A Customized and Automated Assignment Management and Marking System for Evaluating Student Performance in the STEM Disciplines

1st Ashkan Shokri
Bureau of Meteorology
Melbourne, Victoria, Australia
Ashkan.Shokri@bom.gov.au

2nd Veronica Halupka
Faculty of Engineering
Monash University
Clayton, Victoria, Australia
Veronica.Halupka@monash.edu

3rd Valentijn R.N. Pauwels
Department of Civil Engineering
Monash University
Clayton, Victoria, Australia
Valentijn.Pauwels@monash.edu

***Abstract*—This is an Innovative Practice, Work-in-Progress paper focusing on automated marking. The increase in student numbers and decrease in exam performance for a course taught at Monash University, CIV3204, Engineering Investigation (introduction to statistics), has forced the responsible academic to restructure the assessment for this course. A system has been developed to automatically generate and mark individualized assignments for the students. A wide range of questions is possible, including questions of stochastic nature, for which no exact answer exist. Detailed feedback is provided for each student. The system works on all operating systems (Windows, Linux, and Mac), and has received a very positive student evaluation. It has also led to a strongly increased performance in the final exam. This paper presents an overview of the functioning of the system, the results and impact of its implementation, and the lessons that were learned.**

***Index Terms*—educational technology, electronic learning**

I. Introduction

Marking of assignments and exams is a very labor-intensive task and leads to signifant costs [1]. For this reason, efforts towards the development of automated marking systems have been made for decades, with the first attempts focusing on marking computer codes [2]. An overview of the currently available systems for automated computer code marking is provided by [3], who concluded that two issues still remain to be solved: limitations regarding the operating system, and improving the quality of the feedback. Another more recent field in which automated marking has received significant attention is the marking of essays. An overview of the currently available software is provided by [4]. Remaining issues to be solved are the lack of sense of the rater as a person, the potential of deception of the system so it gives a wrong mark, and the limited ability to assess the creativity of ideas and an evaluation of their practicality.

A number of attempts have also been made to automate the marking of assignments in the exact sciences. Already in 1999, [5] developed a web-based system to automatically generate and mark homeworks for statistics courses. A popular method for automatic marking in the exact sciences is Machine Learning. [6] provide an overview of the application of this methodology in the assessment of scientific assignments. Their major conclusion is that machine learning significantly improved the automaticity of examining and scoring complex constructions such as explanation, argumentation, scientific inquiry and problem-solving, and is thus promising for next generation science assessments. From this overview, it is clear that automated marking is a field that will continue to evolve in the future.

At Monash University, CIV3204 (Engineering Investigation), an introduction to statistics course, has faced a number of challenges over the last few years. The number of students increased from 151 in 2013, to 475 in 2019. It should be noted that these numbers refer to the students at the Clayton campus, while the course is simultaneously taught at the campus in Kuala Lumpur, Malaysia, adding roughly another 20% to the student numbers. In 2020 the amount of students enrolled in the Clayton and Kuala Lumpur campuses has moderated to 350 and 66, respectively, and the enrollment numbers are expected to remain at this level. This increase in student numbers has been accompanied by a decrease in the performance of the cohort for the final exam. Anecdotal evidence suggests that cheating in the continuous assessment can be an explanation. This has forced the responsible academic to drastically restructure this continuous assessment.

This paper focuses on the development of an automated assignment generating and marking system for this course. More specifically, a number of requirements had to be met. First, each student must work with individualized data. It has already shown that this reduces the amount of cheating in the answering of assignment questions [7], [8], which is becoming more important due to the unsupervised assessment caused by COVID19. Second, the system must allow a range of question types, including questions of stochastic nature, which inherently include uncertainty in the answers. Third, the students must be able to submit their answers in a user-friendly manner. Fourth, the system must work through the e-learning system, more specifically MOODLE. Fifth, the

We wish to thank the Department of Civil Engineering at Monash University for providing the funding for this project.

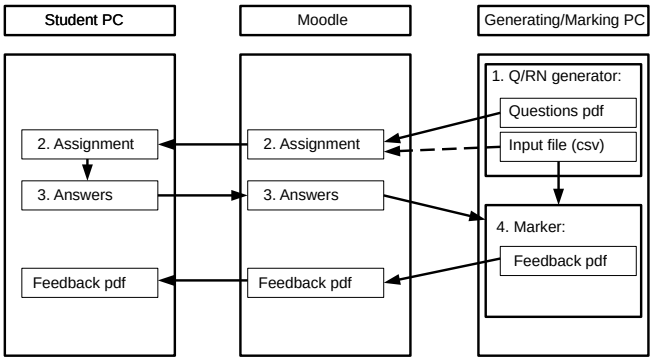

Fig. 1. The workflow of the system. Q/RN stands for question and random numbers. The dashed line implies that input files are only provided to the students if they are necessary, for example if the students need data to perform operations with.

system must lead to challenging questions, and not provide direct information on how to solve the problems. Finally, sixth, the system must work on all platforms (more specifically Windows, Mac, and Linux). This paper describes this system, and the lessons that have been learned from its first application. An overview of the results of the implementation of the system, including an evaluation by the students and an analysis of the impact on their final examination performance, is also provided.

II. Organization of the System

A. The Workflow

The entire system has been coded in python. This ensures it can work on all platforms. The system has been tested on Windows and Linux, and no issues have been found. Figure 1 provides an overview of the organization of the system. The overall principle is that all information between the generator and marker on the instructor side and the students on the other side is passed through the electronic learning system, MOODLE. As a first step, for each student, the questions, random numbers, and input files are generated. The input files contain the random numbers that are part of the question, and any data that the students need in order to answer the question. If the students do not need the input files, they are not passed on to them. For example, a question on probability can be constructed using different random numbers, which can all be in the pdf with the questions. In this case the students do not need the input file with the random numbers. On the other hand, students can be asked to calculate a sampling distribution of a specific data set. In this case they will need the input file.

It is important to note that the random number generator is a pseudo-random number generator, using a seed that depends on the student ID number. This has the advantage that the same random numbers are obtained for each student every time the questions are regenerated.

The system generates a directory for each student, following the conventions by MOODLE. These directories can then be zipped into one file, which can then be uploaded on MOODLE. Figure 2 shows an example of the files that are located inside each of these directories, and that are passed to the students. These files include the pdf with the assignment questions, a Graphical User Interface for each question (see Section II-C), and the input file for each question (if this is necessary).

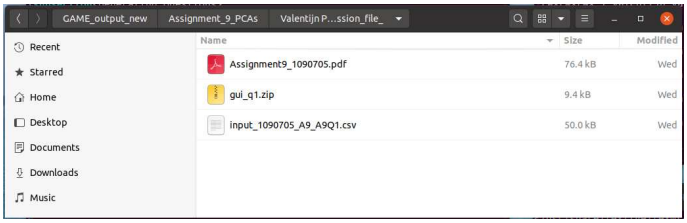

Fig. 2. The files provided to the students in their feedback boxes on MOODLE. The number 1090705 is the student identification number.

The students can then download their assignment files and enter the answers in the Graphical User Interface, which generates excel files following the same naming convention as the input files (Figure 2). The students need to upload these excel files to MOODLE. The submissions of all students can then be downloaded as one zip file. The marker then marks the submissions, and provides feedback in a single file for all questions for each student. These feedback files are then uploaded to moodle, and can be downloaded by the students.

B. Generating Questions

In order to generate a question, two subroutines must be modified. The first subroutine that needs to be modified is input_maker, in which the random numbers are initialized. These could be, for example when doing hypothesis testing, the sample sizes, the means and standard deviations of the samples, and the confidence level. The upper and lower limits for these random numbers need to be specified, as well as the distributions from which they are drawn. The second subroutine that needs to be modified is tex_maker, which uses these random numbers to write the question to be solved by the students. This question is written as a LaTeXstring. The system then uses these subroutines to generate the pdf that lists the questions for the students. These two subroutines can easily be developed by copying example subroutines.

C. The Graphical User Interface

The next step is setting up the Graphical User Interface (GUI) for the students. Two short python codes need to be modified for this purpose. The first is structureMaker.py, which generates a yaml file with the structure of the GUI. Essentially, this program can be set up by copying from example programs, and lists, for each row in the GUI, which variable needs to be entered, and how it should be entered (from a drop-down menu or by manually entering the number). The second program that needs to be modified is the program for the GUI itself. This program specifies the assignment and question number, and the order in which the variables entered by the students must be saved, and can also be developed by copying from example programs. Figure 3 shows the resulting GUI, which the students must then use to enter their results.

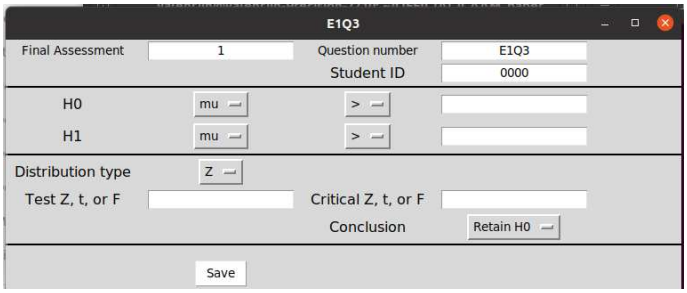

Fig. 3. The Graphical User Interface through which the students can enter their results.

D. Marking Questions

This is the part of the system in which the user has the most freedom. Three subroutines must be modified here.

1) *input_loader*: Here the inputs written in the csv file (for example, the random numbers described in Section II-B) are read in and stored in a vector (or matrix) "inputs".
2) *result_loader*: Here the results from the students are read in, and entered in the vector or matrix "results".
3) *marker*: This subroutine uses the vectors or matrices "inputs" and "results", and generates a LaTeXstring "feedbacks" and a number "mark". As the name suggests, "feedbacks" contains the feedback that is written to the feedback file for the student. "mark" contains the student's mark for this specific question. In this subroutine, the user needs to calculate the correct answers to the question, compare the student's answers to the correct answers, and calculate the mark for the question and generate the feedback.
   It should be noted that the system makes consequentially marking questions very easy. If an intermediate error when solving the question is made, the system can calculate the pseudo-correct answer, compare the student's response to this number, and assign a lower mark.

III. Types of Questions Enabled by the System

The most straightforward questions are questions that require simple calculations. The marker can compute the correct answer, compare the correct answer to the student's response, and mark the answer as correct or wrong using a specified tolerance. This type of questions includes hypothesis testing, analysis of variances (ANOVA), regressions, etc. Other straightforward questions are multiple choice questions, which can be developed using the dropdown menu option.

The interesting aspect of the system is that it can also work with questions that are stochastic in nature, and thus must take into account uncertainty in the answers. One such question is the calculation of a sampling distribution. For this purpose, the students were provided an individualized data set, from which they had to calculate the sampling distribution of the mean, using a sample size of three, and 10,000 repetitions. They were then provided six different sampling distributions, of which one was the correct one. The number of the correct plot and the five other plots were randomized as well.

Another example of a stochastic question is the generation of time series following a specific model. For this purpose, the students were provided with measured river discharge data, from a different station for each student. The students then had to check the validity of the autoregressive (AR), moving average (MA), and autoregressive moving average (ARMA) models. They then had to generate time series following an AR model of order 2, and an ARMA model of orders 1 and 1. The system had checked beforehand that these models were valid for the data that were provided to the students. The students then had to calculate the autocovariances with lag 1 and 2 and the variance of the generated time series. When marking these values entered by the students, the system computed 10,000 time series of the same length, calculated the same statistics, and stored the lowest and highest values. A safety factor of 0.1 times the difference between the maximum and minimum was then applied, and if the values entered by the students were within this range, the answer was marked as correct.

IV. Lessons Learned

By operationalizing the system for the course in the second semester of 2020, a number of lessons could be learned. Nine assignments were generated, leading in total to 25 questions.

A first lesson was that the marking system needed to be made more robust with respect to the numbers entered by the students. Even though it was made very clear to the students to only enter numbers in the GUI cells, and no letters or special characters, in the beginning of the semester this request was consistently ignored. The marker crashed if a string was being read in while a number was expected. A short subroutine was then written that checked, for each cell where a number was expected, if a number was entered. If it was not, the answer was assigned the value -9999 and the answer was marked as incorrect.

A second lesson that was learned was that a number of students kept on modifying the names of the files generated by the GUI's (one student did this up till assignment 4), or uploading the files in a zip file, even though the students were clearly instructed to upload the excel files unzipped without changing the file names. Another minor problem was that students occasionally made errors entering their student ID number in the GUI (see Figure 3). These issues caused the marker not to find the files, and consequently return a mark of zero. A short code was then written to list the missing files for each student. The zip files were then manually unzipped, and the file names with errors manually corrected.

V. Evaluation of the System

The implementation of the system has led to very positive results for the course. Table I shows an overview of the Student Evaluation of Teaching and Units (SETU) for the Clayton campus. SETU is the standard way of evaluating units in Australia, and is an anonymous questionnaire filled out by the students. Of the 350 students enrolled, 67 students participated in the evaluation. The result for the overall satisfaction question is the highest for the eight years in which the responsible academic

TABLE I
STUDENT EVALUATION OF TEACHING AND UNITS (SETU) FOR THE COURSE FOR THE CLAYTON CAMPUS.

Question	Responses	Median	%Strongly Agree or Agree
University Wide Items (Summary)			
The Learning Outcomes for this unit were clear to me	67	4.01	79.10%
The instructions for the assessment tasks were clear to me	67	4.03	76.12%
The Feedback helped me achieve the Learning Outcomes for the unit	67	3.90	70.15%
The Resources helped me achieve the Learning Outcomes for the unit	67	4.01	79.10%
I attempted to engage in this unit to the best of my ability	67	4.22	86.57%
Overall, I was satisfied with this unit	67	3.95	74.63%
Faculty Wide Items (Summary)			
The assessment tasks helped me to develop the knowledge and skills required for this unit	64	4.10	82.81%
I understood the grading criteria used in assessing my work	67	3.89	68.66%
This unit contained a good mix of theory and practical	67	3.89	68.66%
The Moodle site was engaging and enhanced the learning experience	67	3.97	74.63%
The lectures were valuable for my learning	67	4.02	74.63%

taught the course. Important as well are the results of the qualitative analysis. The first quesion is "Which aspect(s) of this unit did you find most effective?" 42 Students answered this question, and 28 students stated they appreciated the nine relatively short assignments, and four students stated clearly that they appreciated that this forced them to keep on track with the course.

The second question is: "Would you suggest any changes to enhance this unit in the future?", which 44 students answered. Seven students replied that the setup of the GUI's could be improved. Improvements to the setup of the GUI's will be made next time the course is taught, in the second semester of 2021. Only five students replied to the third and final question, "Comments". No remarks regarding the system were made.

Another advantage is that student complaints regarding unprepared tutors have disappeared. A week earlier than for the students, an individualized assignment was also generated for the tutors, which they also had to generate the answers for, and for which they also were marked. This forced the tutors to prepare themselves for the tutorials.

A final advantage of the system was that it led to a strongly improved performance of the students in the final exam, even though this was more difficult than in the previous years. For the students enrolled in Clayton in 2019, 107 out of the 475 did not pass the course. In 2020, after the implementation of the system, 40 out of the 350 students enrolled in Clayton failed the course. The failure rate thus decreased from 22% to 11%. One explanation, which is suggested by the answers to the SETU questions, is that the individualization of the assignments has forced the students to do them, and consequentially they were better prepared for the exam.

VI. CONCLUSION

Because of a strong increase in student enrollment and a decreasing student performance in the final exam, an automated individualized assignment generation and marking system has been developed for CIV3204 (Engineering Investigation) at Monash University, an introduction to statistics course. The system is programmed in python and works on all operating systems (Linux, Windows, and Mac). It is very flexible and allows a wide range of questions, including stochastic questions for which no exact answer exists. It has received very positive feedback from the students, and has also led to an increased performance in the final exam. For these reasons, the system will be implemented and improved for the course in 2021.

ACKNOWLEDGMENTS

We wish to thank the Department of Civil Engineering at Monash University for providing the funding to develop this system. We express our special gratitude to the Doctoral Teaching Assitant, Filippo Nelli, and the Practice Class Assistants, Aditya Gupta, Amin Assadzadeh, Aria Moradshahi, Bakab Khadivi, Chathurika Jayasundara, Chi Nguyen, Jihane Elyahyioui, Marcela de Freitas Silva, Milad Bazli, Mobakareh Mohammadpour, and Sobhan Hatami for their continuous support throughout semester 2, 2020. We also thank the unit coordinator in the Monash Malaysia campus, Pat Yeh. We are also thankful to Mayer Melham and Colin Caprani for their help with the transfer of the files to and from MOODLE.

REFERENCES

[1] A. Vista, E. Care, and P. Griffin, "A new approach towards marking large-scale complex assessments: Developing a distributed marking system that uses an automatically scaffolding and rubric-targeted interface for guided peer-review," *Assessing Writing*, vol. 24, pp. 1–15, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.asw.2014.11.001

[2] W. Fleming, K. Redish, and W. Smyth, "Comparison of manual and automated marking of student programs," *Information and Software Technology*, vol. 30, no. 9, pp. 547–552, 1988.

[3] H. Aldriye, A. Alkhalaf, and M. Alkhalaf, "Automated grading systems for programming assignments: A literature review," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 3, pp. 215–221, 2019.

[4] M. A. Hussein, H. Hassan, and M. Nassef, "Automated language essay scoring systems: A literature review," *PeerJ Computer Science*, vol. 2019, no. 8, pp. 1–16, 2019.

[5] D. W. Stockburger, "Automated grading of homework assignments and tests in introductory and intermediate statistics courses using active server pages," *Behavior Research Methods, Instruments, and Computers*, vol. 31, no. 2, pp. 252–262, 1999.

[6] X. Zhai, Y. Yin, J. W. Pellegrino, K. C. Haudek, and L. Shi, "Applying machine learning in science assessment: a systematic review," *Studies in Science Education*, vol. 56, no. 1, pp. 111–151, 2020. [Online]. Available: https://doi.org/10.1080/03057267.2020.1735757

[7] K. B. Menk and S. Malone, "Creating a cheat-proof testing and learning environment: A unique testing opportunity for each student," *Advances in Accounting Education: Teaching and Curriculum Innovations*, vol. 16, pp. 133–161, 2015.

[8] S. Manoharan, "Personalized assessment as a means to mitigate plagiarism," *IEEE Transactions on Education*, vol. 60, no. 2, pp. 112–119, 2017.