Sequential Patterns of Affective States of Novice Programmers

Nigel Bosch[1] and Sidney D'Mello[1,2]

Departments of Computer Science[1] and Psychology[2], University of Notre Dame
Notre Dame, IN 46556, USA
{pbosch1, sdmello}@nd.edu

Abstract. We explore the sequences of affective states that students experience during their first encounter with computer programming. We conducted a study where 29 students with no prior programming experience completed various programming exercises by entering, testing, and running code. Affect was measured using a retrospective affect judgment protocol in which participants annotated videos of their interaction immediately after the programming session. We examined sequences of affective states and found that the sequences Flow/Engagement ↔ Confusion and Confusion ↔ Frustration occurred more than expected by chance, which aligns with a theoretical model of affect during complex learning. The likelihoods of some of these frequent transitions varied with the availability of instructional scaffolds and correlated with performance outcomes in both expected but also surprising ways. We discuss the implications and potential applications of our findings for affect-sensitive computer programming education systems.

Keywords: affect, computer programming, computerized learning, sequences

1 Introduction

Given the unusually high attrition rate of computer science (CS) majors in the U.S. [1], efforts have been made to increase the supply of competent computer programmers through computerized education, rather than relying on traditional classroom education. Some research in this area focuses on the behaviors of computer programming students in order to provide more effective computerized tutoring and personalized feedback [2]. In fact, over 25 years ago researchers were exploring the possibility of exploiting artificial intelligence techniques to provide customized tutoring experiences for students in the LISP language [3]. This trend has continued, as evidenced by a number of intelligent tutoring systems (ITSs) that offer adaptive support in the domain of computer programming (e.g. [4–6]).

One somewhat neglected area in the field is the systematic monitoring of the affective states that arise over the course of learning computer programming and the impact of these states on retention and learning outcomes. The focus on affect is motivated by considerable research which has indicated that affect continually operates throughout a learning episode and different affective states differentially impact per-

© Springer-Verlag Berlin Heidelberg 2011

formance outcomes [7]. Some initial work has found that affective states, such as confusion and frustration, occur frequently during computer programming sessions [8, 9] and these states are correlated with student performance [10].

The realization of the important role of affect in learning has led some researchers to develop learning environments that adaptively respond to affective states in addition to cognitive states (see [11] for a review). Previous research has shown that affect sensitivity can make a measurable improvement on the performance of students in other domains such as computer literacy and conceptual physics [12, 13]. Applying this approach to computer programming education by identifying the affective states of students could yield similarly effective results, leading to more effective systems.

Before it will be possible for an affect-sensitive intelligent tutoring system to be successful in the computer programming domain, more research is needed to determine at a fine-grained level what affective states students experience and how affect interacts and arises from the students' behaviors. Previous work has collected affective data at a somewhat coarse-grained level in a variety of computer programming education contexts. [10] collected affect using two human observers, and were able to draw conclusions about what affective states led to improved performance on a computer programming exam. [14] induced affect in experienced programmers using video stimuli, and found that speed and performance on a coding and debugging test could be increased with high-arousal video clips.

In our previous work [15], we examined the affect of 29 novice programmers at 20-second intervals as they solved introductory exercises on fundamentals of computer programing. We found that flow/engagement, confusion, frustration, and boredom dominated the affect of novice programmers when they were not in a neutral state. We found that boredom and confusion were negatively correlated with performance, while the flow/engagement state positively predicted performance. This paper continues this line of research by exploring transitions between affective states.

Specifically, we test a theoretical model on affect dynamics that has been proposed for a range of complex learning tasks [16]. This theoretical model (Fig. 1) posits four affective states that are crucial to the learning process: flow/engagement, confusion, frustration, and boredom. The model predicts an important interplay between confusion and flow/engagement, whereby a learner in the state of flow/engagement may encounter an impasse and become confused. From the state of confusion, if an impasse is resolved the learner will return to the state of flow/engagement, having learned more deeply. This is in line with other research which has shown that confusion helps learning when impasses are resolved [17]. On the other hand, when the source of the confusion is never resolved, the learner will become frustrated, and eventually bored if the frustration persists.

Researchers have found some support for this theoretical model of affective dynamics in learning contexts such as learning computer literacy with AutoTutor [16], unsupervised academic research [18], and narrative learning environments [19]. We expect the theoretical model to apply to computer programming as well.

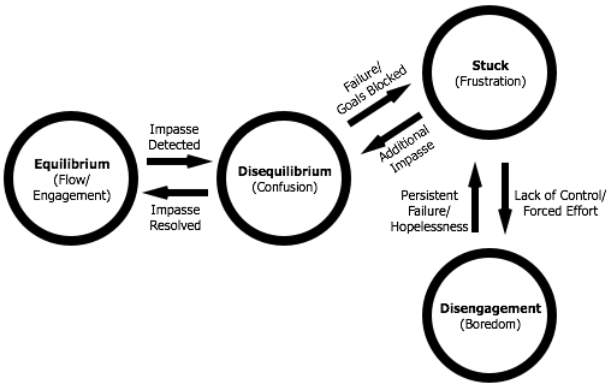

Fig. 1. Theoretical model of affect transitions.

We posit that encountering unfamiliar concepts, syntax and runtime errors, and other impasses can cause confusion in a computer programmer. When those impasses are resolved, the programmer will be better equipped to anticipate and handle such impasses in the future, having learned something. Alternatively, if the impasses persist, programmers may become frustrated and eventually disengage, entering a state of boredom in which it is difficult to learn.

To explore the applicability of this model to the domain of novice computer programming, this paper focuses on answering the following research questions: 1) what transitions occur frequently between affective states? 2) how are instructional scaffolds related to affect transitions? and 3) are affective transitions predictive of learning outcomes? These questions were investigated by analyzing affect data collected in a previous study [15] where 29 novice programmers learned the basics of computer programming over the course of a 40-minute learning session with a computerized environment, as described in more detail below.

2 Methods

Participants were 29 undergraduate students with no prior programming experience. They were asked to complete exercises in a computerized learning environment designed to teach programming fundamentals in the Python language. Participants solved exercises by entering, testing, and submitting code through a graphical user interface. Submissions were judged automatically by testing predetermined input and output values, whereupon participants received minimal feedback about the correctness of their submission. For correct submissions they would move on to the next exercise, but otherwise would be required to continue working on the same exercise.

The exercises in this study were designed in such a way that participants would likely encounter some unknown, potentially confusing concepts in each exercise. In this manner we elicited emotional reactions similar to real-world situations where computer programmers face problems with no predefined solutions and must experiment and explore to find correct solutions. Participants could use hints, which would gradually explain these impasses and allow participants to move on in order to pre-

vent becoming permanently stuck on an exercise. However, participants were free to use or ignore hints as they pleased.

Exercises were divided into two main phases. In the first phase (scaffolding), participants had hints and other explanations available and worked on gradually more difficult exercises for 25 minutes. Performance in the scaffolding phase was determined by granting one point for each exercise solved and one point for each hint that was not used in the solved exercises. Following that was the second phase (fadeout), in which they had 5 minutes to work on a debugging exercise, and 10 minutes to work on another programming exercise with no hints. In this study we will not consider the debugging exercise because it was only 5 minutes long. Performance was determined by two human judges who examined each participant's code, determined the number of lines matching lines in the correct solution, and resolved their discrepancies.

Finally, we used a retrospective affect judgment protocol to assess student affect after they completed the 40-minute programming session [20]. Participants viewed video of their face and on-screen activity side by side, and were polled at various points to report the affective state they had felt most at the polling point. The temporal locations for polling were chosen to correspond with interactions and periods of no activity such that each participant had 100 points at which to rate their affect, with a minimum of 20 seconds between each point. Participants provided judgments on 13 emotions, including basic emotions (anger, disgust, fear, sadness, surprise, happiness), learning-centered emotions (anxiety, boredom, frustration, flow/engagement, curiosity, confusion/uncertainty) and neutral (no apparent feeling).The most frequent affective states, reported in [15], were flow/engagement (23%), confusion (22%), frustration (14%), and boredom (12%), a finding that offers some initial support for the theoretical model discussed in the Introduction.

3 Results and Discussion

We used a previously developed transition likelihood metric to compute the likelihood of the occurrence of each transition relative to chance [21].

$$L(Current \rightarrow Next) = \frac{\Pr(Next|Current) - \Pr(Next)}{1 - \Pr(Next)} \quad (1)$$

This likelihood metric determines the conditional probability of a particular affective state (*next*), given the current affective state. The probability is then normalized to account for the overall likelihood of the *next* state occurring. If the affective transition occurs as expected by chance, the numerator is 0 and so likelihood is as well. Thus we can discover affective state transitions that occurred more ($L > 0$) or less ($L < 0$) frequently than expected by chance alone.

Before computing L scores we removed transitions that occurred from one state to the same state. For example, a sequence of affective states such as *confusion, frustration, frustration, boredom* would be reduced to *confusion, frustration, boredom*. This was done because our focus in this paper is on the transitions between different affective states, rather than on the persistence of each affective state [16, 18]. Furthermore,

although transition likelihoods between all 13 states (plus neutral) were computed, the present paper focuses on transitions between states specified in the theoretical model (boredom, confusion, flow/engagement, and frustration), which also happen to be the most frequent affective states.

What transitions occur frequently between affective states? We found the transitions that occurred significantly more than chance ($L = 0$) by computing affect transition likelihoods for individual participants and then comparing each likelihood to zero (chance) with a two-tailed one-sample *t-test.* Significant ($p < .05$) and marginally significant ($p < .10$) transitions are shown in Figure 2 and are aligned with the theoretical model on affect dynamics.

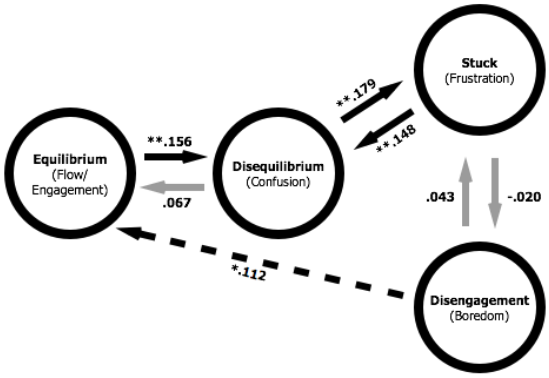

Fig 2. Frequently observed affective state transitions. Edge labels are mean likelihoods of affective state transitions. Grey arrows represent transitions that were predicted by the theoretical model but were not significant. The dashed arrow represents a transition that was marginally significant but not predicted. *$p < .10$, **$p < .05$

Three of the predicted transitions, Flow/Engagement → Confusion, Confusion → Frustration, and Frustration → Confusion, were significant and matched the theoretical model. Confusion → Flow/Engagement was in the expected direction and approached significance ($p = .108$), while Boredom → Frustration was in the expected direction but not significant. The Frustration → Boredom transition was not in the expected direction and was also not significant. Hence, with the exception of the Frustration ↔ Boredom links, there was support for four out of the six transitions espoused by the theoretical model. This suggests that the components of the model related to the experience of successful (Flow/Engagement ↔ Confusion) and unsuccessful (Confusion ↔ Frustration links) resolution of impasses were confirmed. Therefore, the present data provide partial support for the model.

The Boredom → Flow/Engagement transition, which occurred at marginally significant levels ($p = .091$), was not predicted by the theoretical model. It is possible that the nature of our computerized learning environment encouraged this transition more than expected. This might be due to the fast-paced nature of the learning session, which included 18 exercises and an in-depth programming task in a short 40-minute session. Furthermore, participants had some control over the learning environment in that they could use bottom-out hints to move to the next exercise instead of being forced to wallow in their boredom. The previous study that tested this model used a learning environment (AutoTutor) that did not provide any control over the learning activity, which might explain the presence of Frustration → Boredom (dis-

engaging from being stuck) and Boredom → Frustration (being frustrated due to forced effort) links in the earlier data [16].

How are instructional scaffolds related to affect transitions? To answer this question we looked at the differences between the scaffolding and fadeout phases of the study, as previously described. We discarded the first 5 minutes of the scaffolding phase to allow for a "warm-up" period during which participants were acclimating to the learning environment. We also discarded the 25 to 30 minutes portion, which was the debugging task in the fadeout phase. The debugging task was significantly different from the problem-solving nature of the coding portions, and so we excluded it from the current analysis to increase homogeneity. Differences between likelihoods of the five significant or marginally significant transitions from Figure 2 were investigated with paired samples *t*-test (see Table 1).

Table 1. Means and standard deviations (in parentheses) for common transitions in the scaffolding phase (5-25 minutes) and the coding portion of the fadeout phase (30-40 minutes).

Transition	Scaffolding	Fadeout Coding	N
Flow/Engagement → Confusion	**.115 (.308)	**.354 (.432)	20
Confusion → Flow/Engagement	.101(.241)	.029(.331)	27
Confusion → Frustration	.105 (.276)	.184 (.416)	27
Frustration → Confusion	.047 (.258)	.116 (.445)	21
Boredom → Flow/Engagement	.096 (.166)	.226 (.356)	14

*$p < .10$, **$p < .05$

The likelihood of participants transitioning from flow/engagement to confusion was significantly higher in the fadeout phase compared to the scaffolding phase. This may be attributed to the fact that participants have hints and explanations in the scaffolding phase, so in the event of a confusing impasse, a hint may be helpful in resolving the impasse, thereby allowing participants to return to a state of flow/engagement. With no such hints, confused participants may become more frustrated in the fadeout phase, as evidenced by a trend in this direction. This finding is as expected from the theoretical model, which states that confusion can lead to frustration when goals are blocked and the student has limited coping potential (e.g. being unable to progress on an exercise in this case).

Although not significant, there also appears to be an increase in the Boredom → Flow/Engagement affect transition in the fadeout phase. It is possible that too much readily available assistance prevents students from re-engaging on their own.

Are affective transitions predictive of learning outcomes? To determine what affective state transitions were linked to performance on the programming task, we correlated the likelihood of affect transitions with the performance metrics described in the Methods. In previous work we found correlations between performance and the proportions of affective states experienced by students [15]. Hence, when examining the correlations between affect transitions and performance, partial correlations were used to control for the proportions of the affective states in the transitions.

Table 2 lists correlations between frequent transitions and performance. These include correlations between affect transitions in the scaffolding phase with performance in the scaffolding phase (Scaffolding column) and transitions in the fadeout phase with performance in the fadeout coding phase (Fadeout Coding 1). We also correlated transitions in the scaffolding phase with performance in the fadeout coding phase (Fadeout Coding 2). This allows us to examine if affect transitions experienced during scaffolded learning were related to future performance when learning scaffolds were removed. Due to the small sample size, in addition to discussing significant correlations, we also consider non-significant correlations approaching 0.2 or larger to be meaningful because these might be significant with a bigger sample. These correlations are bolded in the table.

Table 2. Correlations between affect transitions and performance.

Transition	**Scaffolding**	**Fadeout Coding 1**	**Fadeout Coding 2**
Flow/Engagement → Confusion	.046	-.094	-.098
Confusion → Flow/Engagement	**-.274**	**-.256**	***-.365**
Confusion → Frustration	.114	****.499**	****.424**
Frustration → Confusion	***-.368**	.051	**-.275**
Boredom → Flow/Engagement	-.034	.050	-.063

*$p < .10$, **$p < .05$

The correlations were illuminating in a number of respects. The Confusion → Flow/Engagement transition correlated *negatively* with performance. This is contrary to the theoretical model which would predict a positive correlation to the extent that confused learners return to a state of flow/engagement by resolving troublesome impasses with effortful problem solving. It is possible that students who frequently experienced this transition were doing so by taking advantages of hints as opposed to resolving impasses on their own. This would explain the negative correlation between Confusion → Flow/Engagement and performance.

To investigate this possibility we correlated hint usage in the scaffolding phase with the Confusion → Flow/Engagement transition, controlling for the proportion of confusion and flow/engagement. The number of hints used in the scaffolding phase correlated positively, though not significantly, with the Confusion → Flow/Engagement transition in the scaffolding phase (r = .297) and the fadeout coding phase (r = .282). Additionally, hint usage correlated negatively with score in the scaffolding phase (r = -.202) and the fadeout coding phase (r = -.506). This indicates that students using hints tended to experience the Confusion → Flow/Engagement transition more (as expected) but this hindered rather than helped learning because students were not investing the cognitive effort to resolve impasses on their own.

Similarly, the correlation between Confusion → Frustration and performance is inconsistent with the theoretical model, which would predict a negative relationship between these variables. This unexpected correlation could also be explained on the basis of hint usage. Specifically, the number of hints used in the scaffolding phase

correlated negatively, though not significantly, with the Confusion → Frustration transition in the scaffolding phase (r = -.258) and the fadeout coding phase (r = -.171). This finding suggests that although hints can alleviate the Confusion → Frustration transition, learning improved when students are able to resolve impasses on their own, which is consistent with the theoretical model.

Finally, the correlation between Frustration → Confusion was in the expected direction. The Frustration → Confusion transition occurs when a student experience additional impasses while in the state of frustration. This transition is reflective of hopeless confusion, which is expected to be negatively correlated with performance, as revealed in the data.

4 General Discussion

Previous research has shown that some affective states are conducive to learning in the context of computer programming education while others hinder learning. Flow/engagement is correlated with higher performance, while confusion and boredom are correlated with poorer performance [10, 15]. Transitions between affective states are thus important because they provide insight into how students enter into an affective state. Affect-sensitive ITSs for computer programming may be able to use this information to better predict affect, intervening when appropriate to encourage the flow/engagement state and minimize the incidence of boredom and frustration.

We found that the presence or absence of instructional scaffolds were related the affect transitions experienced by students, especially the Flow/Engagement → Confusion transition. Our findings show that this transition is related to the presence of hints, a strategy which might be useful in future affect-sensitive ITS design for computer programming students. Similarly, we found that instructional scaffolds were related to the Boredom → Flow/Engagement transition, which is not part of the theoretical model. Future work on ITS design might also need to take into account this effect and moderate the availability of scaffolds to promote this affect transition.

The affect transitions that we found partially follow the predictions of the theoretical model. Impasses commonly arise in computer programming, particularly for novices, when they encounter learning situations with which they are unfamiliar. New programming language keywords, concepts, and error messages present students with impasses that must be resolved before the student will be able to continue. Unresolved impasses can lead to frustration and eventually boredom. The alignment between the theoretical model and the present data demonstrates the model's applicability and predictive power in the context of learning computer programming.

That being said, not all of the affect transitions we found matched predictions of the theoretical model. This includes lack of data to support the predicted Frustration → Boredom and Boredom → Frustration transitions and the presence of an unexpected Boredom → Flow/Engagement transition. Limitations with this study are likely responsible for some of these mismatches. The sample size was small, so it is possible that increased participation in the study might confirm some of these expected transitions. In particular, the Boredom → Frustration transition was in the predicted

direction but not significant in our current sample. Additionally, we exclusively focused on affect, but ignored the intermediate events that trigger particular affective states (e.g., system feedback, hint requests, etc.). We plan to further explore our data by incorporating these interaction events as possible triggers for the observed transitions between affective states. This will allow us to more deeply understand why some of the predicted transitions did not occur (e.g., Frustration → Boredom) and some unexpected transitions did (e.g., Boredom → Flow/Engagement).

It is also possible that some aspects of the model might need refinement. In particular there appears to be an important relationship between Confusion → Frustration transitions, Confusion → Flow/Engagement transitions, performance, and hint usage. While hints may allow students to move past impasses and re-enter a state of flow/engagement, they may lead to an illusion of impasse resolution, which is not useful for learning. Conversely, resolving impasses without relying on external hints might lead a confused learner to momentarily experience frustration, but ultimately improve learning. Future work that increases sample size and specificity of the data (i.e., simultaneously modeling dynamics of affect and interaction events) will allow us to further explore the interaction of hints with the theoretical model, and is expected to yield a deeper understanding of affect dynamics during complex learning.

Acknowledgment. This research was supported by the National Science Foundation (NSF) (ITR 0325428, HCC 0834847, DRL 1235958). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

References

1. Haungs, M., Clark, C., Clements, J., Janzen, D.: Improving first-year success and retention through interest-based CS0 courses. Proceedings of the 43rd ACM technical symposium on Computer Science Education. pp. 589–594. ACM, New York, NY, USA (2012).
2. Fossati, D., Di Eugenio, B., Brown, C.W., Ohlsson, S., Cosejo, D.G., Chen, L.: Supporting Computer Science Curriculum: Exploring and Learning Linked Lists with iList. IEEE Transactions on Learning Technologies. 2, 107–120 (2009).
3. Anderson, J.R., Skwarecki, E.: The automated tutoring of introductory computer programming. Communications of the ACM. 29, 842–849 (1986).
4. Brusilovsky, P., Sosnovsky, S., Yudelson, M.V., Lee, D.H., Zadorozhny, V., Zhou, X.: Learning SQL programming with interactive tools: From integration to personalization. ACM Transactions on Computing Education. 9, 19:1–19:15 (2010).
5. Cheung, R., Wan, C., Cheng, C.: An ontology-based framework for personalized adaptive learning. Advances in Web-Based Learning–ICWL 2010. pp. 52–61. Springer, Berlin Heidelberg (2010).
6. Hsiao, I.-H., Sosnovsky, S., Brusilovsky, P.: Guiding students to the right questions: adaptive navigation support in an E-Learning system for Java programming. Journal of Computer Assisted Learning. 26, 270–283 (2010).
7. Pekrun, R.: The impact of emotions on learning and achievement: Towards a theory of cognitive/motivational mediators. Applied Psychology. 41, 359–376 (1992).

8. Grafsgaard, J.F., Fulton, R.M., Boyer, K.E., Wiebe, E.N., Lester, J.C.: Multimodal analysis of the implicit affective channel in computer-mediated textual communication. Proceedings of the 14th ACM international conference on Multimodal interaction. pp. 145–152. ACM, New York, NY, USA (2012).
9. Lee, D.M.C., Rodrigo, M.M.T., Baker, R.S.J. d, Sugay, J.O., Coronel, A.: Exploring the relationship between novice programmer confusion and achievement. In: D'Mello, S., Graesser, A., Schuller, B., and Martin, J.C. (eds.) Affective Computing and Intelligent Interaction. pp. 175–184. Springer, Berlin Heidelberg (2011).
10. Rodrigo, M.M.T., Baker, R.S.J. d, Jadud, M.C., Amarra, A.C.M., Dy, T., Espejo-Lahoz, M.B.V., Lim, S.A.L., Pascua, S.A.M.S., Sugay, J.O., Tabanao, E.S.: Affective and behavioral predictors of novice programmer achievement. SIGCSE Bulletin. 41, 156–160 (2009).
11. D'Mello, S., Graesser, A.: Feeling, thinking, and computing with affect-aware learning technologies. In: Calvo, R.A., D'Mello, S., Gratch, J., and Kappas, A. (eds.) Handbook of Affective Computing. Oxford University Press (in press).
12. D'Mello, S., Lehman, B., Sullins, J., Daigle, R., Combs, R., Vogt, K., Perkins, L., Graesser, A.: A time for emoting: When affect-sensitivity is and isn't effective at promoting deep learning. In: Aleven, V., Kay, J., and Mostow, J. (eds.) Intelligent Tutoring Systems. pp. 245–254. Springer, Berlin Heidelberg (2010).
13. Forbes-Riley, K., Litman, D.: Benefits and challenges of real-time uncertainty detection and adaptation in a spoken dialogue computer tutor. Speech Communication. 53, 1115–1136 (2011).
14. Khan, I.A., Hierons, R.M., Brinkman, W.P.: Mood independent programming. Proceedings of the 14th European Conference on Cognitive Ergonomics: Invent! Explore! pp. 28–31. ACM, New York, NY, USA (2007).
15. Bosch, N., D'Mello, S., Mills, C.: What Emotions Do Novices Experience During their First Computer Programming Learning Session? Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED 2013) (in press).
16. D'Mello, S., Graesser, A.: Dynamics of affective states during complex learning. Learning and Instruction. 22, 145–157 (2012).
17. D'Mello, S., Lehman, B., Pekrun, R., Graesser, A.: Confusion can be beneficial for learning. Learning and Instruction. (in press).
18. Inventado, P.S., Legaspi, R., Cabredo, R., Numao, M.: Student learning behavior in an unsupervised learning environment. Proceedings of the 20th International Conference on Computers in Education. pp. 730–737. National Institute of Education, Singapore (2012).
19. McQuiggan, S.W., Robison, J.L., Lester, J.C.: Affective transitions in narrative-centered learning environments. In: Woolf, B.P., Aïmeur, E., Nkambou, R., and Lajoie, S. (eds.) Intelligent Tutoring Systems. pp. 490–499. Springer, Berlin Heidelberg (2008).
20. Rosenberg, E.L., Ekman, P.: Coherence between expressive and experiential systems in emotion. Cognition & Emotion. 8, 201–229 (1994).
21. D'Mello, S., Taylor, R.S., Graesser, A.: Monitoring affective trajectories during complex learning. Proceedings of the 29th annual meeting of the cognitive science society. pp. 203–208. Cognitive Science Society, Austin, TX (2007).