Team Contributions: POC
Software Engineering

Team 2, SyntaxSentinals
Mohammad Mohsin Khan
Lucas Chen
Dennis Fong
Julian Cecchini
Luigi Quattrociocchi

This document summarizes the contributions of each team member up to the POC Demo. The time period of interest is the time between the beginning of the term and the POC demo.

1 Demo Plans

In our proof of concept demonstration, our team will showcase our approach to code plagiarism detection using NLP by comparing two code samples and generating a similarity score.

In our demonstration, we will invoke a command line application. The application will take in two source code files in either Python or Java as input and will return a similarity score as a number from 0.0 to 1.0. A score of 0.0 indicates that the source files have no semantic similarity, and a score of 1.0 means that the two source files are entirely semantically similar. We will invoke this application twice, first with two source files that do obviously different things. These files will have a low similarity score, close to 0.0. The second invocation will be with two source files that do the same thing but differ syntactically. These files will have a high similarity score, close to 1.0. The produced similarity scores should make sense when validated by us as human reviewers.

This demonstration aims to prove that the similarity score produced by our NLP model is a good metric for determining if two pieces of code are plagiarised or not.

2 Team Meeting Attendance

Student	Meetings
Total	3
Mohammad Mohsin Khan	3
Lucas Chen	3
Dennis Fong	3
Julian Cecchini	3
Luigi Quattrociocchi	3

Our team has met on more than just 3 occasions, however we have only had this many official meetings which we recorded. Much of the time we work together and brainstorm ideas is not recorded explicitly.

3 Supervisor/Stakeholder Meeting Attendance

[For each team member how many supervisor/stakeholder team meetings have they attended over the time period of interest. This number should be determined from the supervisor meeting issues in the team's repo. The first entry in the table should be the total number of supervisor and team meetings held by the team. If there is no supervisor, there will usually be meetings with stakeholders (potential users) that can serve a similar purpose. —SS]

Student	Meetings
Total	Num
Mohammad Mohsin Khan	Num
Lucas Chen	Num
Dennis Fong	Num
Julian Cecchini	Num
Luigi Quattrociocchi	Num

[If needed, an explanation for the counts can be provided here. —SS]

4 Lecture Attendance

Student	Lectures
Total	12
Mohammad Mohsin Khan	5
Lucas Chen	5
Dennis Fong	5
Julian Cecchini	5
Luigi Quattrociocchi	5

Every team member has attended every lecture since we started tracking this metric by recording lecture attendance. In the above table, we each have only attended 5 lectures because we didn't track this until a couple weeks into the semester (September 17th is our first recorded lecture). In fact, all team members have attended every lecture, except for the VnV plan lecture on October 23rd, which no one attended because we all had a midterm on that day.

5 TA Document Discussion Attendance

Student	Lectures
Total	3
Mohammad Mohsin Khan	3
Lucas Chen	3
Dennis Fong	3
Julian Cecchini	3
Luigi Quattrociocchi	3

6 Commits

[For each team member how many commits to the main branch have been made over the time period of interest. The total is the total number of commits for the entire team since the beginning of the term. The percentage is the percentage of the total commits made by each team member. —SS]

Student	Commits	Percent
Total	118	100%
Mohammad Mohsin Khan	18	15.3%
Lucas Chen	36	30.5%
Dennis Fong	15	12.7%
Julian Cecchini	15	12.7%
Luigi Quattrociocchi	34	28.8%

[If needed, an explanation for the counts can be provided here. For instance, if a team member has more commits to unmerged branches, these numbers can be provided here. If multiple people contribute to a commit, git allows for multi-author commits. —SS]

7 Issue Tracker

Student	Authored (O+C)	Assigned (C only)
Mohammad Mohsin Khan	40	36
Lucas Chen	29	26
Dennis Fong	20	18
Julian Cecchini	19	16
Luigi Quattrociocchi	28	25

8 CICD

The project will use continuous integration and continuous deployment (CICD) to run tests and deploy the software. The steps in the CICD pipeline are as follows:

1. Developer creates PR and CICD pipeline will trigger phase 0.
2. Phase 0 will run build and check if the build is successful.
3. If the build is successful, reviewer will review the PR.
4. If the reviewer approves the PR, the CICD pipeline will trigger phase 1.
5. Phase 1 will run tests and check if the tests are successful.
6. If the tests are successful, the CICD pipeline will trigger phase 2.
7. Phase 2 will merge and deploy the software.