**Text Processing Report– Assignment (IR)**

Qingyu Yang1

1 Ucard No. 001722400 Registration No. 190256746

1. Assignment Purpose and Aim

This assignemnt requires to build and complete the reserved python class *Retrieve* by finishing three models named Binary, Term Frequency(TF), Term Frequency Iverse Document Frequency(TFIDF). For the other classes like *CommandLine*, *IndexLoader* etc. have been provided for you to transform the data structure and operate the file of store rusult input and output. Otherwise, the efficiency of programme and method are also required to be minimal and marked as a part of the grades.

2. Different Schemes Analysis

1.1 Binary

As the hint in assignemnt guidebook given in binary score, we don’t need to mention about the size of the query vector which is constant across the comparisions which can be dropped without affecting how the documents are ranked. The key to solve problem is to extract the similarity information from the data-structure from *IndexLoader*. The procedure of calculate the binary similarity:

1)Find the terms in each query and utilize the terms to return to the *IndexLoader.index* and obtain all (*doc\_id:count*) from the *Retrieve. getCandidate(self)*.

2)Create a new list to store all the repetitive *doc\_id*, count the number of occurance and this result is the molecule of the model.(∑*qidi*)

3)Obtain the length of the each document(*doc\_id*),we can calculate it by the sum of all *count*’s square.(*Retrieve. doc\_length(self)*)

4) *Retrieve.Binary\_model* figures out all the similarity for each document as the formula(1.1) and finally sort its’ values in descending order.

1.2 Term Frequency (TF)

As for the definition of term frequency, we should not only focus on the count of terms in documents, also consider the number of terms showed in each query.For the length of document, we can use the same method which is meantioned in Binary.The procedure of calculating the TF similarity:

1)Traverse each the query to get the terms and find the terms corresponding basic data set in the *IndexLoader.index*. Create a new dictionary variable to store *doc\_id:{term:count}* to facilitate later calculations.(*Retrieve.* *getCandidate*)

2)Obtain the set of terms in each query, and search for the *tr\_candidate\_all* to calculate the cosine score according to the formula(1.2) and get the result for different document.

1.3 Term Frequency, Inverse Document Frequency

As for the definition of TFIDF, we should primarily calculate the IDF value for each term and it’s dictionary values as the formula(1.3):

The IDFs value are stored in a newly created dictionary and the procedure of calculating the similarity is generally same as the TF. But we should multiply the IDF value for each term. It can decrease the weight of terms which are universal and increase the weight of terms which are rare.As a result, the documents which have so much rare words can be ranked equally.The formula can been seen below.(1.4)

1.1.1 Subsubsection heading.

In in nunc. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Donec ullamcorper fringilla eros. Fusce in sapien eu purus dapibus commodo. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Cras faucibus condimentum odio. Sed ac ligula. Aliquam at eros.

Etiam at ligula et tellus ullamcorper ultrices. In fermentum, lorem non cursus porttitor, diam urna accumsan lacus, sed interdum wisi nibh nec nisl. Ut tincidunt volutpat urna. Mauris eleifend nulla eget mauris. Sed cursus quam id felis. Curabitur posuere quam vel nibh. Cras dapibus dapibus nisl. Vestibulum quis dolor a felis congue vehicula. Maecenas pede purus, tristique ac, tempus eget, egestas quis, mauris. Curabitur non eros. Nullam hendrerit bibendum justo. Fusce iaculis, est quis lacinia pretium, pede metus molestie lacus, at gravida wisi ante at libero. Quisque ornare placerat risus. Ut molestie magna at mi. Integer aliquet mauris et nibh. Ut mattis ligula posuere velit.

Nunc sagittis. Curabitur varius fringilla nisl. Duis pretium mi euismod erat. Maecenas id augue. Nam vulputate. Duis a quam non neque lobortis malesuada. Praesent euismod. Donec nulla augue, venenatis scelerisque, dapibus a, consequat at, leo.

Pellentesque libero lectus, tristique ac, consectetuer sit amet, imperdiet ut, justo. Sed aliquam odio vitae tortor. Proin hendrerit tempus arcu. In hac habitasse platea dictumst. Suspendisse potenti. Vivamus vitae massa adipiscing est lacinia sodales. Donec metus massa, mollis vel, tempus placerat, vestibulum condimentum, ligula. Nunc lacus metus, posuere eget, lacinia eu, varius quis, libero.

Aliquam nonummy adipiscing augue. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet commodo magna eros quis urna. Nunc viverra imperdiet enim. Fusce est. Vivamus a tellus. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Acknowledgements

Proin pharetra nonummy pede. Mauris et orci. Aenean nec lorem. In porttitor. Donec laoreet nonummy augue. Suspendisse dui purus, scelerisque at, vulputate vitae, pretium mattis, nunc. Mauris eget neque at sem venenatis eleifend. Ut nonummy.

References

1. Surname A, Surname B and Surname C 2015 *Journal Name* **37** 074203
2. Surname A and Surname B 2009 *Journal Name* **23** 544