Grade Me

Authors: Kamran Hussain, Zhaozhong (Alex) Wang, Kaz Nakao
Revision: 5.3.22

<u>Introduction</u>:

We propose a deep learning based, server-client application that autogrades essays. This program consists of 3 sub-applications for each aspect of the pipeline, a student interface, teacher interface, and server all written in Java. The application would include a desktop version for students where they can upload their essays to be automatically graded and can see their previous essays. A component teacher app would also be available where student essay scores can be seen by the teacher. Additionally, the teacher can upload new essay assignments for grading and set a rubric. A third component application would be on the server end. This includes a storage database for essays and loads the BERT (Devlin et al. 2017) binary, which is pre-trained and saved to a pickle file.

Our application tackles the tediousness of grading essays when a lot of the grading elements are redundant. Our target audience is high school students and teachers. By automating grading, students can get rapid feedback allowing them to make corrections and continue learning. This also allows teachers to gather more data on Students progress and modify lesson plans accordingly.

Through BERT, our application uses semantic topic extraction to grade papers. This is essentially creating a mathematical representation of the student document and comparing the meaning of student document sentences to the mathematical representation of the rubric. This allows us to test if the student mentions items outlined in the rubric. If gibberish is inputted, the grader would be able to identify the input essay has no semantic relation to the rubric, and would thus grade the essay accordingly.

<u>Instructions</u>:

Students:

Launch the desktop app and create a new assignment. Input the assignment code provided by your teacher in the new essay dialogue. Next copy and paste your essay into the submission area or upload a text file. Finally, press the submit assignment button, you'll see a dialogue that says your essay is in the grading queue. When the assignment is graded, it will pop up on the app.

Teachers:

Launch the desktop app and add a new assignment. Upload your rubric for the essays to be graded along. Next, share the generated assignment code with the students. As students make submissions and their essays are graded, their grades will show up in the students tab. You can submit essays for regrading if necessary.

Server:

This is not a client configurable pipeline. The server end uses a maven built jar to load the grading model and start the server. When the server is up, its status will show up on the client apps.

<u>Features List (THE ONLY SECTION THAT CANNOT CHANGE LATER)</u>:
Must-have Features:

- Text File sending to and from the server via packets
- Desktop app can upload and store student essays
- Teacher app can see student scores from server
- Teacher app shows score distributions
- Server can load the machine learning binary
- Clients can connect to the server

Want-to-have Features:

- Plagiarism checks against other student essays
- Allow for pdf and docx submissions
- Faster server with parallelized processing across multiple CPUs or GPUs
- Dark mode

Stretch Features:

- Add sentiment analysis and grammar checks to the grading
- Summarize the essays for the teacher
- Submit from google drive

<u>Class List</u>:

DataBase
Classroom
Student
Teacher
Submission

Client app
Main
HomeScreen
StudentScreen
SubmissionScreen
TeacherScreen

Server app
Main
Database
Model Loader
BertClassifier

<u>Credits</u>:

- BERT (Devlin et al. 2017)
- Kamran Hussain
    - Wrote ML model
    - Wrote the code required for the grader
- Zhaozhong (Alex) Wang
- Kaz Nakao
    - Wrote basic classes for database types
    - Created classes for the client application
- DeepLearning4J
- DeepJavaLibrary
- Firebase
- John Shelby
    - Firebase demo