Cross-System Transfer of Machine Learned and Knowledge Engineered Models of Gaming the System

Luc Paquette[1(✉)], Ryan S. Baker[1], Adriana de Carvalho[2], and Jaclyn Ocumpaugh[1]

[1] Teachers College, Columbia University, New York, NY, USA
{paquette,jo2424}@tc.columbia.edu,
baker2@exchange.tc.columbia.edu
[2] Google, New York, NY, USA

Abstract. Replicable research on the behavior known as gaming the system, in which students try to succeed by exploiting the functionalities of a learning environment instead of learning the material, has shown it is negatively correlated with learning outcomes. As such, many have developed models that can automatically detect gaming behaviors, towards deploying them in online learning environments. Both machine learning and knowledge engineering approaches have been used to create models for a variety of software systems, but the development of these models is often quite time consuming. In this paper, we investigate how well different kinds of models generalize across learning environments, specifically studying how effectively four gaming models previously created for the Cognitive Tutor Algebra tutoring system function when applied to data from two alternate learning environments: the scatterplot lesson of Cognitive Tutor Middle School and ASSISTments. Our results suggest that the similarity between the systems our model are transferred between and the nature of the approach used to create the model impact transfer to new systems.

Keywords: Gaming the system · Cognitive tutors · ASSISTments · Machine learning · Cognitive modeling · Cross-system transfer

1 Introduction

In intelligent tutoring systems (ITSs) and other learning environment, student disengagement often manifests in a behavior known as gaming the system. This behavior, which is neither clearly off-task nor on-task, is defined as "attempting to succeed in an educational environment by exploiting properties of the system rather than by learning the material and trying to use that knowledge to answer correctly" [1]. Research in multiple learning environments [1, 2, 3, 4, 5, 6, 7] has linked gaming to poor learning outcomes [8, 9, 10, 11], increased boredom [2] and lower long-term levels of academic attainment [12]. Both knowledge engineering [4, 5, 6, 7, 13, 14] and machine learning [1, 3, 7] approaches have been used to create models of gaming the system for specific learning environments. These detectors have been successfully applied (driving interventions) [15, 16] and used in discovery with models analyses [17], but only after a painstaking development process where researchers essentially start from scratch for each new learning environment.

F. Ricci et al. (Eds.): UMAP 2015, LNCS 9146, pp. 183–194, 2015.
DOI: 10.1007/978-3-319-20267-9_15

Little work has focused on the generalizability of gaming detectors. Nearly a decade ago, researchers showed that gaming detectors developed using machine learning could be generalized across different lessons in the same tutoring system [1]. Since that time, others have applied models to multiple learning systems [e.g., 6], but without validating that the model reliably captures gaming behavior across systems. Validating that some gaming detectors can be effectively applied across learning systems would facilitate broader use of gaming detectors. Currently, creating a model of gaming the system is a time consuming process that requires either field observations or the use of text replays, where log files of students interactions with the learning environment are segmented and formatted for presentation to a trained expert human coder who then labels whether each segment includes gaming behavior [18]. After this step, the feature engineering and model development process is time-consuming, complicated, and costly. The development of a model that allows us to skip (or reduce) the time-intensive development process for new learning systems would facilitate greater use of gaming detectors in intervention and analysis.

In this paper, we evaluate the generalizability of four recently published gaming models. Each were initially developed for use within an ITS known as Cognitive Tutor Algebra, but were constructed using different modeling techniques [19]. Specifically, we compare the generalizability of a detector developed with a form of knowledge engineering known as cognitive modeling [20] to that of three detectors that were developed by using machine learning to improve the cognitive model [21]. We assess each model's generalizability by comparing its performance when applied to two other ITSs for mathematics: the scatterplot lesson of Cognitive Tutor Middle School [22] (also called the scatterplot tutor) and ASSISTments [23]. Studying the generalizability of each model to these different systems allows us to investigate the degree to which each captures characteristics of the gaming construct that extend beyond a single system's features. We investigate how the nature of each system and the nature of each model interacts. More broadly, we discuss how the procedures used in this study might facilitate future development of detectors that generalize across learning environments, increasing their applicability and impact.

2 Gaming the System in Cognitive Tutor Algebra

2.1 Data

All four models of gaming behavior discussed in this paper were created using Cognitive Tutor Algebra (CTA) data [1, 21]. Specifically, these models were constructed from data produced by 59 students who used CTA as part of their regular mathematics curriculum throughout an entire school year (Pittsburgh Science of Learning Center DataShop "Algebra I 2005-2006 (Hampton only)" dataset [24]). The Cognitive Tutor environment presents students with complex mathematical problems that have been broken down into component steps, using a cognitive model of the task to assess whether answers correctly map to each step. Struggling students can request help at any time, and the tutor will provide increasingly specific, multi-step hints.

Data from 12 CTA lessons were segmented into short sequences of actions, called *clips*. For this study, clips were defined as sequences of at least 5 actions with a minimum duration of 20 seconds. If a 5-action sequence lasted less than 20 seconds, additional 5-action sequences were added to the clip until the total time duration was greater than 20 seconds. A total of 10,397 clips were randomly selected from this dataset; the chance of a clip being selected was weighted for each lesson based on the total number of clips in that lesson, so that each of the lessons in CTA were equally represented.

These clips were then presented to an expert coder (the 3rd author, henceforth expert #1), who had previously reached an interrater reliability with another expert (Cohen's Kappa > 0.6) through an extensive training process [22]. Cohen's Kappa [26] assesses the degree to which agreement between the expert is better than chance. A Kappa of 0 indicates chance level agreement and a Kappa of 1 indicates perfect agreement.

The clips were presented to expert #1 in the form of *text replays* [25, 18], where each clip is presented in a layout that highlights the relevant information needed to code student behaviors in a contextualized manner. Fig. 1 shows an example of a text replay taken from our CTA data. It displays each action's time (relative to the 1st action in the clip), the step's context, the input entered, the relevant skill (production) and the system's assessment of the action (a right or wrong answer, a help request or a predicted wrong answer, called a "bug"). In this way coders can quickly assess the actions in each clip to determine whether the student's interactions suggest gaming behaviors. In this case, the coder classified 6.81% (708) of clips as involving gaming the system behaviors and 93.19% (9,689) as not gaming.

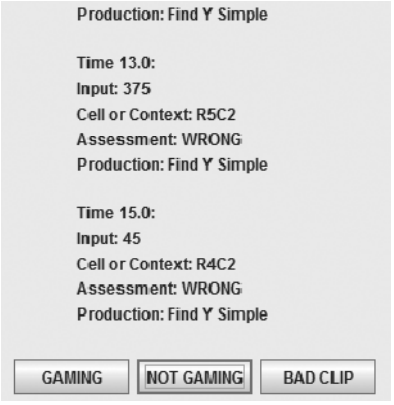

Fig. 1. The last actions of a 5-action text replay clip

2.2 Cognitive Model

The first step toward developing a generalizable model of gaming the system was to achieve a deeper understanding of how experts think about this construct [20]. To this end, we conducted a cognitive task analysis [27, 28] of expert #1's coding task using

text replays from CTA. This involved a combination of active participation [28, 29] (in which the person performing the cognitive task analysis actively participated in the coding of text replays), think aloud observations [30] and interviews to explicate the coding process. Results indicated that the expert's coding method could be classified into two cognitive processes: interpreting the student's individual actions and identifying patterns of gaming across those actions. Although the expert executes these in parallel, our resulting cognitive model executes these as consecutive steps without changing the fundamental reasoning process.

In [1], we modeled expert #1's assessment of individual actions by constructing a list of 19 *constituents*—meaningful units of student behavior that the expert regularly relies upon when making gaming judgments, sometimes across more than one action. For example, the expert interprets a *short pause* between actions as a [*guess*], which is likely to indicate gaming behavior. She is similarly suspicious when a student enters a [*similar answer*] in two consecutive actions. (A detailed list of the 19 constituents we identified is available in [1].) Next, we developed a set of 13 *action patterns*, composed of 2-4 actions, each matching a predefined set of gaming constituents [1]. For example, consider a 2-action sequence where a student's 1st answer is incorrect. If they quickly attempt to answer a different problem step using the same response, the expert is likely to interpret the second action as a [*guess*], which she is trained to recognize as typical gaming behavior. This pattern was modeled as the following sequences of actions and constituents: **incorrect** → [*guess*] & [*same answer/diff. context*] & **incorrect**. In our *Cognitive* model of gaming the system, any clip containing actions that matching these 13 patterns was given a gaming label. These patterns were validated (Kappa = 0.330) on a test set, composed of 25% of the data from 2.1, which was held out during the cognitive task analysis [1]. This *Cognitive* model performed better on new data than a machine-learned model previously built for CTA (Kappa = 0.24) [25].

2.3 Machine Learned Models

In this paper we compare generalizability of the knowledge-engineered model we developed through cognitive modeling [20] to three machine-learned models derived from it [21]. The latter combine the *Cognitive* model's constituents into new patterns that were not apparent to human experts. More specifically, we developed an algorithm to generate and filter a large number of action patterns (similar to the one presented above) from the constituents identified in the *Cognitive* model, selecting those that best detected gaming behaviors, which we termed *pattern features.* We combined these with what we called *count features,* which tallied the number of times each of the 25 action types and constituents were present in a given clip.

Naïve Bayes classification was then used to generate three cross-validated, machine-learned models. The first, *CognitiveHybrid-PF*, contained 22 features, including 20 *pattern features* and 2 *count features* (Kappa = 0.477, A' = 0.770). The second, *CognitiveHybrid-C*, was trained using only the *count features*. The resulting model contained 6 features (Kappa = 0.332, A' = 0.875). The performance differences between these models reflect the nature of their features. Pattern features better capture the sequential nature of gaming behaviors, improving that model's Kappa, but their

binary nature prevents it from achieving a high A'. Likewise, the gradient nature of the count features improves that modes' A' performance despite its lower Kappa. For this reason, we created a third model, *CognitiveHybrid-E*, that *Ensembled* the predictions from both these models. *CognitiveHybrid-E* achieves a high performance for both Kappa (0.457) and A' (0.901).

3 Evaluation of Cross-System Transfer

Because the four CTA gaming models were constructed from features that closely match the expert's conception of gaming, we hypothesized that they might capture characteristics of gaming that extend beyond system-specific behaviors, thereby allowing them to generalize to new systems. In this section we report on the performance of each model when it was applied, without additional training or modification, to data collected from the scatterplot lesson of Cognitive Tutor Middle School and ASSISTments.

3.1 Transfer to the Scatterplot Lesson of Cognitive Tutor Middle School

The 1st system we attempted to transfer our gaming models to was the scatterplot lesson of Cognitive Tutor Middle School [19]. This system was built using the same platform as CTA, meaning that their hint features, bug messages, and common interfaces (e.g., virtual problem worksheets) are quite similar. The primary differences are in the mathematical domain taught by each, which result in some interface differences. For example, the scatterplot lesson has additional interfaces for creating data representations (e.g., histograms) but lacks some of the complex algebraic equation manipulation seen in CTA. Nonetheless, given their similarities, we assumed transfer would be more successful for this system than ASSISTments (discussed below).

Data. The data for evaluating the transfer of the CTA models to the scatterplot lesson of Cognitive Tutor Middle School system was taken from a study of interrater agreement of experts coding text replays [18]. As such, we had access to gaming labels from 2 coders, neither of whom is expert #1, who provided the labels for CTA. These coders coded the same 600 clips (with some disagreement on which text replays constituted "bad clips," which did not get coded). We used these experts' gaming labels to develop three datasets for the scatterplot tutor. The first dataset (N = 595) was coded by expert #2, who labeled 29 as gaming and 566 as not. The second dataset (N = 592) was coded by expert #3, who labeled 33 as gaming and 559 as not. Finally, a third "Agreement" dataset is composed of all clips that were consistently coded by both experts (N=571), which includes 19 labeled as gaming and 552 labeled as not.

One important difference between the text replays from the scatterplot tutor and CTA is in the definition of a clip. Although clips from both systems have at least 20 seconds of data, the scatterplot tutor clips do not enforce a minimum number of actions (so may contain fewer than 5). Since gaming the system is a systematic pattern of behaviors, this could make it more challenging for experts to identify gaming actions within the scatterplot tutor clip.

Performance. We applied each of the four CTA gaming models to the three datasets from the scatterplot lesson of Cognitive Tutor Middle School. Table 1 summarizes the performance achieved by each on the new datasets, using both the Kappa and the A' [31] metrics. A' is the probability that given a pair of two clips, one coded as gaming and the other coded as not-gaming, the model can accurately detect which clip was coded as gaming. A' is equivalent to the area under the ROC curve in signal detection theory [31]. Note that A' could not be calculated for the *Cognitive* model, which does not produce confidences.

Table 1. Comparison of the models performance across the Cognitive Tutor Algebra (CTA) and the scatterplot lesson of Cognitive Tutor Middle School datasets

Model	*Cognitive Tutor Algebra (CTA)*	*Scatterplot Expert #2*	*Scatterplot Expert #3*	*Scatterplot Agreement*
Cognitive	Kappa = 0.330 A' = N/A	Kappa = 0.459 A' = N/A	Kappa = 0.479 A' = N/A	Kappa = 0.483 A' = N/A
CogHybrid-PF	Kappa = 0.477 A' = 0.770	Kappa = 0.440 A' = 0.819	Kappa = 0.438 A' = 0.795	Kappa = 0.451 A' = 0.877
CogHybrid-C	Kappa = 0.332 A' = 0.875	Kappa = 0.345 A' = 0.894	Kappa = 0.360 A' = 0.889	Kappa = 0.331 A' = 0.949
CogHybrid-E	Kappa = 0.457 A' = 0.901	Kappa = 0.430 A' = 0.917	Kappa = 0.427 A' = 0.905	Kappa = 0.438 A' = 0.973

All four models transfer well to the scatterplot lesson of Cognitive Tutor Middle School despite differences in the mathematical domain, in the definition of a clip, and among the experts who labeled the data. The most notable finding was that, when applied to the scatterplot tutor data, the *Cognitive* model achieved a higher Kappa than the machine-learned models. Surprisingly, the *Cognitive* model achieved considerably higher performance for the scatterplot tutor than for CTA. The machine-learned models also transfer well, achieving performance metrics for the scatterplot tutor datasets that are comparable to those for the CTA data. *CogHybrid-PF* obtains a slightly lower Kappa on each of the three scatterplot datasets, whereas its A' performance actually increased. Performance for *CogHybrid-C* slightly increases for both metrics. The ensemble model, *CogHybrid-E*, achieves slightly lower Kappa for the scatterplot datasets than for CTA, similar A' for both experts, and an increase in A' for the Agreement set.

In general, models transfer best to the Agreement dataset. Kappa performance is slightly higher for three of the four models (*Cognitive*, *CogHybrid-PF* and *CogHybrid-E*), and A' performance is considerably greater. This may suggest that the text replays that showed expert disagreement were contributing significant noise to the datasets.

3.2 Transfer to ASSISTments

We also studied the degree to which these models could generalize to ASSISTments [23], a web-based tutoring system for middle school mathematics. Two models of gaming the system have previously been developed for ASSISTments. The first, published in 2006 [7], achieved a Kappa of 0.181 for new data. The second, published in 2013 [11], achieved a Kappa of 0.370 and A' of 0.802 [11].

Although both ASSISTments and CTA are problem-solving intelligent tutors for mathematics, the structure and presentation of problems in ASSISTments is quite different from either of the Cognitive Tutor platforms. In Cognitive Tutor, problems are partitioned in multiple steps that must be completed to finish each problem. The Cognitive Tutor indicates whether each step is right or wrong, providing assistance as necessary. In ASSISTments, when students are presented with an "original" problem, they only need to provide its final answer. Individual steps are not required of students who solve the problem on the first attempt. However, students who do not provide the correct answer may be required to correctly answer scaffolding questions in order to successfully complete the problem.

At the same time, there are considerable similarities between the two systems. Both provide immediate feedback indicting whether their answer was right or wrong, including detailed feedback for "bugs," where the student's error indicates a known misconception. ASSISTments also offers help functionality similar to that found on the Cognitive Tutor platforms.

Data. ASSISTments data produced by 1,367 students was used to test the generalizability of our CTA gaming models. This data includes a total of 822,233 problem-solving steps, which were segmented into 240,450 clips. A selection of these (discussed below) was presented to expert #1 for coding.

Again the definition of a clip for the new system was different than it was for CTA, this time because of differences in how the systems present problems to students. Whereas Cognitive Tutor platform always requires students to solve multiple steps before completing a main problem, ASSISTments problems can be solved in one step if the student's first attempt is correct. As such, we define a clip in ASSISTments as starting from the first action on an original unscaffolded problem to the last attempt before the next original, unscaffolded problem. This definition means that a clip can be composed of only 1 action or it can contain more than 50.

As such, when selecting clips for coding by expert #1 (who also coded the CTA data), we filtered them with respect to length. Clips containing more than 25 actions were removed from the dataset because it was difficult to present such a large number of actions in a text replay. We also felt that it was unlikely that the constituent behaviors (e.g., [20]) that comprise gaming behaviors would be different in clips containing 40 or 50 actions than they would be in clips containing 20-25. On the other hand, if there were longer action patterns present in these clips, which comprised less than 0.7% of the dataset, it could create a serious bias towards a different gaming pattern that was being identified by the expert.

Shorter clips (fewer than 3 actions) presented a different problem. Because gaming is a systematic pattern of behavior that often occurs among students who do not understand the material, it is unlikely to be seen in clips where students solve the problem correctly in very few attempts [20, 21]. On the other hand, shorter clips comprise 73.0% of the original data, so filtering them entirely could have biased the coding.

As such, we created two datasets for ASSISTments. Sample #1 was comprised of 1,000 clips with 1-25 actions each, so that clip length distribution closely matches the original dataset. Sample #2 was comprised of 1,000 clips with 4-25 actions; this

allowance was intended to increase the odds that clips containing gaming behaviors would be presented to the coder.

As in [1], clips were presented to expert #1 in the form of text replays. The order in which the expert coded these replays was randomized, mixing clips from both samples in order to avoid coding biases. A technical glitch kept the expert from completing all 2000 clips (with the replay software hanging after 1063 codes). Randomization ensured that the two samples had balanced numbers ($N_1 = 520$, $N_2 = 543$). The expert identified gaming behaviors in 3.46% of sample #1 and 8.47% of sample #2 ($N_1 = 18$, $N_2 = 46$), in line with the design of the latter, which increased the odds of drawing clips which contain this systematic behavior. Combining samples provided 64 gaming clips (6.02%) 996 non-gaming clips (93.70%) and 3 clips where the replay software or data had an error (0.28%).

Performance. Each of the four models of gaming the system were applied to sample #1, sample #2, and the two samples *combined*. As Table 2 shows, each performed above chance when applied to the ASSISTments datasets, but both Kappa and A' were considerably lower than when these models were applied to either of the Cognitive Tutor platforms.

Performance of the *Cognitive* model was similar across all three ASSISTments datasets (Kappa = 0.228-0.256), and as with the scatterplot lesson of Cognitive Tutor Middle School, it generally outperformed the machine-learned models. This difference seems to be driven primarily by the performance of the detectors on sample #1, which was far more likely to contain shorter clips than sample #2. Although the *Cognitive* model performed slightly worse on sample 1 than it had on the CTA data (Kappa = 0.228 compared to Kappa = 0.330), the Kappa values for the machine-learned models were quite low for this sample (Kappa = 0.075-0.124).

Table 2. Comparison of the models performance across the Cognitive Tutor Algebra (CTA) and ASSISTments datasets

Model	*Cognitive Tutor Algebra (CTA)*	*ASSISTments Sample #1*	*ASSISTments Sample #2*	*ASSISTments Combined*
Cognitive	Kappa = 0.330 A' = N/A	Kappa = 0.228 A' = N/A	Kappa = 0.240 A' = N/A	Kappa = 0.256 A' = N/A
CogHybrid-PF	Kappa = 0.477 A' = 0.770	Kappa = 0.075 A' = 0.565	Kappa = 0.285 A' = 0.694	Kappa = 0.248 A' = 0.665
CogHybrid-C	Kappa = 0.332 A' = 0.875	Kappa = 0.124 A' = 0.890	Kappa = 0.156 A' = 0.763	Kappa = 0.173 A' = 0.810
CogHybrid-E	Kappa = 0.457 A' = 0.901	Kappa = 0.121 A' = 0.892	Kappa = 0.246 A' = 0.803	Kappa = 0.235 A' = 0.829

Among the machine-learned models, Kappa performance was variable. *CogHybrid-PF's* performance was unstable across the three data sets. Performance on sample #2 and the combined dataset was acceptable (Kappa = 0.285, Kappa = 0.248), if lower than its performance on either of the Cognitive Tutor systems. However,

performance on sample #1, where the number of actions per clip most closely matches what is typical in ASSISTments, was low (Kappa = 0.075). *CogHybrid-C* achieved relatively poor but more stable performance across the 3 datasets (Kappa = 0.124-0.173). Kappa results for the ensemble model, *CogHybrid-E*, were similar to *CogHybrid-PF*, performing more poorly on sample #1 than on the other 2 datasets. Its A' values were closely aligned to those of *CogHybrid-C*.

The A' performance of these models (which was not calculated for the *Cognitive* model) was more promising. *CogHybrid-PF's* performance, which had been low even on the CTA data, was particularly low for Sample #1, but other A' values were surprisingly high. For instance, *CogHybrid-C*, which was constructed by counting the number of gaming constituents (action sequences) that were present in a clip had an A' for Sample #1 that was higher than for CTA, and *CogHybrid-E*, had an A' for Sample #1 that was nearly as high as the one it achieved for CTA.

Although performance decreased when transferring from CTA to ASSISTments, our models still performed substantially above chance (Kappa = 0.075-0.256 and A' = 0.665-0.829). Their performance is also comparable to previous models of gaming the system developed specifically for the ASSISTments system. For Kappa, performance was moderately worse than the model in [11] (0.370), but, except for *CogHybrid-C*, our models performed better than that in [7] (0.181). Likewise, one of the previously published gaming model for ASSISTments achieved an A' of 0.802 [11], and two of our machine learned models outperform that. (Note that [7] did not report A'.)

4 Discussion and Conclusions

This paper examines the performance of gaming the system models previously developed for Cognitive Tutor Algebra (CTA) to determine the degree to which each can be reliably applied to other systems without additional modifications. To this end, we compared the degree to which these models, which had performed well on the CTA data used to train them, could transfer to two other intelligent tutoring systems that provide similar mathematics instruction: (1) the scatterplot lesson of Cognitive Tutor Middle School, which shares many of the same design features as CTA and (2), ASSISTments, which has more design differences, but covers content that is similar to that provided in CTA.

The first model we tested was developed using a cognitive task analysis to model expert judgments about gaming behaviors [20], while the other 3 models were developed using machine learning techniques to improve the first [21]. That is, machine learning was used to discover patterns in the behavioral constituents that were tacitly used by the expert human coders who provide the training labels for gaming detectors. It was hypothesized that since all four models were developed from features that map to the expert's coding process (rather than the features that were more specific to the learning system), they would be more likely to transfer.

Results from our study indicate, perhaps unsurprisingly, that learning system differences affected how well models were able to generalize. Models performed better when transferred to the scatterplot lesson of Cognitive Tutor Middle School, which

shares many design features with CTA than when transferred to ASSISTments, which has a more distinctive interface. It is likely that these interface differences lead students using ASSISTments to adopt different gaming strategies than those used in the Cognitive Tutor platforms.

Other results were less predictable. Our findings suggest that the machine learning techniques used by [21] to improve performance within CTA may not be necessary for the development of generalizable detectors. That is, the knowledge engineering model (the *Cognitive* model), when applied to new systems, performed as well as or outperformed the three machine learned models on the Kappa metric. This suggests that machine-learning, which optimizes a model's performance on the system it was trained for, may overfit to the specific system.

On the other hand, the results for *CognitiveHybrid-E* model suggests that this finding warrants further research. This model—which ensembled the confidence values of the machine learned models produced with pattern features and with count features—showed superior performance on the original CTA data set and performed almost as well as the *Cognitive* model on the new systems. What's more, the strong A' performance of the machine learned models (two of which were comparable to previously published models of gaming that were developed specifically for ASSISTments) also suggests that this method deserves further consideration. That said, the *Cognitive* model's performance shows substantial stability across systems, an important consideration, particularly for models that will be used to trigger interventions.

Although our models transferred reasonably well to two new learning systems, both were designed for the tutoring of mathematics problem. The current study does not present evidences that these gaming detectors will transfer to mathematics tutors with different intervention strategies or to tutors that provide instruction in other domains. Such differences are likely to trigger different gaming strategies, making this an important area for future research.

Acknowledgments. This research was supported by a Fonds de Recherche du Québec - Nature et Technologies (FRQNT) post-doctoral fellowship and by National Science Foundation (NSF) grant #SBE-0836012.

References

1. de Baker, R.S.J., Corbett, A.T., Roll, I., Koedinger, K.R.: Developing a Generalizable Detector of When Students Games the System. User Modeling & User Adapted Interaction **18**, 287–314 (2008)
2. de Baker, R.S.J., D'Mello, S.K., Rodrigo, M.M.T., Graesser, A.C.: Better to be Frustrated than Bored: The Incidence, Persistence, and Impact of Learners' Cognitive-Affective States During Interactions with Three Different Computer-Based Learning Environments. Int'l Journal of Human-Computer Studies **68**, 223–241 (2010)
3. de Baker, R.S.J., Mitrović, A., Mathews, M.: Detecting gaming the system in constraint-based tutors. In: De Bra, P., Kobsa, A., Chin, D. (eds.) UMAP 2010. LNCS, vol. 6075, pp. 267–278. Springer, Heidelberg (2010)

4. Beal, C.R., Qu, L., Lee, H.: Classifying learner engagement through integration of multiple data sources. In: Proc. of the National Conf. on Artificial Intelligence, pp. 151–156 (2006)
5. Johns, J., Woolf, B.: A dynamic mixture model to detect student motivation and proficiency. In: Proc. of the National Conference on Artificial Intelligence, pp. 163–168 (2006)
6. Muldner, K., Burleson, W., Van de Sande, B., VanLehn, K.: An Analysis of Students' Gaming Behaviors in an Intelligent Tutoring System: Predictors and Impact. User Modeling and User Adapted Interaction **21**, 99–135 (2011)
7. Walonoski, J.A., Heffernan, N.T.: Detection and analysis of off-task gaming behavior in intelligent tutoring systems. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 382–391. Springer, Heidelberg (2006)
8. Beck, J., Rodrigo, M.T.: Understanding wheel spinning in the context of affective factors. In: Trausan-Matu, S., Boyer, K.E., Crosby, M., Panourgia, K. (eds.) ITS 2014. LNCS, vol. 8474, pp. 162–167. Springer, Heidelberg (2014)
9. Cocea, M., Hershkovitz, A., de Baker, R.S.J.: The impact of off-task and gaming behaviors on learning: immediate or aggregate? In: Proc. of the 14th Int'l Conference on Artificial Intelligence in Education, pp. 507–514 (2009)
10. Fancsali, S.E.: Data-Driven Causal Modeling of "Gaming the System" and Off-Task Behavior in Cognitive Tutor Algebra. NIPS Workshop on Data Driven Education
11. Pardos, Z.A., Baker, R.S., San Pedro, M.O.C.Z., Gowda, S.M., Gowda, S.M.: Affective States and State Tests: Investigating how Affect and Engagement During the School Year Predict End of Year Learning Outcomes. J. of Learning Analytics **1**(1), 107–128 (2014)
12. San Pedro, M.O.Z., de Baker, R.S.J., Bowers, A.J., Heffernan, N.T.: Predicting college enrolment from student interaction with an intelligent tutoring system in middle school. In: Proc. of the 6th Int'l Conference on Educational Data Mining, pp. 177–184 (2013)
13. Aleven, V., McLaren, B.M., Roll, I., Koedinger, K.R.: Towards Meta-Cognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor. Int'l J. of Artificial Intelligence in Education **16**, 101–130 (2006)
14. Gong, Y., Beck, J.E., Heffernan, N.T., Forbes-Summers, E.: The fine-grained impact of gaming on learning. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 194–203. Springer, Heidelberg (2010)
15. Arroyo, I., et al.: Repairing disengagement with non-invasive interventions. In: Proc. of the 13th Int'l Conference on Artificial Intelligence in Education, pp. 195–202 (2007)
16. Walonoski, J.A., Heffernan, N.T.: Prevention of off-task gaming behavior in intelligent tutoring systems. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 722–724. Springer, Heidelberg (2006)
17. de Baker, R.S.J., Yacef, K.: The State of Educational Data Mining in 2009: A Review and Future Visions. Journal of Educational Data Mining **1**(1), 3–17 (2009)
18. de Baker, R.S.J., Corbett, A.T., Wagner, A.Z.: Human classification of low-fidelity replays of student actions. In: Proc. of the Educational Data Mining Workshop at Intelligent Tutoring System 2006, pp. 29–36 (2006)
19. Koedinger, K.R., Corbett, A.T.: Cognitive tutors: technology bringing learning sciences to the classroom. In: Sawyer, R.K. (ed.) The Cambridge Handbook of the Learning Sciences, pp. 61–77 (2006)
20. Paquette, L., de Carvalho, A.M.J.A., Ryan, S.B.: Towards understanding export coding of student disengagement in online learning. In: Proc. of the 36th Annual Cognitive Science Conference, pp. 1126–1131 (2014)
21. Paquette, L., de Carvalho, A.M.J.A., Ryan, S.B., Ocumpaugh, J.: Reengineering the feature distillation process: a case study in the detection of gaming the system. In: Proc. of the 7th Int'l Conference on Educational Data Mining, pp. 284–287 (2014)

22. Baker, R.S., Corbett, A.T., Koedinger, K.R.: Learning to distinguish between representations of data: a cognitive tutor that uses contrasting cases. In: Proc. of the International Conference of the Learning Sciences, pp. 58–65 (2004)
23. Razzaq, L., et al.: The assistment project: blending assessment and assisting. In: Proc. of the 12 Annual Conference on Artificial Intelligence in Education, pp. 555–562 (2005)
24. Koedinger, K.R., et al.: A Data Repository for the Community: The PLSC DataShop (2010)
25. de Baker, R.S.J., de Carvalho, A.M.J.A.: Labeling student behavior faster & more precisely with text replays. In: Proc. of the 1st Int'l Conf. on Educational Data Mining 2008, pp. 38–47 (2008)
26. Cohen, J.: A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement **20**(1), 37–46 (1960)
27. Clark, R.E., Feldon, D., van Merriënboer, J., Yates, K., Early, S.: Cognitive task analysis. In: Spector, J.M., Merrill, M.D., van Merriënboer, J.J.G., Driscoll, M.P. (eds.) Handbook of Research on Educational Communications and Technology, 3rd edn., pp. 575–593 (2008)
28. Cooke, N.J.: Varieties of Knowledge Elicitation Techniques. Int'l Journal of Human-Computer Studies **41**, 801–849 (1994)
29. Meyer, M.A.: How to Apply the Anthropological Technique of Participant Observation to Knowledge Acquisition for Expert Systems. IEEE Transactions on Systems, Man, & Cybernetics **22**, 983–991 (1992)
30. Van Someren, M.W., Barnard, Y.F., Sandberg, J.A.C.: The Think Aloud Method: A Practical Guide to Modeling Cognitive Processes (1994)
31. Hanley, J., McNeil, B.: The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve. Radiology **143**, 29–36 (1982)