Cmpe 493 Introduction to Information Retrieval, Spring 2018
Assignment 3 - Single Document Extractive Text Summarization with LexRank, Due: 04/05/2018 (Monday), 23:55

In this assignment, you will implement a text summarization model with LexRank. LexRank is an unsupervised model which computes the relative importance of the sentences based on cosine similarity and a modified version of the PageRank algorithm. The details of the algorithm are available at the following link. You can also consult the lecture notes (IR-Lec10.pdf). Note that the LexRank paper is for multi-document summatization. In this homework, you will perform single document summarization.

`https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume22/erkan04a-html/erkan04a.html`

Dataset: The Dataset.zip file contains a folder named "Dataset" which contains 1000 plain text files. Each file contains a news story. The sentences in each file are separated by new line. In addition, each file contains a set of summary sentences (gold standard summary) separated with a blank like from the news story sentences. So, the structure of the files is as follows:

news-sent-1
news-sent-2
news-sent-3
news-sent-4
.
.
.
news-sent-n

summary-sent-1
summary-sent-2
.
.
.
summary-sent-k

You should perform the following steps:

1. Calculating IDF scores: You will use the cosine similarities between the news sentences to construct a sentence-sentence graph for a given news story, where the nodes are the sentences and the weight of an arc between two sentences is their cosine similarity score. Note that the PageRank algorithm is for directed graphs, so you should include two arcs (one for each direction) between each sentence pair. In order to compute the cosine similarity scores, you need to create the tf-idf vectors of the sentences. First, you need to calculate the idf scores of the words by using all 1000 files in the data set. You should only use the news sentences, not the summary sentences to compute the idf scores.

2. Building the Graph: You need to perform the LexRank algorithm for each news story in the dataset. The details of how to construct the graph are provided in the link above. You should choose the cosine similarity threshold (t) as 0.10 to create arcs between two sentences. In other words, if the cosine similarity between two sentences is less than t, these sentences are not connected to each other (i.e., the corresponding entry in the adjacency matrix is 0), otherwise they are connected to each other in the graph (i.e., the corresponding entry in the adjacency matrix is 1) . Also, the teleportation rate should be set to 0.15 and the error tolerance (ϵ) in the power method should be set to 0.00001. After obtaining the LexRank scores for each sentence, you will select the three sentences with the highest LexRank scores as your summary.

3. Evaluation: You will use the ROUGE scoring mechanism to compare your sentences to the gold summary in the dataset. You can use a third party library in this part. Python users can use this library in the following github repository:
`https://github.com/pltrdy/rouge`

Input-Output Format: You will calculate the ROUGE scores only for your report. We will evaluate your code based on the LexRank scores of the sentences. You will take two parameters for that task. The first parameter will be the path of the Dataset folder. The second parameter will be the name of the file in the Dataset folder. You will calculate the LexRank score for each sentence in that file. For example:
Input:
lexRank.py /home/User/Desktop/Dataset 13.txt
Output(The numbers are not real, they are just provided as an example):
0.67 0.45 0.21 0.66 0.89 0.13 0.71

So, you will calculate the idf scores based on the 1000 news stories in the dataset folder. Then, you will construct the graph for the news sentences in 13.txt. Then, you will calculate the LexRank scores for each sentence and print them out.

You can use numpy for matrix operations. You can use an equivalent library to numpy in Java or C++ just for the matrix operations. You may use any programming language of your choice. However, we should be able to run your program by following the instructions in your readme file and you should not use any third party libraries except for ROUGE scores and matrix operations (You need to implement the power method yourself).

Submission: You should submit a *".zip"* file named as YourNameSurname.zip containing the following files using the Moodle system:

1. Report:
(i) Please provide a table which shows the average Rouge-1, Rouge-2, and Rouge-L F1 scores obtained for the provided dataset.

2. Source code and executable: Commented source code and executables of your summarization system.

3. Readme: Detailed readme describing how to run your program.

Late Submission: You are allowed a total of 5 late days on homeworks with no late penalties applied. You can use these 5 days as you wish. For example, you can submit the first homework 2 days late, then the second homework 3 days late. In that case you will have to submit the remaining homeworks on time. After using these 5 extra days, 10 points will be deducted for each late day.