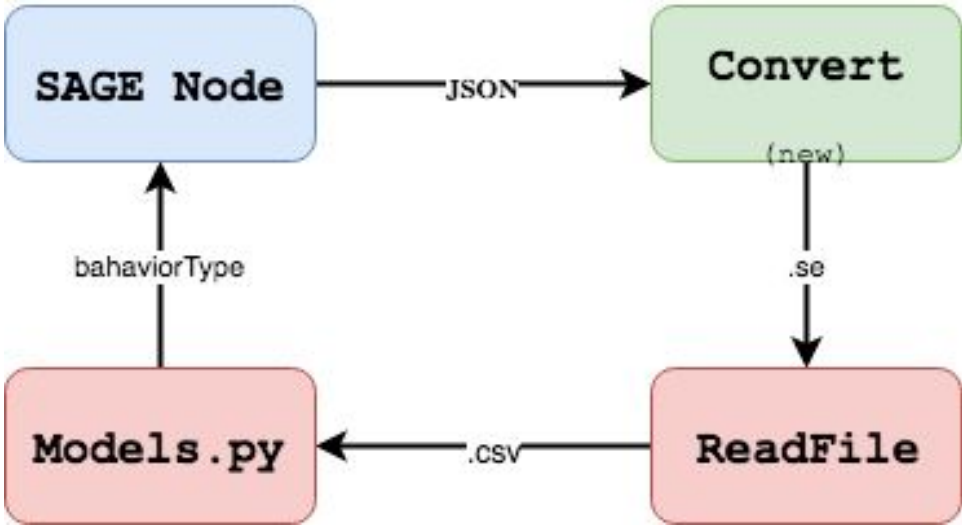
Midterm Report on Behavior Detection

Joe Huang (jch2220)
Wei Han (wh2365)

1. Abstract

This report covers the current progress of Programming Behavior Detection 1.1 under Gameful Intelligent Tutoring. The report primarily discusses the progress in the sage-scratch integration (#290), including its architecture and implementation. The rest of the reports elaborates the possible solutions for replacing local storage with cloud storage of student behavior files to increase scalability for the application.

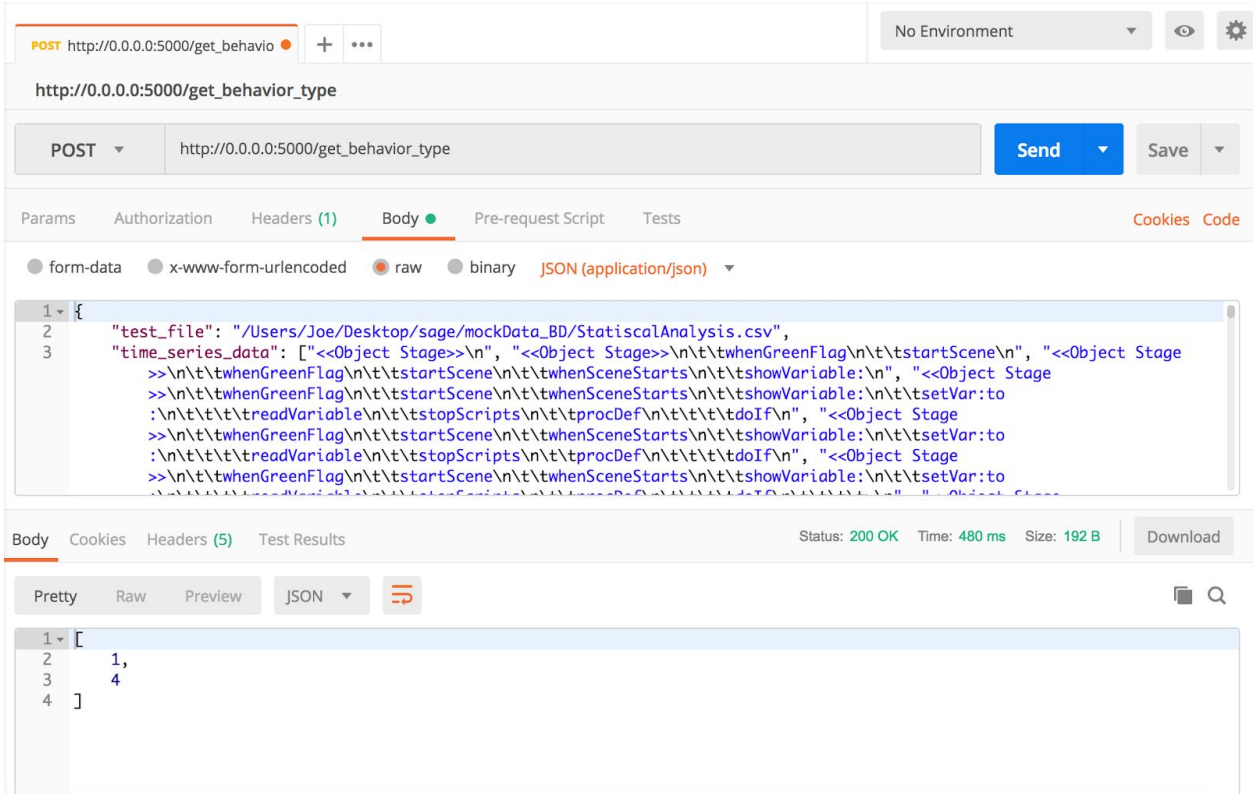
2. Architecture



When the SAGE Node receives a new JSON student behavior file, it passes the JSON file via getBehaviorType API to the new python function we have implemented which simply converts a JSON file to an se file. Then the converted file is passed to the ReadFile.java which generates a csv file of statistical analysis based on the input file. The generated csv file is then passed into Models.py which analyze student learning types based on the input information. At last, SAGE Node receives the latest student behavior type based on the most recently recorded behavior and updates student information in the mLab.

3. Implementation

The described process could be tested by running IntelligentHinting/server_interface/applicaiton.py and send a request using localhost:5000/get_behavior_type along with the json string.



4. Assumptions and Limitations

Currently, we are adapting finished methods from previous semesters, which utilize different languages and complicated data conversions. Besides, all student behavior files are read, processed, generated locally, which requires huge space on the machine when analyzing student behavior types using a large amount of snapshots. Such implementation may not be scalable, inclining to fail when the number of users grows.

5. Future Work

In the future, we propose to add a trigger such that the behavior classification is executed only after student snapshots have accumulated to a certain number k, and a replacement of local storage with cloud storage of student behavior snapshots to increase the scalability of the application.

Student behavior snapshots can stored in mLab, indexed by timestamps. For each student JSON file stored in mLab, in addition to the student's behaviorType, we would add all snapshots IDs collected from the student. Furthermore, in order to avoid using outdated snapshots, we hope to impose a counter on the number of snapshots each individual has on mLab to decide when to run a behavior classification. Whenever the Node receives a new snapshot, it first checks the current count of snapshots associated with the particular student, if the number of snapshots exceeds such certain limit, based on all previous snapshots behavior classification updates the student learning type. The arbitrary limit k would be tested by trial and error.

In addition, we want to implement a simpler workflow such that when the API receives the JSON file, it performs tasks completed currently by ReadFile.java and Models.py, directly outputting the behavior type of the student, to minimize redundancy and the use of local storage.