Documentation of Easy-Compare

Introduction:

Easy Compare is a document comparison application that helps to detect plagiarism. Every document follows a similar text preprocessing approach, every sentence in the document gets hashed using Murmur3 hashing technique and this hash is stored in PostgreSQL. User can now select two documents and get a plagiarism score which is calculated using Jaccard index.

The Jaccard index, also known as Intersection over Union and the Jaccard similarity coefficient is a statistic used for comparing the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Murmur Hash 3:

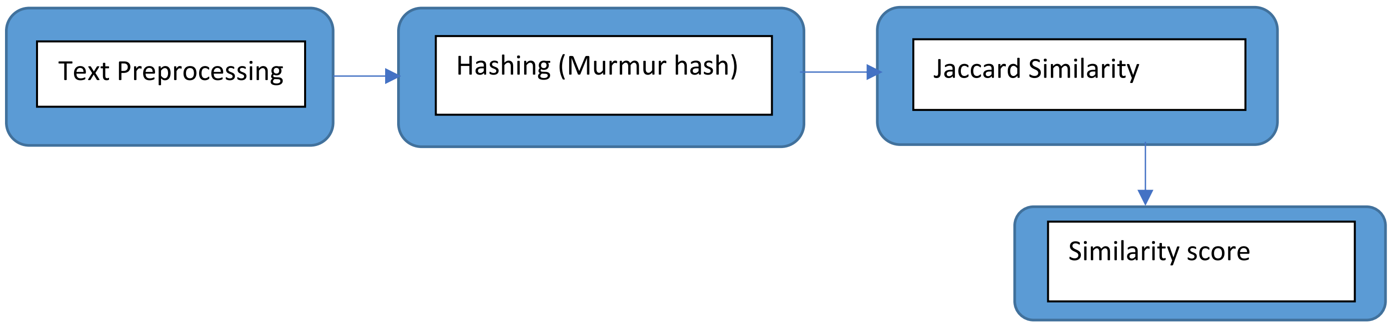
Murmur Hash is a non-cryptographic hash function suitable for general hash-based lookup. The name comes from two basic operations, multiply (MU) and rotate (R), used in its inner loop. https://en.wikipedia.org/wiki/MurmurHash#MurmurHash3

The current version is MurmurHash3 which yields a 32-bit or 128-bit hash value. When using 128-bits, the x86 and x64 versions do not produce the same values, as the algorithms are optimized for their respective platforms. MurmurHash3 was released alongside SMHasher--a hash function test suite.

Text Preprocessing:

Text preprocessing function reads the input file and converts all the text to lowercase and combines every line of the document into a string and removes all the whitespaces, each sentenced is hash and stored in PostgreSQL.

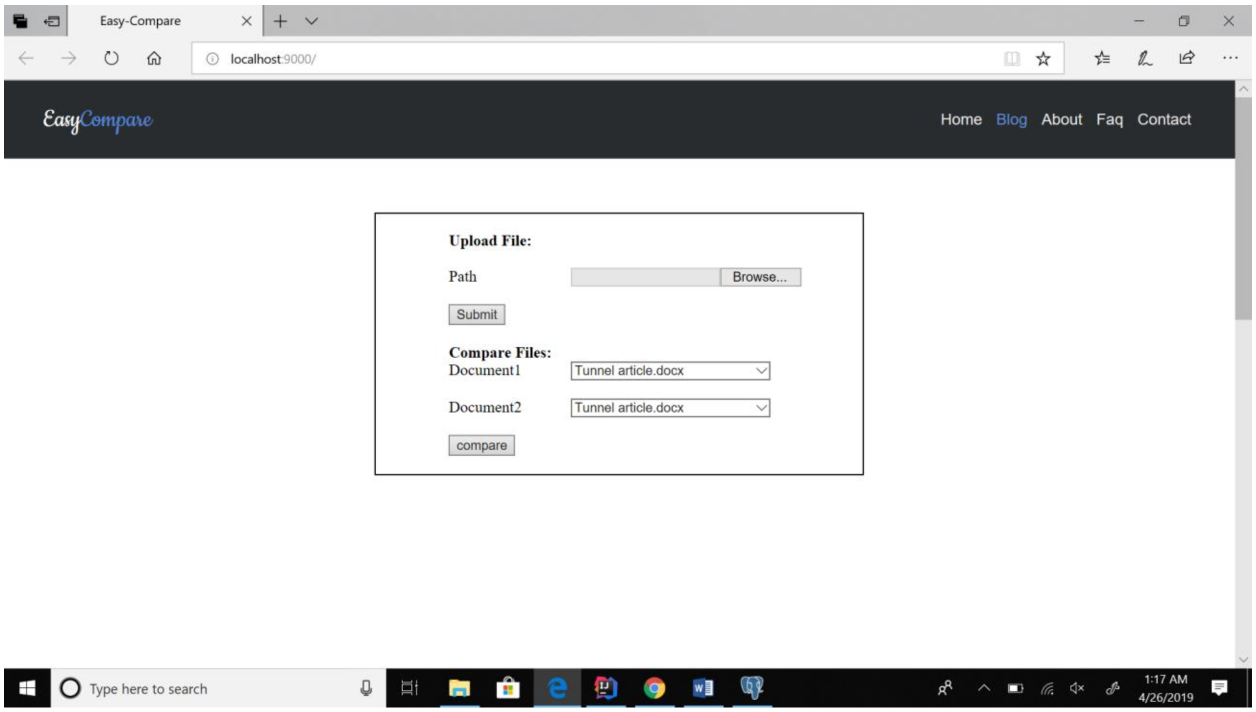
Block Diagram:



How to run and use Easy Compare on your machine:

1. Install PostgreSQL and pgadmin 4
2. Create a table in PostgreSQL using the script given below

   ```
   CREATE TABLE public."tblDocumentInformation"
   (
       "ID" integer NOT NULL,
       "Document_Name" character varying COLLATE pg_catalog."default",
       "Document_Text" character varying COLLATE pg_catalog."default",
       "Sentence_Text" character varying COLLATE pg_catalog."default"
   )
   ```
3. Git clone https://github.com/BaraiKaran/Easy-Compare.git
4. Open the play project in your preferred IDE and in the shell type 'sbt run' and start the play application.
5. Once the application is running click on browse and select the file to upload.
6. Once you have uploaded the files, in dropdown below select the two files which you want to compare.
7. When everything is set press compare and obtain plagiarism score.



Model, View and Controller pattern:

Controller –

Endpoint	**Description**
/	Initial GET request to load a form.
/simpleForm	POST request to store document content in database.
/compare	POST request to compare two documents.

Services	**Description**
BasicForm	Code for form Mapping.
Comparison	Code for comparison between two documents using Jaccard similarity.
Preprocess	Code for initial preprocessing of the text.
Validation	Code for validation of the user input.

Models –

Models	Description
InteractionWithDb	Contains slick code for interaction with PostgreSQL.

Unit Tests –

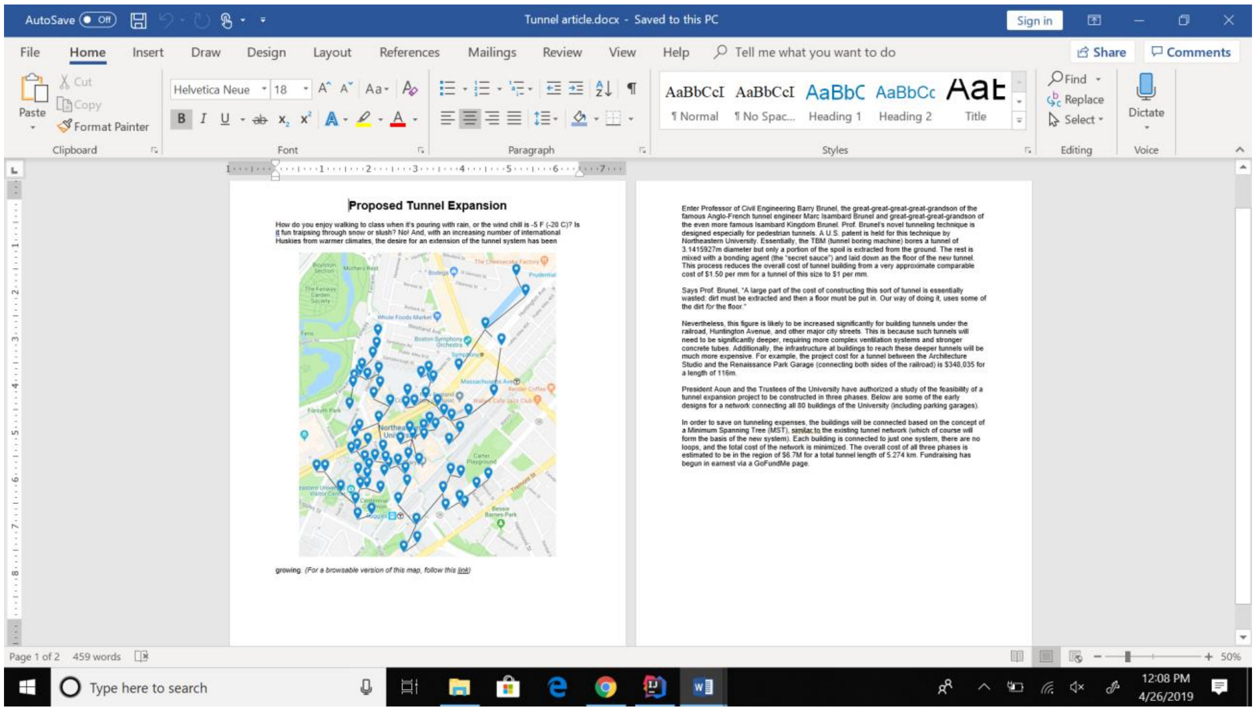
Code has 60% coverage i.e. 60% of the code is unit tested currently. There are few integration tests as well which takes coverage to about 65%. Unit tests can be found in “test” directory.

Test File	Unit tests
preprocessSpec	Covers unit tests for data preprocessing, Functions tested: 1. convertListToString 2. removeWhiteSpaces, removeMultipleWhiteSpaces 3. splitText 4. getFileName 5. readTextFile 6. apply
ComparisonSpec	Compares contents of two documents, functions tested: 1. calculateSimilarityScore 2. hashContentsOfList, hashContentsOfListNotEqual, hashContentsOfEmptyList 3. getListFromOption 4. flattenFuture 5. getSimilarityScore
ValidationSpec	Function tested: 1. getFileType
InteractionWithDbSpec	Functions tested: 1. Sequence, SequenceWhenGivenNull

ControllerSpec (Integeration)	This file tests the initial GET controller (“/”) which brings up the form in user interface.

Demo of the application:

Document 1 -



Proposed Tunnel Expansion

How do you enjoy walking to class when it's pouring with rain, or the wind chill is -5 F (-20 C)? Is it fun traipsing through snow or slush? No! And, with an increasing number of international Huskies from warmer climates, the desire for an extension of the tunnel system has been growing. *(For a browsable version of this map, follow this link)*

Enter Professor of Civil Engineering Barry Brunel, the great-great-great-great-grandson of the famous Anglo-French tunnel engineer Marc Isambard Brunel and great-great-great-grandson of the even more famous Isambard Kingdom Brunel. Prof. Brunel's novel tunneling technique is designed especially for pedestrian tunnels. A U.S. patent is held for this technique by Northeastern University. Essentially, the TBM (tunnel boring machine) bores a tunnel of 3.1415927m diameter but only a portion of the spoil is extracted from the ground. The rest is mixed with a bonding agent (the “secret sauce”) and laid down as the floor of the new tunnel. This process reduces the overall cost of tunnel building from a very approximate comparable cost of $1.50 per mm for a tunnel of this size to $1 per mm.

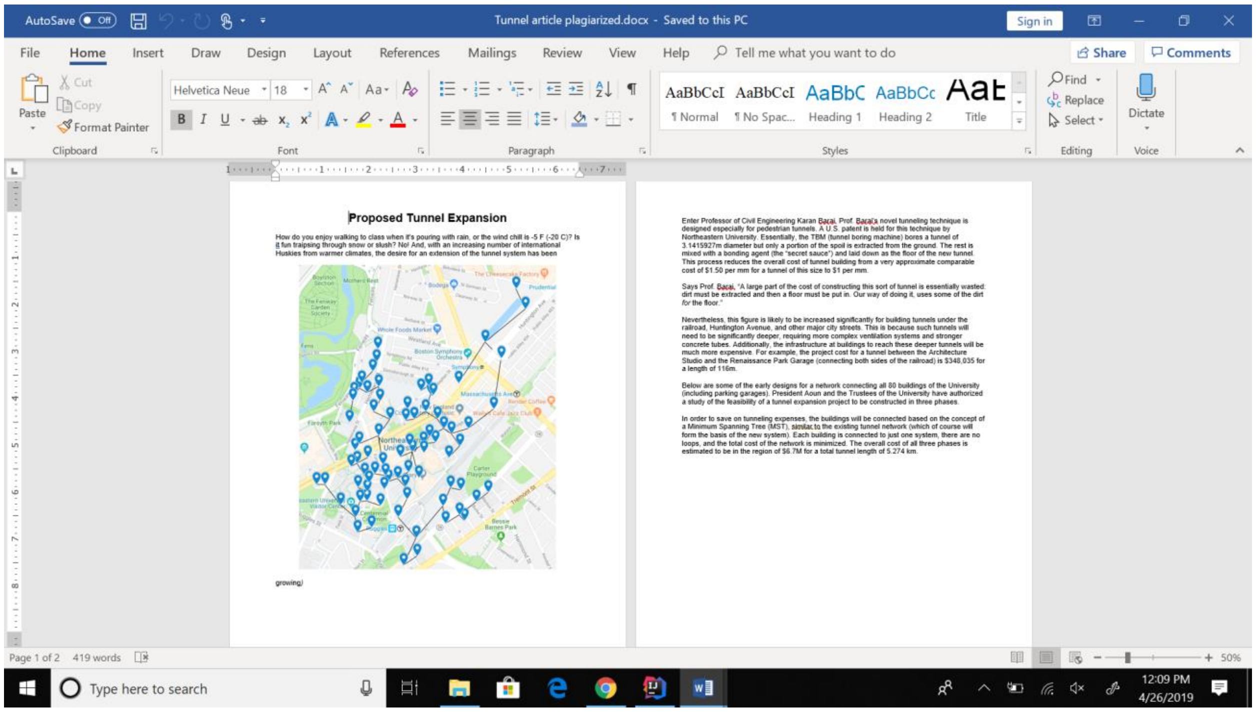
Says Prof. Brunel, “A large part of the cost of constructing this sort of tunnel is essentially wasted: dirt must be extracted and then a floor must be put in. Our way of doing it, uses some of the dirt *for* the floor.”

Nevertheless, this figure is likely to be increased significantly for building tunnels under the railroad, Huntington Avenue, and other major city streets. This is because such tunnels will need to be significantly deeper, requiring more complex ventilation systems and stronger concrete tubes. Additionally, the infrastructure at buildings to reach these deeper tunnels will be much more expensive. For example, the project cost for a tunnel between the Architecture Studio and the Renaissance Park Garage (connecting both sides of the railroad) is $348,035 for a length of 116m.

President Aoun and the Trustees of the University have authorized a study of the feasibility of a tunnel expansion project to be constructed in three phases. Below are some of the early designs for a network connecting all 80 buildings of the University (including parking garages).

In order to save on tunneling expenses, the buildings will be connected based on the concept of a Minimum Spanning Tree (MST), similar to the existing tunnel network (which of course will form the basis of the new system). Each building is connected to just one system, there are no loops, and the total cost of the network is minimized. The overall cost of all three phases is estimated to be in the region of $6.7M for a total tunnel length of 5.274 km. Fundraising has begun in earnest via a GoFundMe page.

Document 2 –



Proposed Tunnel Expansion

How do you enjoy walking to class when it's pouring with rain, or the wind chill is -5 F (-20 C)? Is it fun traipsing through snow or slush? No! And, with an increasing number of international Huskies from warmer climates, the desire for an extension of the tunnel system has been growing.

Enter Professor of Civil Engineering Karan Barai. Prof. Barai's novel tunneling technique is designed especially for pedestrian tunnels. A U.S. patent is held for this technique by Northeastern University. Essentially, the TBM (tunnel boring machine) bores a tunnel of 3.1415927m diameter but only a portion of the spoil is extracted from the ground. The rest is mixed with a bonding agent (the “secret sauce”) and laid down as the floor of the new tunnel. This process reduces the overall cost of tunnel building from a very approximate comparable cost of $1.50 per mm for a tunnel of this size to $1 per mm.

Says Prof. Barai, “A large part of the cost of constructing this sort of tunnel is essentially wasted: dirt must be extracted and then a floor must be put in. Our way of doing it, uses some of the dirt *for* the floor.”

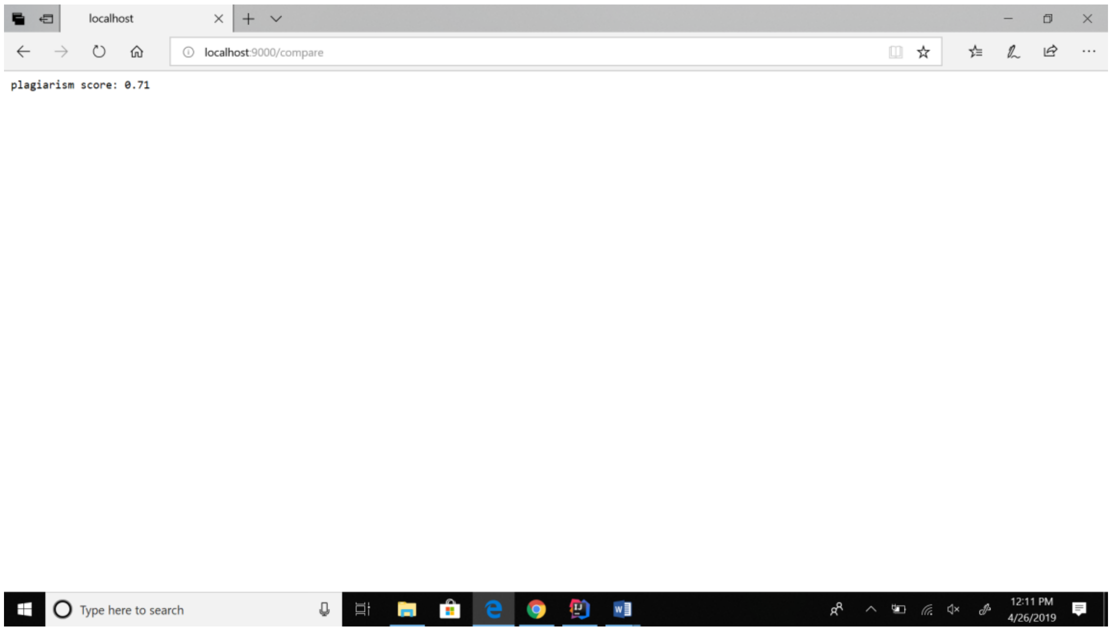
Nevertheless, this figure is likely to be increased significantly for building tunnels under the railroad, Huntington Avenue, and other major city streets. This is because such tunnels will need to be significantly deeper, requiring more complex ventilation systems and stronger concrete tubes. Additionally, the infrastructure at buildings to reach these deeper tunnels will be much more expensive. For example, the project cost for a tunnel between the Architecture Studio and the Renaissance Park Garage (connecting both sides of the railroad) is $348,035 for a length of 116m.

Below are some of the early designs for a network connecting all 80 buildings of the University (including parking garages). President Aoun and the Trustees of the University have authorized a study of the feasibility of a tunnel expansion project to be constructed in three phases.

In order to save on tunneling expenses, the buildings will be connected based on the concept of a Minimum Spanning Tree (MST), similar to the existing tunnel network (which of course will form the basis of the new system). Each building is connected to just one system, there are no loops, and the total cost of the network is minimized. The overall cost of all three phases is estimated to be in the region of $6.7M for a total tunnel length of 5.274 km.

Plagiarism score:

The application returns a score of 71%, that means about 71% of the code is plagiarized.



```
plagiarism score: 0.71
```

Continuous Integration:

To test impact of every commit I have used an integration tool CircleCI that will build the application and run the unit tests to make sure things are working fine after every commit in the git repository.

Codacy:

To keep check on the code quality throughout the entire application I have used an automated code review tool Codacy. This tool checks the code quality after every commit or pull request and gives feedback on ways to improve the code. Currently, Easy Compare gets an “A” in terms of code quality.

Scaladoc –

Entire code is well documented using scaladoc

Technical Specs:

1. Scala 2.12.2
2. Play 2.7
3. PostgreSQL 9.6.5
4. pgAdmin 4
5. Scala Test 3.0.5
6. CircleCI
7. Git and GitHub

Future Scope:

1. Include abstract syntax trees for comparison of code and document.
2. Dockerize the application for ease of installation in any machine.

References:

1. Jaccard_index - https://en.wikipedia.org/wiki/Jaccard_index
2. Murmur hash - https://en.wikipedia.org/wiki/MurmurHash
3. Rabin Karp - https://en.wikipedia.org/wiki/Rabin%E2%80%93Karp_algorithm

License:

1. MIT License