IIT PALAKKAD

CS3100: Paradigms Of Programming

Under the guidance of Dr. Mrinal Kanti Das

Imperative Paradigm Mini Project

Date of submission : Sept 13 2017

Submitted by

Jayaprakash A

Siddhardha SST

Vinay Ande

Aditya M

Chapter 1

Imperative Paradigm

1.1 Problem Statement

Compute distance between two documents.

1.2 Data Structures and algorithms

Some basic data structures like lists and dictionaries were used. Term frequency–inverse document frequency (Tf-Idf), is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. This method has been used to give weightage to more important words rather than just the word frequency.

1.3 Design decisions

Tf-idf weights are used to give weights to various words. Even though Tf-Idf reduces the weight to commonly used words, we opted to remove such words from word frequency vector to remove even minor weightage given to such words.

This way weight of other important words increases. While parsing the text, we decided not to use $ and % as delimiters as they could have some importance as $ could mean money and % could imply percentage

1.4 Methodology

The entire methodology is divide into 4 stages. These include formatting text, creating word frequency vector, computing tf-idf weights and calculation of distance.
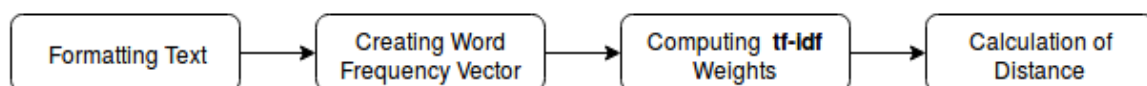

Figure 1.1: The flow chart depicting stages of algorithm

- **Formatting text :** Read the file and extract words from it. Words are obtained by splitting the text by various delimiters (variable `delimiters`).

- **Word frequency vector :** After extracting the words, a vector that contains the frequency of each and every word is created. The vector is saved in a dictionary. This is done for each and every file in the corpus.

- **Computing tf-idf weights :** Tf-Idf weights corresponding to the above word frequency vector is calculated as per the given formula.

$$TF(t) = \frac{\text{Number of times term t appears in a document}}{\text{Total number of terms in the document}}$$

$$IDF(t) = \ln\left(\frac{\text{Total number of documents}}{\text{Number of documents with term t in it}}\right)$$

$$\text{Tf-idf weight(t) = TF(t) * IDF(t)}$$

- **Calculation of distance :** The distance between each and every pair of files is done by calculating the cosine distance and saved in a list. This list is sorted in ascending order and logged to a file.

1.5 Observations and results

The code was run on the testing input and the results were to the `/out_res/eval_res-nearest` file. The results are as follows

Index of file	Index of nearest file
1	6
2	3
3	2
4	1
5	8
6	1
7	8
8	1
9	1
10	1

1.6 Contribution of team members

All the members of the team have put their maximum efforts. Every one helped each other throughout the project, both in debugging of the code as well conceptual part. All the members have put the same amount of man hours for this project. So, one cannot pinpoint of less contribution of any particular member of the team.

Every one has helped in writing the each and every part of code and also in writing of report. However, if one needs to exactly know the division of labour. This is as follows:

Jayaprakash A	Tf-Idf vectorisation
Siddhardha SST	Creating word frequency vector
Mettu Aditya	Text retrieval and mining
Vinay Ande	Calculation of cosine distance