Abstract—This annotated bibliography contains references and very brief descriptions of each reference, for a variety of research material related to the topic of assessing emotions in computer science education and practice.

I. INTRODUCTION

Measuring the emotional affect of students and programmers as they work at programming tasks is potentially useful for the purposes of increasing engagement and productivity, as well as enhancing the ability of students to learn quickly and effectively in a strange, challenging environment. Additionally, by building models of this activity, students of the future may be able to interact with electronic teaching systems that are better suited to tailor assignments and instruction to best suit the emotional needs of students.

II. ANNOTATED REFERENCES

[1] One of the first reports on computer tutoring based on data analysis, this technical report focuses on tutoring specifically in the Socratic domain. That is, it contains a set of rules to be followed in various tutoring situations where the knowledge is imparted to students using a series of directed questions rather than by retelling facts. The idea here is that students will need to be engaged at all times and might learn better even in an automatic environment because they will be forced to discover information on their own at some level.

[2] This article attempts to provide an overview of the problems associated with computerized automatic tutoring systems. It is a highly complicated problem, so this attempts to detail a large variety of these difficulties and ensure that the problem is not underestimated and tackled with naive, insufficient methods that fail to deal with all of the subtleties.

[3] This article brings into the field the beginnings of emotional modelling on computer system. It examines three different emotional models based loosely on psycholgical stereotypes and attempts to address the question of what happens, computationally speaking, in each one of these models, as well as how they differ from each other.

[4] This article examines the efficacy of several different tutoring systems, in particular a spell checking word processor on a computer, a computerized tutoring system, and a traditional textbook-based learning environment. The word processing method turned out to be more effective than the computerized tutor, but also more effective than the traditional teaching method, showing some educational potential for computerized education.

[5] This is a technical report from the US Air Force, in which they investigate the differences that occurred between subjects in a test where they were taught the basics of Pascal computer programming by an automated computerized tutoring system. It was far ahead of most of the research in that field from the era and continues to be highly relevant in studies on computer tutoring systems in modern research.

[6] In this article the authors attempt to revisit some earlier research and correct some of the methodology. Their results show that there is a certain amount of correlation between certain psychological factors of a student and their ability to perform well in computer programming tasks, thus establishing the importance of studying emotional factors as they relate to computer science education and intelligent tutoring systems.

[7] This article uncovers some of the emotional correlations with programming skill acquisition. Specifically, the authors look at how anxiety affects the abilities of students in this field, and how in turn the anxiety may be reduced by further education and the confidence that results from increased knowledge and confidence in capabilities.

[8] This article examines the emotional affect of students at various points during exercises and problem solving activities related to computer science in a programming class at the university level. The authors do so by manual evaluation of students by instructors at coarse intervals. Because of the plural group size and limited capabilities of human judges, this study is limited in its precision, but does a good job of providing classification of emotion in a general way at least.

[9] In this article, the authors produce an extensive modification to the BlueJ code editor which allows them to monitor and record the activity of students in real-time as they compile programs for assignments. The authors attempt to categorize this activity, including the time that they took and the errors they encountered while doing it. They then draw correlations between certain patterns of activity and success or failure of students so that at-risk students may be identified before they fail.

[10] This article categorizes students according to some basic demographic information, and by their learning styles.
They then compared these different categories of students in their abilities to learn computer programming. Interestingly, gender did not play a role in determining ability after controlling for factors that did, such as general academic ability.

[11] The research in this article concerns using predictive models to determine the efficacy of learning for a student according to their mood. Excessively positive or negative emotions can produce negative effects in education, so these models, though quite simple being limited to just positive and negative classification, might be useful to predict when students will enter into a suboptimal learning state and steer them away from it.

[12] This article discusses the difference that doodling on a programming assignment can make for student grades on midterm exams. The authors examine the work of students on exercises that come before the midterm exam in their programming class, and find that those students who “doodle”, by writing additional partial answers and tracing material other than just the correct answer, are likely to score significantly higher than those who did not.

[13] This article uses code analysis and visualization methods on students’ programming assignments in an environment with relatively few restrictions. These techniques for extracting key properties of source code are useful in the article for evaluating the experience and capabilities of students in an automated way that exceeds the capabilities of human observers in some ways.

[14] This article analyzes various types of computer science learning facilitated by automated tutoring agents, and evaluates the efficacy of these different methods. Some methods were found to be much better than others, and the authors found that using the more effective methods resulted in tutoring systems that were able to effectively teach difficult computer programming concepts to novice learners.

[15] This lecture attempts to build a regression model of the emotional states encountered by students while using an automatic SQL tutoring system. In the end, they were able to build a model that correlated well with boredom, although not as well with other emotional affects.

[16] This researchers in this article chose to manipulate the mood of somewhat experienced programmers as they worked, to observe what the effects might be for people who already knew what they were doing, rather than students. Stimuli that was intended to cause emotional arousal in the participants resulted in significantly more correct results than the low arousal stimuli.

[17] The research in this article is relevant to affect-sensitive computer science education because it provides a validated measure of the perseverance, or “grit” of a student. This potentially important facet of programming and education in general is often overlooked at least in an empirical way, and might, in the context of affective computing, provide some predictive capabilities for use in modelling.

[18] This article explores some extremely unique ways to promote computer science education and allow it to become more effective and accessible. These techniques include interesting and highly atypical activities such as dance, music, and other live performing arts. Illustrating difficult concepts in computer science with fun and engaging students might help them to learn more effectively and promote interest and application of principles in a broader scope than merely programming.

[19] In the research in this article, students with programming experience were divided into logical groups according to their ability demonstrated in previous academic work and programming classes, and were then exposed to exercises testing their ability to perform operations on modular code. Higher performing students, unsurprisingly, managed to do better on this task than the other students, but also demonstrated more even emotional temperment and less frustration.

[20] In this article, the researchers measure traditionally psychological aspects of students and relate those to their capabilities in a computer science class. They found that, when looking at the speed with which students moved through life (maintained their focus), and whether or not they tend to focus on people or tasks is a contributing predictor of their capabilities to succeed in computer science education.

[21] This article explores the factors that motivate students of computer science in the framework of an automated e-learning environment. Motivation is important not only for finishing assignments but also for providing impetus to propel students into a habit of practicing their programming skills, which is essential to excelling in the field.

[22] This article is a collection of information regarding a wide variety of automatic programming assignment evaluation software and the techniques that these software packages use. The possibilities are virtually endless for approaching this problem, so the article outlines some of the major approaches that have been used with some degree of success, thus providing a starting point for researchers interested in pursuing these systems further or implementing their own special-purpose systems.

[23] This article looks specifically at the confusion emotion as it relates to computer programming students. By examining logs of the compilation activities of students while they were programming, the researchers were able to identify, both manually and automatically, behavior which appeared to strongly represent confusion. They found that confusion has a negative effect on learning gains, as one might expect, except in the case where the confusion was encountered and then resolved, in which case learning was actually more effective than instances where there was no confusion at all.

[24] This article provides an overview of a pair of tools that work well together for running experiments collecting student data, and which work well for the students as a platform for learning. Specifically, they recommend the use of the extensible editor BlueJ, which is intended for Java, and a schema-less database system that can be paired with it to store data.

[25] This article details a variety of approaches that have been used in different universities to tackle the problem of providing computerized intelligent tutoring systems that are
effective for students. The systems they examined had varying processes with the same ultimate goal of imparting knowledge, so this paper provides an overview of these processes as well as methods that can be used to measure the efficacy of a particular intelligent tutoring system. Such systems are well suited as tools for administering materials and manipulating student affect to observe what behaviors they exhibit.

REFERENCES


