CS60092: Assignment 3

Summarization

[Deadline: 10.11.2024, 11:59 PM IST]

IMPORTANT INSTRUCTIONS

- **Plagiarism:** We will be employing strict plagiarism checking. If your code matches with another student's code, all those students whose codes match will be ***awarded zero marks*** without any evaluation. Therefore, it is your responsibility to ensure you neither copy anyone's code nor anyone is able to copy your code.
- **Code error:** If your code doesn't run or gives an error while running, marks will be awarded based on the correctness of logic. If required, you might be called to meet the TAs and explain your code.
- **Python library restrictions**: You can use simple python libraries like `nltk`, `numpy`, `os`, `sys`, `collections`, `timeit`, etc. However, YOU CANNOT USE LIBRARIES like `lucene`, `elasticsearch`, or any other search API. If your code is found to use any such library, you will be ***awarded zero marks*** for this assignment without any evaluation. You also cannot use parsing libraries either for parsing the corpus and query files (You must write your own code for the same).

SUBMISSION INSTRUCTIONS

Submit the following files:

Assignment3_<ROLL_NO>_summarizer.py
Assignment3_<ROLL_NO>_evaluator.py
Assignment3_<ROLL_NO>_summary.txt
README.txt

in a zipped folder named: **Assignment3_<ROLL_NO>.zip (or tar.gz)**

Your README.txt file should contain the following information-

1. **[Mandatory]** Mention your **Roll Number** on the first line of your README.
2. **[Mandatory]** Any specific library requirements to run your code and the specific Python version you are using.
3. **[Mandatory]** Provide details of your design.
4. **[Optional]** Any other special information about your code or logic that you wish to convey.

IMPORTANT: PLEASE FOLLOW THE EXACT NAMING CONVENTION OF THE FILES AND THE SPECIFIC INSTRUCTIONS IN THE TASKS CAREFULLY. ANY DEVIATION WILL RESULT IN DEDUCTION OF MARKS. PLEASE NOTE THE EVALUATION GUIDELINES ON PAGE 4.

NOTE: YOUR Assignment3_<ROLL_NO>_summary.txt FILE WILL NOT BE EVALUATED UNLESS YOUR CODE WORKS PROPERLY.

This assignment is on multi-document summarization using an exact Integer Linear Programming formulation. Specifically, you will be implementing algorithms from McDonald [3].

You have to use the Python programming language for this assignment. You may use the GLPK library [1] (download the relevant Python compatible library), or any other non-commercial Integer Linear Programming libraries available.

The total marks for this assignment is 50.

Dataset:

For this assignment, you will be using the CNN/DailyMail Dataset [2]. The relevant file will have to be downloaded from the given link-

- Datafile– https://drive.google.com/file/d/1UW-hA5xIeRbXYq541J1oyA6gThSFrnAA/view?usp=sharing

IMPORTANT: The dataset may contain rows with the long summaries (K > 200 words). You are to IGNORE all such rows and print the total number of rows that you use for your experiments.

[1] https://www.gnu.org/software/glpk/
[2] Nallapati, Ramesh, et al. "Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond." *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*. 2016.
[3] McDonald, Ryan. "A study of global inference algorithms in multi-document summarization." *European Conference on Information Retrieval*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

Task A (Summarization Using ILP)

For this assignment, you have to first create an ILP formulation of the problem statement. Refer to Fig. 2 in McDonald [3] for the ILP formulation. Use scoring functions in Sec. 3.1 for scoring relevance and redundancy. We will be using sentences as our textual units. The value of ***K*** should be set to 200 words.

$$\text{maximize } \sum_i \alpha_i Rel(i) - \sum_{i<j} \alpha_{ij} Red(i,j)$$

$$\begin{aligned}\text{such that } \forall i,j: \quad &(1)\ \ \alpha_i, \alpha_{ij} \in \{0,1\} \qquad (4)\ \ \alpha_{ij} - \alpha_j \leq 0 \\ &(2)\ \ \textstyle\sum_i \alpha_i l(i) \leq K \qquad (5)\ \ \alpha_i + \alpha_j - \alpha_{ij} \leq 1 \\ &(3)\ \ \alpha_{ij} - \alpha_i \leq 0\end{aligned}$$

Fig. 2. ILP formulation of global inference.

$$Rel(i) = POS(t_i, D)^{-1} + SIM(t_i, \boldsymbol{D}) \quad \text{(where } t_i \in D \text{ and } D \in \boldsymbol{D})$$

$$Red(i,j) = SIM(t_i, t_j)$$

where $POS(t, D)$ is the position of textual unit t in document D and $SIM(a, b)$ is the cosine similarity between two vectors. Relevance scores prefer sentences that are near the beginning of documents and are maximally informative about the entire document collection. Again, these score functions are general and we only use these particular scoring criteria for simplicity and because we evaluate our approach on news stories[2].

Save the following file in your main code directory:
Assignment3_<ROLL_NO>_summary.txt

- Name your code file as: ***Assignment3_<ROLL_NO>_summarizer.py***

- Running the file: Your code should take the path to the dataset, as input, and it should run in the following manner:
  `$>> python Assignment3_<ROLL_NO>_summarizer.py  <path to data file>`

Task B (Evaluation)

- You will be using ROUGE-1 and ROUGE-2 scores to report the performance metrics of your algorithm.

- Your code should output the ROUGE-1 and ROUGE-2 scores for each document to the console.
- Name your code file as: ***Assignment3_<ROLL_NO>_evaluator.py***

- Running the file: Take the output of Task-A and the original dataset file as input and it should run in the following manner:

`$>> python Assignment3_<ROLL_NO>_evaluator.py <path_to_data file> <path to Assignment3_<ROLL_NO>_summary.txt>`

EVALUATION GUIDELINES

1. Task A **[30 marks]**
    a. Correctly implementing code for the Relevance and Redundancy scores: **5*2 = 10 marks**
    b. Correctly implementing the ILP: **20 marks**
2. Task B **[15 marks]**
    a. Calculating ROUGE-1 and ROUGE-2 scores : **7.5*2 = 15 marks**
3. README **[5 marks]**
4. Deductions:
    a. Plagiarism: **-50 marks**
    b. Using libraries that are not allowed: **-50 marks**
    c. Not following naming conventions: **-2 marks for every violation**
    d. Small bugs in code, etc. that are beyond the overall logic of the code workflow, e.g., not following the input format specification for running code or anything else that does not fall into the marking scheme above: **-2 marks for every violation**
