

Automatic Detection of Off-Task Behaviors in Intelligent Tutoring Systems with Machine Learning Techniques

Suleyman Cetintas, Luo Si, Yan Ping Xin and Casey Hord

Abstract— Identifying off-task behaviors in intelligent tutoring systems is a practical and challenging research topic. This paper proposes a machine learning model that can automatically detect students' off-task behaviors. The proposed model only utilizes the data available from the log files that record students' actions within the system. The model utilizes a set of time features, performance features and mouse movement features and is compared to i) a model that only utilizes time features and ii) a model that uses time and performance features. Different students have different types of behaviors; therefore personalized version of the proposed model is constructed and compared to the corresponding non-personalized version. In order to address data sparseness problem, a robust Ridge Regression algorithm is utilized to estimate model parameters. An extensive set of experiment results demonstrate the power of using multiple types of evidence, the personalized model and the robust Ridge Regression algorithm.

Index Terms—K.3.1 Computer Uses in Education, N.4 Adaptive and intelligent educational systems

1 INTRODUCTION

AN increasing trend in computers' utilization for teaching has led to the development of many intelligent tutoring systems (ITSs). ITSs have been shown to increase students' involvement and effort in the classroom [26] as well as improve students' learning [16]. However students' misuse or lack of motivation can reverse the positive effect of ITSs. Therefore there have been considerable efforts to model and understand the behaviors of students while they use the system [3], [4], [7], [27]. The vast majority of the prior work focused on the interaction between students and the tutoring environments within the software which is called "gaming the system". Gaming behavior is performed by systematically taking advantage of properties, hints, or regularities in a system to complete a task in order to finish the curriculum rather than think about the material and only happens when a student is working with a system. However students' behavior outside the system may also affect the learning opportunities that ITSs provide.

Off-task behavior means students' attention becomes lost and they engage in activities that neither have anything to do with the tutoring system nor include any

learning aim. Surfing the web, devoting time to off-topic readings, talking with other students without any learning aims [3], and preventing other students' learning [28] are among typical off-task behavior examples.

Although off-task and gaming the system behaviors are quite different in nature, it has been noted by Baker that off-task behaviors are associated with deep motivational problems that also lead to gaming the system behaviors. Baker also suggested that these behaviors should be carefully studied together during system design since decreasing off-task behaviors can lead to an increase in gaming behaviors especially in the case of immediately warning a student to cease an off-task behavior [2]. Long term solutions, rather than immediate warnings (e.g. students' self monitoring) have been shown to decrease the off-task behaviors in traditional classrooms [11]. For intelligent tutoring systems, increasing the challenge and giving rewards for quickly solving problems without exhibiting gaming the system behavior have been suggested to decrease off-task behaviors [2].

Off-task behaviors occur not only in educational systems but in various types of interactive systems that require continued attention and engagement of a user. As noted in [2], driving a car is among such technology supported tasks. Being able to detect when users of such systems are not paying necessary attention to their tasks might also make these systems more effective, increase security, labor quality, etc.

Detection of student's off-task behavior in environments where it's not practical to utilize equipment such as microphones, gaze trackers, etc., is a challenging task [2]. Utilization of such equipment would provide instructional systems with audio and video data (e.g.

- Suleyman Cetintas is with the Department of Computer Sciences, Purdue University, West Lafayette, IN, 47907, USA (e-mail: scetinta@cs.purdue.edu).
- Luo Si is with the Department of Computer Sciences, Purdue University, West Lafayette, IN, 47907, USA (e-mail: lsi@cs.purdue.edu).
- Yan Ping Xin is with the Department of Educational Studies, Purdue University, West Lafayette, IN, 47907, USA (e-mail: yxin@purdue.edu).
- Casey Hord is with the Department of Educational Studies, Purdue University, West Lafayette, IN, 47907, USA (e-mail: thord@purdue.edu).

Manuscript received (October 2009).

xxxx-xxxx/0x/\$xx.00 © 200x IEEE

TABLE 1
DETAILS OF OBSERVED DATA FOR ON-TASK AND OFF-TASK BEHAVIORS

Observation Type	Number of Total Observations	# of Observations Per Student	
		Average	Std. Deviation
On-task	273	22.75	3.5194
Off-task	134	11.16	2.6912

Average number of observations per student can be seen under the Average sub-column of the # of Observations Per Student column and the standard deviation of the number of observations among students can be seen under the Std. Deviation sub-column.

facial cues and postures) [12], and there has been some research incorporating this information into the instructional systems [19]; [21]; [23]. This type of data would make tasks such as off-task detection relatively easier; however, since most K-12 schools are not equipped with the equipment to collect such data, systems must detect off-task behaviors using only the data from students' actions within the tutoring software. This brings its own challenges since it has been found that understanding students' intentions only from their actions within a system can be challenging [25]. Yet, off-task behavior detectors, especially personalized off-task detectors that consider the inter-user behavior variability (e.g. using more or less time to solve the problems; having difficulties with particular types of questions and/or problems or different mouse usage types, etc.) have the potential of improving students' learning experiences.

To the best of our knowledge, there is very limited prior work on the automatic detection of off-task behavior utilizing mouse movement data [10], [13] and there is no prior work on the personalized detection of off-task behaviors with multi-feature models. Prior work mainly focuses on detecting gaming behaviors [3], [4], [7], [27] which is also a quite different task from detecting off-task behaviors and none of researchers in these works utilized mouse movement data. Other prior work that focused on off-task behavior detection only analyzed time and performance features (that can be extracted from the logs of user-system interaction) and did not utilize mouse movement data [2]. One of the works using mouse movement data was done by De Vicente and Pain. They had human participants use mouse movement data as well as data from student-tutor interactions for motivation diagnosis, but did not use it for automated detection of off-task behaviors [13]. Cetintas et al. tracked and analyzed mouse movement data along with performance and time data to automatically detect off-task behaviors; however, these researchers did not consider personalization [10]. In another work, Cetintas et al. also used mouse movement data along with performance, problem and time features to predict the correctness of problem solving, which is a very different task from off-task detection [9].

Although time and performance features are useful for improving the effectiveness of off-task behavior detection, the vast majority of these works ignore: i) an important type of data, namely mouse tracking data, which can be easily stored in and retrieved from user-system logs.

Furthermore, all prior work ignores ii) the approach of personalization to capture different characteristics of students' which can lead to different behavior types, that are hard to identify with non-personalized off-task detectors since these models can't recognize inter-user variability for different behavior types.

This paper proposes a machine learning model that can automatically identify the off-task behaviors of a student by utilizing multiple types of evidence including time, performance and mouse movement features and by utilizing personalization to capture inter-user behavior variability. To address data sparseness problem, the proposed model utilizes a robust ridge regression technique to estimate model parameters. The proposed model is compared to i) a model that only utilizes time features; ii) a model that uses time and performance features. Furthermore all models are compared to each other a) when personalization is not used and b) when data sparseness problem is not addressed (i.e., when model parameters are not learned with Ridge Regression). We show that utilizing multiple types of evidence, personalization and the robust Ridge Regression technique improves the effectiveness of off-task detection.

The rest of the paper is arranged as follows: Section 2 introduces the dataset used in this work. Section 3 describes the Least Squares and Ridge Regression techniques. Section 4 proposes several approaches for modeling off-task behaviors as well as the personalization of those modeling approaches. Section 5 discusses the experimental methodology. Section 6 presents the experiment results; and finally Section 7 concludes this work.

2 DATA

Data from a study conducted in 2008 in an elementary school was used in this work. The study was conducted in mathematics classrooms using a math tutoring software (that had been developed by the authors). The tutoring software taught problem solving skills for Equal Group and Multiplicative Compare problems. These two problem types are a subset of the most important mathematical word problem types that represent about 78% of the problems in fourth grade mathematics textbooks [20]. First, in the tutoring system, a conceptual instruction session is studied by the students followed by problem solving sections to test their understanding. Both

the conceptual instruction and the problem solving parts require students to work one-on-one with the tutoring software and if students fail to pass a problem solving session, they have to repeat the corresponding conceptual instruction and the problem solving session. The tutoring software has a total of 4 conceptual instruction sessions and 11 problem solving sessions that have 12 questions each. The problem solving sessions include 4 sessions for Equal Group worksheets, 4 sessions for Multiplicative Compare worksheets and 3 sessions for Mixed worksheets, each of which include 6 EG & 6 MC problems. The tutoring software is supported with animations, audio (with more than 500 audio files), instructional hints, exercises, etc.

The study included 12 students that include 4 students with learning disabilities, 1 student with emotional disorder and 1 student with emotional disorder combined with a mild intellectual disability. Students used the tutor for several sessions which last about 30 minutes. Students used the tutor for an average of 18.2500 sessions with standard deviation of 3.3878 sessions. The evidence about students' on and off task behaviors was observed during these sessions. Outside observations of behavior were used for collecting the evidence. Self-report was not used to assess students' on- and off-task behaviors due to the concern that this measure might influence students' off-task behaviors and learning outcomes. A specific student is coded as either on-task or off-task by the observer as in past observatory studies of on-task and off-task behavior [15], [17], [18].

Most of the observations were carried out by a single observer. An observation includes watching a student solve a problem (i.e., starting from the time s/he starts to solve it until s/he submits her/his final answer). In an observation, if a student has been seen in an off-task activity such as i) talking with another student about anything other than the subject material, ii) inactivity (e.g., putting her/his head on the desk, staring into space, etc.), iii) off-task solitary behavior (i.e., any activity that doesn't involve the tutoring system or another student such as surfing the web) for more than 30 seconds (for any of these 3 types of behaviors), the behavior was coded as off-task. Brief reflective pauses and "gaming the system" behaviors were not treated as off-task behaviors. In order to avoid any observer bias, students were observed sequentially. Synchronization of field observations (which included the information about the problem number, student, time and date) and the log data is straightforward since the log data includes the date and time pairs for every student action such as mouse movements, answer inputs, starting to solve a problem, finishing solving a problem etc. A total of 407 observations were taken, corresponding to approximately 34 observations per student. Details about the on-task and off-task observations are given in Table 1. Data from 6 students were used as training data to build the models for making predictions for the other 6 students (i.e., who are used as the test data). Please note that students with learning disabilities and emotional disorders are splitted between the training and test groups uniformly in order

TABLE 2
DETAILS ABOUT THE TRAINING AND TEST SPLITS OF THE
OBSERVED DATA FOR ON-TASK AND OFF-TASK
BEHAVIORS

	Observation Type		Total
	On-Task	Off-Task	
Training	139 (68.1%)	65 (31.9%)	204
Test	134 (66.1%)	69 (33.9%)	203
Total	273 (67.1%)	134 (32.9%)	407

Number of observed on-task behaviors for training and test splits can be seen under the On-Task column; number of observed off-task behaviors can be seen under the Off-Task column and the total number of observed behaviors for training and test splits can be seen under the Total column. The percentages in the parenthesis indicate the ratio of positive and negative data for training and test splits as well as the total.

not to have any bias on the learned models. Particularly, data from 2 students with learning disabilities and 1 student with emotional disorder was used along with the data from 3 normal achieving students for training. In the same way, data from the remaining 2 students with learning disabilities and 1 student with emotional disorder combined with a mild intellectual disability was used for testing along with the data from the 3 remaining normal achieving students. Details about the training and test splits can be seen in Table 2.

3 METHODS: LEAST SQUARES AND RIDGE REGRESSION

This section first describes the technique of Least Squares, then introduces the problem of over-fitting and finally talks about the technique of Ridge Regression.

The linear model fit has been a very important method in statistics for the past 30 years and is still one of the most powerful prediction techniques. The simplest linear model for regression involves a linear combination of input variables as follows:

$$y(\mathbf{x}_n, \mathbf{w}) = w_0 + w_1 x_1 + \dots + w_n x_n = \mathbf{w}^T \mathbf{x} \quad (1)$$

where $\mathbf{x} = (1, x_1, \dots, x_n)^T$ is an instance of training data of D+1 dimensions and $\mathbf{w} = (w_0, w_1, \dots, w_n)^T$ are model coefficients (w_0 is the bias or the intercept). To fit a linear model to a set of training data, the method of the *Least Squares* is one of the most popular techniques. It has been noted in prior work that for machine learning agents that act as "black box" when making predictions, the exact mechanism used by the agent is secondary (i.e., any machine learning method that performs function approximation will work and that determination of the model's inputs/outputs is the critical issue) [5], [6]. Since this paper uses a machine learning agent acting as "black box" for off-task prediction, the Least Squares is used as the machine learning agent.

The Least Squares method determines the values of model coefficients \mathbf{w} by minimizing the sum of the

squares error between predictions $y(\mathbf{x}_n, \mathbf{w})$ for each data point \mathbf{x}_n and the corresponding target value t_n . The sum of squares error is defined as follows:

$$E_D(\mathbf{w}) = \sum_{n=1}^N \{t_n - y(\mathbf{x}_n, \mathbf{w})\}^2 = \sum_{n=1}^N \{t_n - \mathbf{w}^T \mathbf{x}_n\}^2 \quad (2)$$

which can be minimized with a maximum likelihood solution that gives the Least Squares solution of the model parameters as follows:

$$\mathbf{w}_{ML} = (\Phi^T \Phi)^{-1} \Phi \mathbf{t} \quad (3)$$

where Φ is an $N \times D$ design matrix whose elements are given by $\Phi_{nj} = \mathbf{x}_{nj}$ (i.e., j^{th} dimension of the n^{th} training instance).

An important and common problem in statistical learning is the problem of *over-fitting*. Over-fitting as the name implies is the problem of having an excellent fit to the training data which may not be a precise indicator of future test data [8], [14]. Over-fitting especially occurs in the case of data sparseness that is caused by using limited training data to learn the parameters of a model. Regularization is a technique to control the over-fitting problem by setting constraints on model parameters in order to discourage them from reaching large values that lead to over-fitting [8], [14]. *Ridge Regression* is a technique that better controls over-fitting by adding a quadratic regularization punishment of $E_W(\mathbf{w}) = 1/2 \mathbf{w}^T \mathbf{w}$ to the data-dependent error $E_D(\mathbf{w})$. After the addition of $E_W(\mathbf{w})$, the total error function that the technique of the Ridge Regression aims to minimize becomes:

$$\begin{aligned} E_{TOTAL} &= E_D(\mathbf{w}) + \lambda E_W(\mathbf{w}) \\ &= \sum_{n=1}^N \{t_n - \mathbf{w}^T \mathbf{x}_n\}^2 + \frac{\lambda}{2} \mathbf{w}^T \mathbf{w} \end{aligned} \quad (4)$$

where λ is the regularization coefficient that controls the relative importance of data-dependent error $E_D(\mathbf{w})$ and the regularization term $E_W(\mathbf{w})$. The regularization coefficient in this work is learned with 2-fold cross validation in the training phase. The exact minimizer of the total error function can be found in closed form as follows:

$$\mathbf{w}_{RIDGE} = (\lambda I + \Phi^T \Phi)^{-1} \Phi \mathbf{t} \quad (5)$$

which is the Ridge Regression solution of the parameters of the model.

4 MODELING APPROACHES AND PERSONALIZATION

This section describes several modeling approaches for off-task behavior detection as well as the personalized versions of the models.

4.1 Several Modeling Approaches

This subsection describes the models that are used for evaluation: i) a model that only considers time features (i.e., Time Only Modelling), ii) another modelling

TABLE 3
DETAILS ABOUT THE MEAN AVERAGE NUMBER OF RELATIVE AND PERSONAL DATA PER STUDENT

	Mean Average Number of (Relative or Personal) Data Per Student	
	Mean Average Num_Rel_Data	Mean Average Num_Per_Data
On-Task	20.87	1.82
Off-Task	22.31	2.19

Each student has a Num_Rel_Data and Num_Per_Data value for each problem he solves. The average value of these numbers for all the problems of that student is Average Num_Rel_Data and Average Num_Per_Data. The means of these averages for all students are Mean Average Num_Rel_Data and Mean Average Num_Per_Data; both of which are shown under the column with the same name).

approach that considers performance features as well as time features (i.e., Time and Performance Based Modelling), and finally iii) a more advanced model that incorporates mouse movement features with time and performance related features (i.e., Time, Performance and Mouse Tracking Based Modelling).

4.1.1 Time Only Modeling (TimeOnly_Mod)

Modeling students' off-task behaviors just by considering the time taken on an action has been considered a useful approach in the prior work [4], [7], [22]. This modeling approach only considers time related features as a good discriminator of on-task and off-task behaviors. Setting a cut-off on how much time an action/problem should take and categorizing all actions that last longer than that cut-off is one of the simplest and most intuitive ways that have previously been applied to determine whether a student is reading hints carefully [22], to determine whether a student is using guessing to solve a problem [7] and by Baker as a baseline for his multi-feature off-task detection model in his recent work [2]. Baker uses an absolute time feature which is the time that the action of interest takes the user as well as a relative time feature that is expressed in terms of the number of standard deviations the actions' time was faster or slower than the mean time that actions takes for all other students. Idea of relative time features is also quite intuitive since some actions/problems might take more time for many students while others take relatively much less time depending on factors such as difficulty level, familiarity, etc.

In this work, both an *absolute time feature* and a *relative time feature* are used for time only modeling. The relative time feature used in this work is defined as the time spent by a user minus the average time spent on the same problem by all other students.

Time only modeling in this work serves as the baseline for all other models and will be referred as

TimeOnly_Mod.

4.1.2. Time and Performance Based Modeling (TimePerf_Mod)

Time related features are useful in many situations; however, there are lots of other possible data that can be good indicators of off-task behaviors such as the probability that the student possesses the prior knowledge to answer the given question correctly. The percentage of correctness across all previous problems for a student has recently been used as an indicator of this feature for off-task gaming behavior detection [27] and a similar measure has been used in Baker's recent off-task behavior detection effort [2].

In addition to 2 time features that have been mentioned, this modeling approach incorporates 8 more features consisting of 4 main features each of which has 1 absolute and 1 relative version that are calculated by student's feature value minus the mean feature value across all students for the current worksheet. These 8 new features are used as a measure of the probability that the student knew the skills asked in a question. The first feature is the *percentage of correct answers so far* in a problem solving worksheet. Each problem solving worksheet consists of 12 math word problems and a problem is counted as correct only if all question boxes for the problem are filled correctly. The percentage of correctly solved problems up to a current problem in a worksheet is a good indicator of students' success for the current problem. Second, third and fourth features help to assess students' partial skills that are needed for the solution of a problem when they cannot give a full answer. Such partial skills for a problem include answers to i) diagram boxes which check students' mapping of the information given in a problem into an abstract model; ii) an equation box which checks whether a student can form a correct equation from the information given in a problem; iii) a final answer box which checks whether a student can solve the asked unknown in a problem correctly. The corresponding features are *percentage of correct diagram answers so far*, *percentage of correct equation box answers so far* and *percentage of final answers so far*. The values of these features are calculated for a current problem solving worksheet and provide the percentage of correct answers given by a student for the associated partial skill boxes (of each feature) of all the solved problems of the current worksheet.

The approach that uses time and performance features will be referred as TimePerf_Mod.

4.1.3. Time, Performance and Mouse Tracking Based Modeling (TimePerfMouseT_Mod)

Incorporation of performance features into the time only modeling is an effective way of improving the off-task behavior detector; however, there is still more room to improve. Both time-only modeling and time & performance based modeling approaches ignore an important data source: mouse movement. Students are almost always in interaction with mouse when using the tutoring systems. As far as we know, there is very limited

prior research on the detection of gaming the system or off-task behaviors that utilize mouse tracking data [10], [13]. More details about the prior work on this modeling approach, as well as utilizing mouse tracking data, can be found in the Introduction section.

In addition to the 2 time features and 8 performance related features that have been mentioned, this modeling approach incorporates 6 more features consisting of 3 main features each of which has 1 absolute and 1 relative version that are used as mouse tracking data. The first feature is the *maximum mouse off time* in a problem which provides the knowledge of the biggest time interval (in seconds) in which mouse is not used for a current problem. Second and third mouse tracking features are the *average x movement* and *average y movement* respectively. They basically assess the average number of pixels the mouse is moved along the x and y axes in 0.2 second intervals. Two versions of all mouse movement features are used (i.e., absolute and relative).

The approach that uses time, performance and mouse tracking features will be referred as TimePerfMouseT_Mod.

4.2 Personalization

This subsection describes the approach of personalizing all the models explained in the previous subsection.

Utilizing absolute versions of features, along with relative versions (which basically assess the value of particular feature with respect to other students), has been shown to be effective for off-task detection task by Baker [2]. Although incorporating students' relative information with respect to other students is an intuitive way of improving the accuracy of off-task behavior detection, there is still an important issue to consider: a student's current performance with respect to her/his past performance. Different students have different types of behavior (e.g., using more or less time to solve the problems; having difficulties with particular types of questions and/or problems, different mouse usage styles, etc.). Therefore introducing personalized versions of each feature into off-task detection models makes these models more flexible and adaptive to different student types (i.e., makes them personalized).

In addition to all the absolute and relative versions of the features of each model that were described in the previous subsection, personalized approach also considers personal versions of each feature. Personal version of a feature in this work is defined as the absolute value of a feature minus the average value of this feature on the same problem by the same student so far. To reiterate, data from a student's past trials on a particular problem is used to generate the personal version of each feature while predicting his/her behaviors for the current trial of the same problem. However this approach becomes problematic if there is a limited number of past trials or none at all, in which case personalized features will not be a good representation of students' past behavior trend. Yet, please note the statistics shown in Table 3 which show that students usually repeat the problems and, therefore, personalization is practical in

TABLE 4
RESULTS OF THE NON-PERSONALIZED VERSION OF TIMEPERFMOUSET_MOD MODEL IN COMPARISON TO NON-PERSONALIZED VERSIONS OF TIMEONLY_MOD AND TIMEPERF_MOD MODELS

Models	Technique & Version of the Feature Space	
	Least_Squares_NPers	Ridge_Regression_NPers
TimeOnly_Mod	0.5800	0.5631 (-02.91%)
TimePerf_Mod	0.5872 (+01.24%)	0.6275 (+08.18%)
TimePerfMouseT_Mod	0.7731 (+33.29%)	0.8500 (+46.55%)

Note that the results for each model for the technique of Least Squares are shown under the Least_Squares_NPers column, and the results for each model for the technique of Ridge Regression are shown under the Ridge_Regression_NPers column. The percentages in the parenthesis show the relative improvements of each model with respect to the Least Squares version of the Non-Personalized version of the TimeOnly_Mod model. The performance is evaluated by the F_1 measure.

general.

In this work, we use a weighted combination of relative and personalized versions of each feature in a way that if there is very limited personalized data, relative version of each feature dominates the value of this combined version. If there is enough personalized data available, personalized version of each feature dominates the combined version. Specifically the weighted combination is as follows:

$$\begin{aligned}
 RelPersComb_{p_i} = & \\
 & \left(\frac{(Num_Rel_Data_p / C)}{(Num_Rel_Data_p / C) + Num_Pers_Data_p} \right) * Rel_{p_i} \\
 & + \left(\frac{Num_Pers_Data_p}{(Num_Rel_Data_p / C) + Num_Pers_Data_p} \right) * Pers_{p_i}
 \end{aligned} \quad (6)$$

where $RelPersComb_{p_i}$ is the weighted (linear) combination of the relative version of the i^{th} feature of the p^{th} problem (Rel_{p_i}) and the personalized version of the i^{th} feature of the p^{th} problem ($Pers_{p_i}$). $Num_Rel_Data_p$ is the number of training data instances available for the current (p^{th}) problem (i.e., number of relative data instances available from the training data). $Num_Pers_Data_p$ is the number of personal data instances available for the current (p^{th}) problem (i.e., data from student's past trials on the p^{th} problem). C is a constant that is set to 20. Some statistics about *mean average values of Num_Rel_Data and Num_Pers_Data* are shown in Table 3.

Personalized version of each modeling approach uses the above combined version of relative and personal version of each feature along with the absolute version of each feature.

5 EXPERIMENTAL METHODOLOGY: EVALUATION METHOD

To evaluate the effectiveness of the off-task behavior detection task, we use the common F_1 measure which is

the harmonic mean of precision and recall [1], [24]. Precision (p) is the ratio of the correct categorizations by a model divided by all the categorizations of that model. Recall (r) is the ratio of correct categorizations by a model divided by the total number of correct categorizations. A higher F_1 value indicates a high recall as well as a high precision.

6 EXPERIMENT RESULTS

This section presents the experimental results of the models that are presented in the Methods, and Modeling Approaches, and Personalization sections. All the models were evaluated on the dataset described in Data section.

An extensive set of experiments are conducted to address the following questions:

1. How effective are the following 3 models compared to each other: i) TimeOnly_Mod model that utilizes time features, ii) TimePerf_Mod model that utilizes time and performance features; and iii) TimePerfMouseT_Mod model that utilizes time, performance and mouse tracking features?
2. How effective is the approach of utilizing the Ridge Regression technique to estimate the model parameters?
3. How effective is the approach of utilizing personalization?

6.1 The Performance of Several Modeling Approaches

The first set of experiments was conducted to measure the effect of including the performance features in the TimeOnly_Mod model as well as including the mouse tracking data in the TimePerf_Mod model. The details about this approach are given in the Several Modeling Approaches section.

More specifically, TimePerf_Mod model is compared with TimeOnly_Mod model on the off-task behavior detection task. The performance of TimePerf_Mod model is shown in comparison to TimeOnly_Mod in Table 4 for non-personalized versions of these models, and in Table 5 for personalized versions of these models. It can be seen from both tables that the TimePerf_Mod model outperforms TimeOnly_Mod model for both personalized and non-personalized versions. The lesson to learn from

TABLE 5
RESULTS OF THE PERSONALIZED VERSION OF TIMEPERFMOUSET_MOD MODEL IN COMPARISON TO PERSONALIZED VERSIONS OF TIMEONLY_MOD AND TIMEPERF_MOD MODELS

Models	Technique & Version of the Feature Space	
	Least_Squares_Pers	Ridge_Regression_Pers
TimeOnly Mod	0.5686	0.5794 (+01.89%)
TimePerf_Mod	0.6061 (+06.60%)	0.6367 (+11.98%)
TimePerfMouseT_Mod	0.7971 (+40.19%)	0.8834 (+55.36%)

Note that the results for each model for the technique of Least Squares are shown under the Least_Squares_Pers column, and the results for each model for the technique of Ridge Regression are shown under the Ridge_Regression_Pers column. The percentages in the parenthesis show the relative improvements of each model with respect to the Least Squares version of the Personalized version of the TimeOnly_Mod model. The performance is evaluated by the F_1 measure.

TABLE 6
RESULTS OF THE LEAST SQUARES VERSION OF THE PERSONALIZED VERSION OF ALL MODELS IN COMPARISON TO NON-PERSONALIZED VERSIONS OF ALL MODELS

Methods	Technique & Version of the Feature Space	
	Least_Squares_NPers	Least_Squares_Pers
TimeOnly Mod	0.5800	0.5686 (-01.96%)
TimePerf_Mod	0.5872 (+01.24%)	0.6061 (+04.50%)
TimePerfMouseT_Mod	0.7731 (+33.29%)	0.7971 (+37.43%)

Note that the results for the Non-Personalized version of each model for the technique of Least Squares are shown under the Least_Squares_NPers column, and the results for the Personalized version of each model are shown under the Least_Squares_Pers column. The percentages in the parenthesis show the relative improvements of each model with respect to the Least Squares version of the Non-Personalized version of the TimeOnly_Mod model. The performance is evaluated by the F_1 measure.

this set of experiments is that performance related features are very helpful when they are combined with time features for off-task behavior detection. This explicitly demonstrates the power of incorporating the performance related features into the time only modeling.

In the same way, TimePerfMouseT_Mod model is compared to TimeOnly_Mod and TimePerf_Mod models. The performance of TimePerfMouseT_Mod model is shown in comparison to TimePerf_Mod and TimeOnly_Mod models in Table 4 for non-personalized versions of these models, and in Table 5 for personalized versions of these models. It can be seen from both tables that the TimePerfMouseT_Mod model substantially outperforms both TimePerf_Mod and TimeOnly_Mod models for both non-personalized and personalized versions. Paired t-tests have been applied for this set of experiments, and statistical significance with p-value of less than 0.05 has been achieved in favor of using mouse movements (in different configurations). These sets of experiments show that mouse movement features are very helpful when they are combined with time features and performance features for off-task behavior detection. This explicitly demonstrates the power of incorporating the mouse tracking features into time and performance based modeling.

6.2 The Performance of Utilizing the Ridge Regression Technique

The second set of experiments was conducted to measure the effect of utilizing the technique of Ridge Regression

for learning the model parameters for each of the models. The details about this approach are given in Methods section.

More specifically, Ridge Regression learned models are compared to Least Squares learned models for both non-personalized and personalized versions. The performance of Ridge Regression learned version of each model is shown in comparison to Least Squares learned versions in Table 4 for non-personalized versions of these models, and in Table 5 for personalized versions of these models. It can be seen that the Ridge Regression learned version of each model outperforms Least Squares learned versions with its regularization framework for both of non-personalized and personalized models. Paired t-tests have been applied for this set of experiments, and statistical significance with p-value of less than 0.05 has been achieved in favor of using ridge regression against using least square regression with mouse movements.

6.3 The Performance of Utilizing the Approach of Personalization

The last set of experiments was conducted to measure the effect of utilizing the approach of personalization to better capture different behavior types of different students. The details about this approach are given in Personalization section.

More specifically, personalized version of each model is compared to its corresponding non-personalized version. The performance of the personalized version of each model is shown in comparison to non-personalized

TABLE 7
RESULTS OF THE RIDGE REGRESSION VERSION OF THE PERSONALIZED AND NON-PERSONALIZED VERSIONS OF ALL MODELS
IN COMPARISON TO THE LEAST SQUARES VERSION OF THE NON-PERSONALIZED VERSION OF ALL MODELS

Methods	Technique & Version of the Feature Space		
	Least_Squares_NPers	Ridge_Regression_NPers	Ridge_Regression_Pers
TimeOnly_Mod	0.5800	0.5631 (-02.91%)	0.5794 (-00.01%)
TimePerf_Mod	0.5872 (+01.24%)	0.6275 (+08.18%)	0.6367 (+09.78%)
TimePerfMouseT_Mod	0.7731 (+33.29%)	0.8500 (+46.55%)	0.8834 (+52.31%)

Note that the results for the Non-Personalized version of each model for the technique of Least Squares are shown under the Least_Squares_NPers column; the results for the Non-Personalized version of each model for the technique of Ridge Regression are shown under the Ridge_Regression_NPers column; and the results for the Personalized version of each model for the technique of Ridge Regression are shown under the Ridge_Regression_Pers column. The percentages in the parenthesis show the relative improvements of each model with respect to the Least Squares version of the Non-Personalized version of the TimeOnly_Mod model. The performance is evaluated by the F_1 measure.

version of that model in Table 6 for Least Squares learned versions. The performance of personalized version of each model is shown in comparison to non-personalized version of that model in Table 7 for Ridge Regression learned versions. It can be seen that for both of Least Squares and the Ridge Regression learned versions of each model, the personalized version outperforms the non-personalized versions in most cases with its capability to better capture the different usage styles of different students. Paired t-tests have been applied for this set of experiments. Although the results were not shown to be statistically significant (i.e., with p-value of less than 0.05), the personalization approaches outperform corresponding approaches without personalization consistently in different configurations, which demonstrates the robustness and effectiveness of using personalization. This explicitly demonstrates the power of personalized modeling for off-task behavior detection in intelligent tutoring systems.

7 CONCLUSION AND FUTURE WORK

This paper proposes a novel machine learning model to identify students' off-task behaviors (which involves neither the system nor a learning task) while using an intelligent tutoring system. Only the data that is available from the log files from students' actions within the software is used to construct the model; therefore, the model does not need sophisticated instrumentation (e.g., microphones, gaze trackers, etc.) that are unavailable in most school computer labs. The proposed model makes use of a set of evidence such as time, performance, and mouse movement features and is compared to i) a model that only utilizes time features, ii) a model that uses time and performance features together. Different students have different types of behavior; therefore, personalized versions of each model are constructed and compared to their corresponding non-personalized versions. To address data sparseness problem, the proposed model utilizes a robust Ridge Regression technique to estimate model parameters.

An extensive set of empirical results show that the proposed off-task behavior detection model substantially outperforms the model that only uses time features as

well as the model that utilizes time and performance features together. It is also shown by the experiment results that the personalized version of each model outperforms the corresponding non-personalized version indicating that personalization helps to improve the effectiveness of off-task detection. Furthermore empirical results also show that the proposed models attain a better performance by utilizing the technique of Ridge Regression over the standard Least Squares technique.

There are several possibilities to extend the research. For example, features that explicitly model the difficult levels of available problems will help the system better identify students' behavior. Future research will be conducted mainly in this direction.

ACKNOWLEDGEMENTS

This research was partially supported by the NSF grants IIS-0749462, IIS-0746830 and DRL-0822296. Any opinions, findings, conclusions, or recommendations expressed in this paper are the authors', and do not necessarily reflect those of the sponsor.

REFERENCES

- [1] Baeza-Yates, R. & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Addison Wesley, pp. 75-82.
- [2] Baker, R. S. (2007). Modeling and understanding students' off-task behavior in intelligent tutoring systems. *Proceedings of the SIGCHI conference on Human factors in computing systems, ACM*, 1059-1068
- [3] Baker, R. S., Corbett, A. T., Koedinger, K. R. & Wagner, A. Z. (2004). Off-task behavior in the cognitive tutor classroom: when students "game the system". *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM*, 383-390
- [4] Baker, R. S., Roll, I., Corbett, A. T. & Koedinger, K. R. (2005). Do performance goals lead students to game the system *Proceedings of the 12th International Conference on Artificial Intelligence and Education (AIED2005)*, 57-64.
- [5] Beal, C., Woolf, B. P., Beck, J., Arroyo, I., Schultz, K. & Hart, D. M. (2000). Gaining confidence in mathematics: Instructional technology for girls. *Proceedings of International Conference on Mathematics/Science Education and Technology*.
- [6] Beck, J.E. & Woolf, B.P. (2000). High Level Student Modeling with Machine Learning. *Proceedings of the Intelligent Tutoring Systems Conference*, 584-593.

- [7] Beck, J. (2005). Engagement tracing: using response times to model student disengagement. *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED2005)*, 88-95
- [8] Bishop, C.M. (2006). *Pattern recognition and machine learning*. Springer, pp. 6-10, 144-145.
- [9] Cetintas, S., Si, L., Xin, Y. P. & Hord, C. (2009). Predicting correctness of problem solving from low-level log data in intelligent tutoring systems. *Proceedings of the 2nd International Conference on Educational Data Mining (EDM2009)*, 230-239.
- [10] Cetintas, S., Si, L., Xin, Y. P., Hord, C. & Zhang, D. (2009). Learning to identify students' off-task behavior in intelligent tutoring systems. *Proceedings of the 14th International Conference on Artificial Intelligence and Education (AIED2009)*, 701-703.
- [11] Dalton, T., Martella, R.C. & Marchand-Martella, N.E. (1999). The effects of a self-management program in reducing off-task behavior. *Journal of Behavioral Education* 9, 3-4, 157-176.
- [12] Davis, F. (1976). *La Comunicaci'on No Verbal*, volume 616 of *El Libro de Bolsillo*. Alianza Editorial, Madrid, Spain. Translated by Lita Mourgliaer from: "Inside Intuition - What we Knew About Non-Verbal Communication"; McGraw-Hill Book Company, New York.
- [13] De Vicente, A. & Pain, H. (2002). Informing the detection of the students' motivational state: an empirical study. *Intelligent Tutoring Systems*, 933-943.
- [14] Hastie, T., Tibshirani, R. & Friedman, J. (2001). *The elements of statistical learning*. Springer, pp. 61-68.
- [15] Karweit, N. & Slavin, R.E. (1982). Time-on-task: Issues of timing, sampling, and definition. *Journal of Experimental Psychology*, 74 (6), 844-851.
- [16] Koedinger, K. R. & Anderson, J. R. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, 8, 30-43.
- [17] Lahaderne, H.M. (1968). Attitudinal and intellectual correlates of attention: a study of four sixth-grade classrooms. *Journal of Educational Psychology*, 59 (5), 320-324.
- [18] Lee, S.W., Kelly, K.E. & Nyre, J.E. (1999). Preliminary report on the relation of students' on-task behavior with completion of school work. *Psychological Reports*, 84, 267-272.
- [19] Litman, D. J. & Forbes-Riley, K. (2004). Predicting student emotions in computer-human tutoring dialogues. *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*.
- [20] Maletsky, E. M. et al. (2004). *Harcourt Math*, Indiana Edition, Chicago: Harcourt.
- [21] Merten, C. & Conati, C. (2006). Eye-tracking to model and adapt to user meta-cognition in intelligent learning environments. *Proceedings of the 11th international Conference on Intelligent User interfaces*. ACM, New York, NY, 39-46.
- [22] Murray, R. C. & vanLehn, K. (2005). Effects of dissuading unnecessary help requests while providing proactive help. *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED2005)*, 887-889.
- [23] Person, N., Klettke, B., Link, K. & Kreuz, R. (1999). The integration of affective responses into AutoTutor. In *International Workshop on Affect in Interactions. Towards a New Generation of Interfaces*.
- [24] Rijsbergen, C. J. v. (1979). *Information retrieval*, 2nd edition. University of Glasgow.
- [25] Schofield, J. W. (1995). *Computers and classroom culture*. Cambridge University Press.
- [26] Suchman, L. (1987). *Plans and situated actions: The problem of human-machine communication*. Cambridge University Press.
- [27] Walonoski, J. A. & Heffernan, N. T. (2006). Prevention of off-task gaming behavior in intelligent tutoring systems. *Intelligent Tutoring Systems*, 722-724
- [28] Ziemek, T. R. (2006). Two-D or not Two-D: gender implications of visual cognition in electronic games. *Proceedings of the 2006 symposium on Interactive 3D graphics and games, ACM*, 183-190.



Suleyman Cetintas received his B.S. degree in Computer Engineering from Bilkent University, Turkey. He is currently working toward the Ph.D. degree in Computer Science at Purdue University. His primary interests lie in the areas of information retrieval, machine learning, intelligent tutoring systems and text mining. He has also worked in the area of privacy preserving data mining. He is a member of ACM, ACM SIGIR and the International Artificial Intelligence in Education (IAIED) Society.



Luo Si is an Assistant Professor in Computer Science Department and Statistics Department (by courtesy), Purdue University. Before this, he got his Ph.D. degree from Carnegie Mellon University in 2006. His main research interests include: information retrieval, knowledge management, machine learning, intelligent tutoring systems, and text mining. His research has been supported by National Science Foundation, State of Indiana, Purdue University and industry companies.

He received NSF career award in 2008.



Yan Ping Xin received her Ph.D. in Special Education from Lehigh University. Currently, she is an associate professor at Purdue University. Her research interests include effective instructional strategies in mathematics problem-solving with students with learning disabilities/difficulties, cross-culture performance and curriculum comparison, and meta-analysis. She pioneered Conceptual Model-based Problems Solving (COMPS) approach that facilitates algebra readiness in elementary mathematics learning. She is currently the Principal Investigator (with Dr. R. Tzur in math education and Dr L. Si in computer science) of a 5-year multi-million dollar grant project (funded through National Science Foundation, 2008-2013) that aims to develop a computerized conceptual model-based problem-solving system to nurture multiplicative reasoning in students with learning difficulties. She has published her work in prestigious journals such as *Exceptional Children*, *The Journal of Special Education*, *Journal for Research in Mathematics Education*, and *The Journal of Educational Research*. Her work is cited in detail in the recent Instructional Practices Report from the National Mathematics Panel.



Casey Hord is currently working toward a Ph.D. degree in Special Education at Purdue University. His primary interests lie in the area of interventions for students who struggle with mathematics, particularly, students with learning disabilities or mild intellectual disabilities. He has also worked in the area of software development for helping elementary and middle school students understand and solve word problems. His research has contributed to the development of methods

designed to help students at-risk for failure in mathematics utilizing techniques such as model-based instruction and the concrete-semiconcrete-abstract teaching sequence. He taught middle school in special education and general education settings for a total of six years.