Bogazici University

CmpE493
Homework 3

Spring 2018

Irmak Kavasoglu
2013400090

May 4, 2018

1. Implementation Summary

For this project, I have used Java programming language with Eclipse IDE. In this chapter, I will try to explain the steps I have taken to finish the project and explain details of the implementation. You can find the project progress steps in my github.

1.1. Reading Stories

The *readStories* function along with *readStory* function, reads the stories from the Dataset directory, which is taken as the first argument to the program. Within this directory, files from 1.txt to 1000.txt are expected. Each file will have the story as one sentence per line. After the story, there will be a blank line, and following that, there will be a summary, again as one sentence per line.

1.2. Calculating tf-idf Scores

The *calculateTfIdf* function returns an array. Each element of this array represents one document. Each element is made up of an array, which stands for every sentence in that document. And corresponding to every sentence, there is a HashMap, containing the word and its tf-idf value.

While calculating tf-idf scores, the text is normalized by removing all punctuation and ignoring single character words.

1.3. Building Similarity Matrix

The *buildSimilarityMatrices* function takes in the stories and the tf-idf values. It returns a three-level array such as the element at index <i, j, k> represents the similarity of sentence j to sentence k in document i.

The similarity is calculated only for the original sentences and not the summary sentences. The tf-idf vectors are first normalized and then multiplied to get the similarity.

1.4. Building Adjacency Matrix

The *buildAdjacencyMatrix* function takes in the similarities matrix. It returns a three-level array such as the element at index <i, j, k> is 1 if the similarity of sentence j to sentence k in document i is greater than 0.10 and 0 otherwise.

1.5. Building Transition Matrix

The *buildTransitionMatrix* function takes in the adjacency matrix and the teleportation rate. It returns a three-level array such as the element at index <i, j, k> is the possibility of arriving at sentence k from sentence j in document i.

The calculation is as follows. The teleportation rate is divided to the number of sentences and the transition vector is filled with that value. Than the remaining possibility is divided into the number of 1s for that sentence in the adjacency matrix and the corresponding values are updated with the addition of this amount.

1.6. Applying Power Method

The *applyPowerMethod* calls the recursive *powerMethod* function with the initial vector of <1, 0, 0, 0, …, 0> and the transition matrix. The power method multiplies the given distribution with the transition matrix until the result converges. The convergence is defined as follows: if the distance between new distribution and the given distribution is less than 0.00001, we call this the stable state and not iterate any further.

1.7. Resulting Distributions

In the end, the distribution for a document is printed and this document is selected by the user and received via the second argument of the program.

2. Rouge Calculations and Results

For Rouge, I have used the Java library in the following link:

http://rxnlp.com/rouge-2-0-usage-documentation/#.Wu3-9dNuaCQ

This library requires two sets of files to calculate Rouge values. First set is the reference files and the second set is the system files. The creation of these two are in the code, but commented out. As a summary, we create the newsxx_reference1.txt files for each news file, which contains the summary sentences given in the documents; and we create the newsxx_system1.txt files for each news file, which contains 3 highest scoring sentences from our calculations.

Then we run the library with these data. The results file contains the statistics for each newsxx file. Within the code, again, commented out, we take this results file and calculate the average values. The results are as follows:

	Recall	Precision	F-score
Rouge-1	0.37641753999999983	0.18220568999999975	0.23703178999999966
Rouge-2	0.15474222999999987	0.06965682	0.09235376999999988
Rouge-L	0.3366736299999998	0.17959062999999992	0.2266147399999998