Modelling the way: Using action sequence archetypes to differentiate learning pathways from learning outcomes

Kelvin H R Ng
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
e140025@e.ntu.edu.sg

Kevin Hartman
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
khartman@ntu.edu.sg

Kai Liu
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
kliu006@e.ntu.edu.sg

Andy W H Khong
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
andykhong@ntu.edu.sg

ABSTRACT

During the semester break, 36 second-grade students accessed a set of resources and completed a series of online math activities focused on the application of the model method for arithmetic in two contexts 1) addition/subtraction and 2) multiplication/ division. The learning environment first modeled and then supported the use of a scripted series of steps for solving mathematical word problems. As students completed the activities, the learning environment captured their event-related data. We then used a combination of Affinity Propagation, an automated form of clustering, and sequential pattern mining to convert the activity logs into interpretable activity sequences. Analysis of the activity sequences identified distinct patterns of behavior that strongly predicted which students would transit from the familiar addition/subtraction word problem activity to the unfamiliar multiplication/division word problem activity. Students who showed the greatest and least compliance with the script were the least likely to attempt the multiplication/division activity. Students who showed more of a schematic problem solving process were more likely to continue to the multiplication/division activity.

Keywords

Sequential pattern mining, affinity propagation, cognitive models

1. INTRODUCTION

1.1 Mathematics Learning via the Model Method

In Singapore, early-elementary students are taught to solve arithmetic word problem via the model method [1]. This systematic approach is based on Polya's problem solving techniques [2]. The method can be broken into five steps known as the RIGHT sequence. When applying the RIGHT sequence, students 1) read the word problem, 2) identify the nouns, numeric values, and unknown variable to be solved, 3) graph these values in a box diagram, 4) indirectly perform the appropriate calculation by reasoning through the diagram, and 5) review their work.

The RIGHT sequence, as a learning mnemonic, provides students with a script for executing the model method. Scripts are collections of discrete actions that, when followed, achieve a goal or specific outcome [3]. Ordering food at a restaurant serves as the classic example of following a cognitive script [3]. In most dining establishments, the same set of steps, with some allowance for minor deviations, will lead the patron to receive a meal. Similarly, following the RIGHT sequence will lead students to the correct answer to a word problem. Scripts have been found to reduce cognitive load for novice learners by lessening the mental resources needed for planning and completing the plan. Scripts also lead to greater expressions of automaticity by experts [4]. However, cognitive psychologists also view scripts as the most nascent form of schemas [5]. The application of scripts is contextually bound and rather inflexible. Schank and Abelson [6] refers to scripts as event schemas which are task specific and order dependent. The previous restaurant script may work for purchasing food at most dining establishments, but it could not be used successfully to purchase food at a supermarket. To negotiate the supermarket, one would need to apply either a different script or rely on a more generalizable schema.

Generalizable schemas consolidate the steps of an event schema under a larger label [7]. Rather than simply ordering a meal at a restaurant, a generalizable schema for acquiring food would include all of the known methods of gaining nourishment. What generalizable schemas sacrifice in terms of automaticity, they make up with flexibility [5].

Returning to the original example of the model method, the intent behind introducing students to using box diagrams to solve algebraic word problems is to give them a generalizable schema for solving real-world problems [1]. In practice, students often instantiate the schema in the form of a word problem solving script [8]. When looking at problem solving accuracy, teachers cannot diagnose whether a student has internalized the model method as a generalizable schema or as a problem solving script because both strategies work in the short term. However, only the generalizable schema prepares students to flexibly transfer the model approach to new situations. In this study, we sought to generate an algorithm to classify students as exhibiting script-like or generalizable schema-like behaviors in the context of a series of online math enrichment activities. We then tested whether script-like behaviors, generalizable schema-like behaviors, or problem solving accuracies were more predictive of students seizing future learning opportunities.

1.2 Machine Learning and Temporal Sequencing

In the context of this paper, we define an action as a single line item in a log file and action sequences as the collection of actions that can be described with a more general semantic label. For instance, entering a number into a text box constitutes an action. All of the various combinations of actions that lead to the calculation of that

number being entered into the text box constitute a single action sequence.

When attempting to identify meaningful action sequences while preserving the temporal relationships between those actions, educational data miners use techniques like process mining and sequential pattern mining. With process mining, the learning pathways students take within a learning environment are identified and visualized [9, 10]. Deviations in these pathways from the intended pathways can then be analyzed for meaning [9, 10]. Alternatively, sequential pattern mining identifies frequently occurring subsequences within a temporal dataset for further analysis. Recently, Ye et at. used a hierarchical variant of SPAM to analyze data collected from Betty's Brain OELE [11]. The analysis illustrates the importance of using temporal relationships between user activities to make predictions about future learning behaviors [11]. Veeramachaneni, Adl, and O'Reilly [12, 13] also highlight the significance of incorporating a range of temporal dependencies into features when predicting student traits. Applying a crowd sourcing technique, they obtained lists of complex features that, when divided, seem obvious to experienced teachers and data scientists. However, neither group could have generated the entire list of the features on its own [12].

When extracting frequent patterns from unstructured data, sometimes the patterns are composed of short sets of actions which actually belong to longer action sequences. These algorithms have a tendency to obscure the temporal relationships between the extracted features. Additionally, sequencing combinations of actions and filtering out rare patterns rather than using the complete action sequences can result in the loss of rare action combinations that achieve a common action sequence [14]. The potential for losing rare actions belonging to common action sequences is magnified in learning environments populated by novice learners. Novice learners who are introduced to a learning environment have the dual task of learning to navigate the environment as well as gaining competency with the concepts central to the learning activities. In such situations, data mining techniques that analyze learner actions more schematically, rather than in scripted terms, may actually yield more parsimonious models.

With the goal of aligning our data mining techniques with learners' mental schemas, we propose conceptually reframing individual actions as words and action sequences as sentences. With this recasting, we can apply a combination of string distance measures that take into account the vocabulary and word order within the sentences to make pair-wise comparisons. We used an Affinity Propagation (AP) [15] algorithm to recover distinct action sequences that translated to learning behaviors and the sequence exemplars are referred to as action sequences archetypes (ASAs). Sequential pattern mining is applied to cluster members to summarize the temporal deviations within each cluster. The described method preprocesses the data for analysis and interventions to steer learners towards desired educational outcomes.

AP is useful for our particular context because it simultaneously considers all data points in relation to a shared preference to determine a suitable number of output clusters. This structure independence lends AP to situations where there is no a priori expectation about the output cluster size or number [15]. In our case, the number of sequences within the dataset varies greatly between sessions. Beyond accommodating this variability, the algorithm's input, a similarity matrix defined by the pairwise similarities between two sequences, is not limited to symmetrical pairwise similarities. This freedom creates opportunities to differentiate the discrete ordered lists using different distance measurements. We augmented the AP algorithm with a tree-based sequential pattern mining algorithm for its ability to handle multiple minimum supports and rare item filtering [14]. The algorithm is used to extract maximal sequences, which are longest sequences that satisfy the minimum frequency threshold, for each cluster.

2. Data Collection

36 second grade students completed the first phase of activities in the online learning environment during the school holidays. The activities were part of an ``out of school" enrichment opportunity. At the onset of data collection, all of the invited participants had previously received formal instruction from their teachers on using the model method to solve addition and subtraction word problems. The students had not yet received instruction within the school curriculum on using the model method with multiplication and division word problems.

The online learning environment offers two phases of content. During Phase 1, students' complete addition and subtraction activities. In Phase 2, students encounter multiplication and division activities. Each content phase is divided into four sets of activities: 1) video tutorials, 2) structured activities, 3) unstructured activities, and 4) multiple choice questions (MCQ). The video tutorials explain the RIGHT sequence and the use of the model method in a pen-and-paper context. After each video, students receive a set of practice exercises related to the content of the video tutorial. Additional video segments at the start of each practice question introduce the recommended sequence of steps to solve the word problems using the model method and the representational supports found within the learning environment. The representational supports include using the highlighted noun blocks and the RIGHT checklist while answering the word problems.

The structured activity focuses on the "G" in the RIGHT sequence. Each question in the activity is presented with a practice word problem. The problem is displayed with four multiple choice options showing different bar diagrams and a checklist in the right corner of the workspace. The checklist shows the first three steps of the RIGHT sequence. Students are advised to tick off the respective check boxes as they complete each step in the RIGHT sequence. In the structured activity, the checklist is limited to the first three steps of the RIGHT sequence as students are not expected to take their model to completion.

After students identify the model they think matches the content of the word problem, they are given feedback about their choice before moving on to the next question. They are presented with options to review, ask for hints or proceed to the next question. Choosing to review the question returns students to the last snapshot of the question before the answer submission. Requesting a hint provides students with a partially completed model as a guide. Hints are given progressively until the complete model is revealed. Two hints can be requested for each question. If a student chooses to proceed to the next question without reviewing errors after submitting an error, the learning environment logs the action as ignoring an error.

In the unstructured activity, students solve the problems using the RIGHT sequence. A snapshot of the learning environment for this activity prior to any attempt is shown in Figure 1. Model templates for all four arithmetic operations are made available for students to complete with the correct numerical values. Nouns mentioned in the problem are also presented as colored blocks for labeling the relevant model. Students can drag and drop the blocks to their

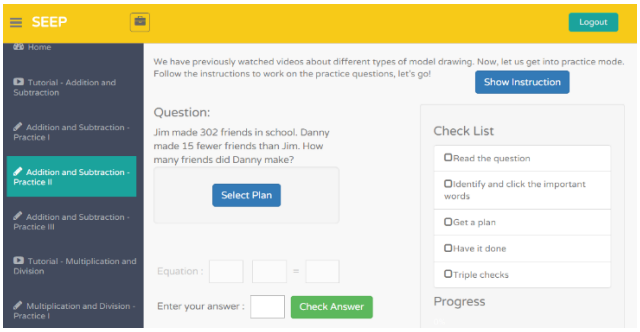

Figure 1: Workspace for unstructured activity.

selected model. Students may also enter mathematical expressions in the provided text boxes. Alternatively, students may forego performing any or all of these actions. However, they must submit a final answer before receiving feedback about their answer and proceeding to the next question.

For the MCQs, students are presented with a page containing ten multiple-choice questions. Each question requires inputting a numerical answer into a textbox. Students again have the option of using the RIGHT checklist that floats in the right margin of the screen. The checklist resets whenever a student interacts with a different question. Students must complete all of the activities before proceeding to Phase 2.

3. Data Preprocessing

Only clickstream and navigation information occurring within the online module was recorded to the log file as students worked through the activities. Beyond navigation and interface information like mouse clicks and text entries, off-task behavior like leaving the learning environment by activating another browser tab and returning to the online module was also collected. A total of 23233 log entries were collected. Table 1 lists the recorded actions.

The log entries were preprocessed to indicate the use of the different learning resources within the learning environment. For example, highlighting a keyword within a question is recorded as one log entry per keyword. However, only the first instances of highlighting and canceling of highlights are retained for each question attempt to signal that the highlighting resource was used. In addition, while learners navigate through the model template selection, we only analyze the final template selection instead of considering all of the navigation activity within the selection area. Filtering out these events greatly reduces the amount of variability within the action sequences and makes them more schematic. To identify revision of answers, first selections for the MCQs are labelled as *mcq_select.* Additional selections are labelled as *mcq_alter*. Following the described procedure reduced the size of the dataset to 9918 entries, or 275 entries per student. The maximum number of analyzed actions for a student was 868. The final list of actions for each type of activity is shown in Table 1.

In the reduced dataset, each action sequence is identified and labelled. For videos, an action sequence constitutes the actions taken from the start of a video to terminating the video either by completing the video or navigating away from the current page. For the exercises, the action sequences span from the initiation of a question until the user proceeds to the next question.

Table 1: List of all log actions

Action	Video	Structured	Unstructured	MCQ
leave_page	✓	✓	✓	✓
return_to_page	✓	✓	✓	✓
phase_start	✓	✓	✓	✓
phase_stop	✓	✓	✓	✓
video_start	✓			
video_stop	✓			
video_pause	✓			
video_scrub_foward[1]	✓			
video_scrub_back[1]	✓			
video_end[1]	✓			
video_end_full[1]	✓			
video_replay	✓			
video_select_same	✓			
video_select_diff	✓			
attempt_qn		✓	✓	✓
highlight		✓	✓	✓
undo_highlight		✓	✓	✓
check_checklist		✓	✓	✓
mcq_select[2]		✓		✓
mcq_alter[2]		✓		✓
confirm_model[2]			✓	
mouse_drag[2]			✓	
label_model[2]			✓	
label_eq			✓	
submit[2]		✓	✓	✓
review_error		✓	✓	✓
ignore_error		✓	✓	✓
show_hint		✓	✓	✓

[1] Actions are inferred from clickstream data due to limitation of YouTube's application programming interface (API).

[2] Actions are recorded but filtered out for the purpose of this analysis.

4. Techniques

4.1 Distance Measures

To differentiate action sequences as one would differentiate sentences, it is necessary to consider the vocabulary (actions) of each action sequence and the order of those words. Our proposed distance measure includes four components, a modified version of the common word order measure [16], Jaccard distance, length difference, and vocabulary rarity. The features capture different aspects of action sequences for differentiation. The distance measure between two action sequences S_1 and S_2 is given by the weighted sum of all four features. In this paper, a constant weight is assigned across the four features.

$$\begin{aligned} dist(S_1, S_2) = {} & w_1 * JaccardDist(S_1, S_2) \\ & + w_2 * CWO(S_1, S_2) \\ & + w_3 * \max\left(idf_{t_j \notin S_1 \cap S_2}(t_j, D)\right) \\ & + w_4 * abs\big(length(S_1) - length(S_2)\big) \end{aligned} \quad (1)$$

where

$$w_1 = w_2 = w_3 = w_4 = 1 \quad (2)$$

Jaccard distance defined by

$$JaccardDist(S_1, S_2) = 1 - JaccardSim(S_1, S_2) \quad (3)$$

where

$$\text{JaccardSim}(\text{S}_1, \text{S}_2) = \frac{|\text{S}_1 \cap \text{S}_2|}{|\text{S}_1 \cup \text{S}_2|} \quad (4)$$

captures the degree of dissimilarity between two sequences through the number of unique terms that are not common to both. The Jaccard distances are derived from Jaccard similarity which determines the ratio of unique common actions between two action sequences. Jaccard similarity and distances are bounded between zero and one.

In our context, the common word order measure reflects the similarity of the order in which actions appear between two action sequences. The measure equals zero when the common actions of two sequences occur in the same order and reaches a maximum of one when the common actions appear in reverse order. Given two sequences $A = \{a_1, a_2, \dots, a_n\}$ and $B = \{b_1, b_2, \dots, b_m\}$ composed of l common action, where $l \leq n \leq m$. Retaining only the common actions, sentence $A = \{a_1, a_2, .., a_l\}$ is transformed into a numerical representation $X = \{1,2, \dots, l\}$ by substituting the actions with its indices. The same actions in B are replaced with the same numerical indices to form B. The common word order measure can then be computed by

$$\text{CWO}(\text{S}_1, \text{S}_2) = \begin{cases} 1 - \dfrac{\left(2\sum_{i=1}^{l}|x_i - y_u|\right)}{l^2}, & if\ l\ is\ even \\ 1 - \dfrac{\left(2\sum_{i=1}^{l}|x_i - y_u|\right)}{l^2 - 1}, & if\ l\ is\ odd \\ 1, & if\ l\ is\ odd\ and\ l = 1 \end{cases} \quad (5)$$

The common word order measure is designed for sentences where a bag-of-words representation has a large number of words, most of which have low frequencies. Due to the constraints of the learning environment, our data set contained many actions with high frequencies. Retaining the common terms within action sequences may result in substrings of inequivalent lengths. Therefore, there may exist more than one combination of mapping between these sentences. To remedy this possibility, we adapted the concept of a common word order measure to obtain a distance estimate for the action sequences by first filtering the reduced sequences to remove actions occurring at a specific position that do not contribute to the distance metric. We then match the remaining actions based on their position within the reduced sequence.

The vocabulary rarity is defined as the maximum of the inverse document frequency (idf) [17] of terms that are not common to both sentences. This measure allows us to distinguish sequences that have actions that are less likely to occur from sequences involving trivial navigational patterns. The inverse document frequency of each term t_i in a set of documents D is computed by the logarithmic inverse of the ratio of document counts containing t_i to the total number of documents in the document set D.

$$idf(t_i, D) = \log \frac{|D|}{\{|d \in D : t_i \in d|\}} \quad (6)$$

4.2 Affinity Propagation

The AP algorithm [15] is a message passing clustering algorithm used in image recognition, text comparison and gene clustering. Unlike centroid-based clustering like k-means clustering, AP does not require users to pre-specify the number of clusters and it is less sensitive to parameter initialization [15]. The algorithm takes a pair-wise similarity matrix and a set of shared preferences as inputs to determine the suitability of data points as cluster centroids. Without prior knowledge of the centroids, shared preferences may be set uniformly across all items. When shared preferences are assigned to the minimum value of the pairwise similarity, the number of resulting clusters will also be at its lowest. The inverse is also true. The number of clusters generated by the different shared preference values for the structured activity are shown in Figure 2.

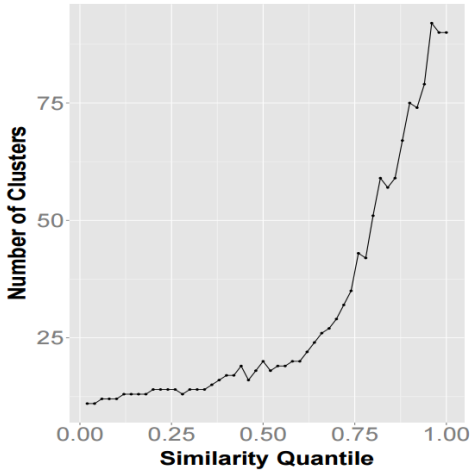

Figure 2: Number of generated clusters based on shared preferences for structured activity.

For our purposes, clusters are determined by passing messages between data points (action sequences) to simultaneously determine their suitability as cluster centroids. The provided similarity matrix may contain unknown pair-wise similarities. However, messages are passed only between points with known similarities. There are two types of messages passed between data points -- responsibility and availability. Responsibility $r(i,k)$, sent from data point i to data point k, dictates the amount of evidence that k is suitable to serve as the exemplar for i, while availability $a(i,k)$, sent from k to i, determines the appropriateness for point i to choose point k as its exemplar. Availabilities are initialized as zeroes and the messages are updated iteratively using

$$r(i,k) = s(i,k) - \max_{k' \neq k}\{a(i,k') + s(i,k')\}, \quad (7)$$

$$a(i,k) = \min\left\{0, r(k,k) + \sum_{i' \notin \{i,k\}} max\{0, r(i',k)\}\right\} \quad (8)$$

$$a(k,k) = \sum_{\{i' \neq k\}} \max\{0, r(i',k)\} \quad (9)$$

At the end of each iteration, exemplars are determined from

$$exemplar(i,k) = \underset{k}{\text{argmax}}\{a(i,k) + r(i,k)\} \quad (10)$$

Pairs (i,k) identified from equation (10) state that either data point i will serve as an exemplar for data point k or vice versa. The algorithm terminates only when either a predefined number of iterations is completed or the changes in the messages falls below a certain threshold.

Essentially, the AP algorithm seeks to identify action sequence archetypes (ASA) around which to cluster the remaining action

sequences. After identifying the ASAs, the similar cluster sequences inherit the index of their closest archetype.

4.3 Sequential Pattern Mining

The position coded pre-order linked web access pattern tree mining (PLWAP) algorithm with multiple minimum supports (MMS) [14] is a tree-based sequential pattern mining algorithm. A PLMS-tree is constructed from the logs by adding actions for each learning opportunity sequentially. Each node holds four variables, the label, the frequency count, a binary position code, and a minimum multiple item support (*minMIS*).

The binary code is similar to Huffman coding as it uniquely identifies nodes and subtrees. The root node of the tree is labelled as 0. The leftmost child of any node has a position code of 1 appended to the back of the position code of the node. The position codes of other children are derived from the position codes of their nearest sibling to the left by appending a 0 to the position code.

The support determines the lower bound for frequencies that sequences must satisfy to qualify as a frequent pattern. For multiple minimum support, a minimum support is computed for each unique item in the dataset. In the case of our action sequences, the items in sequential pattern mining correspond to actions in the action sequences. The global minimum support is dictated by the smallest of the minimum supports. Each node maintains its *minMIS* which defines the support required by itself and the suffix tree to qualify as frequent.

As the nodes are added to the tree, a header table is maintained. The header table contains the unique node labels with a list of corresponding binary code of nodes for the same label within the tree. The table is then sorted by order of decreasing frequencies. An example of the PLMS-tree and its corresponding header table is shown in Table 2 and Figure 3.

Table 2: Header table example for Figure 3

Label	Support	Position Code
video_start	10	{01}
video_end_full	6	{011}, {0110001}
video_end	4	{0110}, {011001}, {011000101}
video_pause	2	{011000}
video_scrub_back	1	{01100010}
video_scrub_forward	1	{01100}

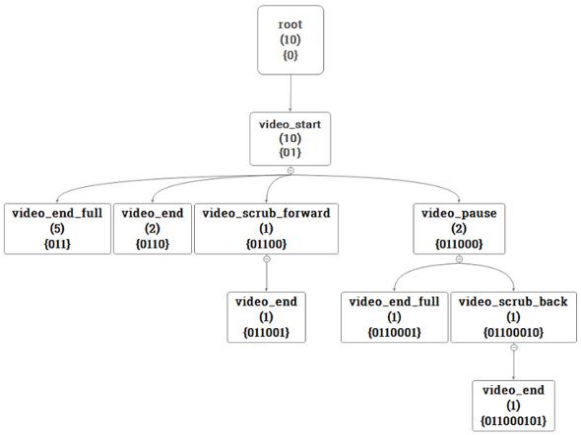

Figure 3: Example of a PLMS-tree for the video activity.

Once the tree is populated, it can be traversed to mine the sequential patterns in the dataset. The mining algorithm proceeds as follows:

1. For each of the entries in the header table, the nodes are identified from the tree using the position codes and the total occurrences is consolidated from the counts of individual nodes. A *k*-sequence is an ordered list of *k* items.
   a. If this sequence satisfies its *minMIS*, it qualifies as a 1-sequence.
   b. If the frequency of this node satisfies the global minimum but not its *minMIS*, the label qualifies as a 1-sequence candidate. Candidates are kept as candidates for mining because a subsequent item of lower *minMIS* may qualify these sequences as frequent sequences.
2. The algorithm proceeds to identify the next item in the sequence by scanning the header table.
3. Position codes in the header table containing the position code of the last found node as its prefixes are identified as descendants for that node.
4. The frequencies of the newly identified nodes are aggregated and a new *minMIS* is updated to be the lower of the *minMIS* from previous nodes and the identified node.
5. The algorithm proceeds to search for possible extensions and validates the frequency of these sequences against the *minMIS*.
6. The algorithm terminates when no more descendants are identified from the header table or if the frequencies of the newly identified nodes are less than the value of the global minimum support.

Table 3: Action Sequence Profiles

Activity	No. of Attempts	No. of Profiles
Video	89	10
Structured	286	11
Unstructured	303	11
MCQ	33	12

5. Clustering

Sequences of attempts for each of the four activities are clustered with the AP algorithm. The shared preferences of AP are set to the maximum of the similarity matrices. We use the R package AP for this analysis. PLWAP is then used to retrieve a descriptive summary for each cluster. We restrict the algorithm to only identify contiguous sequences.

We manually merge the clusters into ASAs based on their compositions. The compositions are determined by indicators signaling the use of certain actions between defined checkpoints, similar to the process mentioned in [10]. During the merging process for each archetype, we consider the actions spanning from the onset of a question to the first submission of the question attempt. Descriptions of the ASAs identified in the video, structured and unstructured activities are presented in Table 4, Table 5, and Table 6 respectively.

Table 4: Action sequence archetypes for the video activity

ASA	Description
V1	Offtask
V2	Pre-mature termination
V3	Complete video without other actions
V4	Complete video with pauses
V5	Complete video with off-task
V6	Complete video with pauses and off-task
V7	Complete video with pauses and scrub back
V8	Incomplete video
V9	Incomplete video with pauses
V10	Incomplete video with scrub forward

Table 5: Action sequence archetypes for the structured activity

ASA	Description
S1	Pre-mature termination
S2	Direct answer with off-task
S3	Direct answer
S4	Direct answer with alter of choice
S5	Answer with highlights
S6	Answer with highlights and alter of choice
S7	Answer with checklist
S8	Answer with checklist and highlights
S9	Answer with highlights and alter of choice and checklist
S10	Submission with checklist and highlights but no answer
S11	Submission without answer

Because the MCQ activity presents multiple word problems on the same page, students may freely switch between the problems without signaling their intent. This freedom of choice yields ASAs with indefinite boundaries. The nebulousness of the ASAs associated with each question provides little inferential utility and will not be addressed in the following section beyond attempting to use each student's MCQ accuracy to predict the probability of persisting to Phase 2.

6. Results

6.1 Score-based Prediction of Persistence

We calculated the percentage of questions students correctly answered for the structured, unstructured, and MCQ activities. As shown in Table 7, only a student's MCQ performance is associated with persisting into Phase 2. Knowing students' performances for the structured and unstructured activities leads to a prediction accuracy level similar to that of assuming no student persists from Phase 1 to Phase 2.

As a caveat, the deterministic appearance of the association between MCQ performance and persisting to Phase 2 is misleading. The high correlation is due to the MCQ activity being a prerequisite for Phase 2. The mere presence of an MCQ submission, rather than the score itself, is predictive of persisting to Phase 2. Students who do not make an MCQ submission effectively earn a score of zero for the activity and do not have the possibility to continue to Phase 2. Additionally, all students who do persist to Phase 2 must have scored above a zero on the MCQ activity.

Table 6: Action sequence archetypes for the unstructured activity

ASA	Description
U1	Attempts with no submission
U2	Attempts with off-task and no submission
U3	Submission without attempts
U4	Submission without answer
U5	Submission with answer and highlights
U6	Submission without highlights and drags
U7	Submission without highlights
U8	Submission without drags
U9	Submission without drags with one change of model template
U10	Submission without drags with multiple change of model template
U11	Submission without drags with off-task
U12	Suggested steps
U13	Suggested steps with one change of model template
U14	Suggested steps with multiple change of model template
U15	Suggested steps with off-task

Table 7: Mean activity scores for students who stop during Phase 1 and persist to Phase 2

	Accuracy	
Activity	*Stop-out*	*Persist*
Structured	53.08%	55.23%
Unstructured	54.56%	69.81%
MCQ	0%	91.84%

6.2 Sequence-based Prediction of Persistence

We converted the frequency of each ASA into a percentage of a student's total action sequences. We then used a classification and regression tree (CART) algorithm to predict which students continued on to Phase 2 based on their ASA values. The decision trees associated with progressing based on ASAs from the video, structured and unstructured activities are presented in Table 4, Table 5 and Table 6 respectively.

While persistence cannot be reliably predicted based on video ASAs, it can be accurately predicted by the structured and unstructured ASAs. The predictability of these features is

determined using a logistic regression classifier for each activity. The results are presented in Table 8.

Table 8: Logistic regression classification for stop-out prediction.

Variable Set	Variables	Accuracy	Kappa Statistics
Score-based	Structured	48.00%	-0.06
	Structured + Unstructured	66.67%	0.43
	MCQ	100.00%	1.00
Sequence-based	Videos	75.00%	0.48
	Structured	81.48%	0.61
	Unstructured	81.82%	0.63
	MCQ	82.35%	0.56

The stop-out prediction accuracy increases as more activity scores are included in the logistic regression models. The accuracy of these models is highly dependent on the inclusion of the MCQ activity scores. The MCQ activity is the last activity students must complete before proceeding to Phase 2.

The decision tree for the video activity, as shown in Figure 4, identify premature termination (ASA V2) as the best criterion for determining if students are likely to stop the activities. Prematurely terminating attempts at a frequency higher than 25% of the student's attempts is predictive of stopping the activity 83% of the time. In addition, a low compliance with incomplete video watching by fast-forwarding (ASA V10) and completing the video without additional actions (ASA V3) are indicative of students who stop out of the learning environment.

For the structured activity, a high compliance with the recommended process but without submitting an answer (ASA S10), answering questions with highlighting of keywords (ASA S5) and answering questions with the scripted steps, as shown in Figure 5, all indicate students who are likely to proceed to Phase 2. Students who tend not to provide an answer for these attempts are likely to not proceed to Phase 2.

While the unstructured activity gives more freedom to participants, the number of splitting criteria is minimal. Learners who do not proceed to Phase 2 are characterized by submitting more than 13% of their questions without any attempt to solve them (ASA U3). Also students who complied more than with the scripted steps more than 56% of the time also tended to stop out (ASA U12). We note that the lower compliance with the RIGHT sequence in unstructured activity in Phase 1 is associated with 86% probability of learners proceeding to Phase 2.

7. Conclusions

In this study, we presented a framework for converting clickstream data into action sequence archetypes. ASAs provide insight into how students approach learning activities by consolidating similar plans of action under a common label. For us, having a common label to refer to different patterns of actions facilitates discussion and interdisciplinary collaboration between the computer sciences and the learning sciences. This collaboration led us away from trying to analyze learning outcomes with click counts and time on task measures and toward ASAs. ASA frequencies identify how often a learner attempts to reach a goal via a particular method.

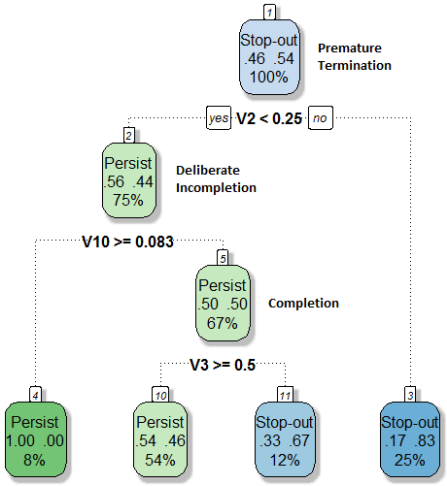

Figure 4: Decision tree for the video activity.

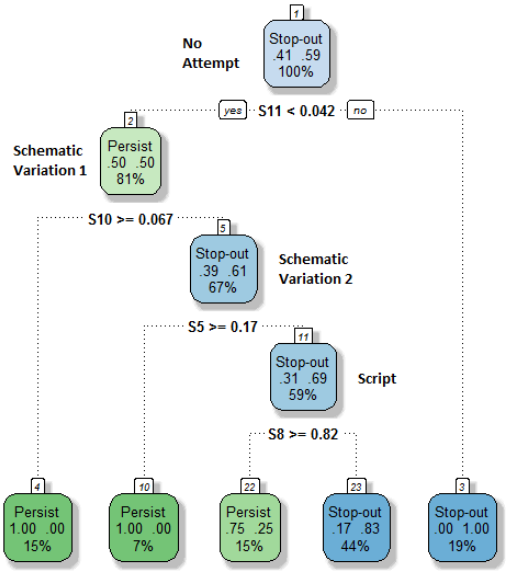

Figure 5: Decision tree for the structured activity.

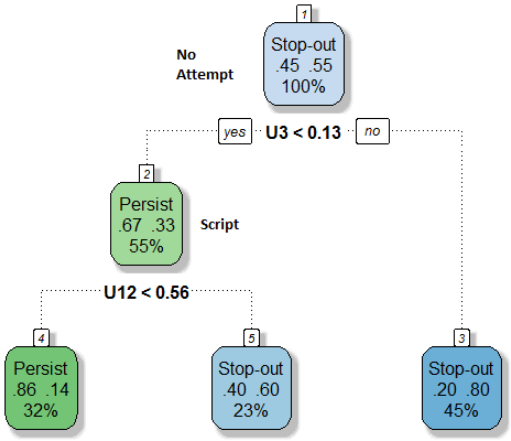

Figure 6: Decision tree for the unstructured activity.

Looking at our decision trees, ASAs can be used to quickly identify whether a learner is using on-task or off-task behaviors. However, they also can also be used to separate different approaches to achieving the same goal.

In our case, students whose action sequences aligned more strongly to the archetype representing the RIGHT sequence presented in the online videos were less likely to persist to the second phase of activities. In one sense, it is counterintuitive to suggest that students who follow a taught script more closely would be less likely to persist in an activity. However, if script use is a way of minimizing cognitive load, novices who consistently exhibit script-like behaviors could be indicating more routinization and less assimilation of new concepts. What these students may have learned from their classroom instruction and the online material is a series of steps for completing the structured and unstructured activities and not the generalizable schema that underlies those activities.

Using the ASAs to separate script users from generalizable schema users gives us a method of predicting a student's likelihood of persisting through the first phase of activities and attempting the second phase composed of unfamiliar math models. This method of prediction identifies students who are likely to stop out before the second phase much earlier than looking at how accurately the students solve the word problems. By the end of the second activity, our model could predict with high accuracy whether a student would continue on to Phase 2. Using a more traditional method of performance assessment and analyzing accuracy levels to predict future behavior required students to complete all of Phase 1 before the model could accurately predict whether the student would persist. In short, using ASAs to analyze how students approach the activities is more diagnostic of future performance than looking at past performance measures.

Finally, it is not lost on us that we developed an algorithm that converts action sequences (scripts) into action sequence archetypes (schemas) to measure students' use of scripts and generalizable schemas. For this project, the machine learning goals and the students learning goals happened to overlap. We plan to continue developing the parallels by integrating our ASA analysis into a student feedback engine that can shift students away from off-task behaviors and toward on-task behaviors. We also seek to lead on-task students toward more productive action sequences that foster the development of generalizable problem solving schemas rather than specific problem solving scripts.

8. ACKNOWLEDGMENTS

This project is supported by a start-up grant from the Centre for Research and Development in Learning (CRADLE@NTU).

9. REFERENCES

[1] Kho, T. H. 1987. Mathematical models for solving arithmetic problems. In Proceedings of the Fourth Southeast Asian Conference on Mathematics Education (ICMI-SEAMS), 345-351.

[2] Polya, G., Kaddouch, R., Renou, M., Comtois, M., and Dubois, M. J. 1957. How to solve it: a new aspect of mathematical method. Princeton University Press, Princeton, NJ.

[3] Abelson, R. P. 1981. Psychological status of the script concept. American Psychologist, 36, 7 (Jul. 1981), 715-729.

[4] Schoenfeld, A. H. 1999. Models of the teaching process. The Journal of Mathematical Behavior, 18, 3, (Mar. 1999), 243-261.

[5] Rumelhart, D. E., and Ortony, A. 1977. The representation of knowledge in memory. In Schooling and the Acquisition of Knowledge, R. C. Anderson, R. J. Spiro, and W. E. Montague, Eds. Erlbaum Associates, Hillsdale, NJ.

[6] Schank, R. C., and Abelson, R. P. 1977. Scripts, Plans, Goals, and Understanding: An inquiry into human knowledge structures. Erlbaum Associates, Hillsdale, NJ.

[7] Sweller, J., and Chandler, P. 1994. Why some material is difficult to learn. Cognition and Instruction, 12, 3, (Sep. 1994), 185-233.

[8] Cheong, Y. K. 2002. The model method in Singapore. The Mathematics Educator, 6, 2, 47-64.

[9] Emond, B., and Scott Buffett, S. 2015. Analyzing student inquiry data using process discovery and sequence classification. In Proceedings of the 8th International Conference on Data Mining, (Atlantic City, NJ, USA, November, 14-17, 2015), ICDM'15. 412-415.

[10] Southavilay, V., Markauskaite, L., and Jacobson, M. J. 2013. From "events" to "activities": Creating abstraction techniques for mining students' model-based inquiry processes. In Proceedings of the 6th International Conference on Educational Data Mining, (Memphis, TN, USA, July 6-9, 2013), EDM'13, 280-283.

[11] Ye, C., Kinnebrew, J. S., Segedy, J. R., and Biswas, G. 2015. Learning behavior characterization with multi-feature, hierarchical activity sequences. In Proceedings of the 8th International Conference on Educational Data Mining, (Madrid, Spain, June, 26-29, 2015), EDM'15, 380-383.

[12] Veeramachaneni, K., O'Reilly, U. M., and Taylor, C. 2014. Towards feature engineering at scale for data from massive open online courses. arXiv preprint arXiv:1407.5238.

[13] Taylor, C., Veeramachaneni, K., and O'Reilly, U. M. 2014. Likely to stop? Predicting stopout in massive open online courses. arXiv preprint arXiv:1408.3382.

[14] Hu, Y. H., Wu, F., and Liao, Y. J. 2013. An efficient tree-based algorithm for mining sequential patterns with multiple minimum supports. Journal of Systems and Software, 86, 5, (May 2013), 1224-1238.

[15] Frey, B. J., and Dueck, D. 2007. Clustering by passing messages between data points. Science, 315, 5814, (Feb. 2007), 972-976.

[16] Islam, A., and Inkpen, D. 2008. Semantic text similarity using corpus-based word similarity and string similarity. ACM Transactions on Knowledge Discovery from Data (TKDD), 2, 2, (Jul. 2008), 10.

[17] Sparck Jones, K., 1972. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, 28, 1, (Jan. 1972), 11-21.