Psychological Test and Assessment Modeling, Volume 65, 2023 (1), 76-124

Detecting Atypical Test-Taking Behavior with Behavior Prediction Using LSTM

Steven Tang, Siju Samuel, Zhen Li

Educational Testing Service

Abstract

A clickstream is a precise log of every user action taken in a software application. Clickstreams recorded during test taking experiences can be analyzed for behavior patterns. In this paper, we introduce a statistic, known as the Model Agreement Index (MAI), that quantifies how typical or atypical an examinee's clickstream's behaviors are relative to a sequence model of behavior; this model is trained to emulate student behaviors using a Long Short-Term Memory (LSTM) network. MAI is intended to be used as a simple statistic to detect instances of atypical user behaviors so that further analysis can be conducted to identify whether the atypical behaviors need to be mitigated in the future. One of the empirical results from this study is that certain examinees with low MAI scores were floundering on the opening and closing of certain tool widgets. This floundering caused wasted time for the examinee, and the discovery of this phenomenon can enable an improvement in the test user interface, showing a good use for the proposed methodology. The study details the processes needed to train the LSTM along with a comparison between the LSTM and a "most common next action" baseline model. Additionally, correlations of MAI with other indicators, such as answer changing, are explored. The use of MAI to identify test user interface issues is demonstrated. Real data from a statewide testing program is used in this study.

Keywords: test security, atypical behavior detection, clickstream analysis, behavior modeling, scalable detection methods

Author Note

Correspondence concerning this article should be addressed to: steven@emetric.net

Introduction

In the context of K-12 large-scale standardized testing, test integrity involves multiple facets: ensuring that the environment is uniform across each testing location across the state, that students are taking the test with purpose and attention, and that cheating, in its many forms, does not occur. The pursuit of preventing or detecting cheating is an inherently challenging task, as those who do cheat attempt to avoid detection. Therefore, methods to detect cheating do not have the luxury of having a clear and obvious indicator of when cheating truly occurs. In the testing industry, many statistical indicators and other proxies for cheating detection are used, and often a method can "flag" an individual for further investigation, relying on an arbitrator to make an ultimate judgment on the case.

In this study, a sequence behavior model that leverages all tracked events in a clickstream is proposed. The goal of this approach is to give a straightforward quantification of how typical an examinee's behaviors are within a testing context. The sequence behavior model is trained on data that includes all trackable actions in a computer-based test environment, including navigations, multiple-choice response selections, tool usage like calculator or notepad, and accommodations such as screen contrast toggling. Note that the proposed model does not currently incorporate time into the model directly, but rather only focuses on the chronological ordering of behaviors represented through the clickstream.

Such an approach differs from existing detection methods, which typically use one or two features of process data, like counting the number of wrong-to-right answer changes (Bishop & Egan, 2017) or modeling response times (van der Linden & Guo, 2008; Wise & Kong, 2005) for aberrance detection. Using raw clickstream data, the proposed approach can idealistically model all possible typical behavior patterns, allowing for the flagging of any behavior sequences that contain fewer typical behavior patterns. This can be a beneficial property for stakeholders interested in investigating any test-takers that are exhibiting atypical behaviors.

Operational Definition of Atypical Behavior

Suppose that a predictive model of student test-taking behaviors exists, with inputs being past clickstream actions and outputs being possible future actions. With this predictive model, one can define an "atypical clickstream" to be a clickstream that is not well predicted by the proposed model by comparing each observed action in the clickstream to the predicted probability of that observed action by the model's output. Clickstreams that are better predicted by the model are supposedly more "typical" as they are more predictable. In this study, a predictive model of behaviors based on a Long Short-Term Memory (LSTM) architecture is proposed, described later in the *Using LSTM for Behavior Modeling* section. This predictive model is then used to compute a **Model Agreement Index** (MAI) value, which indicates the extent of agreement between observed clickstream actions and model-predicted actions on a likelihood continuum ranging between 0 and 1. Clickstreams with relatively low MAI values are operationally considered more atypical than clickstreams with higher MAI values.

An assumption inherent to this study is that such a predictive model can be generally useful to stakeholders interested in ensuring that typical test-taking operations are observed, and that this model could serve as a system to monitor behavior patterns at scale, focusing on the entirety of a test rather than individual item responses. Monitoring algorithms are intended to flag noteworthy results to some degree of accuracy. For testing, noteworthy events could include "cheating behaviors" and "confusion." It can be challenging to design these monitoring algorithms, as descriptions and signals of the cheating phenomenon and of student confusion are not precisely defined and may be extremely rare in practice. The operational definition of atypical in this paper serves as one lens in identifying "typical" and "atypical" behaviors, with the goal that flagging atypical behaviors using this definition will ultimately add value to stakeholders who want to ensure that typical test-taking processes are observed, and that atypical behaviors can be further analyzed to ensure nothing unwanted is occurring.

Related Work

Test fraud, known to be harmful to test integrity and fairness, has been investigated for decades. Common test breach situations include obtaining advance copies of the test, sharing answers between examinees, and modifying incorrect responses. To monitor test security, researchers have developed post-hoc statistical approaches (Asseburg & Frey, 2013; Reise & Due, 1991) to detect test-takers' aberrant behaviors during testing, using item responses or other data collected during the testing process. Existing data forensics methods usually focus on one specific aspect at one time, e.g., examining if an item-response pattern is congruent with a specified measurement model (Drasgrow, Levine, & Williams, 1985), identifying extremely short or aberrant response times (Li, Wall, & Tang, 2018; van der Linden & Guo, 2008; Wise & DeMars, 2006), or detecting a large number of wrong-to-right answer changes at a group or individual level (Bishop & Egan, 2017).

Motivational and emotional aspects of test-takers, such as test-taking efforts, are also found to be highly related to test performance (Asseburg & Frey, 2013). Their study shows that non-effortful behaviors are more likely to be identified with test-takers with lower ability, which in turn leads to a test performance that lies below the test-takers' maximum performance. Non-effortful test-takers are commonly identified by response-time patterns, with an underlying assumption that less motivated test-takers tend to speed in testing (Wise & Kong, 2005). A group of test-takers' non-effortful behaviors might also exert an influence on the estimation accuracy of measurement models and test scores (Wise & DeMars, 2006). Therefore, understanding the test-takers' behaviors is essential in enhancing the validity of educational and psychological measurement.

More recently, there has been greater emphasis on analyzing clickstream and process data for a variety of purposes. In (Jiao, He, & Veldkamp, 2021), a modern overview is presented, with 29 papers from 77 authors, of recent advances concerning the analysis of process data in educational and psychological measurement. Areas included in that collection of research articles include leveraging process data to explore test-takers' behaviors and strategies, joint models for response accuracy and response times, and statistical models on response processes. Ulitzsch et al. (2021) jointly consider action sequences and timing to identify common response processes by employing cluster edge deletion for identifying well-separated groups of action patterns, where each pattern describes a common response process. In this related work, the proposed method focused on assessing response processes to a single item. Instead of focusing on processes for a single item, the method described in this paper utilizes behaviors from all items, including administrative behaviors such as navigating between items and using tools.

Clickstream analysis has historically been used to determine and summarize user behaviors in web usage contexts (Banerjee & Ghosh, 2011; Heer & Chi, 2002). In these works, users' navigation paths within a website were analyzed to obtain information about users' preferences. Clustering techniques have been used to group together clickstreams with similar behavior usage patterns (Gunduz & Ozsu, 2003; Su & Chen, 2015); these clusters were used to infer user interests and predict future user behaviors. In terms of aberrant and malicious user detection, clickstream analysis has been used to detect potential attackers who create fake identities in social media platforms (Wang, et al., 2017). In that work, sub-sequence counting with clustering is used to categorize clickstreams into different user archetypes, identifying clusters of clickstreams that could potentially be flagged for banning in their respective social media platforms.

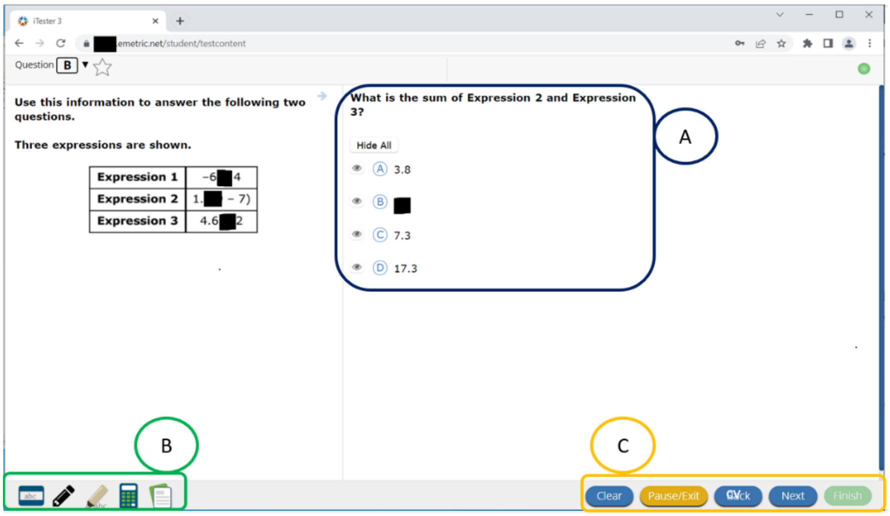
Predicting the next action given an existing sequence is common in the task of language modeling, where the next word in a sequence is predicted based on prior information from the sequence thus far. Existing research in the domain of language modeling has found that sequence models based on Long Short-Term Memory networks have strong performance (Sundermeyer, Schlüter, & Ney, 2012), beating prior approaches based on n-grams or hand-crafted features. Utilizing LSTM networks has also been used to predict student behaviors in Massively Open Online Courses, to better understand usage patterns as well as to possibly identify useful resources based on the resources similar students have utilized in the past (Tang, Peterson, & Pardos, 2017). Similarly, in the current work, the prediction model is used to predict which actions and behaviors are the most likely given a sequence of test-taking behaviors in a test environment.

Dataset

The dataset for this study consists of clickstream data from a state-wide summative test administered to grade 8 students in 2021. Each row in the clickstream log contains key pieces of information: timestamp, click_action, item_id, user_id. The click_action is the actual click or action that was taken. The item_id is used to look up item-specific properties to append to the test state for modeling; for example, certain items might allow for calculator usage, so the item_id can be used to look-up whether calculator is allowed on this item. The user_id identifies which test-taker produced the click-stream.

Table 15 in the appendix shows the 151 possible actions from this clickstream dataset. The approach in the current study has a larger, more complex input space compared to other approaches. The key benefit of using this more complex input space is that every instance of clickstream behavior is modelled, allowing the LSTM model to potentially learn many different patterns of test-taking behaviors.

The input to the behavior model consists of the click_action as well as "test state". Test state is a description of what is currently on the test and what options are available to the examinee. The possible test states for this study can be found in Table 14 in the appendix. Data pre-processing involves concatenating the 'click_action' with the 'test state' at each time step for each individual clickstream.



Use this information to answer the following two questions.

Three expressions are shown.

Expression 1	−6[redacted]4
Expression 2	1.[redacted] − 7)
Expression 3	4.6[redacted]2

What is the sum of Expression 2 and Expression 3?

Ⓐ 3.8

Ⓑ [redacted]

Ⓒ 7.3

Ⓓ 17.3

Figure 1.

Screenshot of Test Application

Figure 1 shows a screenshot of the testing application. To illustrate an example of a "test state", the screenshot is marked with (A), (B), (C). Test state includes elements such as (A) MultipleChoiceItem, (B) a variety of tool options, such as Scientific Calculator, and (C) navigational options, such as Next, Finish, and others. The current test state is described in Table 1, where elements of the test are indicated as 0 or 1. A 0 represents the element is not available in the current test state, and a 1 represents the element is available or on. The model can then leverage this information in making the correct prediction and in avoiding impossible predictions given the current test state.

Table 1:

Sample Test State from Screenshot

Test State Element	
Basic Calculator	0
Scientific Calculator	1
Dictionary	0
Protractor	0
Guideline	1
Sketch And Highlight	1
MultipleChoiceItem	1
OpenEndedItem	0
… (Test state comprises 118 total elements; refer to **Table 14** in appendix)	

Dataset Sample

The dataset used in this study consists of 39,524 Grade 8 examinee records, with a total of 4.9 million clickstream rows, across 5 different test forms from the administration of a state-wide summative assessment in 2021. The 39,524 records represent every "valid" clickstream that was able to be processed.

Table 2 shows the different n-counts across each test form, including the final valid/usable n-count. The first row in the table describes the entirety of the dataset, with all five forms combined, and then each subsequent row describes each individual form. Each form varies the order of questions asked, but all forms contain the same test items. Any clickstream that was incomplete, interrupted, did not have a turn-in event, did not contain a valid login, or was generated by system processes were filtered out. Clickstreams with fewer than 30 actions were additionally filtered out of the data. Most of these short clickstreams were generated by system-testing backend processes and were not related to students taking tests; fewer than 1% of logged click-streams had fewer than 30 actions. The last column in the table shows the average number of logged actions per clickstream for each valid clickstream.

Clickstreams were truncated at 200 logged actions so that actions beyond the 200th action in a clickstream were not considered for modeling. Truncation can be a useful technique when trying to improve model training time for LSTM prediction models if it can be determined that the removal of data does not have a negative impact on the desired outcome effects from the prediction model. Truncation is typically employed when there are a few sequences with abnormally long lengths, as standard training procedures require that all input sequences be of the same standard length, so very long samples might be truncated to alleviate memory requirements and compute time instead of requiring all sequences to be "padded" with null data up to the abnormally long length of the outlier sequences. Only 202 out of the 4,900,088 actions were removed in the truncation procedure for this study; these 202 actions came from 131 unique clickstreams, so the amount truncated per unique clickstream is on the order of only a few actions. Since only a few actions were truncated, the impact of this truncation procedure is likely to be minimal to none. The original purpose of this truncation procedure was to account for extremely long clickstreams; however, once the "valid criteria" was applied, nearly all clickstreams were at most 200 actions, with the longest clickstream coming in at 206 actions, just barely over the original truncation length limit.

Table 2:

Clickstream N Counts Across Different Forms in Test Data

Test/form	Total Unique IDs	N Count: Valid clickstreams (Contains Login action, Turn-in Action, and at least 30 actions)	Average Number of Logged Actions in Valid Click-streams
All forms	92138	39524	122
F.1	20295	8575	123
F.2	20465	8779	123
F.3	20352	8738	123
F.4	20452	8735	123
F.5	10573	4697	121

A model is trained for each individual form, as the shuffling of item orders may impact behaviors. In Table 16 (located in the appendix), the counts for each possible click_action are shown. Note that when interpreting Table 16, readers are only meant to gain a general understanding of the distribution of the observed clickstream actions as well as the distribution of the prediction model's predictions. Notions of model accuracy will be addressed in the model results section, specifically in Table 4 and in Table 5. The most common action taken was ITEM_MULTIPLE_CHOICE_ANSWER, which was taken 1,240,854 times, comprising 25.29% of the dataset. The next most common action was NAVIGATION_ITEM_NEXT, comprising 23.84% of the data. Therefore, 49.13% of the logged clicks were either Multiple Choice Answer or Navigating to the next item. The remaining 50.87%, representing 2.5 million clicks, were distributed among the remaining 149 possible actions. The 3rd most common action was ITEM_DRAG_BOX_DRAG_END, comprising 4.91% of all clicks. Note the large disparity between 23.84% and 4.91%, showing that the top two actions were vastly more common relative to the rest of the possible click_actions.

Using LSTM for Behavior Modeling

The desired model for the task of predicting examinee behavior would 1) be capable of understanding the chronological flow of taking an exam, from starting the test, to answering all test items usually in chronological order, and then finally submitting the test for scoring, and 2) be capable of making a logical next-step behavior prediction based on observations from the past and based on the current test state. This sequential nature of predicting examinee behavior lends itself to the LSTM network architecture, which is especially well suited to generating predictions that are based on historical context. Consider that test-takers that have used a tool in an earlier part of the test may be more likely to use that tool again in a later part of the test, relative to those who did not use the tool earlier in the test. The LSTM architecture is capable of learning this kind of long-term relationship in sequences by leveraging its internal memory network. For an illustrative perspective to understand the LSTM, interested readers are referred to (Olah, 2015), complete with step-by-step figures and illustrative examples. For a lengthier mathematical introduction to the LSTM and its training details, interested readers can see (Greff, Srivastava, Koutnik, Steunebrink, & Schmidhuber, 2016).

LSTM Implementation

The model takes as input each examinee action represented by an index number. These indices correspond to the index in a 1-hot encoding of vectors; a 1-hot encoding is essentially a vector with all 0s except at the index location of interest, which is represented as a 1. Table 15 shows the mapping of indices to the corresponding clickstream action. Additionally, the model takes as input the current test state, represented by a vector of 0s and 1s, corresponding to indices found in Table 14. The concatenation of the one-hot vector and the test-state vector represents the input vector to the LSTM model. The sequence is consumed one element at a time. For example, if a clickstream has 45 actions taken, the LSTM model will consume each input vector (action + test state) one at a time, in sequential order, adding information to its internal representation each time. This one-at-a-time consumption is often referred to as a timestep, where there is a “time” element to the model whereby events that have occurred sequentially earlier in time can influence model predictions later in time, and that each sequential step through the process marks an explicit update to the model’s current state.

The LSTM model was implemented in Keras (Chollet & Others, 2015), an open-source software library that provides a Python interface for artificial neural networks with the machine learning library TensorFlow (Abadi, et al., 2015) serving as the back end. LSTM models have a variety of hyperparameters that can be tuned. In (Tang, Peterson, & Pardos, 2017), an LSTM model was used to predict clickstream actions in a web-based online classroom. In that work, a grid search over the number of hidden layers (1, 2, 3), the number of nodes per hidden layer (64, 128, 256), and learning rate (0.01, 0.001, 0.0001) was conducted. The results showed that the learning rates 0.01 and 0.001 performed the best. Among the experimental models with those two learning rates, test-set accuracy varied by less than 0.01, showing that most configurations of hyperparameters performed equally well. For the current study, it is assumed that similar hyperparameter settings should also perform well, and that a grid search would similarly not achieve substantially different results. For the current study, the number of LSTM cells per hidden layer is set to 100, the number of hidden layers to 2, and the learning rate to 0.001. Increasing the number of hidden LSTM cells and the number of LSTM layers would effectively increase the learning capacity of the model, as each LSTM cell could potentially learn about a different aspect of the mapping between the input to the output. The learning rate parameter controls to what degree the model updates its internal representations and weights when model weights are updated. For a thorough expository treatment of each hyperparameter that can be tuned in LSTMs, interested readers can see (Greff, Srivastava, Koutnik, Steunebrink, & Schmidhuber, 2016).

Backpropagation is the training algorithm used to update the weights in neural networks in order to minimize the error between the expected output and the predicted output for a given input. For sequence prediction problems where there is an order dependence between observations, recurrent neural networks, which include LSTMs, are trained using a variation of the Backpropagation algorithm called Backpropagation Through Time, or BPTT for short. In effect, BPTT unrolls the recurrent neural network and propagates the error backward over the entire input sequence, one timestep at a time. The weights are then updated with the accumulated gradients. For a more in depth explanation of the BPTT training technique, interested readers may view (Brownlee, 2017).

The proposed prediction model outputs 151 values, each value representing one of the possible clickstream actions. These values are converted to probabilities using the softmax function, which normalizes the output values to sum to 1. Each value in the output of the softmax function is interpreted as the probability of membership for each class (Brownlee, 2020). The ground truth for each prediction is the index of the next action; thus, the model attempts to predict the next action given a history of past actions. Categorical cross entropy is used as the loss function; categorical cross entropy is used when true labels are one-hot encoded, meaning that there is only one possible label for each prediction. Categorical cross entropy can also be called "logarithmic loss," whereby output probabilities that are far from the actual expected value are assigned a penalty that is logarithmic in nature.

Categorical cross entropy loss is defined as:

$$L_{CE} = -\sum_{i=1}^{n} t_i \log(p_i)\,,\ \text{for } n \text{ classes}, \tag{1}$$

$$t_i = \begin{cases} 1 \text{ if the } i^{th} \text{class is true} \\ 0 \text{ otherwise} \end{cases}$$

where t_i is the truth label and p_i is the softmax probability for the i^{th} class.

Classical stochastic gradient descent maintains a single learning rate for all weight updates, and the learning rate does not change during training. A potentially more efficient alternative to classical stochastic gradient descent is the Adam (Kingma & Ba, 2015) optimizer. Adam is chosen as the optimizer for this study as it has been shown to work well in practice for a variety of neural network optimization tasks. Interested readers are directed to (Brownlee, 2021) that provides a comparative analysis of different methods for optimizing neural network model training. Dropout layers were added between LSTM layers as a method to curb overfitting. Dropout randomly zeros out a set percentage of network edge weights for each batch of training data (Pham, Bluche, Kermorvant, & Louradour, 2014). The data was randomly split into a training set, comprising 70% of the data, and a validation set, comprising the remaining 30%. Early stopping was used, so that the model was trained until 15 consecutive epochs did not result in improvement on the validation set. An epoch represents the process of passing through all training samples once during training; during this process, the model's weights are updated towards improved prediction performance, but the updates are down weighted by regularization techniques such as the model's learning rate. The final LSTM model has a 269-dimensional input (action + test state) at each timestep and outputs a 151-dimensional output at each timestep. This implies that the model can differentiate 151 different actions in the output. The 269-dimensional input consists of 151 dimensions for the click_action and then an additional 118 dimensions to describe the current test state.

In this study, an nVidia RTX8000 GPU was used for model training and inference. Model training time information is provided so the reader can determine the practicality of the methodology. Table 3 shows the time taken to train the LSTM model on each form, as well as the resulting best epoch based on the holdout validation set. Timing results could vary in the future by adjustments to training parameters such as batch sizes, differences in hardware, and differences in clickstream dataset size. In the current study, optimizing training time was not a primary objective.

Table 3:

Training Epochs and Time by Form

Form	Best epoch	Training Time (Total Hours)	Training Time (Seconds Per Epoch)
F.1	196	4.13	72
F.2	204	4.48	79
F.3	181	3.92	78
F.4	183	3.97	78
F.5	213	2.48	42

Model Training Validation

Figure 2 shows the training and validation loss for each epoch for test form F.1. Training and validation loss do not deviate strongly from each other, indicating that the chosen model is fitting well and generalizing to the validation set. One notable trend is that the validation loss appears to be lower than the training loss until roughly the 125th epoch. This may seem unintuitive at first, as models typically perform better on training data and perform worse on unseen, validation data. This unintuitive phenomenon can occur for two reasons: 1) Training loss is incrementally computed after each mini-batch of training data is processed during an epoch. This means that loss calculations performed earlier in the epoch may have worse performance compared to calculations performed later in the epoch when more weight updates have occurred. The model used during validation loss calculation does not suffer from this shifting performance, as the model has already had all of its updates from the training pass completed, and all predictions on the validation set utilize the fully trained model following the training epoch. 2) Dropout regularization is only applied during training but not during validation. Dropout reduces the input signal during training by randomly masking or removing input elements, forcing the model to make predictions using less data, whereas the model used during validation makes predictions based on full input data, thereby potentially increasing prediction accuracy. As training continues, the training loss curve eventually crosses lower than the validation loss curve after roughly the 125th epoch, which indicates that the model is improving its predictions on the training set at a slightly faster rate than its improvements on the validation set.

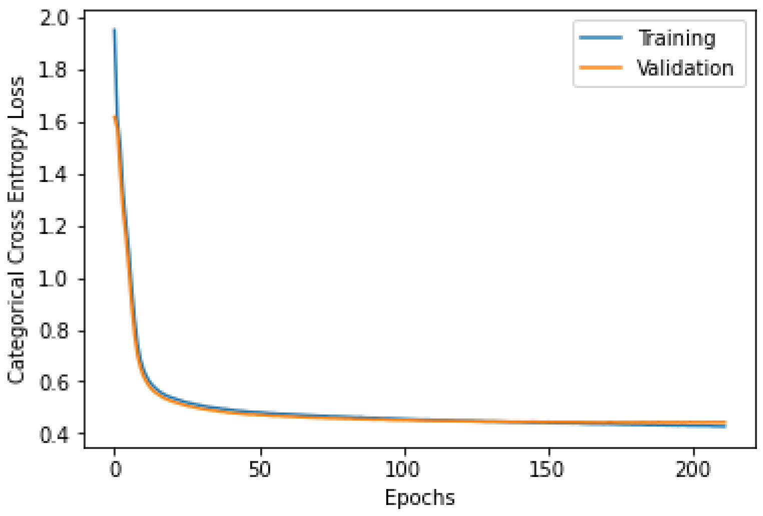


Figure 2.

Training (Blue) and Validation (Orange) Loss during Fine-tuning for Form F.1

Model Agreement Index

The Model Agreement Index (MAI) is a straightforward index of how well an examinee's behaviors align with the trained clickstream behavior model. The index is simply the average probability score of an examinee's observed actions according to the model's predictions of their actions. Therefore, MAI is effectively a summarized weighted probability over all actions taken within an individual clickstream.

A clickstream ***c*** can be defined as a list of vectors. Each vector is a representation of a single click taken by an examinee combined with some state information that describes elements of the test that are present to the examinee at that timestep. The dimensionality of each vector is equal to the number of dimensions needed to create a one-hot encoding representation of the click-action space combined with the number of dimensions in the desired test state description. For the specific number of dimensions unique to the data in this study, see the *LSTM Implementation* section presented earlier in this study.

To calculate MAI for a clickstream ***c***, the corresponding probability from the model output probability distribution for the actual action taken at each timestep is iteratively obtained, summed up, and divided by the length of ***c***.

The MAI formula for a clickstream ***c*** can be described as:

$$MAI_c = \frac{\sum_{s=1}^{S} \sum_{i=1}^{n} t_{si} p_{si}}{S}, \tag{2}$$

$$t_{si} = \begin{cases} 1 \text{ if action } i \text{ is the action observed at timestep } s \\ 0 \text{ otherwise} \end{cases}$$

where *S* is the length of the clickstream, *s* represents a single "step" or "timestep" and iterates from 1 through S, *i* is used to correspond to an index used to represent a particular action, *n* is the total number of possible actions and represents the highest possible value of *i*, t_{si} is a truth label at timestep *s* and for action *i* defined as described in formula (2), and p_{si} is the softmax probability from the model for action *i* at timestep *s*. For the specific dataset used in this study, *n* takes a fixed value of 151, the mapping of each index *i* to a specific action is depicted in Table 15, S is on average 122 across all clickstreams but can vary, and there are a total of 39,524 valid clickstreams that are analyzed.

MAI takes a score range from 0 to 1. Higher scores show stronger agreement between examinee observed behaviors and predicted model actions. Conversely, lower scores mean that the examinee has taken more atypical (and less likely) actions, according to the model's predictions. In general, MAI can be used to identify individual examinee atypical behavior. MAI can also be aggregated for group-level analysis.

Baseline Model – Most Common Next Action

One of the simplest approaches towards a sequence prediction model involves assessing how often an individual's "next action" matches the "most common next action" that the population would have taken given the immediately prior action. This model will be referred to as the Most Common Next Action (MCNA) model. The MCNA model can be further divided into two forms of tabulation: MCNA-Top1 and MCNA-D. MCNA-Top1 only considers the most common following action for a particular behavior and will therefore predict with 100% confidence that the next action will be this most common action. MCNA-D, where the D is short for "distribution", weights each following prediction by its corresponding prevalence; this model can therefore provide a probability in its output prediction based on the proportion of times the "following" action followed the current action.

The MCNA model is effectively a 2-gram, also known as a bigram (Collins, 1996), model. A 2-gram is a specific case of the "n-gram" model, whereby n specifies the length of sequence components to tabulate over when creating the prediction distribution. The two elements in these sequences are the current action and the action immediately following that action. The number of elements to consider is generalized to any n in the n-gram. The proposed LSTM approach could be somewhat thought of as an “infinite” length generalization of the n-gram approach that also leverages additional weighting mechanisms to model a sequence of elements; these weighting mechanisms can learn additional useful information about individual elements to enhance predictions.

Tang et al. (2017) compared a similar LSTM behavior model systematically to a corresponding n-gram approach. The values of n for the n-grams studied in that paper ranged from 2 through 10. The result showed that the n-gram approach stopped improving in accuracy at the 8-gram mark, indicating that 9- and 10-grams showed worsening performance when generalizing from the training data to test data. The 2-gram (akin to the MCNA in this study) approach achieved top-1 accuracy of .630, the highest-performing 8-gram achieved .703, and the LSTM approach achieved .722. This prior work showed that while the n-gram plateaus in performance, the LSTM was capable of improving with stronger predictive power. For the current study, Table 17 in the appendix shows the corresponding prediction for each possible behavior in the input space for the MCNA-Top1 model, tabulated from test form F.1.

LSTM-MAI and MCNA-D Results

Using the proposed LSTM model and the previously described MAI calculation method, Table 4 shows the MAI and the Top1 scores for both the LSTM and MCNA models across all 5 test forms. LSTM-MAI is used to denote the MAI results using the LSTM model. MCNA-D is parallel to the LSTM-MAI calculation, using the MCNA’s weighted prediction space as the output and generating an average accuracy based on these weighted predictions for each examinee clickstream. LSTM-Top1 analyzes how often the top prediction from the LSTM model exactly matches the observed action, without considering the weighted probability from the output distribution. MCNA-Top1 is parallel to LSTM-Top1 but uses the MCNA model instead. Standard deviations are included in parenthesis in each table cell.

Table 4:

MAI And Top1 Performance for LSTM and MCNA

Form Number	LSTM-MAI Average Score (Std. Dev.)	MCNA-D Average Score (Std. Dev)	LSTM-Top1 Average Score (Std. Dev.)	MCNA-Top1 Average Score (Std. Dev.)
F.1	.668 (.08)	.516 (.07)	.774 (.08)	.668 (.10)
F.2	.661 (.08)	.502 (.07)	.766 (.08)	.656 (.10)
F.3	.658 (.08)	.502 (.07)	.765 (.08)	.657 (.10)
F.4	.662 (.08)	.513 (.07)	.770 (.08)	.665 (.10)
F.5	.675 (.09)	.530 (.08)	.778 (.09)	.682 (.11)
Overall Average	.665	.513	.771	.666

From Table 4, the LSTM-MAI outperformed MCNA-D by a wide margin, averaging .665 compared to .513 respectively across all 5 test forms, for an average absolute performance difference of .152. This large average difference shows that the incorporation of longer sequence histories, in addition to the LSTM's internal memory and learning network, offers substantial improvement over just using the most common next action's distribution. Top1 results are also included, whereby the LSTM-Top1 results were also substantially higher than the MCNA-Top1 results, on average .771 to .666, respectively. This indicates that the LSTM approach's most likely prediction was accurate 77.1% of the time, while the MCNA's most likely prediction was accurate 66.9% of the time. Using the MAI metric is more informative compared to using the Top1 approach for model evaluation purposes, as the probabilistic weights from the MAI approach offer an interpretation of "likelihood" rather than assessing each action as simply "right" or "wrong", or analogously "typical" versus "atypical".

Table 5 shows cross-tabulated results comparing LSTM-Top1 to MCNA-Top1 results. This tabulation shows how often the models agreed or disagreed on a correct or incorrect prediction, showing the exact numerical improvement that one model has over the other on the entire dataset.

Table 5:

Cross-Tabulated Prediction Results: LSTM-Top1 compared to MCNA-Top1

	MCNA-Top1 Correct	MCNA-Top1 Incorrect	Totals
LSTM-Top1 Correct	3,048,742	699,026	3,747,768
LSTM-Top1 Incorrect	158,311	994,009	1,152,320
Totals	3,207,053	1,693,035	4,900,088

From Table 5, it is possible to ascertain the positive predictive effect of using the LSTM model compared to the MCNA model. The top left cell shows how often both models correctly predicted an action, which occurred 3,048,742 times. Looking at the cell to the immediate right, there were 699,026 instances where the LSTM model made a correct prediction while the MCNA model made an incorrect prediction. Conversely, the MCNA model was correct 158,311 times where the LSTM model made an incorrect prediction. Taking the difference of these numbers, the LSTM model had a net positive effect of 540,715 predictions in this dataset.

Figure 3 shows the distribution of LSTM-MAI scores in a histogram for Form F.1. The LSTM-MAI values are centered at 0.668 and follow a normal distribution. In practice, the MAI is considered a “relative” measure of normalness, where MAI ought to be discussed relative to other MAI values within the distribution. One possible application of the relative MAI results would be to analyze the tails of the distribution to better understand what types of behaviors can be found at the low and high ends of the MAI spectrum and distribution.

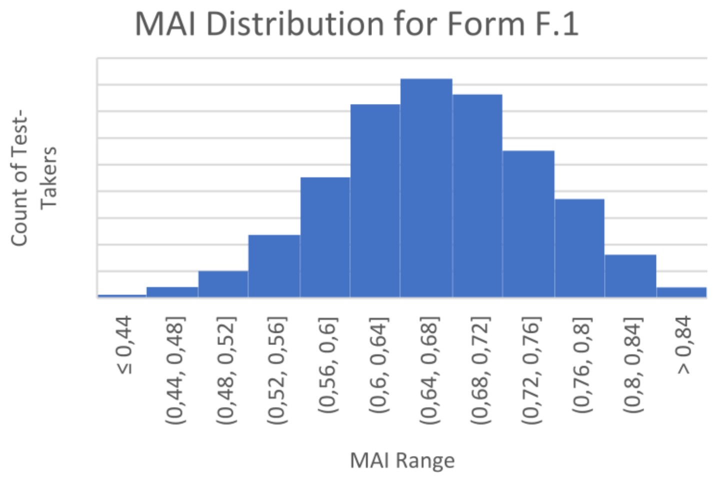


Figure 3.

Histogram of MAI Scores for Form F.1

Analyzing which behavior patterns are prevalent at different MAI levels for this dataset

The overarching framework to compute the proposed LSTM-MAI score is intended to be potentially applied to any computer-based test environment with clickstream logs. For each individual test administration, the nature of student behaviors may vary depending on test-specific and test-taker-population specific characteristics. That being said, it might be illustrative to analyze the kinds of behavior patterns that emerged in the test dataset used for this study, even though such patterns may not generalize to other datasets outside the scope of this paper. In this section, an analysis is conducted that illustrates what kinds of matching and mismatching patterns are observed at different ends of the MAI distribution.

A mismatch occurs when the model's prediction does not match the student's observed action at a timestep. Conversely, a match occurs when the model's prediction does match the student's observed action at a timestep. Table 6 shows the top 3 mismatching and matching predictions across the entire dataset. Each row in Table 6 consists of the following elements in the first data column: "Highest Predicted," "Observed," "Count," "% of all," "Average Probability Predicted," and "Average Probability Observed." *Highest Predicted* refers to the action name that was associated with the highest probability from the prediction model; this action corresponds to the highest predicted value from the model. *Observed* refers to the action name of the actual action taken at a timestep in the clickstream. When the *Highest Predicted* and *Observed* values mismatch, then that means the most likely action predicted by the model did not match the observed action, resulting in a mismatching prediction. *Count* is the number of times the match or mismatch occurred across all possible prediction events, while *% of all* takes the *Count* and divides by the total number of clickstream events across the entire studied dataset. The *Count* and *% of all* metrics are intended to give the reader distributional information about when the prediction model makes matching predictions and mismatching predictions. In the first column, *Average Probability Predicted* refers to the probability that was output by the prediction model for the *Highest Predicted* action. In the case of a mismatch, this is part of the error of that mismatching prediction, since the model's prediction was ultimately not the observed action. The *Average Probability Observed* refers to the model's prediction probability of the *Observed* action. Although the *Observed* action was not the highest probability, the prediction model still assigned some probability to this action (and all other possible actions), so that information is conveyed in this element. This information is intended to give distributional characteristics of the prediction model to better understand how "confident" the model is when it makes predictions. In the final column of Table 6, information about the most common matching patterns is given. In this column, both the *Highest Predicted* action and the *Observed* action were the same, indicating that the most likely prediction from the model matched the observed action in the clickstream. *Count* and *% of all* retain the same interpretation as from the previous column, while the *Average Probability of Action* is just one value because the *Predicted* action is the same as the *Observed* action, compared to the previous column which broke this into two separate probabilities. The information from this table helps the reader obtain a better understanding of the kinds of scores that the prediction model outputs, and how much a matching or mismatching prediction might positively or negatively impact MAI scores.

A match will naturally yield a higher MAI value, as the model's most likely prediction agrees with the observed action. A mismatch will yield a lower MAI value. Since MAI is based on the probability of the action, the degree to which the MAI is affected will depend on how likely the match or mismatch is. Note that the "top 3 mismatches" will generally yield relatively high MAI results; the occurrence of a top 3 mismatch will not necessarily sway an MAI for a clickstream in the atypical direction very strongly. This is because the "top 3 most common mismatches" are relatively likely, even if they are not the most likely action according to the model. For reference, Table 6 also shows the average prediction probability for each match and mismatch; the average probability is used when calculating MAI for a particular clickstream.

Looking at row 1, the most common mismatch occurred when the model predicted NAVIGATION_ITEM_NEXT, but the student was observed selecting a multiple-choice answer. The average MAI value of this mismatch is .23, which will sway an MAI to be lower to a small extent, but not as much as a rarer mismatch would. Looking at row 2, the next most common mismatch in this dataset happened when the student "toggled calculator", but the model predicted a multiple-choice answer. This mismatch had an average MAI value of .19. The 3rd most common mismatch happened when the model predicted a multiple-choice answer event, but the student navigated to the next item, resulting in an average MAI value of .17.

Looking to the last column in the table, the most common match involved correctly predicting NAVIGATION_ITEM_NEXT; 20.44% of all predictions were correctly identified as this navigation event, with an average MAI value of .77. The 2nd most common match was correctly predicting a multiple-choice answer, which comprised 18.69% of all events with an average MAI value of .76. Finally, the 3rd most common match involved finishing a "box drag", which was part of a technology-enhanced-item event. Based on the overall MAI value of .99, this action appears to be a "forced" event within this clickstream, such that this clickstream event appears to always be logged given a particular prior event. Such artifacts can occasionally occur in clickstream data depending on the design of how the clickstream logger records student data.

Table 6:

Top 3 Mismatch and Matches

Rank	Most Common Mismatch Between Predicted and Observed	Most Common Match Between Predicted and Observed
1	Highest Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 229907 % of all: 4.69 Average Probability of Predicted: .71 Average Probability of Observed: .23	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 1001531 % of all: 20.44 Average Probability of Action: .77
2	Highest Predicted: ITEM_MULTIPLE_CHOICE_ANSWER Observed: TOOL_CALCULATOR_TOGGLE Count: 70381 % of all: 1.44 Average Probability of Predicted: .63 Average Probability of Observed: .19	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 915815 % of all: 18.69 Average Probability of Action: .76
3	Highest Predicted: ITEM_MULTIPLE_CHOICE_ANSWER Observed: NAVIGATION_ITEM_NEXT Count: 66877 % of all: 1.36 Average Probability of Predicted: .62 Average Probability of Observed: .17	Predicted and Observed: ITEM_DRAG_BOX_DRAG_END Count: 229506 % of all: 4.68 Average Probability of Action: .99

Analyzing 15 individual clickstream results at different points in the MAI distribution

Table 7 shows results from 15 individual test takers. The examinees are separated into three categories: Low, Mid, and High. Each of the three categories is comprised of five examinees, where one examinee is sampled from each of the 5 test forms. The low category is comprised of examinees approximately two standard deviations below the MAI mean, the mid category is comprised of examinees approximately at the MAI mean, and the high category is comprised of examinees approximately two standard deviations above the MAI mean. Examinees in the “low” category were around .49 MAI; examinees in the mid” category were around .67 MAI; and examinees in the “high” category were around .83 MAI.

The second column in Table 7 shows the most common mismatched prediction, while the third column in Table 7 shows the most common matching prediction. For example, in the first row, the Examinee ID is **Low.F.1.** This indicates that this examinee was in the “low” category, approximately two standard deviations from the MAI mean, sampled from test form #F.1. Examinee Low.F.1 had a total clickstream length of 86. Of the total possible 86 predictions, the most common correct prediction occurred 15 times, where the model correctly predicted a multiple-choice answer would occur. The most common incorrect prediction occurred when predicting a navigation_next, but instead the examinee submitted a multiple-choice answer selection. Looking across all 15 individual clickstreams, in 11 out of 15 of the cases, either NAVIGATION_ITEM_NEXT or ITEM_MULTIPLE_CHOICE was the most common mismatched prediction. The “observed” action from the mismatched prediction varied across 6 different observed actions. Among the high examinee clickstreams, the most common mismatches ranged from 1 to 4 in count, while the low examinee clickstreams ranged from 2 to 17 in count. Conversely, the most common matches ranged from 6 to 24 among the low examinees and ranged from 28 to 55 among the high examinees. Analyzing individual cases can allow for a better understanding of how a particular examinee tended to deviate from the model’s expected behavior.

The resulting trends from Table 7 indicate that there are apparent differences in matches and mismatches across the different levels of “*low*,” “*mid*,” and “*high*” MAI. One notable trend is that the 5 *high* MAI clickstreams only maxed out at 4 occurrences of the most common mismatch. Conversely, for *low* and *mid* MAI clickstreams, *low* maxed out at 17 and *mid* maxed out at 13. This trend can be interpreted in that *high* MAI clickstreams do not exhibit any recurring “mismatch” themes, where the same mismatching action is taken by the student multiple times. Another interesting trend is that within the *low* MAI group, all 5 of the clickstreams had **different** most common mismatching patterns, meaning that the Predicted and the Observed actions within each of the most common mismatched patterns did not duplicate across the sample. This trend can indicate that *low* MAI clickstreams do not necessarily replicate the same mismatching behaviors all the time. In the *mid* MAI category, however, the trend is almost completely reversed. In four out of the five students in the *mid* MAI sample, NAVIGATION_ITEM_NEXT is predicted and ITEM_MULTIPLE_CHOICE_ANSWER is observed as the most common mismatching pattern. This trend could indicate that the "Mid MAI" clickstreams do share more uniformity relative to the *low* MAI clickstreams in mismatching predictions. Finally, in the sample of five *high* MAI clickstreams, the mismatch counts are all very low, indicating there are not many common patterns that are frequently mismatched for this sample. The results shown in this table and study may not necessarily generalize to other datasets; however, such an approach could be employed in analysis of future applications of the proposed methodology to uncover differences in matches and mismatches by MAI levels.

Table 7:

Mismatch and Match Pairs Across MAI Distribution

Examinee ID	Most Common Mismatching Prediction/Observed	Most Common Matching Prediction/Observed	Clickstream Length
Low.F.1	Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 7	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 15	86
Low.F.2	Predicted: ITEM_MULTIPLE_CHOICE_ANSWER Observed: TOOL_ANSWER_MASKING_TOGGLE Count: 17	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 24	163
Low.F.3	Predicted: ITEM_MULTIPLE_CHOICE_ANSWER Observed: NAVIGATION_REVIEW_PANEL_OPEN Count: 4	Predicted and Observed: NAVIGATION_REVIEW_PANEL_CLOSE Count: 6	57
Low.F.4	Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER	55

	Count: 2	Count: 8	
Low.F.5	Predicted: ITEM_MULTIPLE_CHOICE_ANSWER Observed: TOOL_CALCULATOR_TOGGLE Count: 8	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 12	118
Mid.F.1	Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 9	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 26	173
Mid.F.2	Predicted: ITEM_DRAG_BOX_DRAG_START Observed: ITEM_DRAG_BOX_DRAG_END Count: 3	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 30	166
Mid.F.3	Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 8	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 27	123
Mid.F.4	Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 8	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 27	153
Mid.F.5	Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 13	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 54	156
High.F.1	Predicted: NAVIGATION_ITEM_NEXT	Predicted and Observed: NAVIGATION_ITEM_NEXT	91

	Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 2	Count: 28	
High.F.2	Predicted: NAVIGATION_DIRECTIONS_CONTINUE Observed: NAVIGATION_ACCESS_CODE_SUBMIT Count: 2	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 28	95
High.F.3	Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 4	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 28	92
High.F.4	Predicted: ITEM_DRAG_BOX_DRAG_START Observed: NAVIGATION_ITEM_NEXT Count: 1	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 30	89
High.F.5	Predicted: ITEM_DRAG_BOX_DRAG_START Observed: NAVIGATION_ITEM_NEXT Count: 1	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 55	154

Analyzing the ends of the MAI distribution

Table 8 shows the results for the individual clickstream that achieved the lowest MAI score in the dataset. With a total of 56 clicks, this clickstream received an MAI score of 0.35. Table 9 shows the results for the clickstream that had the highest MAI score. The highest MAI result had a total of 83 clicks, with an MAI score of 0.90. Looking at the percentages in both tables, it is evident that the highest MAI result had a larger proportion of observed actions using NAVIGATION_ITEM_NEXT (34.94 %), and ITEM_MULTIPLE_CHOICE_ANSWER (33.73%), compared to 17.86 % and 19.64

% in the lowest MAI result. The highest MAI result conformed to the most common behavior patterns exhibited among all test-takers, while the lowest MAI result often deviated from expected behaviors.

Table 8:

Top Predicted and Observed Actions for Lowest MAI Result

Rank	Observed Action	N-count	Percent	Predicted Action	N-count	Percent
1	ITEM_MULTI-PLE_CHOICE_AN-SWER	11	19.64	ITEM_MULTI-PLE_CHOICE_AN-SWER	17	30.36
2	NAVIGA-TION_ITEM_NEXT	10	17.86	NAVIGA-TION_ITEM_NEXT	9	16.07
3	TOOL_CALCULA-TOR_TOGGLE	10	17.86	TOOL_CALCULA-TOR_OPEN	9	16.07

Table 9:

Top Predicted and Observed Actions for Highest MAI Result

Rank	Observed Action	N-count	Percent	Predicted Action	N-count	Percent
1	NAVIGA-TION_ITEM_NEXT	29	34.94	NAVIGA-TION_ITEM_NEXT	27	32.53
2	ITEM_MULTI-PLE_CHOICE_AN-SWER	28	33.73	ITEM_MULTI-PLE_CHOICE_AN-SWER	27	32.53
3	ITEM_DRAG_BOX_D RAG_START	6	7.23	ITEM_DRAG_BOX _DRAG_START	7	8.43

Table 10 further describes the nature of the mismatching predictions for the lowest MAI result. The top 3 most common mismatching predictions sum to a total of 11 mismatches. Since the lowest MAI result had a total of 56 clicks, there is an 11/56 error rate from the top 3 mismatches. The most common prediction error for this click-stream occurred when predicting the student would OPEN the calculator, while the student instead TOGGLED the calculator tool. Opening a tool in this test application requires two steps. The first step would be to “click” on a calculator icon. After this click occurs, the calculator does not open yet. A separate menu pops up, allowing the student to select the calculator type; in this test, the only option available was labeled

"Scientific". Therefore, clicking on the calculator icon was a TOGGLE, and then clicking "Scientific" from the options was to OPEN the calculator. The vast majority of use cases "toggles" the calculator, and students then immediately "open" the calculator. However, this low MAI result had repeated toggles one after the other, meaning the student was perhaps confused about how to open the calculator, potentially not realizing that a second selection was needed to open the calculator. This could mean that the test application user experience could potentially be improved to remove this kind of confusion. From a practical standpoint, the "atypical" MAI score here was clearly indicative of something abnormal happening, perhaps indicating confusion by the student. This analysis could potentially be used to inform a possible modification to the testing environment to streamline access to the calculator, which could potentially provide a better test experience to confused students during a high-stakes test.

Table 10:

Top Mismatch and Matching Patterns of Lowest MAI Result

Rank	Most Common Mismatch Pattern	Most Common Match Pattern
1	Predicted: TOOL_CALCULATOR_OPEN Observed: TOOL_CALCULATOR_TOGGLE Count: 5	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 8
2	Predicted: ITEM_MULTIPLE_CHOICE_ANSWER Observed: NAVIGATION_ITEM_NEXT Count: 3	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 5
3	Predicted: NAVIGATION_ITEM_NEXT Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 3	Predicted and Observed: TOOL_CALCULATOR_OPEN Count: 4

Table 11 shows the most common mismatches and matches for the highest MAI result. The top 3 mismatches only occurred once each, thus comprising an error rate of just 3/86 from the top 3 mismatches, much lower than the result from the lowest MAI result. The majority of clicks (54/86) comprised NAVIGATION_ITEM_NEXT and

ITEM_MULTIPLE_CHOICE_ANSWER. Given that this clickstream had the highest MAI result in the data, this examinee matched the model's expected behavior the most and rarely deviated from common actions.

Table 11:

Top Mismatch and Matching Patterns of Highest MAI Result

Rank	Most Common Mismatch Pattern	Most Common Match Pattern
1	Predicted: NAVIGATION_REVIEW_PANEL_OPEN Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 1	Predicted and Observed: NAVIGATION_ITEM_NEXT Count: 27
2	Predicted: ITEM_DRAG_BOX_DRAG_START Observed: NAVIGATION_ITEM_NEXT Count: 1	Predicted and Observed: ITEM_MULTIPLE_CHOICE_ANSWER Count: 27
3	Predicted: ITEM_TILE_BOX_DRAG_START Observed: NAVIGATION_ITEM_NEXT Count: 1	Predicted and Observed: ITEM_DRAG_BOX_DRAG_START Count: 6

MAI Comparisons

On its own, MAI summarizes to what extent a test-taker performs a predictable action or pattern of actions, weighted probabilistically. MAI is primarily best interpreted as a *relative* measure of typicalness compared to the population, rather than an absolute measure of something concrete. This property would ideally be utilized as part of a multiple-methods approach towards monitoring test irregularities, as MAI could be considered one more index to rely on in monitoring testing administrations at large. Drawing comparisons from MAI to other indicators would be a primary method to interpret such relative results.

Comparisons to Scale Sores

Each test-taker was assigned to take one of the five online test forms, denoted as F.1 through F.5. A single form is comprised of two testing sessions, denoted as Session 1 and Session 2. Based on response patterns from both Session 1 and Session 2 combined, each test-taker was assigned a scale score that ranges between 200 to 400.

Table 12:

Descriptive statistics of scale scores and MAI on sessions

		Scale Score	MAI Score	
		Both Sessions	Session 1	Session 2
F.1	Mean	265.52	0.64	0.69
	S. D.	33.07	0.08	0.08
F.2	Mean	267.02	0.62	0.69
	S. D.	34.08	0.08	0.08
F.3	Mean	266.63	0.63	0.69
	S. D.	33.05	0.08	0.08
F.4	Mean	265.26	0.64	0.68
	S. D.	33.39	0.08	0.08
F.5	Mean	234.49	0.65	0.70
	S. D.	27.70	0.09	0.09

Table 12 shows the descriptive statistics for scale scores and MAI scores by form and session. Forms F.1 to F.4 have similar means and standard deviations. Form F.5 is a read aloud form, and scale scores were slightly lower relative to the scores from the other four forms. Read aloud is a non-standard accommodation applied only to students with special needs. Form F.5 is included in the current study for a general analysis, but the difference between form F.5 and other forms is not thoroughly investigated in the current study. In the following analysis, Form F.1 will be mainly used as an example to show the relationship between MAI scores and other variables. For Form F.1, the scale scores have a mean of 265.52 and a standard deviation of 33.07. The average MAI score is slightly lower in test session 1 (0.64) than in test session 2 (0.69). Due to the distributional difference between test session 1 and session 2, the relationship between MAI scores and scale scores will be analyzed by test sessions separately.

Figure 4 plots MAI across the deciles of the scale score distribution for Form F.1. A decile splits the distribution of scale scores into 10 ordered groups, with each decile comprising 10% of the total count of test-takers. The first decile is comprised of the lowest scoring 10% of test-takers, while the last and tenth decile considers the highest scoring 10% of test-takers. The x-axis of the figure shows the range of scores that are included in each decile group. Session 1 and Session 2 are plotted separately. For Session 1 results, there appears to be a slightly decreasing trend in median MAI scores up until about the 6th decile group. From the 7th through 10th decile, there is a slightly increasing trend. For Session 2 results, the slightly decreasing trend goes from the 1st through the 8th decile, and then there appears to be a slight increase in MAI scores in the 9th and 10th decile. These results indicate that the relationship between MAI and performance does not appear to be linear. It is also of note that the inter-quartile ranges of each box plot span a relatively wide range, indicating that there is not necessarily a strong or obvious relationship between MAI and scale score, other than the slight dip observed in the distributions from both test sessions.

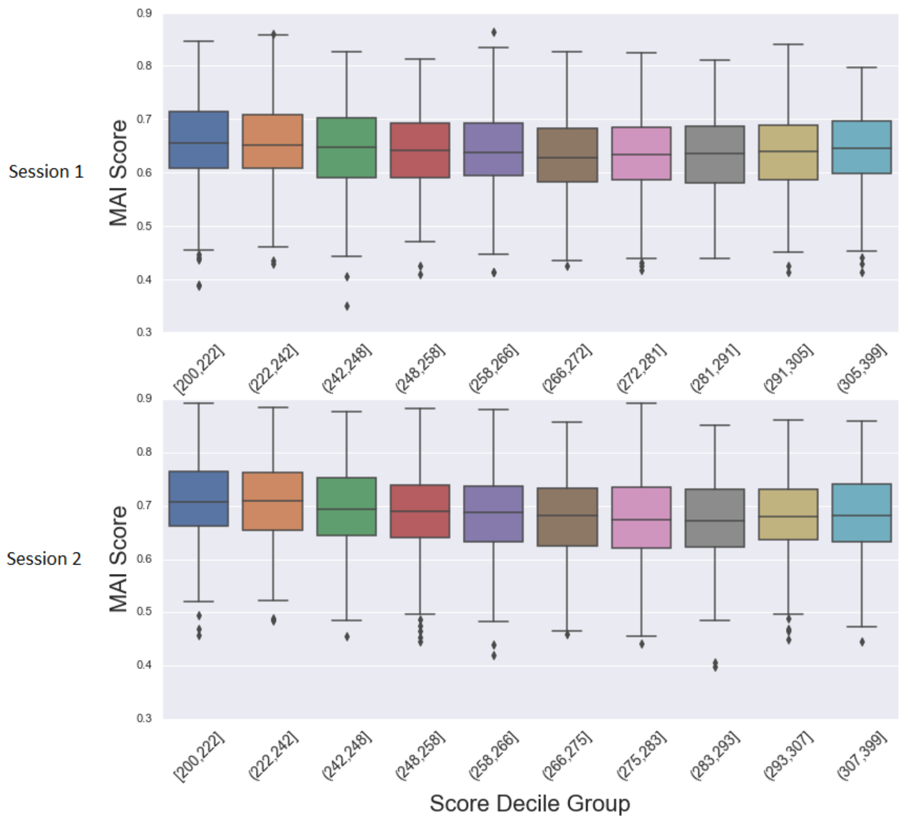


Figure 4.

MAI scores against scale score decile groups for form F.1

Comparing MAI Score to Aberrance Detection Indices

N2 and NC2 (Bishop & Egan, 2017) are two common aberrance indices (Ranger, Schmidt, & Wolgast, 2020) that are relatively straightforward to compute. N2 indicates the number of items on which an examinee changes his/her response at least once. NC2 indicates the number of items on which a test-taker changes his/her response from wrong to right at the last attempt. Other aberrance indices focus on response-time analysis. Based on the lognormal model for response times (van der Linden & Guo, 2008), Li et al. (2018) introduced the statistical indices Z_l and Z_s . Z_l detects whether an examinee has too short or too long response times on each item during the test. Z_s, similar to Z_l, identifies extreme response times, taking person speed into consideration in addition to the distribution of response times on each item. High values of both Z_l and Z_s indicates aberrant response time on an item. The extent of aberrance of an examinee's response time pattern is represented by the mean of Z_l and Z_s across all items. Furthermore, Z_l and Z_s could be calculated based on the response time of examinee's last attempt or the total response time on each item. Therefore, four sets of aberrant RT indices were calculated: Z_l_lastRT, Z_s_lastRT, $Z_l\_totalRT$ and $Z_s\_totalRT$.

Table 13:

Correlation Coefficients Between MAI scores and Traditional Aberrance Detection Indices

	Form F.1, Session One			Form F.1, Session Two		
	MAI_Score	Top1_ Accuracy	Top3_ Accuracy	MAI_Score	Top1_ Accuracy	Top3_ Accuracy
N2	-.35	-.38	-.14	-.39	-.44	-.08
NC2	-.25	-.27	-.13	-.31	-.33	-.10
Z_s_lastRT	-.29	-.31	-.18	-.22	-.25	-.08
Z_l_lastRT	-.20	-.21	-.14	-.10	-.14	-.03
$Z_s\_totalRT$	.02	.00	-.02	.05	.05	-.03
$Z_l\_totalRT$	.12	.10	.04	.16	.15	.04

From Table 13, among the traditional aberrance detection indices, both N2 and NC2 have a weak negative correlation with MAI for both sessions of Form F.1. The correlation between N2 and MAI scores is the highest among the tested statistics; this could be somewhat expected given that both N2 and the current MAI approach do not consider response correctness or response times, while the other models do. The negative correlation shows that, on average, examinees who change answers more frequently have lower MAI scores. The correlation coefficients between MAI scores and response time indices are relatively small. The correlation with Z_s_lastRT and Z_s_lastRT are negative, while the correlation with $Z_l\_totalRT$ and $Z_s\_totalRT$ are very slightly positive. The highest correlation coefficient belongs to Z_s_lastRT, indicating that examinees who have higher response time aberrance on their last attempt on an item tended to have slightly lower MAI values. The current calculation of MAI does not incorporate response time or timing between actions. In future work, if timings were to be included as part of the MAI computation, correlations with aberrance indices that are related to response times could increase.

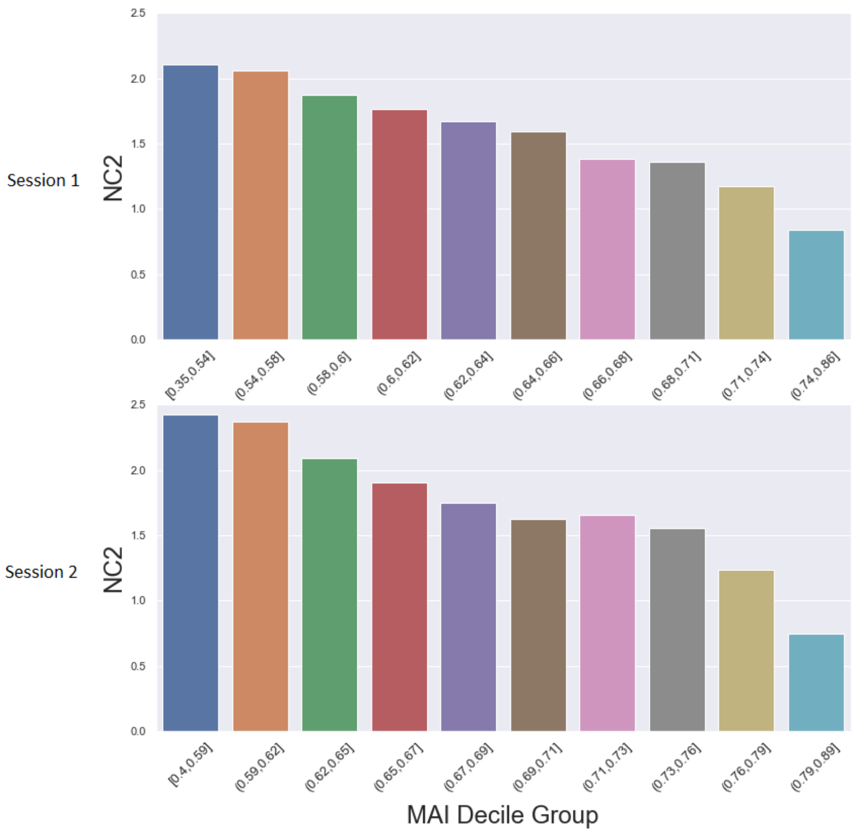


Figure 5.

NC2 Index Across MAI Deciles for Form F.1

Figure 5 plots the average NC2 value across the 10 deciles of MAI scores for Form F.1. Note the downward trend, where the average NC2 value starts at 2.1 in decile 1 and decreases to 0.8 in decile 10 for Session One. The average NC2 value decreases from 2.4 to 0.7 for Session Two. This trend shows that lower MAI scores tended to have higher NC2 values across the entire distribution of MAI scores. In interpretive terms, this means that clickstreams that were identified as relatively more atypical by their MAI values tended to also be relatively more aberrant according to their NC2 values.

The comparisons discussed in this section are a small subset of all possible comparisons to aberrance indices, intended to illustrate the type of analysis that could be done when using an index like MAI in practice. Since each test administration is unique, establishing relationships between indices should be evaluated on a dataset-by-dataset basis, as relationships found in one dataset may not generalize when evaluating other datasets.

Discussion

In this study, a behavior prediction model that predicts test-taking behaviors based on clickstream data was illustrated. The proposed LSTM approach demonstrated substantial accuracy improvements over the baseline MCNA model in terms of predicting the next action given prior actions. The model utilizes past clickstream behaviors as well as a snapshot of the current test state environment. An operational definition of normality and atypical behaviors was established based on the predictions of the model, whereby clickstreams that demonstrate low MAI could be considered atypical since the actions in those clickstreams are predicted to be less likely to happen during testing. When analyzing the behavior patterns of low MAI students, it was found in this study that opening the calculator was a common source of "wasted clicks" for many clickstreams with low MAI. This finding is a practical result which can inform test designers that there may be opportunities to optimize and improve the calculator use workflow for the test application that generated the student clickstreams, as some students may be wasting time or getting stuck on a task that should be unrelated to the test-taking experience. In future studies, analyzing different levels of MAI can help identify common wanted and unwanted behavior patterns for different types of test-takers stratified by different levels of MAI scores. The proposed method is one general approach towards utilizing process data to analyze normal and atypical behaviors unique to a particular test administration.

Identifying testing irregularities in large-scale testing environments is typically approached by using multiple methods, starting from ensuring item bank integrity to ensuring physical test-location uniformity as well as analyzing test-taker process data, such as their response choices, timing, performance irregularities, and behavioral clickstream data. In practice, identifying cheating behaviors is incredibly difficult. In many test administrations, there may actually be zero substantiated occurrences of cheating, making this a sparse phenomenon that is not easily modelled. Despite how rare the objective is, test administrators still want to invest in multiple monitoring techniques to feel confident that testing irregularities are identified and understood. It is within this scope and need that the proposed behavior modeling and corresponding MAI calculation are situated, allowing for a simple statistical index that quantifies the normalness of each examinee’s actions. This simple statistical framework allows for outlier values to be further scrutinized. In this post-hoc analysis, investigators can better understand the typical and atypical behavior patterns that arise during testing. In conjunction with other methods that are typically used, such as the well-known person fit indexes (Reise & Due, 1991) and “rapid guessing” indicators (Wise & Kong, 2005), the MAI score can provide a further vector of monitoring information to create a case that test results are typical.

Limitations and Future Directions

The approach of training directly on granular clickstream data has the benefit of incorporating all behavior patterns and allows the model to freely learn about normal behavior patterns across many different test-taker archetypes and styles. However, this freedom has the drawback that interpreting model results are less straightforward compared to models where input features are more strictly defined. The LSTM model does not explain why one individual’s clickstream achieves a high MAI and a different one achieves a low MAI. Since the model depends entirely on the training data and the distribution of behaviors in the training data, interpretations about what “low” or “high” MAI means in terms of actual behaviors will always depend on post-hoc analysis of examinee behavior clickstreams at varying levels of MAI. In all circumstances, a low MAI indicates that the behaviors of an individual were less expected relative to the population of other test-takers.

Future studies could be conducted to investigate and improve MAI in several directions. First, the effectiveness of MAI in detecting potential cheating behaviors could be further evaluated with simulated data or labeled contaminated data in future studies. Secondly, the general approach towards modeling behaviors can be extended with additional data when available. The use of timing and timestamps, for example, was not analyzed in the current study, but could have a meaningful impact on identifying noteworthy behavior patterns and related analyses in future work. Finally, the potential connections between MAI and other indicators, including performance and aberrance indices, could also be further analyzed, focusing on the specific monitoring needs of the interested party.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., . . . Jia, Y. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Retrieved from tensorflow.org

Asseburg, R., & Frey, A. (2013). Too hard, too esay, or just right? The relationship between effort or boredom and ability-difficulty fit. *Psychological Test and Assessment Modeling, 55*(1), 92-104. Retrieved from https://psycnet.apa.org/record/2013-18917-006

Banerjee, A., & Ghosh, J. (2011). Clickstream Clustering Using Weighted Longest Common Subsequences. *Proc. of the Web Mining Workshop in CDM.* Retrieved from https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.29.427&rep=rep1&type=pdf

Bishop, S., & Egan, K. (2017). Detecting erasures and unusual gain scores: Understanding the status quo. In *G. J. Cizek & J. A. Wollack (Eds.). Handbook of Quantitative Methods for Detecting Cheating on Tests* (pp. 193-213). Washington, DC: Routledge. Retrieved from https://www.taylorfrancis.com/chapters/edit/10.4324/9781315743097-10/detecting-erasures-unusual-gain-scores-scott-bishop-karla-egan

Brownlee, J. (2017, June 28). *How to Prepare Sequence Prediction for Truncated BPTT in Keras*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/truncated-backpropagation-through-time-in-keras/

Brownlee, J. (2020, October 19). *Softmax Activation Function with Python*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/softmax-activation-function-with-python/

Brownlee, J. (2021, January 13). *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

Castelvecchi, D. (2016, October 05). *Can we open the black box of AI?* (Nature) Retrieved from https://www.nature.com/news/can-we-open-the-black-box-of-ai-1.20731

Chang, Z., Zhang, Y., & Chen, W. (2018). Effective Adam-Optimized LSTM Neural Network for Electricity Price Forecasting. *IEEE*. Retrieved from https://ieeexplore.ieee.org/document/8663710

Chollet, F., & Others. (2015). Keras. Retrieved from https://github.com/keras-team/keras

Collins, J. M. (1996). A New Statistical Parser Based on Bigram Lexical Dependencies. *Association for Computational Linguistics*, 184-191. Retrieved from https://aclanthology.org/P96-1025/

Drasgrow, F., Levine, V. M., & Williams, A. E. (1985). Appropriateness measurement with polychotomous item response models and standardized indices. *British Journal of Mathematical and Statistical Psychology, 38*, 67-86. Retrieved from https://psycnet.apa.org/record/1985-24320-001

Greff, K., Srivastava, R. K., Koutnik, J., Steunebrink, B. R., & Schmidhuber, J. (2016, July 8). LSTM: A Search Space Odyssey. 2222-2232. IEEE Transactions o nNeural Networks and Learning Systems. Retrieved from https://ieeexplore.ieee.org/abstract/document/7508408

Gunduz, S., & Ozsu, M. T. (2003). A Web page prediction model based on click-stream tree representation of user behavior. *Proc. of KDD.* Retrieved from https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.3.1067&rep=rep1&type=pdf

Heer, J., & Chi, E. H. (2002). Mining the Structure of User Activity using Cluster Stability. *Proc. of the Workshop on Web Analytics, SIAM Conference on Data Mining.* Retrieved from https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.16.3665&rep=rep1&type=pdf

Jiao, H., He, Q., & Veldkamp, B. P. (2021). *Process Data in Educational and Psychological Measurement, 2nd Edition.* Frontiers in Psychology, Frontiers in Education and Frontiers in Applied Mathematics and Statistics. doi:10.3389/978-2-88974-138-0

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations*. Retrieved from https://arxiv.org/abs/1412.6980

Li, Z., Wall, N., & Tang, H. (2018). A new statistic for detecting aberrant response time patterns in large-scale assessments. *Paper presented at the annual meeting of the National Council on Measurement in Education (NCME).* New York, NY. Retrieved from https://www.emetric.net/Content/pdf/Manuscript-Response%20Aberrance.pdf

Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems*. Retrieved from https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

Mikolov, T., Karafiát, M., Burget, L., & Khudanpur, S. (2010). Recurrent neural network based language model. *Interspeech*. Retrieved from http://www.fit.vutbr.cz/research/groups/speech/servite/2010/rnnlm_mikolov.pdf

Olah, C. (2015, August 27). *Understanding LSTM Networks*. Retrieved from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Pham, V., Bluche, T., Kermorvant, C., & Louradour, J. (2014). Dropout improves recurrent neural networks for handwriting recognition. *Frontiers in Handwriting Recognition (ICFHR)*, 285-290. Retrieved from https://arxiv.org/abs/1312.4569

Ranger, J., Schmidt, N., & Wolgast, A. (2020). The detection of cheating on e-Exams in higher education-The performance of several old and some new indicators. *frontiers in Psychology, 11*. Retrieved from https://doi.org/10.3389/fpsyg.2020.568825

Reise, S. P., & Due, A. M. (1991). The influence of test characteristics on the detection of aberrant response patterns. *Applied Psychological Measurement, 15*(3), 217-226. Retrieved from https://doi.org/10.1177/014662169101500301

Su, Q., & Chen, L. (2015). A Method for Discovering Clusters of E-commerce Interest Patterns using Click-stream Data. *ECRA*, *14,* pp. 1-13. Retrieved from https://doi.org/10.1016/j.elerap.2014.10.002

Sundermeyer, M., Schlüter, R., & Ney, H. (2012). LSTM Neural Networks for Language Modeling. *Interspeech.* Retrieved from https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.248.4448&rep=rep1&type=pdf

Tang, S., Peterson, J., & Pardos, Z. (2017). Predictive Modeling of Student Behavior Using Granular Large Scale Action Data from a MOOC. Retrieved from https://www.researchgate.net/publication/315874906_Predictive_Modelling_of_Student_Behaviour_Using_Granular_Large-Scale_Action_Data

Ulitzsch, E., He, Q., Ulitzsch, V., Molter, H., Nichterlein, A., Neidermeier, R., & Pohl, S. (2021). Combining Clickstream Analyses and Graph-Modeled Data Clustering for Identifying Common Response Processes. *Psychometrika*, 190-214. doi:10.1007/s11336-020-09743-0

van der Linden, J. W., & Guo, F. (2008). Bayesian procedures for identifying aberrant response-time patterns in adaptive testing. *Psychometrika, 73*(3), 365-384. Retrieved from https://doi.org/10.1007/s11336-007-9046-8

Wang, G., Zhang, X., Tang, S., Wilson, C., Zheng, H., & Zhao, B. Y. (2017). *Clickstream User Behavior Models.* ACM Transactions on the Web. Retrieved from https://doi.org/10.1145/3068332

Werbos, P. J. (1988). Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks,* 339-356. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/089360808890007X

Wise, S., & DeMars, C. (2006). An application of item response time: The effort-moderated IRT model. *Journal of Educational Measurement,* 19-38. Retrieved from https://www.jstor.org/stable/20461807

Wise, S., & Kong, X. (2005). Response time effort: A new measure of examine motivation in computer-based tests. *Applied Measurement in Education*, 163-183. Retrieved from https://doi.org/10.1207/s15324818ame1802_2

Appendix

Table 14:

Test State (Concatenation of Tools, Widgets, and Accommodations)

Standard Tools	
1	Basic Calculator
2	Custom Masking
3	Dictionary
4	Graphing Calculator
5	Protractor
6	Ruler
7	Scientific Calculator
8	Standard Calculator
Standard Universal Tools	
9	Eliminator
10	Guideline
11	Screen Zoom
12	Sketch And Highlight
13	Answer Masking
Standard Widgets	
14	Association
15	Bar chart
16	Browser
17	CodeInput
18	Connections
19	Data Table
20	Draggable
21	Electronic Circuit
22	Fill in the blank
23	GoalBox
24	Grapher
25	GridWidget

26	HotSpot
27	LineChart
28	MathInput
29	MathResponse
30	MediaBox
31	Multiple Choice
32	MultipleChoiceItem
33	OpenEndedItem
34	Partition Number Line
35	Pie Chart
36	Rearrange
37	Rich text response
38	Select and Change
39	Select from drop down
40	SelectPoint
41	Selectable Text
42	Selection Number Line
43	SketchPad
44	Slider
45	SpeechRecorder
46	Stimulus
47	TabbedBox
48	Text response
49	TextBox
50	TileBox
Standard Accommodation	
51	allowaccessibilitymodetesting
52	answermasking
53	aslvideoela
54	aslvideomath
55	assistivetechnologydevicepresentation
56	assistivetechnologydevicesresponse

57	basiccalculator
58	braillenotetaker
59	braillewriter
60	colorcontrast
61	commercialwordtoworddictionary
62	customizedduallanguagedictionary
63	custommasking
64	elattsspanish
65	foregroundcolour
66	headphonesnoisebuffer
67	humanreadereng
68	humanreadersp
69	humansigner
70	humansignertestdirections
71	invertcolourchoice
72	magnification
73	mathtools
74	maththsenglish
75	mathttsspanish
76	nativelanguageaccommodations
77	picturedictionary
78	pocketwordtowordtranslator
79	readaloudtoself
80	refreshablebraille
81	scientificcalculator
82	screenreader
83	scribeconstructedresponse
84	scribeselectedresponse
85	spanishlanguageversion
86	speechtotext
87	spokenell
88	spokens504

89	spokenspanish
90	wordprediction
91	wordpredictionemb
92	Read Aloud US History
93	answer masking
94	color contrast
95	ellaccommela
96	ellaccommmath
97	ellaccommscience
98	ellaccommushistory
99	general masking
100	iepaccommela
101	iepaccommmath
102	iepaccommscience
103	iepaccommushistory
104	line reader
105	read aloud math
106	read aloud reading
107	read aloud science
108	read aloud social studies
109	read aloud us history
110	readaloudmathspanish
111	readaloudsciencespanish
112	readaloudushistoryspanish
113	s504accommela
114	s504accommmath
115	s504accommscience
116	s504accommushistory
117	screen zoom
118	turn off all universal tools

Table 15:

Click_action List

0	NULL_RECORD
1	ALERT_DIRECTIONS_EXIT
2	ALERT_DIRE_WARNING_CLOSE
3	ALERT_DIRE_WARNING_RETRY
4	ALERT_FINAL_SCORE_UNAVAILABLE_CLOSE
5	ALERT_INACTIVITY_EXIT
6	ALERT_LOCK_TIMEOUT_EXIT
7	ALERT_OFFLINE_WARNING_CLOSE
8	ALERT_OFFLINE_WARNING_READ
9	ALERT_PROCTOR_PASSWORD_SUBMIT
10	ALERT_PROFILE_EXIT
11	ALERT_SIMULTANEOUS_USER_CLOSE
12	ALERT_START_TEST_ERROR_CLOSE
13	ALERT_START_TEST_ERROR_RETRY
14	ALERT_TIMEOUT_CLOSE
15	ALERT_TTS_FAILURE_CLOSE
16	ITEM_BOOKMARK_OFF
17	ITEM_BOOKMARK_ON
18	ITEM_CLEAR_CANCEL
19	ITEM_CLEAR_COMMIT
20	ITEM_CLEAR_START
21	ITEM_CONNECTION_match
22	ITEM_CONNECTION_unmatch
23	ITEM_DRAG_BOX_DRAG_END
24	ITEM_DRAG_BOX_DRAG_START
25	ITEM_HOTSPOT_select
26	ITEM_HOTSPOT_unselect
27	ITEM_MATH_EQUATION_CANCEL
28	ITEM_MATH_EQUATION_OPEN
29	ITEM_MATH_EQUATION_SELECT
30	ITEM_MULTIPLE_CHOICE_ANSWER

31	ITEM_MULTIPLE_CHOICE_Eliminate
32	ITEM_MULTIPLE_CHOICE_UnEliminate
33	ITEM_OPEN_ENDED_BLUR
34	ITEM_OPEN_ENDED_BOLD
35	ITEM_OPEN_ENDED_COPY
36	ITEM_OPEN_ENDED_CUT
37	ITEM_OPEN_ENDED_FOCUS
38	ITEM_OPEN_ENDED_ITALIC
39	ITEM_OPEN_ENDED_PASTE
40	ITEM_OPEN_ENDED_REDO
41	ITEM_OPEN_ENDED_SPELLCHECK_OFF
42	ITEM_OPEN_ENDED_SPELLCHECK_ON
43	ITEM_OPEN_ENDED_UNDERLINE
44	ITEM_OPEN_ENDED_UNDO
45	ITEM_SELECTTEXT_select
46	ITEM_SELECTTEXT_unselect
47	ITEM_SELECT_DROP_DOWN_select
48	ITEM_STIMULUS_SELECT
49	ITEM_STIMULUS_TOGGLE
50	ITEM_TILE_BOX_DRAG_END
51	ITEM_TILE_BOX_DRAG_START
52	NAVIGATION_ACCESS_CODE_CANCEL
53	NAVIGATION_ACCESS_CODE_SUBMIT
54	NAVIGATION_ACCOMMODATION_OPTIONS_CONTINUE
55	NAVIGATION_DIRECTIONS_ACCOMMODATION_CLOSE
56	NAVIGATION_DIRECTIONS_ACCOMMODATION_OPEN
57	NAVIGATION_DIRECTIONS_CONTINUE
58	NAVIGATION_FINAL_SCORE_CLOSE
59	NAVIGATION_ITEM_BACK
60	NAVIGATION_ITEM_JUMP
61	NAVIGATION_ITEM_NEXT
62	NAVIGATION_LOCK_RESUME
63	NAVIGATION_LOCK_SIGN_OUT

64	NAVIGATION_PAUSE_CANCEL
65	NAVIGATION_PAUSE_COMMIT
66	NAVIGATION_PAUSE_LOCK
67	NAVIGATION_PROFILE_CHOOSE
68	NAVIGATION_PROFILE_LOGIN
69	NAVIGATION_REVIEW_PANEL_CLOSE
70	NAVIGATION_REVIEW_PANEL_OPEN
71	NAVIGATION_SECTION_DENIED_CLOSE
72	NAVIGATION_SECTION_WARNING_CANCEL
73	NAVIGATION_SECTION_WARNING_COMMIT
74	NAVIGATION_SHOW_ANSWER_CLOSE
75	NAVIGATION_SHOW_ANSWER_OPEN
76	NAVIGATION_SHOW_ANSWER_SELECT
77	NAVIGATION_TURN_IN_CANCEL
78	NAVIGATION_TURN_IN_COMMIT
79	NAVIGATION_TURN_IN_START
80	NAVIGATION_trigger_START
81	TOOL_ANSWER_MASKING_DISABLE
82	TOOL_ANSWER_MASKING_ENABLE
83	TOOL_ANSWER_MASKING_TOGGLE
84	TOOL_CALCULATOR_CLOSE
85	TOOL_CALCULATOR_OPEN
86	TOOL_CALCULATOR_TOGGLE
87	TOOL_COLOR_SCHEME_DISABLE
88	TOOL_COLOR_SCHEME_ENABLE
89	TOOL_COLOR_SCHEME_OFF
90	TOOL_COLOR_SCHEME_ON
91	TOOL_COLOR_SCHEME_TOGGLE
92	TOOL_CUSTOM_MASKING_CLOSE
93	TOOL_CUSTOM_MASKING_DISABLE
94	TOOL_CUSTOM_MASKING_ENABLE
95	TOOL_CUSTOM_MASKING_OPEN
96	TOOL_CUSTOM_MASKING_TOGGLE

97	TOOL_DICTIONARY_CLOSE
98	TOOL_DICTIONARY_OPEN
99	TOOL_DICTIONARY_TOGGLE
100	TOOL_Eliminator_DISABLE
101	TOOL_Eliminator_ENABLE
102	TOOL_GUIDELINE_CLOSE
103	TOOL_GUIDELINE_DISABLE
104	TOOL_GUIDELINE_ENABLE
105	TOOL_GUIDELINE_OPEN
106	TOOL_MASKING_DISABLE
107	TOOL_MASKING_ENABLE
108	TOOL_NOTEPAD_BLUR
109	TOOL_NOTEPAD_CLOSE
110	TOOL_NOTEPAD_OPEN
111	TOOL_PROTRACTOR_CLOSE
112	TOOL_PROTRACTOR_OPEN
113	TOOL_REFERENCES_CLOSE
114	TOOL_REFERENCES_OPEN
115	TOOL_REFERENCES_TOGGLE
116	TOOL_REVERSE_CONTRAST_DISABLE
117	TOOL_REVERSE_CONTRAST_ENABLE
118	TOOL_REVERSE_CONTRAST_OFF
119	TOOL_REVERSE_CONTRAST_ON
120	TOOL_RULER_CLOSE
121	TOOL_RULER_OPEN
122	TOOL_RULER_TOGGLE
123	TOOL_SIGNING_DISABLE
124	TOOL_SIGNING_ENABLE
125	TOOL_SKETCH_CLOSE
126	TOOL_SKETCH_OPEN
127	TOOL_SKETCH_SELECT
128	TOOL_TEXT_HIGHLIGHT_CANCEL
129	TOOL_TEXT_HIGHLIGHT_CANCEL_ALL

130	TOOL_TEXT_HIGHLIGHT_SELECTED
131	TOOL_TEXT_HIGHLIGHT_TOGGLE
132	TOOL_TTS_DISABLE
133	TOOL_TTS_ENABLE
134	TOOL_TTS_OFF
135	TOOL_TTS_ON
136	TOOL_TTS_RATE
137	TOOL_TTS_SELECT
138	TOOL_TTS_VOLUME
139	TOOL_ZOOM_DECREASE
140	TOOL_ZOOM_DISABLE
141	TOOL_ZOOM_ENABLE
142	TOOL_ZOOM_INCREASE
143	TOOL_ZOOM_RESET
144	TOOL_ZOOM_SET
145	NAVIGATION_REVIEW_PANEL_START
146	NAVIGATION_TOOLBAR_START
147	TOOL_TTS_PAUSE
148	TOOL_TTS_PLAY
149	TOOL_TTS_RESUME
150	TOOL_TTS_SKIP
151	TOOL_TTS_STOP

Table 16:

Distribution of Clicks and Predictions across entire dataset

Action	Observed N-count	%	Prediction N-count	%
Total	4900088	100	4900088	100
ITEM_MULTIPLE_CHOICE_ANSWER	1238857	25.28	1280009	26.12
NAVIGATION_ITEM_NEXT	1168210	23.84	1338722	27.32
ITEM_DRAG_BOX_DRAG_END	240823	4.91	230116	4.7
ITEM_DRAG_BOX_DRAG_START	228877	4.67	274523	5.6

TOOL_ANSWER_MASKING_TOGGLE	199107	4.06	155601	3.18
TOOL_CALCULATOR_TOGGLE	181773	3.71	84782	1.73
NAVIGATION_REVIEW_PANEL_CLOSE	161839	3.3	156002	3.18
NAVIGATION_REVIEW_PANEL_OPEN	161220	3.29	95612	1.95
ITEM_TILE_BOX_DRAG_END	136896	2.79	134418	2.74
ITEM_TILE_BOX_DRAG_START	134475	2.74	168177	3.43
NAVIGATION_ITEM_BACK	120292	2.45	82532	1.68
TOOL_CALCULATOR_OPEN	117182	2.39	142565	2.91
TOOL_CALCULATOR_CLOSE	116949	2.39	105236	2.15
NAVIGATION_ITEM_JUMP	66372	1.35	69357	1.42
TOOL_SKETCH_SELECT	56437	1.15	64968	1.33
ITEM_SELECT_DROP_DOWN_select	51261	1.05	49090	1.0
NAVIGATION_ACCESS_CODE_SUBMIT	42926	0.88	42025	0.86
NAVIGATION_TURN_IN_START	40195	0.82	46508	0.95
NAVIGATION_PROFILE_CHOOSE	40023	0.82	58001	1.18
NAVIGATION_TURN_IN_COMMIT	39564	0.81	39506	0.81
NAVIGATION_DIRECTIONS_CONTINUE	35018	0.71	36568	0.75
TOOL_REFERENCES_TOGGLE	29340	0.6	9684	0.2
TOOL_TEXT_HIGHLIGHT_TOGGLE	23708	0.48	19918	0.41
ITEM_BOOKMARK_ON	21861	0.45	13915	0.28
TOOL_SKETCH_CLOSE	20647	0.42	9600	0.2
TOOL_SKETCH_OPEN	20646	0.42	11310	0.23
NAVIGATION_PROFILE_LOGIN	20573	0.42	3314	0.07
TOOL_REFERENCES_OPEN	19597	0.4	25837	0.53
ITEM_BOOKMARK_OFF	19326	0.39	14986	0.31
ALERT_PROFILE_EXIT	14968	0.31	17224	0.35
ITEM_CONNECTION_match	14557	0.3	17930	0.37
TOOL_REFERENCES_CLOSE	13703	0.28	15243	0.31
ITEM_HOTSPOT_select	12098	0.25	14756	0.3
TOOL_TTS_PLAY	11069	0.23	10214	0.21
TOOL_TEXT_HIGHLIGHT_SELECTED	10827	0.22	14806	0.3
ITEM_STIMULUS_TOGGLE	7456	0.15	5025	0.1
TOOL_TEXT_HIGHLIGHT_CANCEL_ALL	6851	0.14	5817	0.12

ITEM_HOTSPOT_unselect	6143	0.13	5472	0.11
ITEM_CLEAR_START	4400	0.09	5	0.0
NAVIGATION_ACCOMMODATION_OP-TIONS_CONTINUE	4279	0.09	5288	0.11
TOOL_TEXT_HIGHLIGHT_CANCEL	4237	0.09	4435	0.09
TOOL_GUIDELINE_CLOSE	4165	0.08	4508	0.09
TOOL_GUIDELINE_OPEN	4152	0.08	225	0.0
ITEM_CLEAR_COMMIT	3371	0.07	4082	0.08
TOOL_TTS_PAUSE	2556	0.05	2045	0.04
ITEM_CONNECTION_unmatch	2133	0.04	91	0.0
ALERT_INACTIVITY_EXIT	1961	0.04	102	0.0
TOOL_TTS_STOP	1619	0.03	353	0.01
NAVIGATION_LOCK_RESUME	1606	0.03	1653	0.03
TOOL_TTS_VOLUME	1460	0.03	1166	0.02
NAVIGATION_TOOLBAR_START	1454	0.03	2	0.0
TOOL_TTS_ON	1289	0.03	376	0.01
NAVIGATION_ACCESS_CODE_CANCEL	1272	0.03	47	0.0
TOOL_TTS_RESUME	1265	0.03	1711	0.03
ITEM_CLEAR_CANCEL	1035	0.02	312	0.01
NAVIGATION_PAUSE_CANCEL	856	0.02	768	0.02
NAVIGATION_PAUSE_LOCK	844	0.02	1010	0.02
NAVIGATION_TURN_IN_CANCEL	816	0.02	45	0.0
TOOL_TTS_RATE	812	0.02	379	0.01
TOOL_TTS_SELECT	596	0.01	734	0.01
ALERT_PROCTOR_PASSWORD_SUBMIT	542	0.01	605	0.01
TOOL_TTS_OFF	480	0.01	425	0.01
NAVIGATION_REVIEW_PANEL_START	338	0.01	0	0.0
TOOL_TTS_SKIP	271	0.01	116	0.0
TOOL_COLOR_SCHEME_ON	129	0.0	109	0.0
NAVIGATION_PAUSE_COMMIT	93	0.0	8	0.0
ALERT_DIRECTIONS_EXIT	68	0.0	0	0.0
NAVIGATION_LOCK_SIGN_OUT	46	0.0	4	0.0
NAVIGATION_DIRECTIONS_ACCOMMODA-TION_OPEN	46	0.0	0	0.0

NAVIGATION_DIRECTIONS_ACCOMMODA-TION_CLOSE	46	0.0	47	0.0
ALERT_START_TEST_ERROR_RETRY	39	0.0	0	0.0
TOOL_ZOOM_SET	28	0.0	9	0.0
TOOL_ZOOM_INCREASE	19	0.0	5	0.0
TOOL_COLOR_SCHEME_TOGGLE	18	0.0	17	0.0
TOOL_REVERSE_CONTRAST_OFF	18	0.0	3	0.0
TOOL_ZOOM_DECREASE	18	0.0	9	0.0
TOOL_CUSTOM_MASKING_OPEN	10	0.0	4	0.0
ALERT_OFFLINE_WARNING_CLOSE	8	0.0	10	0.0
TOOL_CUSTOM_MASKING_CLOSE	6	0.0	4	0.0
ALERT_LOCK_TIMEOUT_EXIT	5	0.0	1	0.0
TOOL_REVERSE_CONTRAST_ON	5	0.0	0	0.0
ALERT_SIMULTANEOUS_USER_CLOSE	4	0.0	2	0.0
TOOL_CUSTOM_MASKING_TOGGLE	3	0.0	4	0.0
TOOL_ZOOM_RESET	2	0.0	0	0.0
ALERT_START_TEST_ERROR_CLOSE	1	0.0	0	0.0
TOOL_COLOR_SCHEME_OFF	1	0.0	0	0.0

Table 17:

Most Common Next Action Dictionary (Top-1 Choice among all possibilities)

Action	Most Common Next Action
-1 (begin)	NAVIGATION_PROFILE_CHOOSE
ALERT_DIRECTIONS_EXIT	NAVIGATION_PROFILE_LOGIN
ALERT_INACTIVITY_EXIT	NAVIGATION_PROFILE_LOGIN
ALERT_PROCTOR_PASSWORD_SUBMIT	NAVIGATION_DIRECTIONS_CONTINUE
ALERT_PROFILE_EXIT	NAVIGATION_PROFILE_LOGIN
ALERT_START_TEST_ERROR_RETRY	NAVIGATION_DIRECTIONS_CONTINUE
ITEM_BOOKMARK_OFF	NAVIGATION_REVIEW_PANEL_OPEN
ITEM_BOOKMARK_ON	NAVIGATION_REVIEW_PANEL_OPEN
ITEM_CLEAR_CANCEL	ITEM_MULTIPLE_CHOICE_ANSWER
ITEM_CLEAR_COMMIT	ITEM_MULTIPLE_CHOICE_ANSWER

ITEM_CLEAR_START	ITEM_CLEAR_COMMIT
ITEM_DRAG_BOX_DRAG_END	ITEM_DRAG_BOX_DRAG_START
ITEM_DRAG_BOX_DRAG_START	ITEM_DRAG_BOX_DRAG_END
ITEM_MULTIPLE_CHOICE_ANSWER	NAVIGATION_ITEM_NEXT
ITEM_SELECT_DROP_DOWN_select	ITEM_SELECT_DROP_DOWN_select
ITEM_STIMULUS_TOGGLE	ITEM_STIMULUS_TOGGLE
ITEM_TILE_BOX_DRAG_END	ITEM_TILE_BOX_DRAG_START
ITEM_TILE_BOX_DRAG_START	ITEM_TILE_BOX_DRAG_END
NAVIGATION_ACCESS_CODE_CANCEL	NAVIGATION_PROFILE_CHOOSE
NAVIGATION_ACCESS_CODE_SUBMIT	NAVIGATION_DIRECTIONS_CONTINUE
NAVIGATION_ACCOMMODATION_OPTIONS_CON-TINUE	NAVIGATION_DIRECTIONS_CONTINUE
NAVIGATION_DIRECTIONS_CONTINUE	ITEM_MULTIPLE_CHOICE_ANSWER
NAVIGATION_ITEM_BACK	NAVIGATION_ITEM_BACK
NAVIGATION_ITEM_JUMP	ITEM_MULTIPLE_CHOICE_ANSWER
NAVIGATION_ITEM_NEXT	ITEM_MULTIPLE_CHOICE_ANSWER
NAVIGATION_LOCK_RESUME	NAVIGATION_LOCK_RESUME
NAVIGATION_LOCK_SIGN_OUT	NAVIGATION_PROFILE_CHOOSE
NAVIGATION_PAUSE_CANCEL	ITEM_MULTIPLE_CHOICE_ANSWER
NAVIGATION_PAUSE_COMMIT	NAVIGATION_PROFILE_LOGIN
NAVIGATION_PAUSE_LOCK	NAVIGATION_LOCK_RESUME
NAVIGATION_PROFILE_CHOOSE	NAVIGATION_ACCESS_CODE_SUBMIT
NAVIGATION_PROFILE_LOGIN	NAVIGATION_PROFILE_CHOOSE
NAVIGATION_REVIEW_PANEL_CLOSE	NAVIGATION_ITEM_JUMP
NAVIGATION_REVIEW_PANEL_OPEN	NAVIGATION_REVIEW_PANEL_CLOSE
NAVIGATION_TURN_IN_CANCEL	NAVIGATION_TURN_IN_START
NAVIGATION_TURN_IN_COMMIT	ALERT_PROFILE_EXIT
NAVIGATION_TURN_IN_START	NAVIGATION_REVIEW_PANEL_CLOSE
TOOL_ANSWER_MASKING_TOGGLE	TOOL_ANSWER_MASKING_TOGGLE
TOOL_CALCULATOR_CLOSE	ITEM_MULTIPLE_CHOICE_ANSWER
TOOL_CALCULATOR_OPEN	ITEM_MULTIPLE_CHOICE_ANSWER
TOOL_CALCULATOR_TOGGLE	TOOL_CALCULATOR_OPEN
TOOL_GUIDELINE_CLOSE	TOOL_SKETCH_OPEN

TOOL_GUIDELINE_OPEN	TOOL_GUIDELINE_CLOSE
TOOL_REFERENCES_CLOSE	ITEM_MULTIPLE_CHOICE_ANSWER
TOOL_REFERENCES_OPEN	TOOL_REFERENCES_CLOSE
TOOL_REFERENCES_TOGGLE	TOOL_REFERENCES_OPEN
TOOL_SKETCH_CLOSE	TOOL_TEXT_HIGHLIGHT_TOGGLE
TOOL_SKETCH_OPEN	TOOL_SKETCH_SELECT
TOOL_SKETCH_SELECT	TOOL_SKETCH_SELECT
TOOL_TEXT_HIGHLIGHT_CANCEL	TOOL_TEXT_HIGHLIGHT_CANCEL
TOOL_TEXT_HIGHLIGHT_CANCEL_ALL	TOOL_TEXT_HIGHLIGHT_TOGGLE
TOOL_TEXT_HIGHLIGHT_SELECTED	TOOL_TEXT_HIGHLIGHT_SELECTED
TOOL_TEXT_HIGHLIGHT_TOGGLE	TOOL_TEXT_HIGHLIGHT_SELECTED
TOOL_TTS_RATE	TOOL_TTS_RATE
TOOL_ZOOM_DECREASE	ITEM_MULTIPLE_CHOICE_ANSWER
TOOL_ZOOM_INCREASE	ITEM_MULTIPLE_CHOICE_ANSWER
TOOL_ZOOM_RESET	TOOL_ZOOM_INCREASE
TOOL_ZOOM_SET	NAVIGATION_ACCOMMODATION_OP-TIONS_CONTINUE
NAVIGATION_REVIEW_PANEL_START	NAVIGATION_PAUSE_CANCEL
NAVIGATION_TOOLBAR_START	NAVIGATION_PAUSE_LOCK