Programming with Your Heart on Your Sleeve: Analyzing the Affective States of Computer Programming Students

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

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

Abstract. Students learning computer programming must learn difficult concepts via complex problem-solving activities which elicit strong emotional responses. In this research we explore the affective states that occur while learning computer programming, the events that precede them, and the outcomes that are influenced by them. The data collected in current and future research will be used to create an affect-sensitive intelligent tutoring system which will be better able to maximize learning gains in novice computer programmers and improve their perception of computer science via intelligent handling of emotion.

Keywords: computer programming, affect, emotions, ITSs

1 Introduction

Between the 2000-2001 and 2009-2010 school years, there was a net increase in the number of CS degrees granted, but the percentage of CS degrees compared to all degrees dropped from 9.4% to 7.8% [1]. This drop is surprising given the increasing demand for computer programmers and the growing influence of computer technology in everyday life. Efforts have been made to increase the retention rate of CS students, including tailoring coursework to the special interests of students, providing engaging assignments to improve learning gains, and giving extra opportunities for students to practice programming skills [2]. Researchers have also experimented with using statistical approaches, based on homework submission patterns and similar factors, to identify struggling students who may need special attention to succeed [3].

The low number of students graduating with CS degrees might partially be attributed to the fact that computer programming, which is an essential component of a CS degree, is a difficult skill to acquire because it requires advanced critical thinking, abstract reasoning, and analytical skills. Computer programming is challenging, and often disheartens students due to the impasses that arise. One of the often overlooked factors that contribute to the challenge of programming is the emotional toll that it can inflict on students [4]. Previous work has found that affective states are an important part of conceptual learning and complex problem solving [5] and that computer programming elicits affective states that can be important predictors of performance [6].

adfa, p. 1, 2011.

Improved computer programming education that takes into account affective states and how they influence the learning experience for students could improve the enrollment, retention, and degree production of university computer science and engineering programs. Creating an affect-sensitive ITS to teach novices the basics of computer programming could potentially create a more positive and effective learning experience. This is the major goal of the proposed research project.

2 Previous Research

Over two decades ago, researchers were already exploring the factors that contribute to becoming a good computer programmer and good programming education [7]. Although a number of computerized learning environments to teach programming have emerged, they do not react intelligently in real-time to changes in student affect and do not adjust the material or instruction accordingly.

Recent research has indicated that frustration can be effectively predicted from the code compilation behavior of programming students [8], while a wide variety of sensors and techniques have been used to detect emotion in other contexts [9]. These affect detection techniques can be used to dynamically adjust feedback and instruction based on sensed affective states of computer programming students.

We have done some preliminary work to investigate affective states experienced by novice students while learning to program in the Python language [10]. A computerized learning environment delivered instructional material to 29 students with no prior programming exposure. The system provided them with a series of exercises designed to teach them the fundamentals of computer programming and to test what they had learned. We used a retrospective affect judgment protocol, in which students viewed videos of their face and on-screen interaction and provided affective judgments at approximately 100 points throughout a learning session. Flow/engagement, confusion/uncertainty, frustration, and boredom were the dominant affective states of students when they were not in the neutral state. These four states accounted for 71%, neutral 15%, while other affective states (curiosity, happiness, anxiety, surprise, anger, disgust, fear, and sadness) accounted for a mere 14% of the affect reports. We found that confusion/uncertainty and boredom had a negative impact on performance while flow/engagement had a positive effect. This suggests that confusion/uncertainty and boredom are negative states that an affect-sensitive ITS can focus on regulating.

We also explored some of the effects that instructional materials have on learning. Hints were available to students during problem-solving exercises, to help resolve mental impasses that they were likely to encounter. When using hints, participants did not experience significantly different levels of flow/engagement, but they did experience less boredom, frustration, and confusion/uncertainty.

3 Future Research Plans

To advance this research, we intend to design and implement an ITS that is capable of adapting intelligently to the affective state of novice students as they learn computer programming. This project will consist of three major parts: data collection and analysis, affect detection, and affect responding.

3.1 Data Collection and Analysis

The previous research we have conducted to monitor the affective states of novice programmers will be augmented with additional data collection with a larger sample of students to better explore the links between affective states and programmer actions. Additionally, we will look at how different types of system actions (e.g. feedback, hints, etc.) correlate with affect and performance. We will also explore the time spent on various components of the learning task (reading, typing, testing, off-task behavior, etc.) via refinements to our computerized learning environment to discern what relationships those components have on affective states and performance. The data collected will be analyzed similarly to our previous research and also with Hidden Markov Models (HMMs) to uncover latent (hidden) factors that give rise to time series consisting of student actions, system responses, and student affect.

3.2 Affect Detection

We will use machine learning techniques to build affect detectors that diagnose what affective states will arise based on contextual factors (e.g., problem difficulty), student actions (e.g., code executions, edits), and system responses (e.g., syntax errors, negative feedback). In addition, videos of the faces of students will be analyzed with facial feature detection computer vision algorithms. These facial features will be added as features to enhance the power of the affect detectors. Graphical models that explore the temporal dependencies among features and labels, such as conditional random fields, will also be investigated over more standard supervised learning techniques.

3.3 Affect Responding

We will create an ITS specifically tailored to the emotional needs of novice computer programming students by first integrating the aforementioned affect detectors in the computerized environment. The ITS will then be able to use this information to adjust the instructional material and feedback for students to maintain an emotional state better suited to learning, and to improve students' perceptions of the introductory programming experience. The affect-sensitive ITS will intervene to steer students back toward more productive affective states using interventions at crucial moments when a student is struggling with material but has not yet given up or disengaged. For example, after a student runs their code and encounters an error, it might be useful to provide some encouraging feedback and thus ameliorate affect. Finally, we will compare performance of students with and without affect-sensitive enhancements in the learning environment to determine the efficacy of affect sensitivity in this domain.

4 Conclusion

Computer science can be furthered in many ways through better education. ITS-based learning helps to alleviate problems of availability and cost, and continues to become more technologically advanced as the scientific method is applied to improving techniques. By developing an affect-sensitive computer programming ITS, we hope to improve the learning gains of novice computer programming students and better prepare them for programming education opportunities in the future while simultaneously deepening our understanding of the role of affect in 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 the NSF.

1. Snyder, T.D., Dillow, S.A.: Digest of Education Statistics 2011, http://nces.ed.gov/pubs2012/2012001_0.pdf, (2012).
2. DeClue, T., Kimball, J., Lu, B., Cain, J.: Five focused strategies for increasing retention in Computer Science 1. Journal of Computing Sciences in Colleges. 26, 252–258 (2011).
3. Tabanao, E.S., Rodrigo, M.M.T., Jadud, M.C.: Predicting at-risk novice Java programmers through the analysis of online protocols. Proceedings of the seventh international workshop on Computing education research. pp. 85–92. ACM, New York, NY, USA (2011).
4. Kinnunen, P., Simon, B.: Experiencing programming assignments in CS1: the emotional toll. Proceedings of the Sixth international workshop on Computing education research. pp. 77–86. ACM, New York, NY, USA (2010).
5. Pekrun, R., Stephens, E.J.: Academic emotions. In: Harris, K.R., Graham, S., Urdan, T., Graham, S., Royer, J.M., and Zeidner, M. (eds.) APA educational psychology handbook, Vol 2: Individual differences and cultural and contextual factors. pp. 3–31. American Psychological Association, Washington, DC, US (2012).
6. Rodrigo, M.M.T., Baker, R.S., 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 Bull. 41, 156–160 (2009).
7. Shute, V.J., Kyllonen, P.C.: Modeling Individual Differences in Programming Skill Acquisition. (1990).
8. Rodrigo, M.M.T., Baker, R.S.J. d.: Coarse-grained detection of student frustration in an introductory programming course. Proceedings of the Fifth International Workshop on Computing Education Research. pp. 75–80. ACM, New York, NY, USA (2009).
9. Calvo, R.A., D'Mello, S.: Affect Detection: An Interdisciplinary Review of Models, Methods, and Their Applications. IEEE Transactions on Affective Computing. 1, 18 –37 (2010).
10. Bosch, N., D'Mello, S., Mills, C.: What Emotions Do Novices Experience During their First Computer Programming Learning Session? (in review).