Building Learner Activity Models From Log Data Using Sequence Mapping and Hidden Markov Models

Paras Sharma
University of Pittsburgh
Pittsburgh, PA, USA
pas252@pitt.edu

Angela E.B. Stewart
University of Pittsburgh
Pittsburgh, PA, USA
angelas@pitt.edu

Qichang Li
University of Pittsburgh
Pittsburgh, PA, USA
qil107@pitt.edu

Krit Ravichander
University of Pittsburgh
Pittsburgh, PA, USA
krit.rr@pitt.edu

Erin Walker
University of Pittsburgh
Pittsburgh, PA, USA
eawalker@pitt.edu

ABSTRACT

Open-ended learning environments (OELEs) involve high learner agency in defining learning goals and multiple pathways to achieve those goals. These tasks involve learners transitioning through self-regulated learning (SRL) phases by actively setting goals, applying different strategies for those goals, and monitoring performance to update their strategies. However, because of the flexibility, how learners react to impasses and errors has a critical influence on their learning. An intelligent pedagogical agent (IPA) continuously modeling learner activities could help support learners in these environments. However, this continuous comprehension of behaviors and strategies is difficult in OELEs with evolving goals, ill-defined problem structures, and learning sequences. In this paper, we draw from the literature on SRL phases and cognitive states to investigate the utility of two different methods, Sequence Mapping, and Hidden Markov Models, in building learner activity models from log data collected from a summer camp with 14 middle school girls in an open-design environment. We evaluate the effectiveness of these models separately, and combined, in identifying 7 states: Forethought, Engaged Concentration, Acting, Monitoring, Wheel Spinning, Mind Wandering, and Reflect and Repair. Lastly, we recommend dialogue intervention strategies for an IPA to support learning in OELEs.

Keywords

Learner-State Modeling, Open-Ended Learning Environments, Hidden Markov Models, Sequence Mapping

1. INTRODUCTION

Open-ended learning environments (OELEs) are learner-centered and provide a learning context and scaffolding to help students explore and build solutions to authentic and complex problems [10]. In certain OELEs, the inherent nature of the task makes the learning goals ambiguous and ill-defined, offering learners high agency in defining their goals and strategies to achieve them. We call these environments open-design environments. For example, in our study, we give learners a task to "build a robot protege". While some learners focus on the physical appearance of their robots, others program the robot's functionality. Even when learners have the same goal there is no fixed path to achieve them. For example, some learners first added sensors to their robots and then programmed them, while others had a mix-and-match approach. This openness in goals and strategies encourages learners' creativity and motivates them to bring their experiences and identities into technology design. This makes these environments suitable for a Culturally Responsive Computing (CRC) context that attempts to engage and empower a range of learners, in part by including interactions where learners reflect on the choices made when designing technologies, and how this intersects with their identities and experiences [25]. An intelligent pedagogical agent could help support learner activities in these highly exploratory environments. However, to do so, the agent has to continuously model learner activity states and perform strategic interventions to guide learners. In this paper, we present an attempt at modeling learner activities in open-design environments with the ultimate goal of building pedagogical agents to support learning in these environments.

Previous works in the EDM community have focused on modeling learner behaviors in educational contexts with large sample sizes and well-defined trajectories, e.g. MOOCs [24, 15, 23] and open-ended domains with fixed goals [1, 6, 12]. These models either consider learning systems with fixed pathways or learner performance as a feature in the behavior model construction. However, in open-design domains, the learner models have to come only from the learning process without any pre-defined knowledge of the learning goals or with a more expansive notion of learner performance. As per our knowledge, there is a lack of understanding of how well these traditional learner modeling methods translate to open-design contexts, especially with small sample sizes, such as ours.

In this work, we translate the approaches for learner behavior modeling from more structured educational contexts

P. Sharma, A. E. Stewart, Q. Li, K. Ravichander, and E. Walker. Building learner activity models from log data using sequence mapping and hidden markov models. In B. Paaßen and C. D. Epp, editors, *Proceedings of the 17th International Conference on Educational Data Mining*, pages 584–593, Atlanta, Georgia, USA, July 2024. International Educational Data Mining Society.

https://doi.org/10.5281/zenodo.12729890

to open-design contexts with open-ended goals and small sample sizes. We build learner activity models from multimodal log data in a robotics education curriculum where learners use various tools to design, engineer, and program their robots[4]. Robotics activities involve learners engaging in self-regulated learning (SRL [29, 22]) by actively setting and updating their goals (*Forethought*), applying different strategies for those goals (*Performance*), and monitoring their progress to update their strategies (*Self-Reflection*). We attempt to model these SRL states as they are essential markers for understanding learner progress and can provide ways to support learners as they transition through these metacognitive processes. A second set of states we model is a learner's cognitive states: Engaged-Concentration [6], Mind Wandering [18] and Wheel-spinning [11] that regulate self-learning in open-ended environments. The cognitive shifts affect learning behaviors by inducing productive and unproductive states which might impact the learners' transitions through different learning phases. Understanding these cognitive shifts is crucial for agents supporting learning particularly in open-design settings because the learner is likely to encounter impasses they need to overcome.

We use 2 techniques to model learner activity states in our work. Using 2 different methods makes our findings more reliable in the absence of ground truth data about learner states. We first introduce the Sequence Mapping technique, a combination of "Text Replays" [5], sequence diagrams for learner action mapping [26], and sequence mining, with a human-in-the-loop to build learner activity models in our open-design context by visualizing learner action sequences and theoretically grounding them to extract activity patterns. We then evaluate the transferability of Hidden Markov models, a widely used method for modeling learners' behaviors [23], to our novel open-design context. We further combine these approaches to enhance validity, gain diverse perspectives on learner states, and compare model suitability, recognizing individual models' insufficiency in identifying all cognitive states. Our research questions are:

- What activity states characterizing learner behaviors in open-design contexts can be identified using Sequence Mapping and HMMs?
- How can these methods jointly model learner actions, enabling informed interventions for an IPA?

2. BACKGROUND

Self-regulated learning involves learners continuously transitioning between three cyclical metacognitive phases: Forethought, Performance, and Self-Reflection [29, 22] throughout the task. The Forethought phase demonstrates planning and involves learners analyzing the task, setting their goals, and developing strategies. In the performance phase, learners execute the task and monitor their progress toward their goals, keeping themselves cognitively engaged. The self-reflection phase involves learners assessing their performance and updating their strategies. Supporting learners' SRL activities by building their behavioral models has been shown to enhance project-based learning [9], improve inquiry-based learning by helping in information organization capabilities [20], and promote students' learning achievements using concept maps [16]. Open-design environments, especially those supporting robotics activities, engage learners in these self-regulated learning phases by allowing them to actively set their goals, apply different strategies for those goals, and update their strategies. We model these SRL phases of learner activities when they build their "robot protege" using multimodal log data extracted by our system.

Another line of work studies learners' cognitive state shifts during learning. While engaging in learning activities, learners sometimes end up in Gaming, Mind Wandering, and Wheel Spinning cognitive states [1] or Boredom, Confusion, and Engaged Concentration affective states [6]. These states are usually associated with learner actions and are classified as desirable (Engaged Concentration) or undesirable (Mind Wandering, Gaming, and Wheel Spinning) during the learning process. For example, mind-wandering is associated with task-unrelated thoughts and boredom [18] and wheel spinning is related to long response time in learning systems [11]. Learners can sometimes transition into undesirable cognitive states in open-design environments due to the unstructured nature of the goals and pathways to achieve those goals. We model learners' cognitive states while they build their robots and identify the behaviors leading to transitions in the undesirable states, guiding the design of intelligent agents to support the transition from undesirable to desirable states.

The use of learners' task performance data and correctness of current action [6] and temporal understanding of their system interaction data [21] to build activity models has been prevalent in learning environments with fixed goals and pathways of progress. Researchers have leveraged this learner interaction data to build learner models using various machine learning and data mining techniques. Hidden Markov models have been used to understand learner dialogue processes in collaborative learning [13] and SRL processes in MOOCs [23]. Sequence mining has been used to model cognitive and metacognitive states in open-ended learning [12]. Hierarchical task modeling has been utilized to model learner actions in science learning settings [10].

All these techniques use features common in well-defined learning environments, for example, knowledge of learning goals, fixed ordering of tasks, correctness of learner actions, and time duration of interaction, and often require huge amounts of learner data. However, their translation to open-design environments, with unrestricted learning goals and pathways to achieve those goals, and smaller sample sizes is still unclear. We apply HMMs in our novel open-design context of robotics education and evaluate their effectiveness in this context. We also develop a new visual coding technique called Sequence Mapping using "Text Replays" [5] and learner action mapping using sequence diagrams [26] to build a visual representation of learner actions, followed by a human-coding process to extract behavioral patterns. Our Sequence Mapping technique applies to smaller datasets like ours and is theoretically informed, transparent, and explainable in building learner models.

3. CORPUS AND MEASURES

Task. We conducted a 2-week summer camp with 14 upper-elementary to middle-school (4^{th} to 7^{th} grades) girls, recruited through our community partner from a historically African American neighborhood in a mid-sized US city. The

learners ranged from $8-12$ years old (average = 10.36, SD = 1.2) with 12 identifying as Black and 2 with no answer. 65% of them had moderate experience with computer science and robotics through prior participation in robotics camps. 3 learners were part of a similar theme camp we ran last year. The broader camp followed a Culturally-Responsive Computing pedagogy [25] and included discussions of power and identity, community-building activities, lessons on AI Fairness, a Futuring Day, and a series of Robot-Co-Creation sessions. For this paper, we only focus on our Robot Co-Creation sessions. The learners were asked to: "*Create a robot protege to be presented on a robot runway*". This task is open as it allows learners to bring their identities and experiences to the learning activity, which is one of the tenets of CRC pedagogy [25]. The learners worked on building their robots in 4 one-hour sessions: aesthetic design, brainstorming and coding, sensor addition to the robot, and a final session to finish the development. The ethical evaluation of our study was conducted by our IRB and parental consent and learner assent were taken before the study. Moreover, learners' explicit assent was re-taken every time before they were audio or video recorded during the study.

System Used. We develop a custom system enabling learners to build and program robots using different sensors or actuators (LEDs, TriLEDs, Servos, etc.), coding blocks, design materials, and a modified Hummingbird Robotics kit[1]. Our system enables the learners to design the aesthetics of their robot, add physical sensors to the robot, program different robot functionalities using a customized Blockly interface, voluntarily record videos about their actions and goals, and respond to pre-defined dialogue scenarios using a Chatbox in the frontend. Our system automatically captures these multiple modalities of learner interactions with the robot as log data. The main data collection in the system happened through continuous logging of sensed learner events constituted by sensor activities, programming activities, dialogue activities, and learner-initiated video interaction activities.

Action Categories. We categorize the log data into 5 action categories based on the system component they are associated with. Each action category is further divided into plausible actions. Below is a list of the action categories:
(1) ***Sensors***: The logs associated with this category show the addition or removal of a sensor from the robot, along with the type of the sensor and the port that the sensor is associated with. For our learner activity models, we do not use the sensor type or associated port information and only consider the sensor addition (**SA**) and sensor removal (**SR**) as possible learner actions to focus on learner processes and not the content of their actions.
(2) ***Blocks***: This action category constitutes logs about the coding blocks and repeatedly logs any block addition (**BA**), updates (**BU**), or removal (**BR**). It also logs the values and the blocks' structural arrangement; however, we do not consider them for our final learner action models.
(3) ***Dialogue***: Dialogue Logs consist of the interactions (**D**) between the learner and the robot in the frontend dialogue interface. In the current system, the robot always triggers these interactions based on pre-defined conditions.
(4) ***Video***: Video logs relate to the learner-initiated video interactions with the system and include Video Start (**VS**) and Video End (**VE**) as plausible learner actions.
(5) ***Code Run***: This category corresponds to the action of running the code (**CR**) from the frontend interface and logs whenever a learner tries to test their robot.

We collected 2961 instances of log action data across all the learners (Mean = 199, SD = 141.15). Each of these actions also contains timestamp information associated with them. We believe both the sequence of actions and the time between subsequent actions determine a learner's behaviors and cognitive states they are in, during learning in an open-design environment. We divide the time between subsequent actions under 3 categories: *Fast* (less than 1 minute, 93% of all actions), *Regular* (between 1 min and 5 min, 5% of all actions), and *Long* (more than 5 minutes, 2% of all actions). The data is skewed with most of the actions under 1 minute. A further subdivision of these Fast actions revealed that 60% of these actions are block reordering/updates, which may not necessarily indicate cognitive shifts. Hence, we maintain the initial time segmentation for our analysis.

4. SEQUENCE MAPPING

We introduce a visual technique called sequence mapping to build patterns out of learner action sequences. Our method is based on "Text Replays" [5] which represent segments of log files in textual form to code. We further closely follow [26] to represent individual learner actions using sequence diagrams. We extend these techniques by incorporating timing information in the sequence diagram creation, and by including a qualitative coding of the diagram itself to extract theoretically grounded patterns of learner behaviors. We first individually plot sequence diagrams using the log data for all the learners following [26], then map the commonly identified observable patterns from these diagrams to a defined coding scheme. Coding these visualized sequences of learner actions from the log data could prove more useful than automatic pattern-matching techniques such as sequence mining for smaller datasets like ours. Moreover, this visual process is more transparent and explainable in extracting patterns than other black-box pattern-matching techniques.

Sequence Diagram. Figure 1 shows different clippings of sequence diagrams across learners. We build the sequence diagrams using the individual actions across all the action categories as the observable events. Each action is represented as a separate node (Action Node) in the sequence diagram, and each row in the diagram represents a logged event from the data at a particular time. A learner takes a sequence of actions over time starting from a node "S". The number and order of these actions could vary across learners based on the variability in learner goals and the multiple pathways to achieve those goals.

Each learner action is assigned to an action node. An edge is added from the current node (previous action or "S" at the beginning) to the corresponding action node in the next row, representing an action shift moving forward in time. There are 3 types of edges that could be there between 2 action nodes (A & B) or the same action node(A & A): Edge 1 (Red) represents a *Fast* transition from node A to node B, Edge 2 (Blue) represents a *Regular* time transition from

[1] `https://www.birdbraintechnologies.com/products/hummingbird-bit-robotics-kit/`

Table 1: Table shows the set of rules developed after a series of qualitative coding.

Forethought: Exploration and Planning (4.2%)
At the start of the process, there are a variety of "Fast" or "Regular" time subsequent actions between different action categories.
More than 3 different types of actions across different action categories not under "Long" time category.
The actions are at the beginning of the process.
Engaged Concentration (27.81%)
The learner is under the same action category, but transitioning between states with "Fast" or "Regular" transition times.
Consistent progress with predominantly Fast or Regular actions with minimal periods of inactivity across Sensors and Blocks.
All actions within Regular time are related to building /coding (adding, removing, updating, running code)

node A to node B, and Edge 3 (Violet) represents a *Long* transition from A to B. There could also be weighted edges at the same level between 2 action nodes. This weight represents the number of times the learner continuously transitioned between these action nodes before moving to a different action node.

Qualitative Coding. We performed a qualitative coding of the sequence diagrams to generate a list of codes identifying patterns across different learner action sequences. We used a mix of inductive and deductive coding approaches. Initially, the first author went through the sequence diagrams and generated a codebook based on the patterns in the diagrams. The codes represented a learner's cognitive and metacognitive states and were theoretically grounded in the 4 learning phases (forethought, monitoring, control, and reaction and reflection) in [22] and the metacognitive states in [27]. 2 other authors used this initial codebook to extract patterns from data while adding to the codes or code definitions. After this coding session, all the authors agreed upon a collective list of 7 codes and their definitions. Table 1 shows the codebook generated after the coding process with the percentage of the occurrence of these states across the learners. 3 authors then independently coded the data for 2 learners (Krippendorf's alpha [19] = 0.583) using the codebook.

Rule-based Pattern Matching. We operationalize the fuzzy terms from the code book to convert them to rule-based Pseudocodes. Table 2 shows Pseudocodes for pseudocodes of Forethought and Engaged Concentration states. We then use these pseudocodes in pattern-matching algorithms to automatically determine a learner's state at a particular moment using the log data. These algorithms use the action history from data sequences based on the time categories to identify the patterns occurring over time.

Results. Based on the qualitative coding followed by rule-based pattern matching we identified 7 activity states across all the learners. Figure 1 shows the sequence diagrams for "Forethought" and "Mind Wandering" states. All the other state diagrams are presented in the appendix E. The percentage associated with each state represents the fraction of time that state occurred across the learners.

(1) ***Forethought: Exploration and Planning (***4.2%***)***: This activity state corresponds to the situation when learners are unsure of their goals or the system functionality and hence are trying the different components of the system. This ties back to the "Forethought" phase in Zimmerman's Cyclical Phases model of self-regulated learning [29] where a learner defines an understanding of the task and is ready to enter the goal-setting and planning process.

(2) ***Wheel Spinning (***1.62%***)***: This cognitive state means that the learners are unsure of how to proceed further from the current state. This links to a "Wheel Spinning" cognitive state under unproductive thinking when the student makes an effort but does not succeed, leading to long response times and many help requests [11].

(3) ***Engaged Concentration (***27.81%***)***: This cognitive state signifies that the learner is implementing a strategy towards their goal and is in a concentrated state. It reflects the "enacting study tactics and strategies" phase of studying from Winne and Hadwin's SRL model [27], where a learner is acting upon the strategies to reach their goals.

(4) ***Mind Wandering (***12.29%***)***: This state is associated with the "Mind-wandering" cognitive state and is related to boredom and attention shift toward task-unrelated thoughts [18]. It could also indicate distraction, e.g. performing these actions while talking with another learner (which is not inhibited by our study design), hence not interacting quickly enough with the system. Additional logging by the system might be needed to disambiguate between these states.

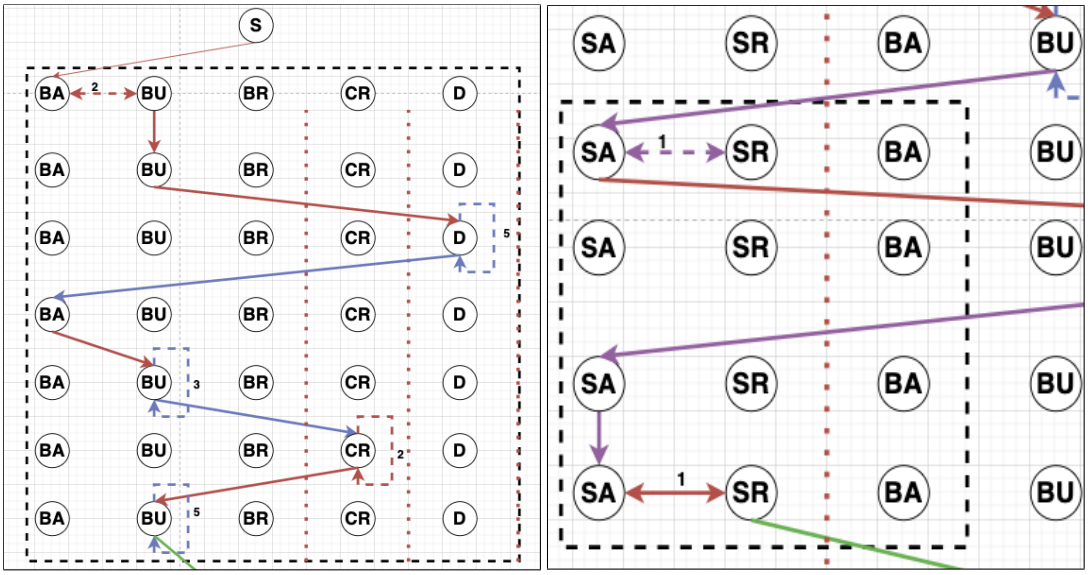
(5) ***Monitoring and Testing (***21.02%***)***: This means that the learner is testing the different system components. This relates to the "Monitoring" phase in [22] Pintrich's SRL model [22] and the "Performance" phase in the cyclical phases model [29] where learners execute the task while monitoring their progress. However, this pattern could have different meanings across different time intervals in our open-design context. If this occurs at the end of the learning process, this could mean that the learner is just checking whether everything is working correctly or showing off the functionality of their robot to other people. A similar pattern in the middle of the learning process following an Acting state might mean that the learners are testing the functionality of the components they just added.

(6) ***Reflect and Repair (***4.92%***)***: This state circles back to the "Reaction and Reflection" phase in [22] and "Self-Reflection" in [29] where students evaluate the task and their performance. This state could also mean debugging or good coding practice. It can have learners in a concentrated engagement making goal progress.

(7) ***Acting (***28.14%***)***: This is the default state for a learner action and signifies that a learner is interacting with the system in some form.

5. HIDDEN MARKOV MODEL

Our sequence mapping technique revealed different activity states that learners transition through in our open-design context. Even though our technique has a qualitative step, we do not have explicit ground truth information about



(a) "Forethought" as observed in learners.

(b) "Mind Wandering" as observed in learners.

Figure 1: Sequence Diagrams. Edge 1 (Red) represents a fast transition ($<$ 1 min), Edge 2 (Blue) Represents a regular transition ($>$ 1 min and $<$ 5 min), and Edge 3 (Violet) is a long transition ($>$ 5 min). The weight on an edge represents the number of times a learner transitioned through that edge. The transition time for an edge represents the total time the transition was repeated.

learner states to verify these findings. Hence, we experiment with a widely used learner behavior modeling method, the Hidden Markov Model (HMM), and evaluate how it translates to our context and complements the sequence mapping results. An HMM is a probabilistic model based on the chain of Markov processes (sequence of events where the next event only depends on the current event) and can be used to describe latent (hidden) events that lead to a sequence of observations. We consider all the 9 action types as observations in our model and further add timing information based on the 3 time categories defined above. For example, if a random log data corresponds to Block update (BU) and the learner took this action after a "Regular" time from the previous action, the feature for this data becomes $[BU, 2]$. To identify the optimal number of hidden states, we train HMMs on these derived features for these sequences with hidden states ranging from 2 to 9 using a leave-one-out cross-validation approach. We identified models with 5 (HMM-5) and 6 (HMM-6) hidden states to perform the best across Akaike Information Criteria (AIC), Bayesian Information Criteria (BIC), and log-likelihood (see Appendix B. However, an extra state in the HMM-6 was a low-probability state and could be collapsed into the other states in the HMM-5 without any information loss. Hence, we only consider the HMM with 5 states for further analysis.

We induced an HMM with 5 hidden states on all the learner data (see Figure 2), and mapped the identified hidden states to learner actions to understand the hidden learner activity states. The HMM identified states 2 and 4 as the only starting point for the learners, with 2 being the most probable. State 4 corresponds to all the visible components (Video Recording, Code Run, Dialogue) of the system. A learner starting in this state means that they are *exploring* the different options present on the interface. Hence, at the beginning of the session, this state corresponds to an "Exploration" state for the learner. However, this is also the only state with a high probability of the "Code Run" action being observed. This means that if a learner enters this state at a later point in the session, e.g. after they perform some actions related to other system components, they could be in a "Testing" phase. Hence, a pedagogical agent making these learner models needs to consider the timing information of when a learner enters a particular state, as a state could have different meanings based on when in the session it occurs. The most probable way to transition out from state 4 is to move to state 1 which mostly emits sensor-related actions. This means that after either "exploring" or "testing" in the state 4 a learner mostly performs sensor actions in their robot. The most probable way to transition into this state is from the state 1, meaning that a learner is immediately testing whatever they are doing with the sensors.

Starting state 2 is mostly associated with block addition or update actions. This indicates that a learner has an initial idea of their programming goals when they start in this state and are making slow actions toward it, e.g. adding a block and then updating it to decide what they want that block to do. Hence, this state combines exploration and goal definition and links back to the "Forethought" phase in [29] and [22]. The learners in this state have formed goals but are still investigating the system's functionality and learning how to navigate through the system. However, this state could also relate to Engaged Concentration later in the sessions, where the learner knows about their goals and works towards updating things to make the different components. This state is almost in a cyclical transition with the state 5 that mostly emits the sensor addition action. This indicates that a learner interacting with the system performs the sensor and the block-related actions sequentially. They

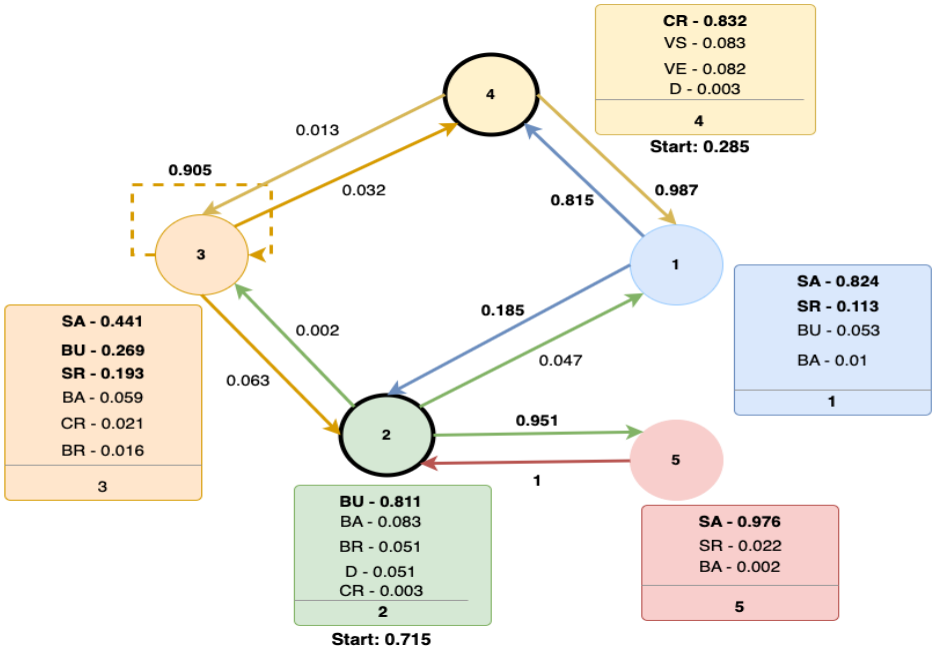


Figure 2: Figure shows HMM with 5 hidden states. The circular nodes represent the hidden states and the transition between the states identified by the models. The rectangular boxes represent the emitted states from a hidden state along with their probability. The hidden states with "Black" borders are the start states identified by the models.

might add a sensor in the state 5 and then move back to the state 2 to add/update/remove/test the corresponding blocks, and this almost happens most of the time, until they either transition to the state 3 or state 1.

The state 1 is mostly associated with the sensor actions and could be considered an "Acting" state for the sensor-related updates. A learner in this state mostly transitions to the state 4 to verify their updates, which is a good coding practice, or with a low probability move to the state 2 for block-related updates. This transition cycle between the states 1, 4, and 2 could be considered an "Engaged Concentration" or "Acting" state. In this joint cognitive state, a learner has already built strategies for their goals and then is transitioning between these states to *enact those strategies* [27].

The hidden state 3 hints towards a stuck or confused state. If a learner enters this state, they mostly remain in it, until they finally transition to the state 2 or state 4. A learner in this state is mostly adding/removing sensors or updating the code blocks. This corresponds to a "Wheel Spinning" cognitive state with the learners stuck over a repeated set of actions for a long time [11]. A learner does not enter this state frequently but could transition into this from either state 4 (if something fails and the learner doesn't know what to do, so they are just stuck trying different things and not moving forward) or from state 2 (learner updated their code but now don't know how to proceed further and are just playing around with sensors/blocks).

State 5 corresponds to a learner taking sensor-related actions. This state combined with the state 2 corresponds to an "Acting" state. A learner enters this state from the state 2 and then immediately transitions back to the state 5, meaning these 2 states jointly represent a single cognitive behavior, related to action.

To summarize, the HMM identified 5 states: Forethought, Acting, Engaged Concentration, Monitoring and Testing, and Wheel Spinning.

6. ASSESSING LEARNER STATE BY COMBINING THE MODELS

Building a joint learner state assessment using the two models depends on the specific learning objectives and the desired level of accuracy in the analysis. Both the models only agree on the detected states 49.25% of the time, with high agreement on Engaged Concentration (63.29%), but lower agreement on Acting (27.34%), Monitoring and Testing (26.81%), and Forethought (16.48%) and a rare agreement on Wheel Spinning. Because Engaged Concentration is accurately detected by both models, either one can be used to detect learner behaviors in this state. But for some states, it may be useful to rely more on one model's predictions over the other's. For example, the Sequence Mapping model does not correctly distinguish between Forethought and Acting states, due to highly similar codes for both states. So if the objective is to decide when learner actions indicate they are in a Forethought state, the HMM would be a better choice. As the HMM model could not capture the Mind Wandering or Reflect and Repair behaviors, Sequence Mapping would be a better choice for those states.

Additionally, HMM only considers the current state to predict the next state. Thus, it can perform better than Sequence Mapping, which requires action history, in the initial learning stages. The predictions from Sequence Mapping

grow better as the actions accumulate over time. Hence, if the objective is to understand a combination of learner states over a period of time, we could start with more confidence in the HMM model's predictions and then gradually add more confidence in Sequence Mapping predictions with time. However, a more advanced study with evaluation metrics to compare these models is necessary to make definite conclusions.

The learner's state assessment also provides intervention opportunities for an intelligent pedagogical agent involved in the learning process using dialogues. Dialogue as an intervention strategy has been used to help students solve problems [14], explain their actions [2], and construct knowledge [17]. Thus, it could be used as a tool for effective interventions by an intelligent pedagogical agent in open-design contexts. For example, the agent could help learners progress toward their goals through dialogues when it detects they are in a "Wheel Spinning" state. It could initiate reflection interactions to encourage learners to explain and reflect on their actions [3, 2]. Moreover, it could try to employ social interactions when it detects learners are in a "Mind Wandering" state, to re-engage them in the activity and motivate them to focus on their goals.

7. DISCUSSION AND CONCLUSION

We present two distinct methods, Sequence Mapping, and Hidden Markov Models, to model learner activities in open-design environments. We give the states identified by each model and compare the effectiveness of these models to understand learner activities. We then recommend dialogue strategies for pedagogical agents supporting learning in open-design settings.

The states that we identify as part of this work are well-grounded in two theoretical frameworks, Self-Regulatory learning, and cognitive state modeling, and thus understanding those states has the potential to contribute back to those frameworks by incorporating empirical findings from our work to potentially extend existing understanding of learner behaviors. One of the states that we do not engage with in our models is Gaming [7], related to quick actions and short response times [8]. It is not relevant in our settings as most learner actions in our system are quick. We also only engage with learner processes and not the content of learner actions in our models. However, we do believe that the specific information about learner actions could provide additional insights into learner behaviors. For example, understanding what learners program their robots to "say" might indicate more about their tastes and preferences.

The Sequence Mapping technique we propose has several distinct advantages. It is both explainable and allows for theory-informed decisions. Hence, any decision that could be made by a pedagogical agent based on these detected states involved in the learning process could be easily justified to all the stakeholders (students, teachers, educators, etc.). This transparency is essential to build models that provide ways for learners/educators to understand the system decisions [28], which is essential to give stakeholders agency in the use of a system that models learner activities using multimodal log data. Moreover, this method is appropriate for contexts with small sample sizes. We argue that it is important to do this kind of data mining research on locally run community programs, rather than solely using data from large-scale online collections so that these methods can be applied to hyper-specific contexts that may be relatively unique. We expect our methods to transfer to similar open-ended contexts.

Compared to sequence mapping, applying HMMs is a more widely used approach due to their ability to model sequential data, applicability to unlabeled datasets, and flexibility in state representation to capture various aspects of the learning process, such as cognitive states, or engagement levels. Nonetheless, the HMM's premise of decision-making solely reliant on the present state may not effectively capture the dynamics of open-design contexts, particularly behaviors necessitating knowledge beyond the current state. Although we integrated time as a variable in our HMM, we think that more temporally sensitive models, such as RNNs, could detect learner states better.

One limitation of our study is the lack of a clear evaluation metric for the states identified. Techniques like think-aloud and field observations have effectively collected ground truth for learner affective and cognitive states in previous works [12]. However, they have limited suitability in our context due to their potential disruption to the learning process. Hence, we introduced a qualitative step in our analysis to align observations with established theories and used two methods to validate our findings. This qualitative step may be time-consuming but aids in visualizing learner behaviors and theoretically grounding observed patterns. Following this qualitative step, learner states are automatically extracted using scripts in real-time, ensuring efficiency and scalability beyond the initial qualitative phase. Since our current work identifies relevant states and related behaviors we could use this information in future system iterations, as a part of learner-agent dialogue interactions or open-learner models. This approach would allow us to verify the correctness of identified states in the future.

Conclusion. This paper offers insights into modeling learner activities in open-design environments and lays the groundwork for future exploration and application of these models to different open-design learning contexts. By contributing in this direction, we hope to enhance the effectiveness of pedagogical support in adaptive learning environments, such as those used in culturally-responsive pedagogies, that empower learners to choose their own goals and strategies to achieve them.

8. ACKNOWLEDGMENTS

We thank Dr. Tara Nkrumah, Dr. Amy Ogan, and Dr. Kimberly Scott for their leadership on this project, as well as the rest of the project team for their support with system design and running the summer camp (Veronica Bella, Angeline Pho, Paige Branagan, Nicole Balay, and Adeola Adekoya). This work is funded by NSF DRL-1811086 and DRL-1935801.

9. REFERENCES

[1] Z. Aghajari, S. Deniz, M. Unal, E. Mesut, L. Gómez, and E. Walker. Decomposition of response time to

give better prediction of children's reading comprehension. 02 2021.

[2] V. Aleven, A. Ogan, O. Popescu, C. Torrey, and K. Koedinger. Evaluating the effectiveness of a tutorial dialogue system for self-explanation. In J. C. Lester, R. M. Vicari, and F. Paraguaçu, editors, *Intelligent Tutoring Systems*, pages 443–454, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[3] V. A. Aleven and K. R. Koedinger. An effective metacognitive strategy: learning by doing and explaining with a computer-based cognitive tutor. *Cognitive Science*, 26(2):147–179, 2002.

[4] J. Angel-Fernandez and M. Vincze. Towards a formal definition of educational robotics. pages 37–42, 07 2018.

[5] R. Baker and A. Carvalho. Labeling student behavior faster and more precisely with text replays. pages 38–47, 01 2008.

[6] R. Baker, J. Kalka, V. Aleven, L. Rossi, S. Gowda, A. Wagner, G. Kusbit, M. Wixon, A. Salvi, and J. Ocumpaugh. Towards sensor-free affect detection in cognitive tutor algebra. *Proceedings of the 5th International Conference on Educational Data Mining*, page 126–133, 01 2012.

[7] R. S. Baker, A. T. Corbett, K. R. Koedinger, and A. Z. Wagner. Off-task behavior in the cognitive tutor classroom: when students "game the system". In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, page 383–390, New York, NY, USA, 2004. Association for Computing Machinery.

[8] R. S. Baker, A. T. Corbett, K. R. Koedinger, and A. Z. Wagner. Off-task behavior in the cognitive tutor classroom: when students "game the system". In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, page 383–390, New York, NY, USA, 2004. Association for Computing Machinery.

[9] M. Barak. From 'doing' to 'doing with learning': reflection on an effort to promote self-regulated learning in technological projects in high school. *European Journal of Engineering Education*, 37:105–116, 03 2012.

[10] S. Basu, G. Biswas, and J. S. Kinnebrew. Learner modeling for adaptive scaffolding in a computational thinking-based science learning environment. *User Modeling and User-Adapted Interaction*, 27(1):5–53, Mar. 2017.

[11] J. E. Beck and Y. Gong. Wheel-spinning: Students who fail to master a skill. In H. C. Lane, K. Yacef, J. Mostow, and P. Pavlik, editors, *Artificial Intelligence in Education*, pages 431–440, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[12] G. Biswas, J. Segedy, and J. Kinnebrew. Smart open-ended learning environments that support learners cognitive and metacognitive processes. volume 7947, 01 2013.

[13] T. V. Earle-Randell, J. B. Wiggins, J. M. Ruiz, M. Celepkolu, K. E. Boyer, C. F. Lynch, M. Israel, and E. Wiebe. Confusion, conflict, consensus: Modeling dialogue processes during collaborative learning with hidden markov models. In N. Wang, G. Rebolledo-Mendez, N. Matsuda, O. C. Santos, and V. Dimitrova, editors, *Artificial Intelligence in Education*, pages 615–626, Cham, 2023. Springer Nature Switzerland.

[14] M. Evens, R.-C. Chang, Y. Lee, L. Shim, C.-W. Woo, and Y. Zhang. Circsim-tutor: An intelligent tutoring system using natural language dialogue. pages 13–14, 01 1997.

[15] Y. Fan, W. Matcha, N. A. Uzir, Q. Wang, and D. Gašević. Learning analytics to reveal links between learning design and self-regulated learning. *Int. J. Artif. Intell. Educ.*, 31(4):980–1021, Dec. 2021.

[16] J.-W. Fang, L.-Y. He, G.-J. Hwang, X.-W. Zhu, C.-N. Bian, and Q.-K. Fu. A concept mapping-based self-regulated learning approach to promoting students' learning achievement and self-regulation in stem activities. *Interactive Learning Environments*, 31(10):7159–7181, 2023.

[17] A. C. Graesser, K. VanLehn, C. P. Rose, P. W. Jordan, and D. Harter. Intelligent tutoring systems with conversational dialogue. *AI Magazine*, 22(4):39 – 51, Dec. 2001.

[18] S. Hutt, J. Hardey, R. Bixler, A. Stewart, E. F. Risko, and S. Mello. Gaze-based detection of mind wandering during lecture viewing. In *EDM, Proceedings of the 10th International Conference on Educational Data Mining*, pages 226–231. ERIC, 2017.

[19] K. Krippendorff. Computing krippendorff's alpha-reliability. 2011.

[20] C.-L. Lai, G.-J. Hwang, and Y.-H. Tu. The effects of computer-supported self-regulation in science inquiry on learning outcomes, learning processes, and self-efficacy. *Educational Technology Research and Development*, 66(4):863–892, Aug 2018.

[21] C. Paans, I. Molenaar, E. Segers, and L. Verhoeven. Temporal variation in children's self-regulated hypermedia learning. *Computers in Human Behavior*, 96:246–258, 2019.

[22] P. Pintrich. The role of goal orientation in self-regulated learning. *Handbook of Self-regulation*, 12 2000.

[23] L. Qiao, W. Zhao, and X. Xu. Mining and analysis of self-regulated learning process model: Based on hidden markov model. In *2021 Tenth International Conference of Educational Innovation through Technology (EITT)*, pages 276–281, 2021.

[24] N. Rohani, K. Gal, M. Gallagher, and A. Manataki. Discovering students' learning strategies in a visual programming mooc through process mining techniques. In M. Montali, A. Senderovich, and M. Weidlich, editors, *Process Mining Workshops*, pages 539–551, Cham, 2023. Springer Nature Switzerland.

[25] K. Scott, K. Sheridan, and K. Clark. Culturally responsive computing: a theory revisited. *Learning, Media and Technology*, 40:1–25, 12 2014.

[26] S. Wang, D. Sonmez Unal, and E. Walker. Minddot: Supporting effective cognitive behaviors in concept map-based learning environments. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–14, New York, NY, USA, 2019. Association for Computing

Machinery.
[27] P. Winne. A cognitive and metacognitive analysis of self-regulated learning. *Handbook of Self-regulation of Learning and Performance*, pages 15–32, 01 2011.
[28] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu. Explainable ai: A brief survey on history, research areas, approaches and challenges. In J. Tang, M.-Y. Kan, D. Zhao, S. Li, and H. Zan, editors, *Natural Language Processing and Chinese Computing*, pages 563–574, Cham, 2019. Springer International Publishing.
[29] B. J. Zimmerman. Attaining self-regulation: a social cognitive perspective. pages 13–39, 2000.

APPENDIX

A. PSEUDOCODE FOR PATTERN MATCHING

Table 2 shows the pseudocode developed for the Forethought and Engaged Concentration states.

Table 2: Pseudocode for rule-based pattern-matching for Forethought and Engaged Concentration states.

Forethought
Action Window is all actions within past 10 minutes **If** count(unique categories in Action Window) $>$ 3 and Current Action **between** first third of all actions: **ForEach** action in Action Window: **Add** "Forethought" code to action
Engaged Concentration
Action Window is all actions within past 10 minutes **If All** categories in Action Window **in** [Block, Sensor, or Code Run] **and** Current Action **within** last two-thirds of all actions: **ForEach** action in Action Window: **Add** "Engaged Concentration" code to action

B. SELECTING IDEAL NUMBER OF HMM STATES

To identify the best number of hidden states, we train a series of models with hidden states ranging from 2 to 9 using a leave-one-out cross-validation approach for all the learner sequence data. We then used Akaike Information Criteria (AIC), Bayesian Information Criteria (BIC), and Log Likelihood (LL) to compare all the trained models to select the best number of hidden states. Figure 4 shows the model comparison results. We find the number of hidden states 5 and 6 to perform best on average across all the 3 evaluation criteria. We then train 2 models one with 5 (HMM-5) and the other with 6 (HMM-6) hidden states with all the sequence data. We compared both the emerging HMMs. HMM-6 had a similar arrangement of certain states to HMM-5. The main difference between the 2 HMMs was that states 3 and 5 from HMM-5 were split across states 1, 2, and 5 in HMM-6, producing the new state 1 in HMM-6. But the chance of a learner transitioning into this state is very low, 0.064 from state 2 and 0.066 from state 3. The transition out of this state is highly likely towards either the start state 6 (similar to the start state 2 in HMM-5) or to the state 2 (similar to state 3 in HMM-5), and hence this state could be collapsed into the states 2 and 5 without losing any extra information. It shows that the HMM-6 is likely overfitting the data (there are a lot of very less likely transitions) and hence would not generalize better.

C. LEARNER CREATION

Figure 3 shows an image of a robot created by a learner and the blocks programmed by the learner from the summer camp.

D. CODES REPRESENTING EACH LEARNER STATE

Table 3 shows the codes for Acting, Monitoring and Testing, Wheel Spinning, Mind Wandering, and Reflect and Repair states.

Table 3: Table shows the set of rules developed after a series of qualitative coding.

Acting (28.14%)
Default Category for all actions.
Monitoring and Testing (21.02%)
The learner is transitioning in the "Code Run" category for a "Fast" or "Regular" time.
More than half of the actions in the time window are "Code Run", combined with at least two kinds of action types(adding, removing, updating blocks or sensors)
Wheel Spinning (1.62%)
The learner is under the same action category and action state for a "Long" time before transitioning to an action state under a different action category
Current action has a long duration and is the same type as the previous action
Mind Wandering (12.29%)
The learner is transitioning continuously between 2 action states under the same action category(except "Code Run") for a "Long" transition time
Irregular action patterns with both "regular" and "long" time between actions.
Reflect and Repair (4.92%)
The learner is continuously transitioning between "Code Run" and "Sensor/Block" action categories with "Fast/Regular" transition times for at least thrice in a "Long" time.

E. SEQUENCE DIAGRAMS

Figures 5, 6, 7, and 8 show the sequence diagrams representing the states Testing, Wheel Spinning, Engaged Concentration, and Reflect and Repair respectively.

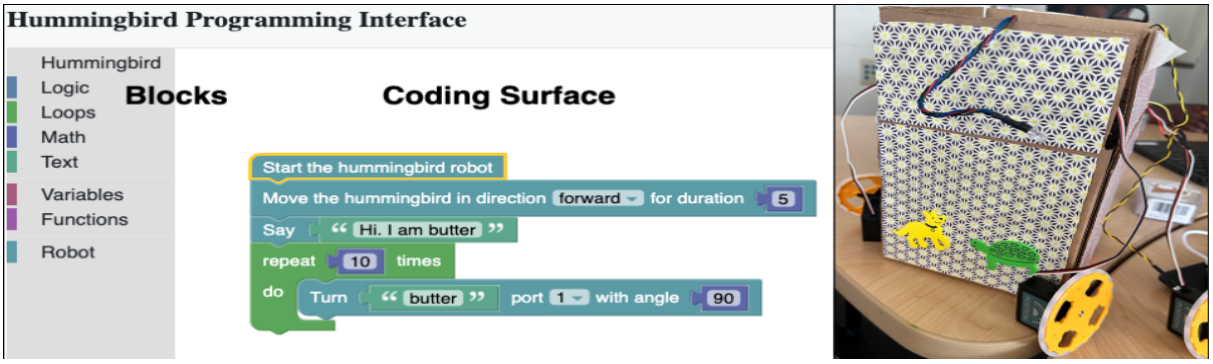


Figure 3: Image of the blocks added and the robot created by a learner from the camp.

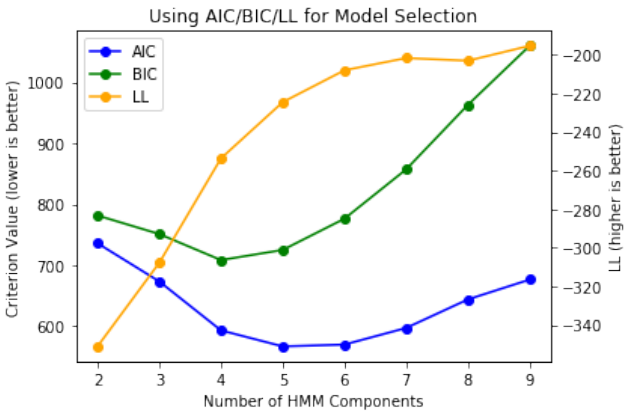


Figure 4: Evaluation metrics AIC/BIC and Log Likelihood to select best HMM model.

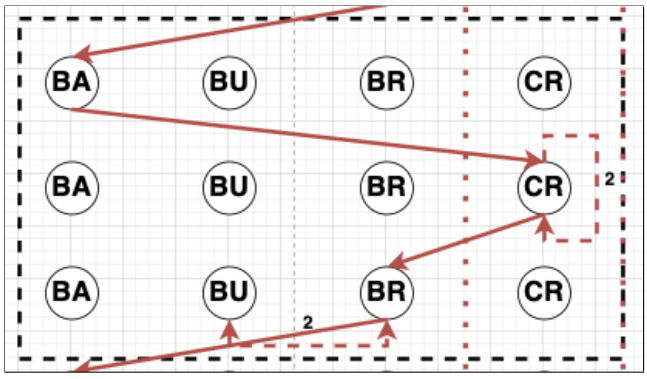


Figure 5: "Testing" as observed in learners.

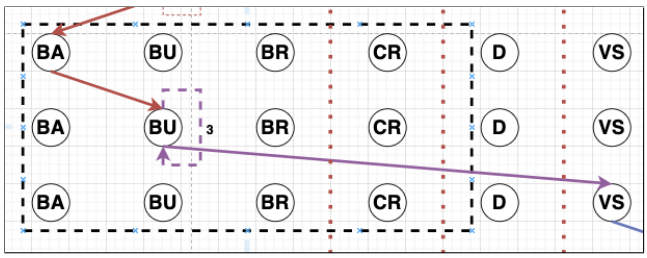


Figure 6: "Wheel Spinning" as observed in learners.

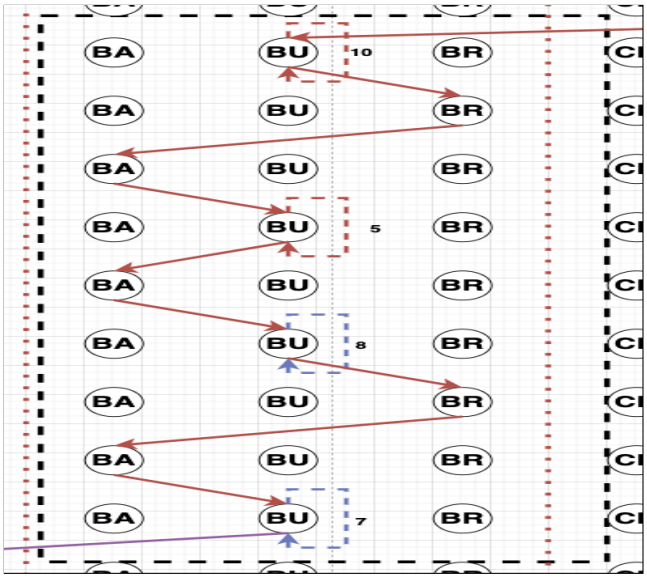


Figure 7: "Engaged Concentration" as observed in learners.

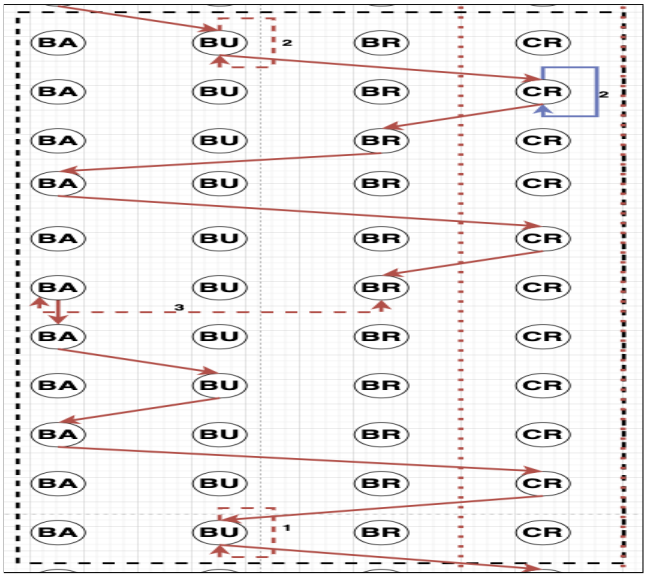


Figure 8: "Reflect and Repair" as observed in learners.