# Improving Student's Engagement Through the Use of Learning Modules, Instantaneous Feedback and Automated Marking

Prapa Rattadilok
*School of Computer Science*
*University of Nottingham*
Ningbo, China
prapa.rattadilok@nottingham.edu.cn

Chris Roadknight
*School of Computer Science*
*University of Nottingham*
Ningbo, China
chris.roadknight@nottingham.edu.cn

***Abstract*—Assessment is central in effective teaching. This research sets out to discover the impact and effectiveness of timely assessment and feedback on student performance and engagement. Qualitative and quantitative data is collected from two cohorts of students with different levels of engagement. We have shown that more regular feedback and engagement resulted in a significantly improved pass rate and average mark. In conclusion, enabling timely assessment and feedback can improve student performance and give educators tools that make this process more manageable.**

***Keywords—instantaneous feedback; student engagement; learning modules; automated marking***

## I. INTRODUCTION

Because it is impossible to predict what students learnt with any certainty, assessment is therefore commonly used to determine whether any teaching activites resulted in intended learning. There are a number of different types of assessment methods, whereby formative and summative assessments are among the most commonly used. While summative assessments are used to record and report on students' overall achievement at a given point, formative assessment is one of the most powerful ways of improving student's learning achievement through feedback [1].

Assessments' feedback is one of the most powerful influences on the learning of students [2]. It can also be considered as a retention strategy particularly during the first year of university by lessening student's performance anxieties and intensifying their perception of academic support that are available. In addition to that, the process of providing feedback itself can also be used to inform the teacher regarding students' understanding gaps for ongoing curriculum development [3].

Various forms of generic feedback can be provided rapidly and cost-effectively. However, benefits to students are constrained to mainly signaling weaknesses in knowledge in comparison to personalised feedback. Timeliness of the feedback is also vital in motivating students in the process of competency achievements. Feedback that shortly follows an assessment allows students to reflect on their own performance while it is fresh in their minds, whether this be regarding their strengths and weakness. This further builds on their capabilities and addresses deficient areas [4]. Nevertheless, personalised and timely feedback can pose challenges in terms of time and logistics particularly for large cohorts.

A virtual learning environment is used to automatically deliver personalised and timely feedback to two different cohorts of first year computer science students. Combining this with the use of learning modules [5] enables a deeper feedback process, this feedback process aims to answer the following three feedback questions: Where am I going? How am I going and Where to next? [2]. This paper compares students' performance of the two cohorts, prior and following the implementation of automated personalised feedback.

## II. MARKING AND FEEDBACK

Marking provides feedback to students and helps teachers identifying areas of student misunderstanding. Research [6] suggested that good feedback should address both cognitive and motivational factors at the same time. For cognitive factors, this includes highlighting what stage they are at in their learning and what to do next. On the other hand, motivational factors develop once the student feels that they have control over their own learning.

Surveys [7][8][9] indicate the differences in guidelines for feedback strategies and feedback content, which vary greatly between institutions. In particular, emphasises the need to inform students as early as possible regarding the differences in the feedback process at university and school levels [10].

Digital feedback has only gained more popularity in recent years [9][11]. The use of text-editing tools, voice-recorded feedback, and web-based feedback are some examples of digital feedback. Although they can be made available to the students relatively promptly, none of these surveys discuss instantaneous and automated feedback.

## III. AUTOMATED FEEDBACK

Instantaneous and automated feedback is an emerging field of research in teaching and learning. The scalability challenge in lecture theaters as well as in Massive Open Online Courses

(MOOC) have fueled the need for automated feedback. Two main areas where automated feedback have been used are language learning [12][13][14] and computer programming [15] [16][17].

National Language Processing (NLP) was used in [12] to extract linguistic features and evaluate the submitted piece of works according to syntax and topical features. Researchers in [13] apply machine learning to the task of evaluating 'English as Second Language' students. Their technique focuses on writing aspects such as grammar, spelling, sentence diversity, structure and organisation. The benefit of having automated feedback in particular is highlighted in [14]. Computerised feedback, which was inserted into student's work, became a productive source for learning as evidenced by the increased quality of each subsequent written draft by students.

Learning computer programming is somewhat similar to language learning. There is a clear set of syntax and structure that needs to be followed to produce a desirable outcome. Although there may be a number of approaches to achieve the same outcome, the complete specifications are known and the mistakes are predictable [15]. In [16], 69 different tools for learning programming were evaluated. Although mistakes are the most common type of feedback, many provide no knowledge on "how to proceed" and do not provide alternative solutions. In [17], the use of a plugin, FrenchPress, was evaluated in helping the learners learn how to program. Rather than focusing on compile-time, run-time or logical errors, FrenchPress targets the programmer's shortcomings whereby the programming environment does not alert them, such as better use of data type e.g. using constant instead of variable, or public instead of local variables.

## IV. Learning Modules

Learning modules [5] is a package of teaching material consisting of a sequence of activities and provisions of evaluations. From students' perspective, learning modules provide benefits in terms of: instant feedback to the learner; optional, self-continuative and recycling paths to achieve the learning objectives; and individualised use of instructions. From the teachers' perspective, learning modules provide benefits in terms of: the tracking student's learning progress; improvement of teaching instruction through behavioural observation of students, resulting from the learning outcomes achieved by the students; and the enabling of learning outside of the teacher's presence. Fig. 1 [18] shows an illustrative presentation of learning modules.

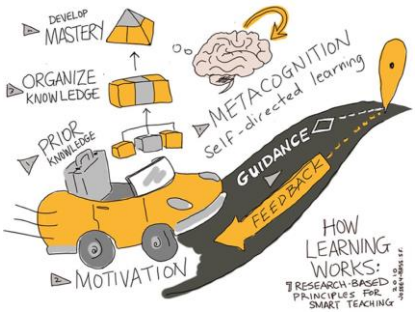



Fig. 1. Illustrative representation of learning module.

The ideal instructional model for learning modules is to itemise the subject content into a collection of concepts for students to work through and their performance can then be evaluated based on their competency. The responsibility for working through these collections of concepts is placed on the students with the teacher's role being to evaluate, comment and guide. A student can work on a single concept repeatedly, or several students can work on concepts simultaneously.

Learning modules maximise the use of technology to effectively enhance the occurrence of learning outside of classrooms. This shifts the teaching focus away from verbalised instruction. The teacher, as the orchestrator of learning activities, advises the students according to the learnt content the students have demonstrated. Rather than focusing on what has been taught, the student's performance and progress are evaluated.

## V. The Three Feedback Questions

An ideal learning experience or environment requires both teachers and students to seek answers to these three questions: 1) Where am I going?, 2) How am I going?, and 3) Where to next?. Unfortunately, teachers very often do not consider feedback given to students as learning possibilities for themselves [2].

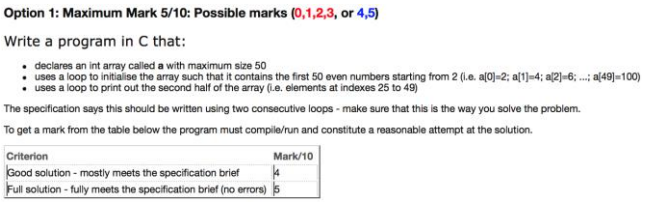
The attainment of learning objectives related to the task or performance, i.e. *"Where am I going?"*, can be judged on many dimensions. For example, direct "passing a test"; comparative "doing better than last time"; or even automatic and triggered outside of specific awareness "seek more challenging tasks". The last dimension promotes goal-directed action thus establishing the conditions for ongoing learning.

The information related to a task or performance can be measured according to some expected standard. This could be in relation to prior performance, or to success or failure on a specific part of the task. This measurement of progress can be given by the teacher, peer, task or themselves. It can be used to infer how the student should proceed i.e. *"How am I going?"*.

The final feedback question to be addressed is *"Where to next?"*. Rather than offering sequential instruction, feedback can be used to provide information that leads to greater fluency, learning and automaticity as well as deeper learning. For example, sign-posting to enhanced challenges, more information about what is and what is not understood.

## VI. The Instanteneous Feedback and Automated Marking Platform

A traditional programming task normally consists of a set of objectives and constraints (Fig. 2). Students often self-assess the software they developed according to the level of similarity of their output based the given objectives. However, problems arise particularly when the students are starting to learn how to program e.g. during practical sessions, as this assumption may not necessarily always be correct.



**Option 1: Maximum Mark 5/10: Possible marks (0,1,2,3, or 4,5)**

Write a program in C that:

- declares an int array called a with maximum size 50
- uses a loop to initialise the array such that it contains the first 50 even numbers starting from 2 (i.e. a[0]=2; a[1]=4; a[2]=6; ...; a[49]=100)
- uses a loop to print out the second half of the array (i.e. elements at indexes 25 to 49)

The specification says this should be written using two consecutive loops - make sure that this is the way you solve the problem.

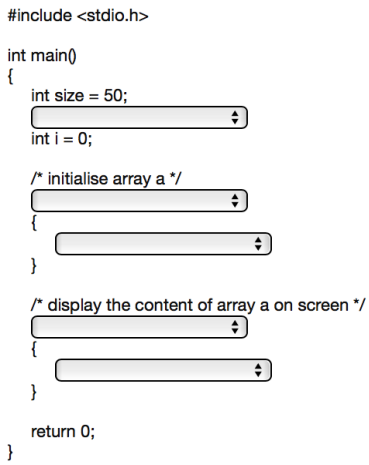
To get a mark from the table below the program must compile/run and constitute a reasonable attempt at the solution.

| Criterion | Mark/10 |
|---|---|
| Good solution - mostly meets the specification brief | 4 |
| Full solution - fully meets the specification brief (no errors) | 5 |

Fig. 2. Example of a traditional programming task.

This may therefore impact the overall levels of engagement and in-class interactions, as students may choose to never interact with their tutors during the practical sessions. Consequently, they miss opportunities to learn skills that could enhance their learning experience and subject-matter knowledge on the practical session topic. This issue can be exacerbated, especially for summative assessments, as students are evaluated based on the number of objectives and constraints they have achieved. In addition to this, their naive self-assessments during their practical sessions may have a long term impact on the efficiency of a piece of code and their coding style.

To evaluate the effectiveness of using learning modules to address the three feedback questions, the instantaneous feedback and automated marking platform (iFaME) was piloted using an off-the-shelf virtual learning environment. iFaME exploits fill-in-multiple-blanks and jumbled-sentences (Fig. 3) to automate the feedback and the marking for each programming task. While fill-in-multiple-blanks focus on evaluating students' understanding of syntax, jumbled-sentences evaluate both students' programming logic and syntax. A set of statements can be selected to address the given instructions similar to what is shown in Fig. 2.



```
#include <stdio.h>

int main()
{
    int size = 50;
    [ ▾ ]
    int i = 0;

    /* initialise array a */
    [ ▾ ]
    {
        [ ▾ ]
    }

    /* display the content of array a on screen */
    [ ▾ ]
    {
        [ ▾ ]
    }

    return 0;
}
```

Fig. 3. Example of a jumbled-sentence task.

Traditional programming tasks for practical sessions are organised into weekly journal entries. Each journal entry is divided into mandatory and supplementary sets of tasks. The supplementary sets of tasks are aimed at catering for the occasions where skilled students can gain additional practice. Students have two options in completing each journal entry. They can choose to complete a journal entry by programing from scratch, or by using iFaME.

While the feedback on iFaME is pre-specified, any journal entries that are completed by programming from scratch are manually assessed and given feedback by the tutors. Although the assessment and feedback for this option is not as instantaneous as iFaME, the tutors delivered the assessment and feedback for all submissions within a week of submission.

Fig. 4 compares the level of engagement between two first year computer science students. To protect their identities, the names of the students are anonymised to student A and student B. Each pie chart illustrates the level of engagement for each student. Three components are monitored, including the lecture slides, practical programming tasks and others (e.g. student's satisfaction surveys). The level of engagement for each of the three components are measured based on the amount of time students spent on different types of content. In Fig. 4, these three components are represented using red, blue and yellow respectively. As shown, both of the students spent most of their time in carrying out their practical tasks. Student B (i.e. the right pie chart) has a more balanced level of engagement between different types of content when compared to Student A (i.e. the left pie chart).

Fig. 4. A comparisons of the level of engagement on different learning contents i.e. the lecture slides (red), practical programming tasks (blue) and others (yellow) between two students: A (left) and B (right).

Having observed this behavioural patterns, and in order to promote student engagement [19], students are permitted to retake all of their summative assessments. The number of permitted attempts is three due to the administrative efficiency. The rest of this paper discusses the implementation outcome for a first year 'Programming in C' module. The cohort consists of 400 students with a mixed background in computer programing. The outcome compares the performance between a previous cohort of students when completing a traditional programming task.

According to our demographic survey, the two cohorts do have similar programming background. There are 5% of the students with experience in C programming language, 10% of the students with experience in C and at least one other programming language, the rest of the students either have no programming experience at all or having only non-C programming experience.

## VII. Implementation Outcome

Fig. 5 compares the performance of the two cohorts using histograms. When comparing the performance from the previous cohort in the top graph, the number of students who

failed the module, (i.e. scoring below 40), was lower and the average score of the students increased by 14%.

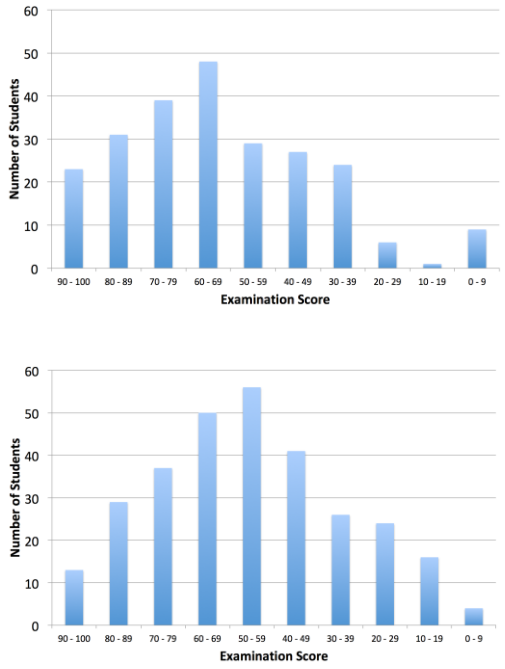



Fig. 5. A comparisons of the performance between the two student cohorts: previous year (top) and this year (bottom).

Fig. 6 compares behavioural patterns of the two cohorts, where x-axis represents the timeline and the y-axis represents the count of students that engage with the learning materials. As illustrated, the previous cohort (i.e. the top graph) focused their study closer to their summative assessment period, whereas the behavioral pattern for the current cohort is much more evenly distributed.

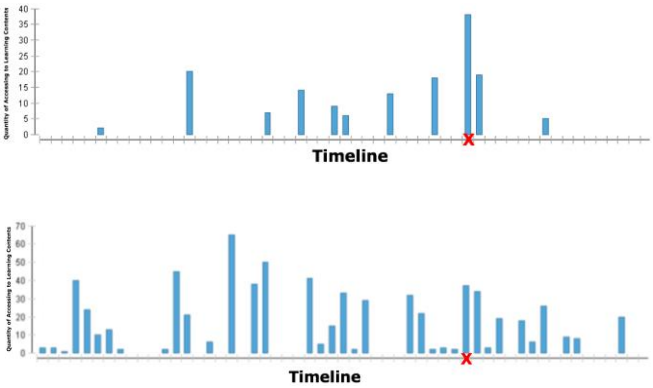



Fig. 6. A comparisons of the quantity of accessing learning contents between the two student cohorts: previous year (top) and this year (bottom).

Fig. 7 demonstrates student feedback for the 2 questions: "Journal entries keep me engaged with the learning?" (i.e. pie chart on the left), and "Being able to resit summative assessments helps me work harder on my learning progress" (i.e. pie chart on the right) respectively.

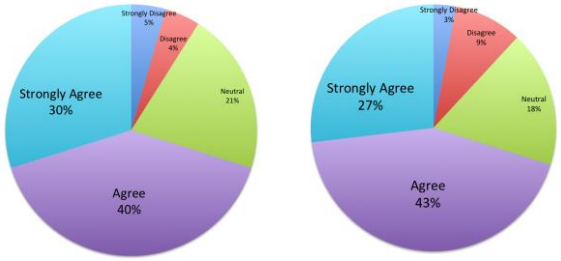



Fig. 7. Survey results on the question "Journal entries keep me engage with the learning?" (left), and "Being able to resit summative assessments helps me work harder on my learning progress" (right).

At the end of year qualitative surveys, students were asked "What have you most liked about this module?". The answers are largely positive. Example responses related to the learning modules and iFaME include "the journal entries are very intuitive and progressive", "the use of resits and the amount of provided supplementary tasks give me opportunities to study harder on the topics I realised I am not very good at.", "Both the automated assessment and programming from scratch help me practice my coding skills in different ways", "the range of automated assessments available are impressive".

Students were also asked "Would you prefer a mixture of different assessment methods? (i.e. fill-in-multiple-blanks, jumbled-sentences and programming from scratch)", 60% responded "Yes". The survey also asked if the students think they should be involved in the design of their assessment methods. 80% of the students answered "Agree" and "Strongly Agree".

## VIII. Conclusions and Future Work

The paper presents the use of learning modules, instantaneous feedback and automated marking in addressing the three feedback questions 1) Where am I going?, 2) How am I going?, and 3) Where to next?. The implementation outcome indicates positive changes in students' studying behaviour. iFaME allows students to revisit their prior knowledge, organise their knowledge and eventually develop their programming mastery, which is in line with the learning module discussed in section IV. Students appear to be more engaged with the learning materials as well as having a better performance based on their summative assessment results.

According to the qualitative survey, the feedback from students is also largely positive. Benefits for students include enhanced learning experience through active-learning, improved accuracy of students' problem solving abilities in programming, feedback is provided more efficiently and therefore increases the number of opportunities where students can tackle a variety of programming exercises. Due to the success of this pilot study, the technique is now being implemented on a second year JAVA programming module. Many of the students indicated their excitement towards

learning programming even when that was initially their main worries having had no previous programming experience. It is the authors aim to develop an in-house software suite specifically to implement this iFaME platform to enhance the features available and to include additional programming languages such as C++ and C#. The ease of use and design features help clarify tasks for both staff and students.

Specific investigative objectives for this in-house software suite include optimal design iFaME workbenches for both staff and students, the viability of automatically generating iFaME based programming tasks, coursework or examinations by adding certain common mistakes to each model-answer codes, and the possibility of applying iFaME in other STEM subjects, such as engineering, and mathematics.

From the teacher's assessment and feedback workload perspective, having developed and applied iFaME for two consecutive years, it was apparent that the workload was significantly reduced in the second year. This was most apparent in the reduction of time taken to provide feedback.

## REFERENCES


[1] D. Wiliam, "Assessment: The Bridge between Teaching and Learning," Voices from the Middle, vol. 21, no. 2, December 2013.

[2] J. Hattie, and H. Timperley, "The Power of Feedback," Review of Educational Research, vol. 77, no. 1, pp. 81-112, March 2007.

[3] K. Weston-Green, and M. Wallace, "A method of providing engaging formative feedback to large cohort first-year physiology and anatomy students, " Adv Physiol Educ, vol. 40: 393–397, 2016.

[4] T. Zehra, M. Tariq, S. K. Ali, A. Motiwala, J. Boulet, "Challenges of providing timely feedback to residents: Faculty perspectives," Journal of Pakistan Medical Association, 65(10), 1069-1074, 2015.

[5] J. W. Robinson, and W. B. Crittenden, "Learning Modules: A Concept for Extension Educators?," Journal of Extension, 35-44, 1972.

[6] S. M. Brookhart. How to Give Effective Feedback to Your Students: Alexandria, ACDC, 2008.

[7] Write Now, "Assessment, Feedback and Marking Practices," no date. [Online]. Available at: http://www.writenow.ac.uk/wp-content/uploads/2010/12/writenowguide_assessment-designmarking-and-feedback_final.pdf. [Accessed 18 August 2018].

[8] D. Copping, "Eliminating Unesccary Workload Around Marking," March 2016. [Online]. Available at: http://2fv5d843v9w22sxtto1ibxtu-wpengine.netdna-ssl.com/wpcontent/uploads/2016/03/Marking_report_240316.pdf. [Accessed 18 August 2018].

[9] V. Elliott, J. Baird, T. N. Hopfenbeck, J. Ingram, I. Thomson, N. Usher, M. Zantout, J. Richardson, and R. Coleman, "A Marked Improvement? A Review of the Evidence on Written Marking," April 2016. [Online]. Available at: https://educationendowmentfoundation.org.uk/public/files/Publications/EEF_Marking_Review_April_2016.pdf. [Accessed 18 August 2018].

[10] H. Jones, A. Bavage, A. Gilbertson, M. Gorman, J. Lodge, K. Philips, and K. Yeoman, "Increasing the Quality of Feedback on Assignements while Altering Student Perceptions of Good Feedback based on Their School Experience," June 2009. [Online]. Available at: https://www.heacademy.ac.uk/system/files/jones_final_report.pdf. [Accessed 18 August 2018].

[11] Curtin University, "Assessment and Student Progression Manual: Principles and Requirements," no date. [Online]. Available at: https://clt.curtin .edu.au/local/downloads/assessment/Feedback.pdf. [Accessed 18 August 2018].

[12] J. Wilson and G. Andrada. "Using Automated Feedback to Improve Writing Quality: Opportunities and Challenges," Handbook of Research on Technology Tools for Real-World Skill Development, pp.678-703, 2016.

[13] M. Liu, Y. Li, W. Xu, and L. Liu. "Automated Essay Feedback Generation and Its Impact on Revision," IEEE Transactions on Learning Technologies, Vol. 10 (4), 2017.

[14] E. Cotos, "Potential of Automated Writing Evaluation Feedback," 2011. [Online]. Available at: https://pdfs.semanticscholar.org/8c62/9ecefde59520845205 201df89c434615ea2e.pdf. [Accessed 18 August 2018].

[15] R. Singh, S. Gulwani, A. Solar-Lezama, "Automated Feedback Generation for Introductory Programming Assignements," PLDI, June 2013.

[16] H. Keuning, J. Jeuring, and B. Heeren, "Towards a Systematic Review of Automated Feedback Generation for Programming Exercises," ITiCSE, July 2016.

[17] H. Blau, "Automated Style Feedbacl for Advanced Beginner Java Programmers," PhD Thesis, University of Massachusetts, 2015.

[18] Extend, "Teacher for Learning," no date. [Online]. Available at: https://extend.ecampusontario.ca/wp-content/uploads/teacher_for_learning_module.pdf. [Accessed 18 August 2018]

[19] J. P. Hausknecht, C. O. Trevor, and J. L. Farr, "Retaking Ability Tests in a Selection Setting: Implications for Practice Effects, Training Performance, and Turnover ," Journal of Applied Psychology, Vol. 87, No. 2, 243–254, 2002.