An architecture for identifying and using effective learning behavior to help students manage learning

Paul Salvador Inventado*, Roberto Legaspi, Koichi Moriyama, Ken-ichi Fukui and Masayuki Numao
The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka, Japan, Osaka, 567-0047
{inventado,roberto,koichi,fukui}@ai.sanken.osaka-u.ac.jp, numao@sanken.osaka-u.ac.jp

ABSTRACT

Self-regulated learners are successful because of their ability to select learning strategies, monitor their learning outcomes and adapt them accordingly. However, it is not easy to measure the outcomes of a learning strategy especially while learning. We present an architecture that allows students to gauge the effectiveness of learning behavior after the learning episode by using an interface that helps them recall what transpired during the learning episode more accurately. After an annotation process, the profit sharing algorithm is used for creating learning policies based on students' learning behavior and their evaluations of the learning episode's outcomes. A learning policy contains rules which describe the effectiveness of performing actions in a particular state. Learning policies are utilized for generating feedback that informs students about which actions could be changed or retained so that they can better adapt their behavior in future learning episodes. The algorithms were also tested using previously collected learning behavior data. Results showed that the approaches are capable of building a logical learning policy and utilize the policy for generating appropriate feedback.

Keywords

delayed feedback, self-regulated learning, profit sharing

1. INTRODUCTION

Students often learn on their own when they study for tests, make assignments and perform research as part of their academic requirements. They also learn by themselves when they investigate topics which may not be directly related to class discussions but are interesting to them. When students learn alone, they encounter many challenges related to the learning task, as well as challenges that are meta-cognitive and affect related.

Students who can self-regulate are capable of overcoming these challenges better compared to those who cannot. One reason for this is that self-regulated students know how to select and adapt their learning strategies depending on the current situation. However, this is a complex task because it requires attention and sophisticated reasoning to know which learning strategies to apply, to monitor the outcomes of a learning strategy and to know when a strategy needs to be changed [13].

In this research, we discuss an architecture for helping students manage their learning behavior by helping them become aware of the outcomes of the learning strategies they employed and by helping them identify which strategy is effective in a particular situation.

2. RELATED WORK

Self-regulated learners can be differentiated from less self-regulated learners by looking at the learning behaviors they exhibit. They are characterized by their diligence and resourcefulness, their awareness of the skills they possess, their initiative to seek out information and their perseverance to continue learning and find ways to overcome obstacles [13].

Research such as that of Kinnenbrew, Loretz and Biswas [8] has shown these differences in behavior. In their work, they investigated students' learning behavior while using Betty's Brain, a computer-based learning environment in the science domain that helped students develop learning strategies. They processed log data from student interactions and mapped them to canonical actions. Action sequences were then mined using sequential pattern mining and episode mining to discover learning behaviors. Their results showed that high performing students showed systematic reading behavior and frequent re-reading of relevant information which was not seen in low performing students.

In the work of Sabourin, Shores, Mott and Lester [10], the authors also observed differences in the students' behavior as they interacted with Crystal Island, a game-based learning environment developed for the microbiology domain. While interacting with the environment, students were prompted to report their mood and status. These were later processed and used to categorize the students' goal setting and goal

*also affiliated with: Center for Empathic Human-Computer Interactions, College of Computer Studies, De La Salle University, Manila, Philippines

reflection behavior. They were then given an overall self-regulated learning (SRL) score based on their reports and assigned into low, medium or high SRL category. Students in the high SRL category frequently used in-game resources that provided task-related information and resources that allowed them to record notes. They also spent less time using resources for testing their hypothesis and had higher learning gains.

MetaTutor is a hypermedia learning environment developed for the biology domain that identifies students' SRL processes and also helps them use these processes [2]. Students who used the system indicated the SRL processes they used by selecting it from the list of SRL processes in the system's interface. Pedagogical agents also gave them prompts to use certain SRL processes depending on the current situation (i.e., student information, time on page, time on current sub-goal, number of pages visited relevance of the current page to the sub-goal, etc.) and also gave them feedback regarding how they used these processes. Students who used the version of the system with prompts and feedback were reported to have higher learning efficiencies compared to students who used a version of the system without prompts and feedback.

3. SYSTEM ARCHITECTURE

Learners often have difficulty in selecting, monitoring and adapting learning strategies because of its high cognitive load requirement. This is especially true for complex domains such as science, math, engineering and technology. The approach we take in this work involves helping students understand the outcomes of their learning behavior better by helping them recall what transpired in a recently concluded learning episode. The advantage of recalling is that after the learning episode, students do not need to worry about the learning task and can focus on analyzing their learning behavior. Students will also have a more complete and accurate measurement of their learning behavior's effectiveness because they can observe both short and long term effects on learning. This information will be useful for students in future learning episodes because when they monitor and adapt learning strategies, they can base their decisions on the current context as well as their predictions of what could happen according to their reflections from previous learning episodes.

Asking students to recall a recently concluded learning episode presents two issues. First, students will not be able to completely remember what transpired during the learning episode. We addressed this in our previous work wherein we developed a tool called Sidekick Retrospect, which took screenshots of the students' desktop and video frames from a video of their face during a learning episode [7]. Students who used the software in our experiment reported that they were able to discover things about their behavior that they were previously unaware of. It was also enough to help them reflect on what transpired so that they were able to identify problems with their learning behavior and think of probable solutions. Figure 1 shows a screenshot of the system's interface which are presented to the students after the learning episode. A timeline of the entire learning episode is shown together with desktop and webcam video screenshots relative to the mouse's position in the timeline.

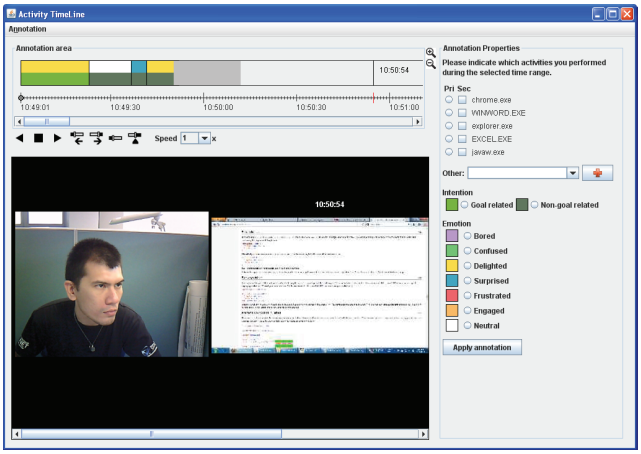

Figure 1: Sidekick Retrospect Annotation Interface

An issue we encountered from our previous work was that students who used the software seemed to focus only on the most significant aspect of the learning episode. They did not reflect as much on other instances during the learning episode even when they employed other learning strategies that also had an impact on their learning. This may have been the case because students were already too tired to spend more time analyzing each event in depth.

The architecture presented in Figure 2, integrates the methodology we used in our previous work with our current approach for helping students recall what transpired during the learning session and helping them discover more insights about their learning behavior. We designed our system so that students would not be bound by a specific environment or domain and keep the learning environment as natural as possible. Students were allowed to learn using any tool or application on or off the computer. However, they had to stay in front of the computer so it could take desktop and webcam video screenshots of their activities and so they could annotate the data after the learning episode. The entire process was split into three phases which are each discussed in the following subsections.

3.1 Interaction Phase

The interaction phase begins by first asking students to input their learning goals for the current learning episode. Data collection starts right after students finish inputting their goals. The system then starts logging the applications used by the students, the title of the current application's window and the corresponding timestamps. Screenshots of the desktop and the webcam's video feed are also taken and stored using the same timestamp as that of the log data.

3.2 Annotation Phase

In the annotation phase, students are asked to annotate their *intentions*, *activities* and *affective states*. Intentions can either be goal related or non-goal related relative to the goals that were set at the start of the learning episode. Activities referred to any activity the student did while learning which could either be done on the computer (e.g., using a browser) or out of the computer (e.g., reading a book). Two sets of affect labels were used for annotating affective states

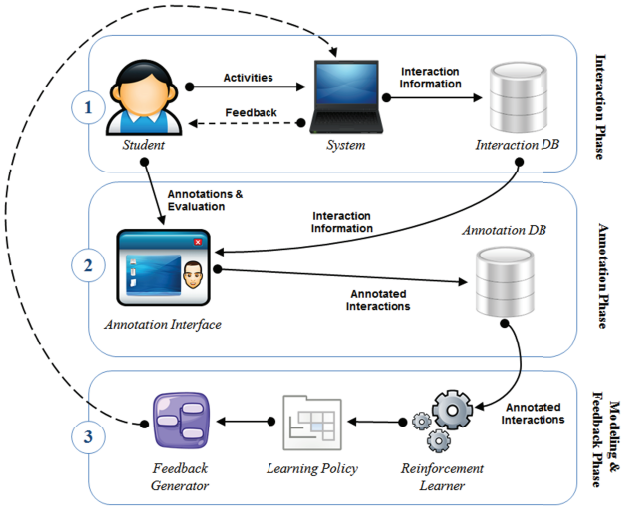

Figure 2: System architecture

wherein goal-related activities were annotated as either delighted (DEL), engaged (ENG), confused (CNF), frustrated (FRS), bored (BRD), surprised (SRP) or neutral (NUT) and non-goal related activities were annotated as either delighted (DEL), sad (SAD), angry (AGY), disgusted (DIS), surprised (SRP), afraid (AFR) or neutral (NUT). Academic emotions [4] are used for annotating goal related intentions because they give more contextual information about the learning activity. However, academic emotions might not capture other emotions outside of the learning context so Ekman's basic emotions [5] were used to annotate non-goal related intentions.

The system's annotation interface helps students recall what transpired during the learning episode by showing desktop and webcam screenshots depending on the position of the mouse on the timeline. The actual annotations can be created by using the mouse to select a time range then clicking on the corresponding intention, activity and affective state buttons. Students are also allowed to input a description of the activity when it was done outside of the computer.

While annotating, students inherently recall what transpired allowing them to identify the appropriate annotation. Going through the entire learning episode sequentially also helps the students annotate more accurately because they can see how and why their activities change. Furthermore, they also see the outcomes of these activities. It is possible that students might not annotate the data correctly for fear of judgment or lower scores. However, reassuring them that the results will not be used as part of their grades or telling them that accurately annotating their data will help them become more self-regulated and effective could help minimize these cases.

After the annotation process, students are asked to give a learning effectiveness rating between one to five, indicating how good they felt the learning episode was. This rating is likely to be accurate because of the level of detail in which students reviewed their learning episode.

3.3 Modeling and Feedback Phase

In the modeling and feedback phase, students' data are analyzed to create and update the student's list of effective learning behavior or policy. Students' behavior in the current learning episode can be compared to the policy to identify effective and ineffective behavior that can be adapted in succeeding episodes.

3.3.1 Learning policy creation

Self-regulation can be viewed as cyclic phases of forethought, performance and self-reflection [14] wherein reflections about the outcomes of behavior after a learning episode can be used to increase the effectiveness of future learning episodes (e.g., discarding or modifying ineffective behavior). The ideal effect of would be for learning outcomes to continually improve over time.

We fit this incremental perspective of adapting behavior into a reinforcement learning (RL) problem in machine learning which searches for the best actions to take in an environment (i.e., learning behavior) to maximize a cumulative reward (i.e., learning effectiveness) [11].

Profit sharing is a model-free RL approach that is capable of converging even in domains that do not satisfy the Markovian property [1]. We decided to use this approach primarily because we deal with human behavior in a non-deterministic and uncontrolled environment. Profit sharing's reinforcement mechanism allows it to learn effective, yet sometimes non-optimal, policies quickly compared to other algorithms. This is ideal for our situation because we need to give policy-based feedback using minimal data.

Profit sharing differs from other RL techniques because it reinforces effective rules instead of estimating values from succeeding sequential states. A rule consists of a state-action pair (O_t,A_t) which means performing A_t when O_t is observed. We consider these rules as learning behaviors. An episode n is a finite sequence of rules wherein the entire sequence is awarded the reward R based on its outcome. After each episode, the weights of each rule in the sequence is updated using (1) where function $f(R,t)$ is a credit assignment with t being the rule's distance from the goal. Note that it is possible for a rule's weight to be updated more than once if it appears more than once in a sequence. The set of all rules and their corresponding weights is called a policy. A policy is rational or guaranteed to converge to a solution when the credit assignment fuction satisfies the rationality theorem (2) with L being the number of possible actions. In our work, we used a modified version of the rational credit assignment function (3), which was adapted from [1] so that the rules' weights will be bound by the reward value.

$$W_{n+1}(O_t, A_t) \leftarrow W_n(O_t, A_t) + f(R, T) \tag{1}$$

$$\forall t = 1, 2, 3..., T. \quad L\sum_{j=0}^{t} f(R, j) < f(R, t) \tag{2}$$

$$f_{n+1}(R, t) = (R - W_n(O_t, A_t))(0.3)^{T-t} \tag{3}$$

According to Winne's [12] SRL model, students adapt their

strategies based on the results of metacognitive monitoring and evaluation. When the outcome of a task satisfies a student's expectations, then they may continue performing the current task or proceed to the next task. On the other hand, when a task does not achieve its expected outcomes, students can adapt their strategies accordingly. Unfortunately, we did not have access to students' metacognitive evaluations in our data. However, Carver and Scheier's [3] model theorized that the results of metacognitive evaluations can be observed in students' emotion. When the outcome of a task is according to a student's expectation, then neutral affect is experienced. However, when the outcome does not satisfy expectations then negative affect is experienced. On the other hand, when the outcome exceeds expectations then positive affect is experienced. Based on these assumptions, we represented our states using the triple <activity, affect, duration>. Apart from affect which approximated students' metacognitive evaluation, we included activity to indicate the task performed by the student and duration to indicate how long it was performed by the student.

The data showed that students performed similar activities but used different applications (e.g., browsing websites with Google Chrome vs. Mozilla Firefox). Instead of treating these separately, we categorized the students' activities into six types: information search [IS] (e.g., using a search engine), view information source [IV] (e.g., reading a book, viewing a website), write notes [WN], seek help from peers [HS] (e.g., talking to a friend), knowledge application [KA] (e.g., paper writing, presentation creation, data processing) and off-task [OT] (e.g., playing a game).

Durations were even more varied ranging from one second (e.g., clicking a link from a search results page) to 53 minutes (e.g., watching a video). Using this directly will result in a large state space so we categorized them into short, medium or long duration. The duration values were positively skewed so evenly partitioning the data according to the number of elements or frequency would cause both short and medium groups to have small and similar values. The long duration group on the other hand, would have values with high variation. We decided to use k-Means to categorize the duration values into three clusters (i.e., $k = 3$) and using a Euclidean distance formula as described in [6]. Clustering produced groups with elements having similar duration values and whose values were different from the other groups. Elements in the cluster with the smallest values were labeled short duration, elements in the cluster with the biggest values were labeled long duration and the elements in the remaining cluster were labeled with medium duration. The centroids identified by k-means for short, medium and long durations were 69.4 seconds (1.15 minutes), 614.5 (10.2 minutes) seconds and 1999.4 seconds (33.3 minutes) respectively. 90.83% of the duration values were short, 8.17% were medium and 0.10% were long.

In the learning context, actions would refer to changing from one activity to the other. So, we used the same eight activity categories as actions. However, we added a change information source [CS] action to handle cases when students would either view a different website or change to or from a physical information source (e.g., book, printed conference paper).

In this representation, there would be no consecutive rules with states having the same values unless they were paired with different actions. Otherwise, these rules were merged and their durations added. An example of a rule would have the form (<IV, CNF, short>, CS).

The student's rating of the learning episode's effectiveness can directly be used as the reward value. Data from learning episodes can then be converted into rule sequences and be used to update each rule's weight incrementally using (1) with the corresponding reward values. The rules' weights are expected to converge to the reward value it is commonly associated with.

3.3.2 *Learning policy-based feedback*

According to Pressley, Levin and Ghatala [9], adult students who were given information regarding the utility of two learning strategies and a chance to practice them were capable of validating its outcomes and were reported to use the more effective strategy. In our case, the utility of performing an action in a certain state is its weight value (i.e., applying the rule will likely lead to a learning effectiveness rating that is at least the weight value). Information about the utility of two or more competing rules (i.e., rules referring to the same state but with different actions) can be used to give students feedback at the end of a learning episode so they can verify and adapt them accordingly in succeeding episodes. When students used more effective rules, it is assumed to result in better learning effectiveness ratings which will reinforce the rule in the learning policy.

As more rules are observed and added into the learning policy, some rules may not be relevant to a particular learning episode. The rules with their corresponding utilities should first be filtered before they are presented to the student. In the first learning episode, the learning policy will still be empty so feedback will be unavailable. When a policy already contains rules, each rule employed in the current learning episode can be compared to the rules in the learning policy and provide relevant feedback. The pseudo code presented below describes how three types of feedback can be given to the student. First, when students perform an action with a worse utility based on the policy, the system can remind the student to select the better action. Second, if the student performs an action which isn't in the policy but has lower utility than the best action in the policy, the student is told that the action may be ineffective. Lastly, if the student performs an action which isn't in the policy but has a higher utility than the best action, the student is informed that a better action has been found compared to the previous best action. Whenever a student performs the best action according to the policy, feedback is no longer given because it is assumed that the student already knows this and is the reason why the action was selected. In cases when the student performs an action in an unknown state, feedback cannot be given as well because of insufficient information.

Initialize set of weighted rules X
Copy old policy P into P'
For each (O_t, A_t) in the current learning episode
    Update $W(O_t, A_t)$ in P' using (1)

```
    For each (O_p, A_p) in policy P
        If O_t = O_{p,i}
            Add W(O_{p,i}, A_{p,i}) into X
        End
    End
End
For each (O_t, A_t) in the current learning episode
    If (O_t, A_t) not in X
        Unknown utility
    Else if (O_t, A_t) not in P
        If W(O_t, A_t) < max(W(O_{p,i'}, A_{p,i'})) in X
            Inform student that A_{p,i'} > A_t
        Else
            Inform student that A_t > A_{p,i'}
        End
    Else
        If A_t <> A_{p,i'} where max(W(O_{p,i'}, A_{p,i'})) in X
            Inform student that A_{p,i'} > A_t
        End
    End
End
```

A cause for concern is that the learning policy might not have converged yet resulting in incorrect feedback (e.g., telling the student to perform an action which is actually ineffective). Again according to Pressley *et. al.* [9], despite being given incorrect utility information adults are able to select better strategies after practice wherein they are able to observe the strategy's actual utility. As students constantly select effective actions (i.e., as a result of their own evaluation), the policy will be updated to reinforce these actions and decrease the chance of providing incorrect feedback. This emphasizes the need for students in this environment to explore other actions so that they can find the best actions which will also be reflected in the policy. It also then becomes necessary for other mechanisms to encourage exploration such as looking at other students' learning policies for possible actions or using expert knowledge.

4. LEARNING BEHAVIOR DATA

The methodology described in the interaction and annotation phases of the architecture was used in collecting the data in our previous work [7]. The data was collected from four students aged between 17 and 30 years old, conducting research as part of their academic requirements. Three of the students were taking Information Science while one student was taking Physics. During the data collection period, two of the students were writing conference papers and two made power point presentations about their research. They all processed and performed experiments on their collected data, searched for related literature and created a report or document. Although their topics were different, they performed similar types of activities. Two hours of annotated learning behavior data in five separate learning episodes were collected from each student over a one week period. The annotation data was processed using the method described in Section 3.3.1 resulting in five separate learning episodes for every student and each episode consisting of the sequenced rules. On average, students used 54.35 rules per session (N=20; σ=27.71) including repeated rules.

Table 1: Rule Categories

#	Type	State	Action	Reward
1	PRL	ENG, IV, short	KA	0.360000
2	PRL	ENG, IV, short	CS	0.004154
3	CDH	CON, IV, short	CS	0.441939
4	CDH	CON, IV, short	KA	2.34E-05
5	CDH	CON, IV, short	OT	9.16E-15
6	RLX	ENG, KA, long,	OT	1.830000
7	RLX	ENG, KA, long,	HS	0.009720
8	RLX	ENG, KA, long,	IV	2.13E-06
9	RSL	DEL, OT, short	KA	0.389484
10	RSL	DEL, OT, short	IV	2.00E-18
11	RSL	DEL, OT, short	HS	9.57E-26

5. RESULTS AND ANALYSIS

The learning policies generated by the profit sharing algorithm on the learning behavior data consisted of rules based on the state and action representation used. There were many rules due to our selected state-action space, but we observed four categories after analyzing the data– Prolonged learning (PRL), Cognitive disequilibrium handling (CDH), Relaxation (RLX) and Resumed learning (RSL). Table 1 presents examples of each category which were taken from the learning policy of the doctoral physics student who was experimenting with her data and used its results for writing a conference paper.

PRL rules refer to states wherein students feel engaged while performing a learning-related activity and switch to another learning-related activity. It describes how long a certain type of activity could be effective and what other activities may complement it. Taking the physics student's data as an example, let us consider that she was looking into different concepts for data manipulation because she needed it for writing her conference paper. According to rules 1 and 2, it was better for her to try and run an experiment on her data (i.e., apply knowledge), before shifting to a different concept (i.e., view information source). This would allow her to have a better understanding of the concept and allow her to write the paper more easily.

CDH rules refer to states wherein students adapt their behavior to handle negative affect (e.g., confusion or boredom) while learning. These give an idea how long to stay in a confusing or bored learning state before shifting to an activity that will probably alleviate the problem. For example, rule 3 indicates that it is probably better to find a different information source if it is confusing instead of spending a lot time trying to understand it. Rule 5 also indicates that it is not a good idea to just engage in off-task activities when it is difficult to understand a certain information source.

RLX rules refer to states wherein students relax or shift to off-task activities after learning. According to rule 6, it was effective for the student to relax after spending a long time learning. This supports claims that off-task activities or relaxation are important for continued learning [7].

RSL rules refer to cases wherein students shift back to learning from an off-task activity. It seemed that the utility for performing actions in this category are context-dependent.

Table 2: Rule correctness over learning episodes

Ep	+	-	New^+	New^-	Unknown	Reward
2	0	0	1	0	3	4
3	1	0	2	1	1	3
4	12	0	5	0	1	4
5	4	51	0	1	6	2

For example, according to rule 9, it was more effective to apply knowledge probably because the goal was to write a conference paper. Spending too much time reading information sources would help, but not directly lead to the achievement of the goal. This effect is important to consider because if students change their goals, the policy may not be directly applicable to the new goal. A separate experiment needs to be conducted to observe how the architecture will handle such scenarios. We think however that the speed in which the algorithm adjusts the learning policy is a good factor that can make it capable of handling such changes.

After a student completes a learning episode, an updated learning policy can now be used to generate feedback. The feedback will be based on five cases: the student chooses the best action according to the policy (+), the student does not choose the best action according to the policy (-), the student tries a new action which has better results than the best action in the policy (New^+), the student tries a new action which has worse results than the best action in the policy (New^-) and the student performs the only action associated to a state in the policy or the student performs an action in an unknown state for the first time such that the policy will not be able to identify if there is a better action (Unknown).

We simulated how feedback would be generated for these five cases by testing the algorithm on data from the same student. The student's actions in the first learning episode were used to build an initial policy. No feedback was generated at this point because learning policy would only contain rules based on the current episode. Feedback for the second episode could now be generated because it can be compared with the learning policy created using data from the first learning episode. The learning policy was updated using data from the second episode, and was used to generate feedback for the third learning episode. This was repeated for all remaining learning episodes. Table 2 presents the number of times each case is encountered as new learning episodes are experienced by the student.

The table shows that the student implemented a few rules in episode two which was caused by the student spending a long time performing an activity. We see that her learning policy was updated with three new rules as well as a new effective action (i.e., performing an off-task activity after spending a long time experimenting with data). The high reward value indicates that the student did well because all actions, including those unknown actions, were effective. This was confirmed by checking her updated learning policy generated in the fifth episode. The unknown actions were in fact the best actions in their corresponding states (i.e., performing an off-task activity after spending some time experimenting with data, resume data experimentation after a short off-task activity and consulting a friend about the experiment after a short off-task activity). The student also performed few actions in the third episode but gave it a smaller reward value probably because she spent too much time talking to a friend even though the other actions were effective (i.e., resuming data experimentation after a short off-task activity and viewing a paper after some time experimenting with data). In the fourth episode, the student constantly performed effective actions and even discovered a new action which probably caused the increase in reward. Finally in the fifth episode, the student performed a lot of ineffective actions which probably caused the big decrease in the reward value. Specifically, as we have discussed earlier, she spent short amounts of time repeatedly viewing different information sources. The policy indicated that it would have been better for her to apply knowledge, which in her context would mean either writing the paper or experimenting with her data. This could in fact be an effective strategy because she could verify and learn more about the concept by applying it rather than moving on to another concept right away.

Our results also showed that there was a relationship between the number of times students correctly followed rules in their learning policy and their learning effectiveness rating. Figure 3 presents graphs corresponding to each student showing this relationship. The learning effectiveness ratings were expressed as ratios relative to the highest rating (i.e., five) and the number of correct actions were expressed as ratios relative to the total number of actions in the learning episode. The trend indicates that the learning policy was able to identify effective actions from the students' behavior such that when the students selected more effective actions (i.e., based on the learning policy), they also had a more effective learning episode. This means that if the student will be able to follow the feedback provided by the system in succeeding learning episodes, it is likely for them to have more effective learning experiences.

6. CONCLUSION AND FUTURE WORK

We have presented an architecture for collecting students' learning behavior data, uncovering effective learning behaviors and using them to help students manage their learning. The approach does not require a specific learning environment so the student's behavior is naturalistic and captures how he/she actually learns. However, it does require students to annotate their data. Annotation is done after learning so it does not require additional cognitive load during the learning episode. Desktop and web cam screenshots can help students recall the context in which they learned and can likely improve annotation accuracy.

The profit sharing algorithm was used for building learning policies that contained rules describing an action's effectiveness in a particular state. Learning policies generated from previous learning episodes can be compared with data from the current learning episode to identify which actions were effective or ineffective and generate feedback accordingly. Feedback about possible improvements can be useful for students to adapt their actions in future learning episodes.

Simulations from actual data showed that updating the learning policy also changed the resulting feedback such that

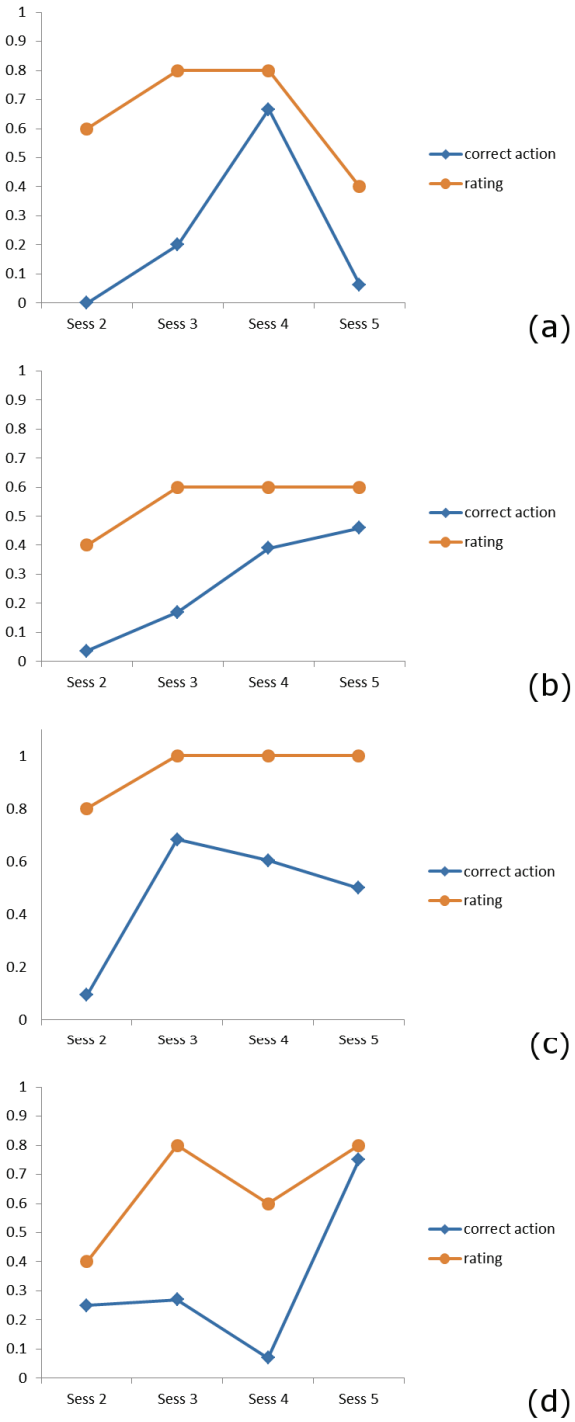

Figure 3: Relationship between action correctness and student rating

newer, more effective actions were presented to the student. This helps ensure that the student will always be prompted to select the most effective learning behavior. The relationship between the number of effective rules followed by the student and their learning effectiveness ratings indicate that the learning policy-based feedback will have a good chance of helping students learn more effectively.

The architecture we have designed still has some issues that need to be addressed. Our state representation did not contain information regarding students' metacognitive evaluations. Although we used emotions to approximate these evaluations, asking students to annotate them will be more accurate and create better policies. The reward values we used were based on students' self-evaluations and it would be interesting to see the difference when using learning gains instead (e.g., asking students to take a pretest and posttest). Combining both learning gains and self-evaluation to create the reward value may be a better measurement because it will consider both the student's preferred learning behavior and knowledge gained.

Our architecture also faces a common problem in RL called the exploration-exploitation problem. In order for the policy to be optimal, students need to try as much actions as possible. Due to the approach's reliance on the student's learning behavior, it cannot suggest actions outside of the current learning policy. This would require mechanisms for suggesting actions not in the learning policy such as using other students' learning policies or using expert knowledge.

Even though the approach can create policies that span across learning episodes, it has only been tested with learning episodes having the same goal. In the case of our data, students were either writing a conference paper or creating a power point presentation. It will be more useful if it could also be used across different learning goals. The current approach needs to be tested to see how well it fares in such a case and necessary modifications need to be applied accordingly.

The data we used was collected from adult learners and may be effective for them. However, according to Pressley *et. al.* [9], children have difficulty in verifying learning strategy utility even after practice. It is possible that additional feedback may be needed to fit this approach to younger learners.

Acknowledgements

This work was supported in part by the Management Expenses Grants for National Universities Corporations from the Ministry of Education, Culture, Sports, Science and Technology of Japan (MEXT) and JSPS KAKENHI Grant Number 23300059. We would also like to thank all the students who participated in our data collection.

7. REFERENCES

[1] S. Arai and K. Sycara. Effective learning approach for planning and scheduling in Multi-Agent domain. In *6th International Conference on Simulation of Adaptive Behavior*, pages 507–516, 2000.

[2] R. Azevedo, R. S. Landis, R. Feyzi-Behnagh, M. Duffy, G. Trevors, J. M. Harley, F. Bouchet, J. Burlison, M. Taub, N. Pacampara, M. Yeasin, A. K.

M. M. Rahman, M. I. Tanveer, and G. Hossain. The effectiveness of pedagogical agents' prompting and feedback in facilitating co-adapted learning with MetaTutor. In *Intelligent Tutoring Systems*, pages 212–221, 2012.

[3] C. S. Carver and M. F. Scheier. Origins and functions of positive and negative affect: A control-process view. *Psychological Review*, 97(1):19–35, 1990.

[4] S. D. Craig, A. C. Graesser, J. Sullins, and B. Gholson. Affect and learning: An exploratory look into the role of affect in learning with AutoTutor. *Journal of Educational Media*, 29(3):241–250, 2004.

[5] P. Ekman. Are there basic emotions? *Psychological Review*, 99(3):550–553, 1992.

[6] G. Gan, C. Ma, and J. Wu. *Data clustering: theory, algorithms, and applications*, volume 20. Society for Industrial and Applied Mathematics, 2007.

[7] P. S. Inventado, R. Legaspi, R. Cabredo, and M. Numao. Student learning behavior in an unsupervised learning environment. In *20th International Conference on Computers in Education*, pages 730–737, 2012.

[8] J. S. Kinnebrew, K. M. Loretz, and G. Biswas. A contextualized, differential sequence mining method to derive students' learning behavior patterns. *Journal of Educational Data Mining*, in press.

[9] M. Pressley, J. R. Levin, and E. S. Ghatala. Memory strategy monitoring in adults and children. *Journal of Verbal Learning and Verbal Behavior*, 23(2):270–288, 1984.

[10] J. Sabourin, L. R. Shores, B. W. Mott, and J. C. Lester. Predicting student self-regulation strategies in game-based learning environments. In *Intelligent Tutoring Systems*, pages 141–150, 2012.

[11] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. A Bradford Book, 1998.

[12] P. H. Winne. Self-regulated learning viewed from models of information processing. *Self-regulated learning and academic achievement: Theoretical perspectives*, 2:153–189, 2001.

[13] B. J. Zimmerman. Self-regulated learning and academic achievement: An overview. *Educational psychologist*, 25(1):3–17, 1990.

[14] B. J. Zimmerman. Becoming a Self-Regulated learner: An overview. *Theory Into Practice*, 41(2):64–70, 2002.