**Add First CSV File or Reset Currently Parsed CSV Data**

Copy desired file into “csv\_files” folder in root directory

In the *config.properties* file (/src/main/resources/config.properties), specify *FILE\_NAME* to be the name of the first csv file that you would like to add.

In the *config.properties* file (/src/main/resources/config.properties), specify *INPUT\_CSV\_LOC* to be the name of the directory containing your csv files (from the base as the root directory of the project): “csv\_files”

In the *config.properties* file (/src/main/resources/config.properties), specify *OUTPUT\_CSV\_LOC* to be the name of the directory where you want the csv data parsed (from the base as the root directory of the project).

In the *config.properties* file (/src/main/resources/config.properties), specify *CSV\_PARSED* to be “false” so that it overrides previously generated data

In the *build.gradle* file in the root directory of the project, put “cs1980viz.GetCSVData” after *mainClassName =*

In a terminal opened in the root directory of the project, enter “gradle run”

**Add Additional CSV Files to Currently Parsed CSV Data**

Copy desired file into “csv\_files” folder in root directory

In the *config.properties* file (/src/main/resources/config.properties), specify *FILE\_NAME* to be the name of the new csv file that you would like to add.

In the *config.properties* file (/src/main/resources/config.properties), specify *INPUT\_CSV\_LOC* to be the name of the directory containing your csv files (from the base as the root directory of the project): “csv\_files”

In the *config.properties* file (/src/main/resources/config.properties), specify *OUTPUT\_CSV\_LOC* to be the name of the directory where you want the csv data parsed (from the base as the root directory of the project).

In the *config.properties* file (/src/main/resources/config.properties), specify *CSV\_PARSED* to be “true” so that it does not override previously generated data

In the *build.gradle* file in the root directory of the project, put “cs1980viz.GetCSVData” after *mainClassName =*

In a terminal opened in the root directory of the project, enter “gradle run”

**Calculate File Similarity- Cosine**

After all relevant csv files have been parsed (using previous methods):

In the *config.properties* file (/src/main/resources/config.properties), specify *SIMILARITY\_TYPE* to be “cosine”

In the *build.gradle* file in the root directory of the project, put “cs1980viz.CalculateCosine” after *mainClassName =*

In a terminal opened in the root directory of the project, enter “gradle run”

The similarity matrix is then calculated and put into the “/output” directory as “output.csv”

**Calculate File Similarity- Overlap**

After all relevant csv files have been parsed (using previous methods):

In the *config.properties* file (/src/main/resources/config.properties), specify *SIMILARITY\_TYPE* to be “overlap”

In the *build.gradle* file in the root directory of the project, put “cs1980viz.CalculateOverlap” after *mainClassName =*

In a terminal opened in the root directory of the project, enter “gradle run”

The similarity matrix is then calculated and put into the “/output” directory as “output.csv”

**Calculate File Similarity- Wu Palmer**

After all relevant csv files have been parsed (using previous methods):

In the *config.properties* file (/src/main/resources/config.properties), specify *SIMILARITY\_TYPE* to be “wupalmer”

In the *build.gradle* file in the root directory of the project, put “cs1980viz.CalculateWuPalmer” after *mainClassName =*

In a terminal opened in the root directory of the project, enter “gradle run”

The similarity matrix is then calculated and put into the “/output” directory as “output.csv”

**HTML Demo**

After similarity matrix is calculated (using previous methods):

In order to get the highest common cluster between any pair of documents (this is necessary for the HTML demo), you need to call ComputeHAC which computes, maps, and writes to files all clusters and their respective member elements as well as the highest common cluster matrix as a subroutine.

In the *build.gradle* file in the root directory of the project, put “cs1980viz.ComputeHAC” after *mainClassName = .*

In a terminal opened in the root directory of the project, enter “gradle run”

The highest common cluster between any 2 documents is serialized and put into the “/output” directory as “array”.

The clusters, identified by their binary path through the dendrogram (it’s a binary tree), are written out to their respective html files. Each cluster contains all member elements.

In the *config.properties* file (/src/main/resources/config.properties), specify *MATCH\_TYPE* to be either “t” or “p” based on whether you want to display the top number of matches or anything over a threshold percentage.

In the *config.properties* file (/src/main/resources/config.properties), specify *NUM\_TOP\_FILES* to be an integer greater than 0 that you want to be the number of top matching files (if you selected “t” from the previous instruction) or specify *PERCENT\_MATCH* to be an integer greater than 0 that you want the cutoff matching percentage to be (if you selected “p” from the previous instruction).

In the *build.gradle* file in the root directory of the project, put “cs1980viz.CalcHtmlDemo” after *mainClassName =*

In a terminal opened in the root directory of the project, enter “gradle run”

The HTML files are then calculated and outputted into the “/html” directory.