Auto-Grading Jupyter Notebooks

Hamza Manzoor
mhamza1@vt.edu
Department of Computer Science,
Virginia Tech
Blacksburg, VA

Amit Naik
amit94@vt.edu
Department of Computer Science,
Virginia Tech
Blacksburg, VA

Clifford A. Shaffer
shaffer@vt.edu
Department of Computer Science,
Virginia Tech
Blacksburg, VA

Chris North
north@cs.vt.edu
Department of Computer Science,
Virginia Tech
Blacksburg, VA

Stephen H. Edwards
edwards@cs.vt.edu
Department of Computer Science,
Virginia Tech
Blacksburg, VA

ABSTRACT

Jupyter Notebooks are becoming more widely used, both for data science applications and as a convenient environment for learning Python. Currently, grading of assignments done in Jupyter Notebooks is typically done manually. Manual grading results in students receiving feedback only long after the assignment is complete. We implemented support for auto-grading programs written in Jupyter Notebooks within the Web-CAT auto-grading system. Scores received are directly reported to the Canvas gradebook. A Jupyter notebook extension allows students to upload their notebook files to Web-CAT directly. Survey results from class use show that 80% of students believe that getting immediate feedback from Web-CAT improved their performance. Instructors report that this implementation has significantly reduced their workload.

KEYWORDS

Learning tools interoperability, auto-grading, Jupyter notebooks, Python

ACM Reference Format:
Hamza Manzoor, Amit Naik, Clifford A. Shaffer, Chris North, and Stephen H. Edwards. 2020. Auto-Grading Jupyter Notebooks. In *The 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11–14, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3328778.3366947

1 INTRODUCTION

Many Computer Science courses would benefit by giving students the ability to integrate code editing and execution, data visualization, equations, and prose explanations. Jupyter Notebooks provide a platform where all these different tasks can be combined in one place. Instructors can provide self-contained notebooks with all the instructions and starter code in one place [3]. Jupyter notebooks are gaining popularity, especially for data science [5] and are a common choice for many data science courses and for Python programming courses [7].

Instructors can create notebooks and distribute them to students through a Learning Management System (LMS) such as Canvas. In this context, students easily download the assignment, write their solutions, and submit their files back to the LMS. However, the LMS cannot then provide auto-grading or assessment services. The instructor or teaching assistants must download the submissions and manually grade them. They also have to manually enter the points scored by each student back to the LMS. In this workflow, feedback can be provided to students only after the assignment is complete [1].

Our aim is to provide auto-grader support for Jupyter notebook assignments, thereby giving immediate feedback to students on their submissions as they progress through the assignment. Auto-grading notebooks can also reduce the workload for instructors and TAs.

2 CHOOSING AN AUTO-GRADER SYSTEM

Many auto-graders exist for programming assignments, such as Web-CAT [2], BOSS [6], CourseMaker [4] and Bottlenose [9]. Previously, none of these supported auto-grading of Jupyter Notebook assignments. Nbgrader [3] was developed by the creators of Jupyter Notebooks to auto-grade Jupyter Notebook assignments.

Instructors can use nbgrader's "formgrader" extension or the command line to create an assignment. During assignment creation, instructors can define both auto-graded and manually graded cells in the notebook. They have to provide the solution code and unit tests for the auto-graded cells. Finally, an instructor will generate a student version of the assignment that does not include solution code and unit tests. When a student completes their assignment, they upload their solution files to their LMS. An instructor or teaching assistant downloads those files and moves them to the "submissions" directory to auto-grade those assignments through the formgrader extension or nbgrader commands. However, nbgrader does not submit scores to an LMS, nor does it provide automated feedback to students. It does generate feedback for students, but this has to be manually delivered to each student, and therefore, it only happens after the final submission.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGCSE '20, March 11–14, 2020, Portland, OR, USA
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-6793-6/20/03…$15.00
https://doi.org/10.1145/3328778.3366947

Web-CAT [2] is an auto-grading system with support for assignments in various programming languages. Instructors create support for an assignment in Web-CAT, and separately creating the assignment definition in their LMS. Adding the resulting Canvas assignment URL to the Web-CAT assignment definition links these two together. When students are done with their assignment, they can use a plugin in their programming environment (such as Eclipse) to directly upload the assignment to Web-CAT. Web-CAT shows them feedback right away and reports the score directly to Canvas.

Our initial plan was to modify nbgrader to support the LTI protocol [8], which would have helped in sending scores to the LMS after auto-grading. However, adding LTI support does not solve the remaining shortcomings of nbgrader. Teachers would still have to collect the assignments, move them to the nbgrader *submissions* directory, and manually share feedback with students. Web-CAT already has the ability to send scores to Canvas (via LTI), and provides immediate feedback to the student (via a web page). Web-CAT also offers other relevant features such as giving extra credit for early submissions and defining rules for late submissions. Therefore, supporting Jupyter notebook auto-grading in Web-CAT would provide access to all of these features. Since Web-CAT provides an API for submission of assignments, we were able to create a Jupyter Notebook extension to upload notebooks directly to Web-CAT. The limitation of Web-CAT is that the only LMS that it supports sending scores to right now is Canvas. Scores still have to be manually exported to any other LMS.

3 AUTO-GRADING JUPYTER NOTEBOOKS WITH WEB-CAT

As discussed in the last section, we decided to use Web-CAT to auto-grade and provide immediate feedback to students. We implemented functionality within Web-CAT to handle Jupyter's "ipynb" file format. Nbgrader is used during assignment creation, because nbgrader provides a toolbar through which instructors can select the type of each cell within the notebook. Instructors mark the cells with different options such as *Autograded Answer* and *Autograded Tests*. The toolbar also allows associating points to the *Autograded Tests* cells.

3.1 Preparing Jupyter Notebook Assignments

The first phase of this project involved adding the functionality in Web-CAT to support auto-grading of Jupyter notebooks. The instructor creates a Notebook assignment, which includes solution code and unit tests (which is which is indicated by using the nbgrader toolbar). Web-CAT automatically generates a student version of the assignment by removing the solution code and unit tests from the notebook. The student version of the notebook file can be downloaded from Web-CAT and distributed to students through the LMS. The following steps are performed in Web-CAT once the instructor uploads the Jupyter Notebook file.

(1) Insert a new cell to the front with the assignment description info for the extension toolbar. This is described in detail in the next section.
(2) Strip out any auto-grading test cells
(3) Strip out any solution blocks and replace them with *# Insert your work here* comments.
(4) Strip out any output content from code cells, since they might be from the instructor solution.

Figure 1 shows an auto-graded cell generated from the nbgrader toolbar. Web-CAT removes the code written between *### BEGIN SOLUTION* and *### END SOLUTION* and replaces these comments with *# Insert your work here* comments. Figure 2 shows the auto-graded test cell that is removed from the student's version. These tests are used for validating student submissions.

Figure 1: nbgrader cell indicating auto-graded solution cell.

Figure 2: nbgrader cell indicating auto-graded tests.

Once students complete their assignments or projects and submit them to Web-CAT, Web-CAT runs the tests which were stripped from the student's version and calculates points. A message is displayed to students when they fail a particular unit test.

3.2 Jupyter Notebook Extension

When a student wants their assignment evaluated, they must upload it to Web-CAT. One way to upload an assignment is to log in at the Web-CAT website, navigate to the assignment page, and upload submission files manually. The score and other feedback is provided directly in the browser. This process must be repeated for each submission. As an easier alternative, we created an extension that allows a student to submit their assignment from within the Jupyter environment. This process directly uploads the file to Web-CAT, and opens a browser window to show Web-CAT's feedback.

Web-CAT provides an internet API to support programmatic submission of assignments. The API requires the following three parameters along with the submission files:

(1) **course:** Unique identifier for a course in Web-CAT.
(2) **a:** Web-CAT's name for the assignment.
(3) **d:** Name of the institution that this course belongs to.

The Jupyter Notebooks system is installed locally on the user's computer, and consists of two major components: A browser-based interface (using HTML and JavaScript), and a python-based application that is known as "the server". We created a Jupyter extension in two corresponding parts, a front-end extension, and a server extension. The front-end extension (written in JavaScript) modifies the default Jupyter notebook interface by adding a submission button on the notebook. However, services such as access to the local file system are provided by Jupyter's python-based server, not in the notebook interface. Hence, our front-end extension cannot access the assignment file and post it directly to Web-CAT. Therefore, the second part of our extension is written in Python, as part of the

Jupyter server. The front-end extension sends a request to the server extension, which then fetches the file from the file system and posts it to Web-CAT with the necessary parameters as described above.

3.2.1 Front-end Extension: The front-end extension adds a button labeled "*Submit to Web-CAT*" at the top of a Jupyter notebook, as shown in Figure 3. The assignment is submitted to Web-CAT when the user clicks on this button.

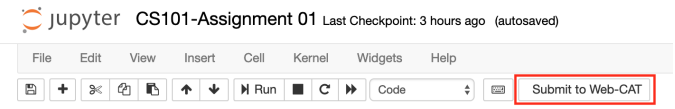

Figure 3: Submit to Web-CAT button

The three parameters required by the Web-CAT API (course, assignmentID, institutionID) are added as comments in the first cell of the assignment notebook, as shown in Figure 4.

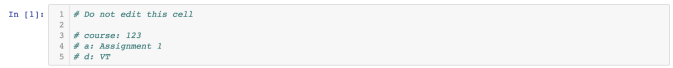

Figure 4: Web-CAT assignment parameters in the first cell of a notebook

The front-end extension reads this first cell and parses these three parameters. If these parameters are not present, the extension throws an error message, as shown in Figure 5.

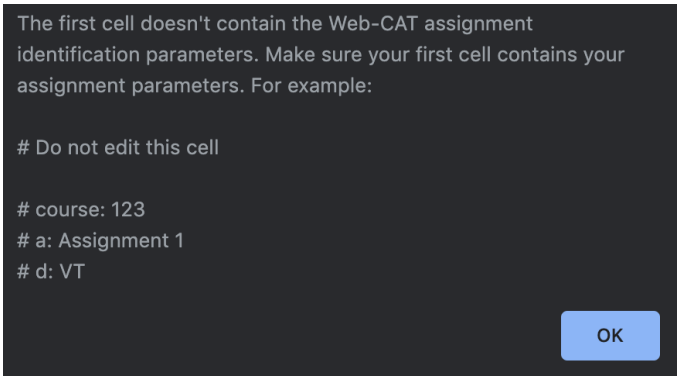

Figure 5: Error message shown when Web-CAT assignment parameters are not present in the first cell

3.2.2 Server Extension: The Jupyter server component of our extension creates a URL and constantly listens for requests sent to this URL. The front-end extension sends the file path and Web-CAT parameters to this URL. When the server extension receives the request, it reads the parameters from the received request and fetches the student's notebook file from the local file system. It then sends the request to the Web-CAT API with the notebook file, along with the assignment identification parameters. Web-CAT sends a response back with a URL on which the scores and feedback can be viewed. The server extension sends the response containing this URL back to the front-end extension running in the Jupyter notebook interface. The front-end extension opens this URL in a pop-up browser window. Figure 6 shows the students' output.

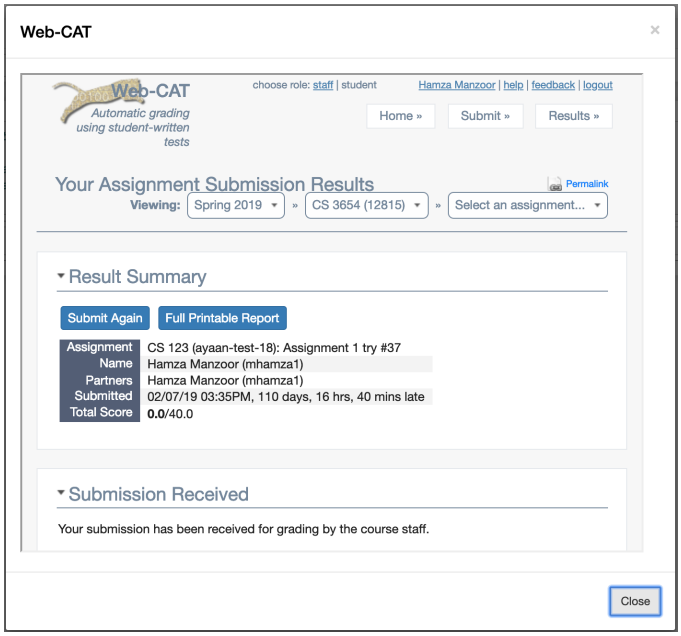

Figure 6: Pop-up showing the scores received on the assignment.

3.3 Putting it all together

Initially, instructors create their assignment notebook, including the solution code and unit tests. They mark the cells containing the solution code and unit tests using the nbgrader toolbar's cell types *Autograded Answer* and *Autograded Tests*, respectively. Then, they create an assignment in Canvas. If one does not already exist, the instructor will have to make an entry for the course at Web-CAT. They then create an assignment within that course by filling out a form where they select *JupyterPlugin* for the assignment type, and provide the Canvas assignment URL.

Web-CAT will take the completed assignment Jupyter Notebook and generate a student version that does not include solution code or reference unit tests. The instructor can download this student version from Web-CAT *Browse Raw Files* section. The instructor can then distribute the students' assignment notebook, perhaps through the LMS. When a student completes their assignment, they can upload their submissions using the Jupyter Notebook extension, which will return their scores and associated feedback by opening a browser page. Web-CAT also submits scores directly to the gradebook of the LMS. Web-CAT will allow students to improve their code and resubmit as many times as they like, as long as they are within the deadline of the assignment.

4 RESULTS AND FEEDBACK

Two sections of the course CS3654: Intro Data Analytics & Visualization at Virginia Tech used Web-CAT for auto-grading Jupyter

Notebooks in Spring 2019. We designed two surveys to get feedback on the effectiveness of using Web-CAT for auto-grading Jupyter Notebooks. One survey was designed for students and the other for instructors and teaching assistants. Another section of the same course (CS3654) used Web-CAT for auto-grading Jupyter Notebooks in Fall 2019. We used the same survey to get feedback on the effectiveness of using Web-CAT for auto-grading Jupyter Notebooks.

4.1 Student survey

Our student survey contained 15 questions. In Spring 2019, students were asked whether they preferred continuous feedback through Web-CAT (during the course, they had assignments with and without automated submission and feedback). They were also asked if they believe their performance improved with continuous feedback. The instructor of one section asked students to complete the survey in class, and the other instructor asked students to complete the survey through a Canvas announcement. There were 55 students in each section. We got 99 responses, but 5 responses had incomplete data and were removed from our analysis.

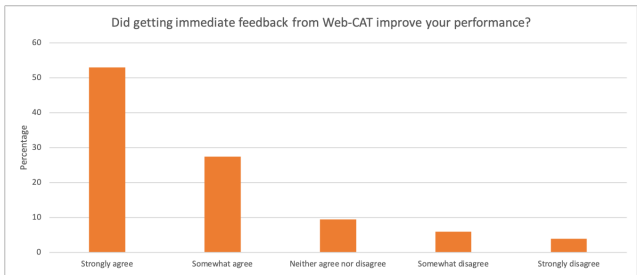

Figure 7: Spring 2019 - Responses to question asking if getting immediate feedback improved performance.

Figure 7 shows the responses (as percentages) when students were asked if getting immediate feedback from Web-CAT improved their performance. Note that students this semester had completed both auto-graded and manually graded Jupyter Notebook assignments, so they had experience on which to make such an opinion. From the survey responses, we can say that 80% of students believe that getting immediate feedback from Web-CAT improved their performance.

This also aligns with our empirical Web-CAT data. Students on average had 11.4 submissions on each homework. Only 3% of students have a single submission. 14% of students made over 20 submissions to improve their scores. One student had 69 submissions on Homework 3 and finally managed to score 100%.

Similarly, 75% of students say that they will recommend future instructors to use Web-CAT for auto-grading and immediate feedback in their courses. Figure 8 shows the distribution of responses on a 10-point scale.

The same survey was provided to students in Fall 2019. Figure 9 shows the responses (as percentages) when students were asked if getting immediate feedback from Web-CAT improved their performance. From the survey responses, we can say that 89.7% of these students believe that getting immediate feedback from Web-CAT improved their performance.

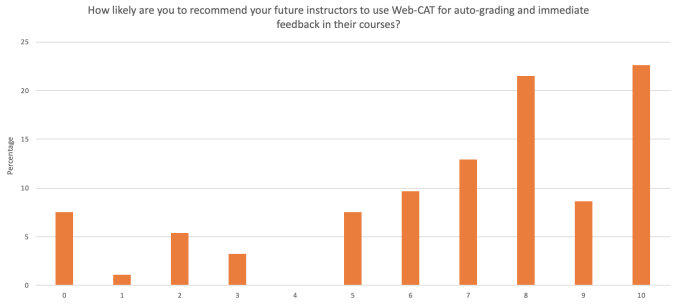

Figure 8: Spring 2019 - Responses to survey question asking students how likely are they to recommend using Web-CAT for Jupyter Notebooks to other instructors.

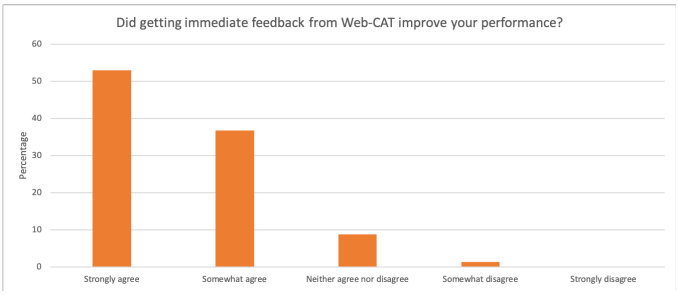

Figure 9: Spring 2020 - Responses to question asking if getting immediate feedback improved performance.

Similarly, 91.17% of Fall 2019 students say that they will recommend future instructors to use Web-CAT for auto-grading and immediate feedback in their courses. Figure 10 shows the percentage of responses on a 10-point scale.

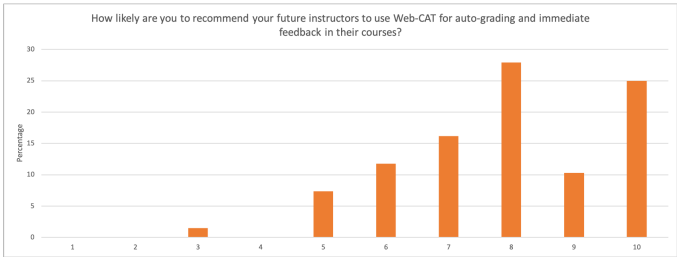

Figure 10: Spring 2020 - Responses to survey question asking students how likely are they to recommend using Web-CAT for Jupyter Notebooks to other instructors.

We also received valuable feedback from free-response questions. Most students say that getting feedback on homework is useful and tremendously improved their scores on assignments, while others say that the feedback they received is too generic and should be more specific.

4.2 Instructor's survey

A 16-question survey was written and given to instructors and Teaching Assistants from this same course. It asked questions such as whether the number of students asking for feedback significantly reduced or not on the auto-graded assignments. There were seven teaching assistants and two instructors. We received responses from five of them.

When asked if they believe that their load has been reduced using Web-CAT for auto-grading Jupyter Notebooks, four of five agreed and one had a neutral response. Similarly, three responses tell us that the number of students asking for feedback was noticeably reduced.

Three instructors/TAs indicated that Web-CAT with the Jupyter extension is better than nbgrader. When asked if they will recommend using Web-CAT for auto-grading Jupyter notebooks to other instructors, all five responses were "Yes".

We also received feedback from instructors in free-response questions. A couple indicated that the process of creating assignments on Web-CAT is cumbersome and the UI should be improved. One instructor provided detailed feedback explaining why Web-CAT is better than nbgrader, but he also suggested that the feedback provided by nbgrader has the advantage that it embeds the feedback directly into the student's notebook format as an HTML page.

One instructor mentioned that a big advantage of Web-CAT's immediate grading feedback was that it helped with ambiguity in the homework specifications. Without Web-CAT, instructors have to write the homework instructions carefully to inform students about the format of their answers, which is difficult in data science exercises where answers might be complex tables of data. Students can easily misunderstand and submit answers in the wrong format and lose points for a non-pedagogical reason. This results in instructors having to deal with many students coming to get corrections to their grades. Whereas, the immediate feedback enables students to see that they have the wrong format immediately, fix it, resubmit, and get full points.

4.3 Comparison with a past course

One of the instructors taught the same course in Fall 2018 without auto-grader and with auto-grader in Spring 2019. 5 homework assignments in both courses were essentially identical except for the auto-grader system. We compared the scores of these homework assignments against one another and 4 out of 5 homeworks had statistically significantly higher mean scores ($p < 0.05$). Table 1 contains the mean scores of homework assignments and the p-values from Student's t-test.

#	Mean nbgrader	Mean Web-CAT	p-value
HW3	58.76	87.23	1.88263E-08
HW4	88.31	95.23	0.016632693
HW5	82.50	90.80	0.028080726
HW6	83.33	88.84	0.100647859
HW8	67.78	90.54	2.28732E-06

Table 1: Mean scores of identical homework assignments in Fall 2018 (without auto-grader) and Spring 2019 (with autp-grader)

5 THREATS TO VALIDITY

We need a better evaluation to compare our system with the existing one (nbgrader). We asked students and instructors to answer survey questions after the end of the course. Only one of the assignments used nbgrader in that course, and only 2 of the 5 instructors had used nbgrader prior to this course. To compare if instructors prefer nbgrader or Web-CAT, we need to have a better comparison. Secondly, we do not have a comparison to show if students performed better with continuous feedback through Web-CAT. We need comparison with a course that did not use auto-grading to compare how students performed on the same assignments when there was no constant feedback.

6 CONCLUSION

Past research tells us that auto-grading is helpful for both students and instructors. Given the popularity of Jupyter Notebooks, providing a solution for auto-grading Jupyter Notebooks is an important step forward. Prior to our work, only one auto-grader (nbgrader) for Jupyter Notebooks was available. Along with other pain points, nbgrader does not provide immediate feedback because the assignments are graded by nbgrader after students have submitted their final version. The scores are also submitted manually by instructors to the LMS. We provide a solution to address these pain points. We implemented the auto-grading of Jupyter Notebooks in the Web-CAT auto-grading system, and provided Jupyter extensions to allow students to upload their notebook files to Web-CAT directly. As a result, students get immediate feedback on their submissions and their scores are directly reported in Canvas. We used this auto-grading system in two sections of a course at Virginia Tech and conducted surveys at the end. The survey results provided positive feedback from both students and instructors. Most students had a large number of submissions on each assignment, which shows that they used the system to get feedback and improved it. Most students indicated that getting immediate feedback from Web-CAT improved their performance. Similarly, instructors (from a sample of five) also said they prefer this system and it has significantly reduced their load. The comparison of similar assignments from two courses also shows a significant increase in average scores on assignments when auto-grader was used to provide immediate feedback. Hence, we can say that using Web-CAT for auto-grading Jupyter Notebook assignments is useful for both instructors and students.

7 FUTURE WORK

Web-CAT provides feedback separate from the original notebook file. Students get their feedback on Web-CAT's website, or in a pop-up if they use the extension to upload notebook files to Web-CAT. Instructors who used both Web-CAT and nbgrader for auto-grading in their courses told us that nbgrader provides better feedback because it embeds feedback directly in the original notebook files. We can try to follow the same approach of providing feedback within a notebook file, but it is not as simple in the case of Web-CAT. Nbgrader provides feedback at the very end when students have

submitted their final version of the assignment. Embedding feedback within the notebook file makes sense in the end. Whereas in the case of Web-CAT, students get feedback right away. They might want to improve their submissions after getting feedback. Therefore, changing the original notebook file through the extension might not be the best approach. We will probably have to create a new notebook file or generate an HTML file of the notebook with feedback embedded in it, as nbgrader does.

Another comment from the instructors is that in the Data Science course, unit tests do not assess all answer types. For example, a student might be asked to generate a table. There is no assert statement to compare the student table with a model answer, so instructors have to write extra code to do this. In the future, we can also provide code samples to illustrate commonly used output formats.

ACKNOWLEDGMENTS

We would like to thank Ayaan Kazerouni for his help in providing the Web-CAT data. This research was partially funded by National Science Foundation grants DLR-1740775 and DLR-1740765, and by Virginia Tech's Technology-enhanced Learning and Online Strategies (TLOS) unit.

REFERENCES

[1] Maha Aziz, Heng Chi, Anant Tibrewal, Max Grossman, and Vivek Sarkar. 2015. Auto-grading for Parallel Programs. In *Proceedings of the Workshop on Education for High-Performance Computing*. 8 pages.

[2] Stephen H Edwards and Manuel A Perez-Quinones. 2008. Web-CAT: Automatically Grading Programming Assignments. *ACM SIGCSE Bulletin* 40, 3 (2008), 328–328.

[3] Jessica B Hamrick. 2016. Creating and Grading IPython/Jupyter Notebook Assignments with NbGrader. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. 242–242.

[4] Colin A Higgins, Geoffrey Gray, Pavlos Symeonidis, and Athanasios Tsintsifas. 2005. Automated Assessment and Experiences of Teaching Programming. *Journal on Educational Resources in Computing (JERIC)* 5, 3, Article 5 (Sept. 2005).

[5] Tony Hirst. 2018. The Growing Popularity of Jupyter Notebooks. https://blog.ouseful.info/2018/09/10/the-growing-popularity-of-jupyter-notebooks/

[6] Mike Joy, Nathan Griffiths, and Russell Boyatt. 2005. The BOSS Online Submission and Assessment System. *Journal on Educational Resources in Computing (JERIC)* 5, 3, Article 2 (Sept. 2005).

[7] Jeffrey M. Perkel. 2018. Why Jupyter is Data Scientists' Computational Notebook of Choice. https://www.nature.com/articles/d41586-018-07196-1

[8] Charles Severance, Ted Hanss, and Josepth Hardin. 2010. IMS Learning Tools Interoperability: Enabling a Mash-up Approach to Teaching and Learning Tools. *Technology, Instruction, Cognition and Learning* 7, 3-4 (2010), 245–262.

[9] Mark Sherman, Sarita Bassil, Derrell Lipman, Nat Tuck, and Fred Martin. 2013. Impact of Auto-grading on an Introductory Computing Course. *Journal of Computing Sciences in Colleges* 28, 6 (2013), 69–75.