What the Mouse Said:

How Mouse Movements Can Relate to Student Stress and Success

Natalie Culligan
Department of Computer Science
Maynooth University
natalie.culligan@mu.ie

Kevin Casey
Department of Computer Science
Maynooth University
kevin.casey@mu.ie

Abstract

Stress in students may be a useful indication for when a student is struggling and in need of academic intervention. Investigating differences in student behaviour in stressful and comparatively less stressful environments could be helpful in understanding the processes involved in learning to code, and combatting the high levels of drop-out and failure in undergraduate computer science. In this paper we will discuss the mouse movement data gathered from Maynooth University Learning Environment (MULE), our in-house, browser-based pedagogical environment for novice programmers, during the time period February to May of 2019. This included 5 supervised, scheduled lab sessions and two in-lab examinations. The data was used to examine 21 different measurements of student behaviour, for example, by measuring efficiency of the mouse path, or the time between mouse click-down and mouse click-up. These features were used to build a Deep Neural Net that classifies sequences of mouse movements as being either from a more stressful environment or a less stressful one by training the classifier on data from examination situations and regular weekly lab situations, with the goal of comparing how students behave in environments with different levels of student comfort. The classifiers had an average accuracy of 61.9% but was more successful with students who performed poorly in their lab examinations. To further examine this connection between mouse movement, stress and student outcome, a second classifier was built to classify students as being in the high or low 50% of lab-exam grades in the module, with an accuracy of 69%.

1. Introduction

In this study we use data collected by Maynooth University Learning Environment, or MULE (Culligan, Casey 2018). MULE is an online, browser-based pedagogical desktop environment which has been used in multiple first-year coding modules. We received clearance from the University Ethics Committee to collect mouse movements from students as they learn to code from the 29th of February until the 3rd of May in the Introduction to Programming II module (taught in Java) with 250 students completing the module. The students were informed about the use of their data and were asked to consent at the beginning of the semester. All students who completed the module chose to participate in the study.

Using the mouse movement data collected by MULE, a Deep Neural Net (DNN) binary classifier was built to detect if a sequence of mouse movements is from a stressful (in-examination) or less stressful environment (in-lab). The classifier is not universal. It needs to be trained on a student's own data and does not work on all students. This was expected, as stress and comfort are subjective and not all students will experience stress in the same way during an examination. Students may also have different mouse use "styles", which makes it harder to generalise mouse behaviour caused by stress. We must also consider that some students are not stressed during an examination and may even be less stressed than in a normal lab situation.

The classifier works very well for some students and poorly for others, with an average increase of 11%-12% over the accuracy baseline of 50%, an average accuracy of 61.9% for classifying both in-lab and in-examination sequences. The classifier was moderately successful but interestingly the classifier was more successful for students who did poorly in the module. To further explore this, we built a second DNN to investigate if the mouse movement data could be used to classify students as being in the top or bottom 50% of module grades. This classifier was more successful than the stress

classifier, classifying students as being in the top or bottom 50% of the module Continuous Assessment grades with an accuracy of 69%, over an accuracy baseline of 50%.

In this paper, the following questions will be explored in relation to the gathered mouse movement data.

> 1. *Are there differences in mouse movement behaviour of students between lab and exam situations, and can this be a first step in a classifier for stressed students?*
> *2. Are there differences in student mouse behaviour and stress in students in CS1 between students who perform well in-lab examinations and written exams, and those who perform poorly?*

The null hypothesis for these questions are as follows:

> 1. *The results from the Deep Neural Net for classifying sequences of mouse movements sequences as being from stressful or not stressful environments performed no better, or not significantly better than random chance.*
> *2. The results from the Deep Neural Net for classifying individuals as being in the top or bottom performing 50% of students performed no better, or not significantly better than random chance.*

2. Motivation and Related Research

Stress in students may be a useful indication for when a student is struggling and in need of academic intervention. Intervention for students experiencing unusual amounts of stress could be helpful in combatting the high levels of drop out and failure in undergraduate computer science (Beaubouef et al, Biggers et al, Giannakos et al, Hembree et al, Kinnunen et al). This is the first of our studies into student behaviour as they learn to code, and in this study we focus on mouse movement. There are studies that suggest that mouse movement is linked to stress and mood (Sun *et al.*, Wahlström, *et al.*, Yamauchi). In this paper, we are interested in examining student mouse movement in stressful and less stressful environments to try and gain insight into behaviours that indicate stress, and investigate if this is related to student performance.

2.1. Stress Levels in Students

Computer science courses have been reported to have low levels of retention in comparison to other subjects (Giannakos *et al.*, Kinnunen, *et al.*). Research suggests that student comfort is a useful signifier of student success and retention (McCracken *et al.*, Tenenberg, *et al.*, Wilson and Shrock), and that stressful situations such as examinations can cause a student to preform below their ability (Beilock and Carr).

Beilock and Carr discuss the connection between anxiety and a loss in academic performance, and suggest that situation-related worries – such as examination stress or anxiety – can result in a loss of focus on task at hand as the working memory is occupied. Alternatively, it has also been suggested that over-attending to performance, overthinking tasks usually performed automatically, can lead to underperforming in an uncomfortable or stressful situation. Beilock *et al.* discuss how a more stressful or anxious state can also affect tasks that are usually performed in an automated fashion, without the subject thinking too much about it – their paper mentions soccer players' dribbling. We propose that mouse movement could be considered in a similar manner.

Connolly *et al.* found that in their study of 86 computing undergraduate students, 44.4% reported not feeling relaxed when using computers, suggesting that research into this area would be beneficial to a significant portion of the student population.

Bergin and Reilly examined 15 factors in predicting if a student is likely to pass or fail. One of the most statistically significant factors in predicting success was comfort level, in relation to how the student felt about the course. This was measured through cumulative responses to questions about the students' understanding and difficulty completing lab assignments.

2.2. Mouse Movement and Stress

There is prior evidence of a link between student stress and comfort level and their mouse movements. Sun *et al.* constructed a Mass Spring Damper model for the human arm - essentially a model for approximating arm motion and stiffness which could be fed with data from mouse movements. Using arm stiffness as a proxy for stress in the user, the authors report that their method was tested across a variety of prescribed stress tasks. The classifier worked when generalised but was more effective when trained and tested separately for each user.

Yamauchi claims there is both psychological and neurological evidence to suggest that mouse trajectories can be used to assess affective states, such as anxiety. The results of their study show that temporal features, such as speed of mouse movement, and spatial features such as direction change were both indicative of the user's state of anxiety. The researchers in this paper ran a separate analysis for male and female users and found different indications of state anxiety, with female subjects being more inclined to use a less efficient mouse path when anxious, and male subjects being more likely to change their mouse velocity.

Kapoor *et al.* use a specialised pressure mouse with additional sensors to detect frustration in subjects as they attempt to complete a towers of Hanoi puzzle computer game. The game includes an "I'm frustrated" button for the users, which is used to associate behaviour with frustrated state. The resulting classifier can predict frustration at an accuracy of 79%, outperforming the random classifier (58%).

3. Research Design

The goal of this study was to examine the relationship between student mouse behaviour, student outcome, and comfort level in students in CS1, an introduction to programming module. Using the data from MULE, we constructed a Deep Neural Net binary classifier to classify sequences of mouse movements as being from a stressful environment or a less stressful one.

Data Type	Description
userID	The anonymous ID assigned to the student
dumpID	The ID of the dump from student session to the database
sessionID	An ID assigned to the session when a student logs in until they log out
Time	Timestamp of when the event took place, not when it was stored
Type	Mousemove, mouseup or mousedown
X	X co-ordinates of the mouse's current position
y	Y co-ordinates of the mouse's current position

Table 1: Mouse movement data features

MULE was used to collect mouse movement data from students as they learned to code in an authentic learning environment. To use the system, the students sign in through their Moodle accounts from any internet browser on any machine, they do not need to be in the university computer labs. The system is a desktop-like environment simulated within the browser, where they can view assignments from a designated application, use a text editor to write code for the assignments, and compile, run and automatically evaluate their code, receiving a grade and automated feedback if their code has errors. The students use the mouse to navigate the system, to open assignments, open the code editor, and to save, compile, run and evaluate from drop down menus. As the student works, the system automatically stores their mouse movements, along with a timestamp and an anonymised user key to allow for cross session comparisons. Stored mouse movements are sent to the database every 30 seconds, or as soon as the user tries to log out or close the system tab. The system collects mouse movement data as shown in Table 1. Anonymised data on students' performance in the module was also collected, specifically how they performed in the written examination, in weekly labs and in-lab examinations. The total number of students who completed the second semester was 250, of which 196 are included in this study. We removed data from students who did not participate enough for their data to be used in the study, including:

1. Students who did not take both in-lab examinations
2. Students who did not complete the course
3. Students who participated in less than two lab sessions

Students have labs for 3 hours once a week for 12 weeks per semester. The students began using the system in the first semester of the academic year 2018/2019 and used the system for the rest of the academic year. The mouse movement data set we are examining in this paper is from the second semester, from the 29th of February until the 3rd of May. This time period includes 5 regular weekly labs and 2 in-lab examinations. We compare mouse data from students in a regular lab situation versus mouse data from an examination situation, to examine the differences between coding when in situations with different levels of comfort. Both situations are in the same physical space, but with different rules. The students are not allowed to speak to each other, ask for help from demonstrators or look back at their previous work during the examination situation, but are encouraged to do so during regular labs. One of the authors worked as a demonstrator in the labs where this research took place to ensure the coding environment was working correctly, and to assist the students.

We recorded mouse data from students as they worked in scheduled labs, scheduled examinations, and outside of these times. The data from outside of the lab is not discussed in this paper. Data outside scheduled labs and examinations may be the result of users other than the signed-in student and/or very different mouse set up (touch screen, touch pad, or different desk size, for example). Students may also be working in very different situations due to environmental noise, distractions, or caretaking responsibilities, for example.

The mouse data from each student is divided into sequences to be assessed by the classifier. Each sequence begins with any mouse movement and ends with a mouse click-up, and any sequence that is longer than 1450ms is rejected to avoid evaluating sequences from when the student is idle. This time limit was chosen though trial and error, and found the classifier worked best with sequences under this time limit. Tests are run on each sequence to find various metrics for the users' behaviours. Metrics include SequenceSpeed, ClickTime and Efficiency. Each sequence also has an identifier, as in-lab, in-examination or out-lab. Once we have the metrics for each of the sequences, they are used to train and test the Deep Neural Net.

We used a total of 21 different features in our classifier.

Features

1. *AngleVariance1:*

Finds all the different angle changes from one movement to the next (with precision of 2 digits) within a sequence and returns the total number of unique angles.

2. *AngleVariance2*

Same as above, but the total number of angles returned.

3. *AngleVariance3*

The ratio of total angles to unique angles.

4. *VarianceDistance1*

Finds the optimal distance between every set of two mouse movements to 1 decimal place and returns the number of all unique distances.

5. *VarianceDistance2*

Same as above but returns the number of all distances.

6. *VarianceDistance3*

The ratio of all unique distances and all distances in the sequence.

7. *Overshoot-x*

Measures how far a user "overshoots" with the mouse in the direction they are moving the mouse in, along the X axis. If a user moves from point a to point b within a small window of time, point b being where they click the mouse, if at some point during this journey they move further along the x-axis then where they ended, this is recorded as an *Overshoot-x*.

8. *Overshoot-y*

Same as *Overshoot-x*, but along the y axis.

9. *Overshoot*

The square root of *Overshoot-x* and *Overshoot-y* squared and added.

10. *OvershootDirectionAngle*

Finds the angle of the overshoot.

11. *SequenceSpeed*

The total distance travelled divided by the total time.

12. *SequenceDuration*

The time duration of the sequence.

13. *DistanceTravelled*

The true distance travelled during the sequence.

14. *OptimalDistance*

The distance in a straight line between the start and end points of the sequence.

15. *Efficiency*

Optimal distance divided by total distance travelled.

16. *Direction*

The direction from the first point in the sequence to the last.

17. *DirectionAngle*

The direction angle between the starting point and the ending point of the sequence.

18. *AngleDifference*

The absolute value of DirectionAngle subtracted from *OvershootDirectionAngle*.

19. *ClickTime*

The time between click down and click up.

20. *Hesitate*

The amount of time the mouse stalls before the user clicks.

21. *ClickRatio*

This is *Hesitate* divided by *ClickTime*

Yamauchi's paper '*Mouse Trajectories and State Anxiety: Feature Selection with Random Forest*' found that speed and direction were indicators of a subject's emotional state. Our features are chosen to examine this connection, with features such as `DistanceTravelled` and `ClickTime` relating to speed, and `DirectionAngle` and `OvershootDirectionAngle` relating to direction. The paper also discusses tracking direction change, x-overshoot, y-overshoot, which we replicated in our experiment with features such as `Overshoot-x`, `Overshoot`, `DirectionAngle` and `DirectionAngle`. Beilock et al discuss how a more stressful or anxious state can also affect tasks that are usually preformed in an automated fashion. We investigated this with the features `VarianceDistance1`, `VarianceDistance2`, `VarianceDistance3`, to give us insight into how much the subject changed their speed, and the features `AngleVariance1`, `AngleVariance2` and `AngleVariance3` to investigate how often the subject changed direction, perhaps due to confusion or indecisiveness as a result of stress or discomfort.

As per Sun *et al.*, we trained our classifier per user, instead of building a generalised stress classifier. Our initial experiments involved a general classifier using a large subsection of the data from all students, but this classifier did not perform significantly better than random chance. To build a classifier for a user, we selected all the sequences from in-examination, and then a random selection of sequences of an equal amount from in-lab, or vice-versa, depending on the imbalance of data categorised as in-lab or in-examination. The features we get from the mouse movements of each student are then used to train and test a deep neural net, built in Python using TensorFlow (Abadi, Martín, *et al.*).

For most students, we have much more in-lab data than in-examination, so we take a random sample of the in-lab data equal to the size of the in-examination data. We used TensorFlow's DNNclassifier module, with 3 hidden layers of 10 units, a batch size of 5 and 2000 epochs. The classifier outputs a 1 if the mouse movement sequence is classified as in-lab and 0 if the sequence is classified as in-exam. When running the classifier for each student, we wanted to ensure that the results were not due to chance, or a "lucky" selection of test data from the total data set. To combat this, we selected a subsection of the data as test data, and rejected it if it was not 50/50 in-lab and in-exam, again to avoid

good results that are just the result of a classifier only choosing one classification, regardless of feature input. To check that the variance for the classifier results was low, and we were not reporting outliers, the classifiers were run in sections of ten, and the variance within results was checked. The variance for all users was 0.05 or less, with one exception that had a larger variance of 0.13. We performed multiple sets of ten, checking the variance on the cumulative results. For each student, the classifier was run 60 times, with a different random division of training and test data with no increase in variance over 0.016 between the first 10 and the final 60.

4. Discussion of Classifier Performance

The classifier works very well for some students and poorly for others, with an average increase of 11% to 12% over the accuracy baseline and an average accuracy of 62.9% for classifying both in-lab and in-examination sequences. However, for some students that performed poorly in their lab examinations, we found the classifier could work 30% over baseline. On examination of the results, it became apparent that the classifier was more successful with the students who performed poorly in the module than those who performed well. One of the possible reasons for student stress during exams is that they may be unable to use their usual method of solving coding problems. Some students will take previously written code, copy it and rewrite it to complete the given task. During exams the students no longer have access to their previous code. They may panic when they find they cannot use their usual strategy (though they are informed beforehand of the format and rules of the exam), or they may be experiencing additional strain on their working memory. This strain may come from the extra work now being performed by the student. For example, they can't copy a while loop from previous work, so instead they struggle to remember how to write one. The student is not comfortable and familiar with the computer science concepts needed to construct the code to solve the exam question and has been relying on 'tinkering', a technique used by students as described by Perkins *et al.* and Jadud.

4.1. Stress Classifier

When examining the results of the classifiers, differences between the high-performing and low-performing students became apparent. Table 2 shows the average classifier of two groups, the top 50% of grades and bottom 50% of grades. This was done for Continuous Assessment, written exam and total module grade, and repeated with the top and bottom 40%, 30%, 20% and 10%.

	Module High	Module Low	Written Exam High	Written Exam Low	CA High	CA Low
50%	61.875%	62.7913%	61.3518%	62.627%	60.8315%	63.1473%
40%	60.8037%	62.6183%	61.1019%	61.949%	60.7157%	63.8293%
30%	60.1556%	62.6878%	60.9173%	62.6955%	59.7906%	63.9333%
20%	60.0027%	62.5255%	59.8544%	62.8666%	59.7657%	63.4562%
10%	58.8089%	62.8847%	58.2153%	62.643%	59.4642%	65.3041%

Table 2: Comparison of the high and low performing students

In all groups, and with all three grade types, the lower grades group have more successful classifiers, with the difference becoming more pronounced as we look at smaller subsections. We suspect that the reason students in the lower-grade groups are easier to classify is because these students may experience additional strain when writing code, perhaps due to exam anxiety, or a lack of comfort with the material. In the paper "*On the causal mechanisms of stereotype threat: Can skills that don't rely heavily on working memory still be threatened?"*, Beilock, *et al.* claim that while overloaded working memory does not directly affect procedural skills because it is not reliant on working memory, over-attention to procedural skills does impact the subject's performance – a worried student may overthink their behaviour, causing changes in their mouse movement.

4.2. High Low Grade Classifier

We were interested in the possible connection between mouse movements and student grades, from the apparent relation between classifier success and the students' performance in the module shown in Table 2. We suspected that the results indicated a relation between mouse movements, specifically indications of stress in exams, and student grades. There is previous work (Casey) to suggest that low-

level keystroke data can be used to improve grade classifiers, so we wanted to examine if mouse movement data could also be used. To investigate this, a trio of DNN classifiers were created to predict the outcome of students in:

1. *Continuous Assessment (coding exercises, and lab exams),*
2. *End of year written exams*
3. *The module overall.*

The DNN uses the same configuration as the stress classifier. We tried other configurations, including increasing the number of hidden units, but found this was the most successful setting. The DNN classifies each student into one of two categories – either the higher or lower 50% of the class, divided by the results in order. For this dataset we calculated the average of each of the features in the table for in-lab and out-lab. We found this gave the best results, possibly because the indicator of a student who does well or poorly is the difference, or the similarity of the behaviour between regular labs and exams, in line with the findings that the students who did poorly were more easily classed by the classifier.

Grade	Higher 50%	Lower 50%	Classifier Results
Written Exam	63% and over	61% and under	0.588333333
Module Total	59% and over	58% and under	0.656666667
Continuous Assessment	54% and over	53% and under	0.693333333

Table 3: Results of classification

Like the previous classifier, the high/low classifier was run 60 times, each time randomly selecting the training set and the testing set. Like the stress classifier, the randomisation was written to ensure that the testing data set would always be 50% from each classification, to avoid misleadingly high or low results from a classifier only choosing one classification.

5. Discussion of Research Questions

1. *Are there differences in mouse movement behaviour of CS1 students between lab and exam situations, and can this provide insight to the different comfort levels experienced by students in these environments?*

The DNN classifier was mildly successful, implying that there is at least a weak link between mouse movement and comfort level. Students may still be stressed in lab situations, but because the classifier was more successful with students who did poorly in their lab examinations, we believe this is evidence that the classifier works as an indicator of stress – we believe that students who are taking examinations that they are not doing well in are more likely to be experiencing stress than others. We can reject the null hypothesis, as the classifier is more successful than a random chance classifier.

2. Are there differences in student mouse behaviour and comfort level in students in CS1 between students who perform well in-lab examinations and written exams, and those who perform poorly?

The classifier is more effective with students who perform poorly than those who perform well. We would expect students who do poorly in the module to be more stressed in examinations than students who are comfortable with the material and are performing well. To examine this further we built a second DNN classifier and found that we were able to classify students into high/low performing groups with 69% accuracy. We reject the null hypothesis as the high/low classifier is more successful than random chance.

6. Conclusions and Future Research

In this paper, we have reported on the construction of a moderately successful Deep Neural Net that classifiers sequences of mouse movements as being from a stressful or less stressful environment. While other researchers have published work on the connection between mouse movements and stress, to our knowledge this is the only study of mouse movements and stress that uses mouse data gathered

outside of closed experimental environments. From the analysis of the results, we found a connection between mouse movements and a student's grades, especially grades for practical coding assignments.

The classifiers in their current state are not a useful mechanism for detecting stress in students, or for predicting if students will be in the high or low 50% of grades. However, in the construction of these classifiers, we have found mechanisms that will contribute to the construction of models of successful students, and classifiers for students in need of academic intervention. This study is part of a larger project to examine the relationship between student behaviour when learning to code and student success and retention. Our coding environment gathers data beyond mouse movement, including keystrokes, compilation and run results, and returned errors. Other research in this area has used data such as keystrokes to predict student outcome (Casey), and from our work in this paper, which suggests a connection between student success and comfort-level, we believe this data will give further insight to student behaviour in stressful situations. Further work can be done in relation to the mouse analytics performed so far. We are currently refactoring our recording of mouse data so that we can capture additional data in order to attach more meaning to mouse sequences. This would, for example, allow us to distinguish between a mouse sequence that led to a file being saved, versus a mouse sequence that led to a compilation of student code.

We believe there is huge potential for study of this data, which is gathered from an authentic learning environment, as students learn to code. With continued research, we plan to build a larger model of the behaviour of novice programmers as they learn to code, with the potential for an integrated classifier in our coding environment that will alert course coordinators to a student in need of intervention. We hope the construction of a model of successful students will be a useful way to inform and build curriculums that best help students achieve their potential.

5. References

Abadi, Martín, *et al.* (2016) Tensorflow: A system for large-scale machine learning. 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16).

Beaubouef, Theresa, and John Mason. (2005) Why the high attrition rate for computer science students: some thoughts and observations. ACM SIGCSE Bulletin 37.2 103-106. DOI: https://doi.org/10.1145/1083431.1083474

Beilock, Sian L., and Thomas H. Carr. (2005) When high-powered people fail: Working memory and "choking under pressure" in math. Psychological science 16.2 01-105. DOI: https://doi.org/10.1037/e537052012-380

Beilock, Sian L., *et al.* (2006) On the causal mechanisms of stereotype threat: Can skills that don't rely heavily on working memory still be threatened?. Personality and Social Psychology Bulletin 32.8 1059-1071. DOI: https://doi.org/10.1177/0146167206288489

Bergin, Susan, and Ronan Reilly. (2005) Programming: factors that influence success. ACM Sigcse Bulletin 37.1 411-415. DOI: https://doi.org/10.1145/1047344.1047480

Biggers, Maureen, Anne Brauer, and Tuba Yilmaz. (2008) Student perceptions of computer science: a retention study comparing graduating seniors with cs leavers. ACM SIGCSE Bulletin. Vol. 40. No. 1. ACM. DOI: https://doi.org/10.1145/1352135.1352274

Casey, Kevin. (2017) Using keystroke analytics to improve pass-fail classifiers. Journal of Learning Analytics 4.2 189-211. DOI: https://doi.org/10.18608/jla.2017.42.14

Connolly, Cornelia, Eamonn Murphy, and Sarah Moore. (2008) Programming Anxiety Amongst Computing Students—A Key in the Retention Debate?. IEEE Transactions on Education 52.1 52-56. DOI: https://doi.org/10.1109/te.2008.917193

Culligan, N., & Casey, K. (2018). Building an Authentic Novice Programming Lab Environment. Irish Conference On Engaging Pedagogy

Giannakos, Michail N., *et al.* (2017) Understanding student retention in computer science education: The role of environment, gains, barriers and usefulness. Education and Information Technologies 22.5 2365-2382. DOI: https://doi.org/10.1007/s10639-016-9538-1

Hembree, Ray. The nature, effects, and relief of mathematics anxiety. Journal for research in mathematics education (1990): 33-46. DOI: https://doi.org/10.2307/749455

Kapoor, Ashish, Winslow Burleson, and Rosalind W. Picard. (2007) Automatic prediction of frustration. International journal of human-computer studies 65.8 724-736. DOI: https://doi.org/10.1016/j.ijhcs.2007.02.003

Jadud, M. C. (2006). An exploration of novice compilation behaviour in BlueJ (Doctoral dissertation, University of Kent). DOI: https://doi.org/10.1080/08993400500056530

Kinnunen, Päivi, and Lauri Malmi. (2006) Why students drop out CS1 course?. Proceedings of the second international workshop on Computing education research. ACM. DOI: https://doi.org/10.1145/1151588.1151604

Lister, Raymond. Concrete and other neo-Piagetian forms of reasoning in the novice programmer. Proceedings of the Thirteenth Australasian Computing Education Conference-Volume 114. Australian Computer Society, Inc., 2011. DOI: https://doi.org/10.1215/9780822381525-005

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B. D., ... & Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. In Working group reports from ITiCSE on Innovation and technology in computer science education (pp. 125-180). ACM. DOI: https://doi.org/10.1145/572139.572181

Perkins, D. N., Hancock, C., Hobbs, R., Martin, F., & Simmons, R. (1986). Conditions of learning in novice programmers. Journal of Educational Computing Research, 2(1), 37-55. DOI: https://doi.org/10.2190/gujt-jcbj-q6qu-q9pl

Sun, D., Paredes, P., & Canny, J. (2014, April). MouStress: detecting stress from mouse motion. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 61-70). ACM. DOI: https://doi.org/10.1145/2556288.2557243

Tenenberg, Josh D., *et al.* (2005) Students Designing Software: a Multi-National, Multi-Institutional Study. Informatics in Education 4.1 143-162.

Wahlström, J., *et al.* (2002) Influence of time pressure and verbal provocation on physiological and psychological reactions during work with a computer mouse. European journal of applied physiology 87.3 257-263. DOI: https://doi.org/10.1007/s00421-002-0611-7

Wilson, B. C., & Shrock, S. (2001). Contributing to success in an introductory computer science course: a study of twelve factors. Acm sigcse bulletin, 33(1), 184-188. DOI: https://doi.org/10.1145/364447.364581

Yamauchi, Takashi. (2013) Mouse trajectories and state anxiety: feature selection with random forest. 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction. IEEE, 2013. DOI: https://doi.org/10.1109/acii.2013.72