.33%. The precision and F1-score are undefined here since the baseline method predicts the most frequent label for all instances, while producing no other labels.

5.2.2 Random Forest Method

The random forest technique generates multiple decision trees using different subsets of the training data using bagging. A random forest tree is generated by trying a random subset of available features at each split. It then classifies each point in the test set using all the trees and uses the majority vote for classifying the test point. We use the set (10, 25, 50, 100, 200) to tune the number of trees for the model. Using a 5-fold cross-validation approach on our training set we found 25 to be the best number of trees for the full feature set.

Random forest classifiers are subject to randomness when being trained on a dataset. They perform feature bagging (i.e., a random selection of the features at each candidate split), and thus the predictive performance of random forests trained utilizing the same set of hyperparameters can vary depending on the random procedure. As a result, each round of training and evaluation on the same training/test sets will result in slightly different accuracies. Hence, we report the average result of 100 rounds of training and

evaluating the classifier. The mean and standard deviation of the results are shown in Table 2. The results of applying the random forest classifier on the full feature set achieve an average accuracy of 50.43%, an average precision of 52.20%, an average recall of 50.98%, and an average F1-score of 51.03%. The precision, recall and F1 measures are calculated using a macro-average of all three classes (i.e., simple average of the relative measurement of all three classes).

5.2.3 SVM Method

Support vector machines (SVMs) can be used for both regression and classification tasks. In classification tasks for which data are not linearly separable, data will be transformed to a higher-dimensional space for linear separability, and SVMs are applied to classify the transformed data. For this classification task, we use a third-degree polynomial kernel. We tune *C* as the hyperparameter of our SVM model. *C* is the regularization parameter that controls models' tolerance for incorrect classifications during training. We explore a set of values (0.01, 0.1, 1, 10, 15) to tune *C* on the full feature set. Using a 5-fold cross-validation approach on the training set, we found $C = 1$ to be the best parameter to be used in the model. The results of applying the SVM model on the test set show an accuracy of 41.17%, a precision of 44.14%, a recall of 39.25%, and an F1-score of 35.44%. Similar to the RF classifier we report the average result of 100 rounds of training and evaluating the trained classifier. Since there is no random parameter for this method, the standard deviation for the estimated accuracies is 0.

Both the random forest and SVM approaches achieve higher accuracies compared to the simple majority class baseline, suggesting that these methods are effective for stealth assessment. We hypothesize that the accuracy could be increased by explicitly modeling the temporal relationships across students' sequential problem-solving tasks. We next describe the LSTM-based approach and the results it produces.

5.2.4 LSTM Method

LSTMs are a type of recurrent neural networks (RNNs), a class of deep learning methods that are capable of learning temporal patterns in data. This characteristic makes LSTMs a promising candidate for classifying sequential data, such as time-series data of students' strategy uses across the 11 binary challenges they solve during gameplay. A sequence of cluster types (i.e., in-game problem-solving strategies for the 11 in-game binary representation tasks) can reveal students' problem-solving progressions as they unfold over learning sessions to predict students' learning outcomes. We investigate LSTMs to model dynamic changes in students' problem-solving strategies, motivated by LSTMs' ability to preserve long-term dependencies through their three gating units (i.e., input, forget, and output gates).

We tune the number of LSTM layers and the number of hidden units within each layer by conducting a 5-fold cross validation on the training set. We explore 15 different hyperparameter combinations with different numbers of hidden layers (1, 2, 3) and different numbers of hidden units in each layer (10, 15, 25, 50, 100). We found that networks with 2 layers with 15 units per produced the best results for predictive accuracy.

Like random forest models, the LSTM-based approach also results in different models each time it is trained on the same training set. Hence, evaluating these models on the same test-set generates slightly different outputs. This is due to the fact that deep learning approaches are sensitive to the random weights used to initialize the network. In addition, these types of techniques are trained on batches and the input order of the batches influence the models that are generated. We report an average of 100 runs of training and evaluating the LSTM classifier on the same training and test set. The results of applying this LSTM on the held-out test set achieve an average accuracy of 64.82%, an average precision of 63.88%, an average recall of 65.14%, and an F1-score of 63.68%.

Table 2 provides a summary of the results of the classification methods. The highest score per metric is indicated in bold. The baseline and SVM approaches are deterministic so their metrics' standard deviations are zero. All classification methods outperform the majority class baseline. Because reasoning about students' problem-solving strategy adoption *over time* can inform predictions about the strength of their learning as measured by post-test performance, the LSTM-based evidence model yields considerable improvement over the other approaches. The results indicate that the LSTM model appears to successfully capture the latent temporal dependencies among features in students' problem solving.

Table 2. Performance (± standard deviation) of classifiers

Method	Accuracy	Precision	Recall	F1
Baseline	35.7(±0.0)	N/A	33.3(±0.0)	N/A
RF	50.4(±2.5)	52.2(±2.7)	51.0(±2.4)	51.0(±2.6)
SVM	41.2(±0.0)	44.1(±0.0)	39.3(±0.0)	35.4(±0.0)
LSTM	**64.8**(±2.7)	**63.9**(±2.8)	**65.1**(±2.8)	**63.7**(±2.5)

5.3 In-game Strategy for Stealth Assessment

To further investigate the effectiveness of the in-game strategy features in predicting students' post-test performance, we compare two versions of the LSTM-based model, our best performing classification technique. We create a version of the LSTM-based model trained on the full feature set (pre-test features together with in-game strategy features) and compare it to a partial feature set version (pre-test features only). The results of this evaluation are shown in Table 3, where the highest score per metric is indicated in bold.

Table 3. Results of applying LSTM on pre-test only, in-game strategy, and full features feature sets

Feature set	Accuracy	Precision	Recall	F1
Full FS	**64.8** (±2.7)	**63.9**(±2.8)	**65.1**(±2.8)	**63.7**(±2.5)
Pre-test FS	44.7(±7.9)	N/A	42.8(±8.3)	N/A

The results demonstrate that incorporating the in-game strategy features into the model significantly contributes to predictive accuracy. Compared to the 44.66% accuracy achieved by the partial feature set version (pre-test features only), the model that uses in-game strategy features in addition to pre-test features achieves an accuracy of 64.82%. The significantly higher accuracy achieved by the full-set model suggests that the strategy-based approach that uses sequences of strategies as represented by strategy clusters appears to capture an important quality of students' problem-solving strategies that are predictive of learning performance.

6. DISCUSSION

Stealth assessment relies on accurate evidence models inferred from student behavior traces, and we found that student behavior

traces can serve as the foundation for evidence models that are driven by students' in-game problem-solving strategies.

Building on previous work on stealth assessment we have presented a novel problem-solving-strategy-based temporal analytics framework leveraging a clustering approach, which notably does not require a labor-intensive process of labeling data. While the previous work focused on computational methods to model evidence within ECD using deep learning networks, we have investigated temporal evidence derived from students' dynamic in-game strategy uses throughout their game play, and have demonstrated the effectiveness of LSTM-based evidence models that predict students' post-test performance.

For each of the 11 problem-solving tasks in the ENGAGE game-based learning environment, we first transformed sequences of student behavior interactions into sequences of *n*-gram features to capture the temporal information that spans interaction sequences and clustered them with EM Clustering. The results revealed clear distinctions in students' approaches toward solving these computational thinking challenges. The clustering grouped students into those who solved the problem in a few flips and a few attempts, those who solved the problem with a moderate number of flips and with thoughtful trial-and-error, and those who solved the problem with a long sequence of flips and with seemingly random trial-and-error. While in our game settings students could try the problems as many times as they wanted, other game environments might take number of trials into account using a point system that could affect players' problem-solving strategies.

We then used students' cluster memberships across different tasks as an indicator of their in-game problem-solving strategy and used these problem-solving strategies to inform the evidence model for predicting students' post-test performance. The results demonstrated that the in-game strategy features provide strong predictive capacity for LSTM-based evidence models and more generally for the use of stealth assessment. It has been shown that LSTM-based ECD evidence models with in-game strategy features effectively capture the temporal relationships between strategies, as supported by the models' highest predictive accuracy rate, precision rate, recall rate, and F1 scores outperforming competitive non-sequential baseline approaches in predicting students' post-test performance. We used a relatively small dataset, 168 students for this analysis. after collecting more data, we can further verify our results.

It is important to note that the in-game strategy features are derived directly from log data and are generated based on an unsupervised method, EM Clustering. This automated process of extracting students' in-game problem-solving strategy makes it a promising approach for evidence modeling. The approach can be readily used for evidence modeling design for learning environments that center on students solving problems by performing sequences of actions from a limited pool of available actions. However, the proposed approach is not appropriate for analyzing ill-defined problems where players are not bound to certain actions.

Evidence models such as those induced in this paper can be used by intelligent game-based learning environments to infer students' problem-solving strategies from trace data analysis. When the learning environments are signaled by the evidence models that a student is following a strategy associated with a poor learning outcome, it can intervene to guide students towards more productive strategies. In addition to strategy scaffolding, the evidence models can also work in tandem with knowledge modeling to support knowledge scaffolding. For example, in the ENGAGE game-based learning environment, students' generating a desired binary sequence through long series of flips and random trial-and-error might be an indicator of lack of knowledge about digit weights in a binary string, which could be addressed with a timely explanation of binary digit weights. The results of the work reported here, as well as those found in related work on inferring student problem-solving strategies from behavior trace data [18], suggest that modeling students' problem-solving strategies may contribute to improved assessment and also lead to learning environments that can adapt more effectively to students' needs.

7. CONCLUSION

Stealth assessment holds considerable potential for game-based learning. Although high volumes of dynamic student interaction data can be readily captured from game-based learning environments, effective stealth assessment poses significant challenges. We have introduced a strategy-based temporal analytics framework for stealth assessment that uses an LSTM-based evidence model trained on sequences of student problem-solving strategies learned from clustering *n*-gram representations of student in-game behaviors. In an evaluation of predictive accuracy for student learning, the strategy-based temporal analytics framework outperformed baseline models that did not capture the temporal dependencies of strategy use. In future work, it will be important to investigate multiple granularities of strategy representations that may lend themselves to hierarchical deep learning methods. It will also be instructive to incorporate the LSTM-based models into game-based learning environments to explore how they can provide classic stealth assessment functionalities while simultaneously supporting adaptive scaffolding.

8. ACKNOWLEDGMENTS

This research was supported by the National Science Foundation under Grants CNS-1138497 and DRL-1640141. Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

9. REFERENCES

[1] L. Breiman. Random forests. *Machine Learning*, *45*(1):5-32, 2001.

[2] P. Brusilovsky and E. Millán. User models for adaptive hypermedia and adaptive educational systems. *The Adaptive Web*, pages 3–53, 2007.

[3] P. Buffum, M. Frankosky, K. Boyer, E. Wiebe, B. Mott, and J. Lester. Collaboration and gender equity in game-based learning for middle school computer science. *IEEE Computing in Science and Engineering*, 18(2), 18-28, 2016.

[4] G. Chen, S. Gully, and D. Eden. Validation of a new general self-efficacy scale. *Organizational Research Methods*, 4(1):62–83, 2001

[5] M. Cheng, L. Rosenheck, C. Lin, and E. Klopfer. Analyzing gameplay data to inform feedback loops in the radix endeavor. *Computers & Education*, 111:60–73, 2017.

[6] F. Chollet. Keras. https://github.com/keras-team/keras, 2015

[7] D. Clark, E. Tanner-Smith, and S. Killingsworth. Digital games, design, and learning: A systematic review and meta-analysis. *Review of Educational Research*, 86(1):79–122, 2016.

[8] D. Cordova and M. Lepper. Intrinsic motivation and the process of learning: Beneficial effects of contextualization,

personalization, and choice. *Journal of Educational Psychology*, 88(4):715–730, 1996.

[9] C. Cortes and V. Vapnik, Support-vector networks. *Machine Learning*, *20*(3):273-297, 1995.

[10] M. d'Aquin and N. Jay. Interpreting data mining results with linked data for learning analytics: motivation, case study and directions. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, pages 155–164, 2013.

[11] M. Eagle and T. Barnes. Exploring differences in problem solving with data-driven approach maps. In *Proceedings of the 7th International Conference on Educational Data Mining*, pages 76–83, 2014.

[12] M. Falakmasir, J. Gonzalez-Brenes, G. Gordon, and K. DiCerbo. A data-driven approach for inferring student proficiency from game activity logs. In *Proceedings of the Third ACM Conference on Learning@ Scale*, pages 341–349. 2016.

[13] J. Harley, C. Carter, N. Papaionnou, F. Bouchet, R. Landis, R. Azevedo, and L. Karabachian. Examining the predictive relationship between personality and emotion traits and learners' agent-direct emotions. In *proceedings of the 7th International Conference on Artificial Intelligence in Education*, pages 145–154, 2015.

[14] S. Hochreiter, J. Schmidhuber. Long short-term memory. *Neural Computation*, *9*(8):1735–1780, 1997.

[15] G. Jackson and D. McNamara. Motivation and performance in a game-based intelligent tutoring system. *Journal of Educational Psychology*, 105(4):1036–1049, 2013.

[16] T. Käser, N. Hallinen, and D. Schwartz. Modeling exploration strategies to predict student performance within a learning environment and beyond. In *Proceedings of the 7th International Conference on Learning Analytics and Knowledge*, pages 31–40, 2017.

[17] M. Kebritchi, A. Hirumi, and H. Bai. The effects of modern mathematics computer games on mathematics achievement and class motivation. *Computers & Education*, 55(2):427–443, 2010.

[18] D. Kerr and G. Chung. Identifying key features of student performance in educational video games and simulations through cluster analysis. *Journal of Educational Data Mining*, 4(1):144–182, 2012.

[19] Y. Kim, R. Almond, and V. Shute. Applying evidence-centered design for the development of game-based assessments in physics playground. *International Journal of Testing*, 16(2):142–163, 2016

[20] J. Lester, E. Ha, S. Lee, B. Mott, J. Rowe, and J. Sabourin. Serious games get smart: Intelligent game-based learning environments. *Artificial Intelligence Magazine*, *34*(4):31–45, 2013.

[21] A. Liaw and M. Wiener. Classification and regression by randomForest. *R News*. 2(3):18–22, 2002.

[22] L. Malkiewich, R. Baker, V. Shute, S. Kai, and L. Paquette. Classifying behavior to elucidate elegant problem solving in an educational game. In *Proceedings of the 9th International Conference on Educational Data Mining*, pages 448–453, 2016.

[23] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and Friedrich Leisch. e1071: Misc functions of the Department of Statistics, Probability Theory Group, TU Wien, 2017.

[24] W. Min, M. Frankosky, B. Mott, E. Wiebe, K. Boyer, and J. Lester. DeepStealth: Leveraging deep learning models for stealth assessment in game-based learning environments. *In proceedings of the 7th International Conference on Artificial Intelligence in Education*, pages 277-286, 2015.

[25] W. Min, M. Frankosky, B. Mott, E. Wiebe, K. Boyer, and J. Lester. Inducing stealth assessors from game interaction data. In *proceedings of the 9th International Conference on Artificial Intelligence in Education*, pages 212-223, 2017.

[26] W. Min, B. W. Mott, J. P. Rowe, B. Liu, and J. C. Lester. Player goal recognition in open-world digital games with long short-term memory networks. In *proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 2590–2596, 2016.

[27] R. Mislevy, L. Steinberg, and R. Almond. Focus article: on the structure of educational assessments. *Measurement: Interdisciplinary Research and Perspectives*, 1(1):3–62, 2003.

[28] N. Nagappan, L. Williams, M. Ferzli, E. Wiebe, K. Yang, C. Miller, and S. Balik. Improving the CS1 experience with pair programming. *In Proceedings of 34th SIGCSE Technical Symposium*, pages 359–362, 2003.

[29] B. C. Nelson, Y. Kim, C. Foshee, and K. Slack. Visual signaling in virtual world-based assessments: The save science project. *Information Sciences*, 264:32–40, 2014.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, and J. Vanderplas. Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 2825-2830, 2011.

[31] E. Quellmalz, M. Timms, M. Silberglitt, and B. Buckley. Science assessments for all: Integrating science simulations into balanced state science assessment systems. *Journal of Research in Science Teaching*, 49(3):363–393, 2012.

[32] D. Quigley, J. Ostwald, and T. Sumner. Scientific modeling: using learning analytics to examine student practices and classroom variation. In *Proceedings of the 7th International Conference on Learning Analytics and Knowledge*, pages 329–338, 2017.

[33] E. Rowe, R. Baker, J. Asbell-Clarke, E. Kasman, and W. Hawkins. Building automated detectors of gameplay strategies to measure implicit science learning. In *Proceedings of the 7th International Conference on Educational Data Mining*, pages 337–338, 2014.

[34] J. Rowe, L. Shores, B. Mott, and J. Lester, Integrating learning, problem solving, and engagement in narrative-centered learning environments. *International Journal of Artificial Intelligence in Education*, *21*(1-2):115-133, 2011.

[35] A. Rupp, M. Gushta, R. Mislevy, and D. Shaffer. Evidence-centered design of epistemic games: Measurement principles for complex learning environments. *The Journal of Technology, Learning and Assessment*, 8(4), 2010.

[36] J. Sabourin and J. Lester. Affect and engagement in game-based learning environments. *IEEE Transactions on Affective Computing*, 5(1):45–56, 2014.

[37] S. Sahebi, Y. Huang, and P. Brusilovsky. Predicting student performance in solving parameterized exercises. In *proceedings of the 12th International Conference on Intelligent Tutoring Systems*, pages 496–503, 2014.

[38] V. Shute. Stealth assessment in computer-based games to support learning. *Computer Games and Instruction*, 55(2):503–524, 2011.

[39] V. Shute and M. Ventura. *Measuring and supporting learning in games: Stealth assessment*. The MIT press, 2013.

[40] P. Wouters, C. Van Nimwegen, H. Van Oostendorp, and E. Van Der Spek. A meta-analysis of the cognitive and motivational effects of serious games. *Journal of educational psychology*, *105*(2), 249–265, 2013.

[41] Z. Xing, J. Pei, and E. Keogh. A brief survey on sequence classification. *ACM SIGKDD Explorations Newsletter*, 12(1): 40–48,2010