COSC 320 – 001

Analysis of Algorithms

2022/2023 Winter Term 2

Project Topic Number: 1

Project Third Milestone (Group 32)

Keyword Replacement in Corpus

Group Lead: Anitej Isaac Sharma

Group Members:

Anitej Isaac Sharma

Abdirahman Hajj Salad

Yuki Isomura

Abstract

This is the Third Milestone for our Keyword Replacement (in Corpus) Algorithm. In this milestone, we succeeded in coming up with the Dataset details, Implementation of our algorithm, Results, Unexpected Cases/Difficulties, and Task Separation and Responsibilities.

Dataset

There are two datasets in this project.

Abbreviation list

- This dataset is one csv file that contains abbreviations in the first column and full forms of the corresponding abbreviation in the second column.
- The raw file of this dataset is extracted from webopedia using the code in DATA1_ExtractionCode.py . It is processed by eliminating the abbreviations that are poorly used, do not make sense, overlap the commonly used English words, and are difficult to recognize in this algorithm.
- There are about 1,500 rows and the size of the file is about 33KB.
- The formats of the abbreviation are various where they include single words, alphanumeric characters, and some other special characters. However, there are some restrictions and limitations which are mentioned in the Unexpected Cases/Difficulties section.

Documents to be processed

- This dataset consists of 11 csv files that contain the information of reviews of different apps.
- The raw files of this dataset are extracted from UBC OneDrive. They were originally 3676 files where size of each file is inconsistent. Each file folds multiple reviews of the same app where each of them are stored in one row.
- They have the following 11 columns.
    - reviewId … ID of the review.
    - userName … User name of the user who posted the corresponding review.
    - userImage … Link to the profile picture of the user.
    - content … Review comment.
    - score … Rated score out of five.
    - thumbsUpCount … Number of thumbs up given by other users.
    - reviewCreatedVersion … Version of the app when the review is posted.
    - at … Detailed time of when the review is posted.
    - replyContent … Reply comment to the review posted by the developer.
    - repliedAt … Detailed time of when the reply comment is posted.
- Since the maximum size of files that are under the Git support is 100MB, we have combined raw files into 11 files where size of each file is slightly less than 100MB.
- Each file holds approximately 300,000 rows with 11 columns.
- In this project, we only need to consider the content column. Each cell in this column contains a review comment. Some of the words in the cells are abbreviated. Some cells contain commas, newline characters, double quotation characters which make it difficult when retrieving the dataset into a list before handling.

Implementation

Detailed implementation of the algorithm can be checked through Git repository.
Overall, I created five files, Main, Tool, AlgorithmA, AlgorithmB, and MainTest. Main is a driver of any methods that are created in other files. Tool contains some methods that can be commonly used both in AlgorithmA and AlgorithmB. Methods implemented in Tool and their roles are as follows. All of those methods are tested in MainTest.

- `keywordList()`
  reads the abbreviation list from csv file and returns 2D ArrayList of String where inner ArrayList holds a keyword and corresponding phrase.
- `partialKeywordList(int keyword_length)`
  reads the abbreviation list from csv file by the specified number of pairs and returns 2D ArrayList of String same as `keywordList()`.
- `readCSV(String path)`
  reads csv file from specified path and returns 2D ArrayList of String where inner ArrayList represents the row.
- `documentsPath()`
  returns all the paths of csv files of documents.
- `processedPath(String org_path, String cur_folder, String new_folder)`
  returns a new path of file in String by modifying the specified path. Generally, it is used to convert file name such that “data/document/raw/document0.csv” to “data/document/processed/document0_processed.csv”.
- `deleteProcessedDocuments()`
  deletes all the files existing in data/processed.
- `writeCSV(ArrayList<ArrayList<String>> data, String path)`
  writes specified 2D ArrayList of String as a file in the specified path.
- `trimWord(String word)`
  Trims the specified word into 3 components. If the word contains a special character at the beginning or end, those letters are stored at indexes 0 and 2. Trimed word is stored at index of 1. If the original word ends by one of period, interrogation mark, exclamation mark, colon or semicolon, “true” is stored at index of 4 to represent that the next word should start with uppercase.

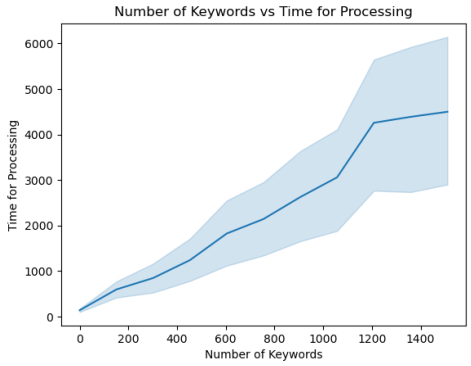
AlgorithmA and AlgorithmB implements the algorithms to process the documents. In this milestone I only implemented AlgorithmA. Below is the detailed process of the algorithm. Methods in Tool are called here.

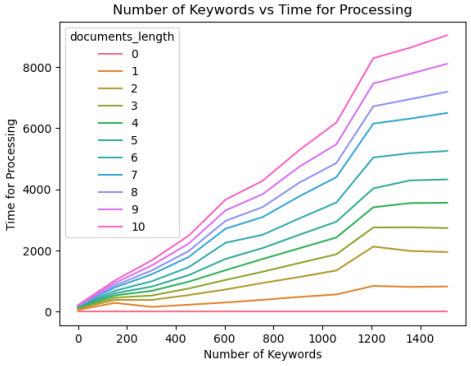
1. AlgorithmA takes the list of abbreviations, data which is 2D ArrayList of String read from file, and path of the file in String.
2. Go through each inner ArrayList of data by for-loop. In other words, go through each row of data.
3. Within the for-loop in 2, retrieve the String at index 3, because it is the cell of content.
4. Separate String by space and create an array of String.
5. Go through Each word by for-loop.

6. Within the for-loop in 5, trim the word to eliminate special characters.
7. Go through the list of abbreviations by for-loop.
8. Within the for-loop in 7, check if the word and keyword matches letter by letter by for-loop.
9. Within the for-loop in 8, if unmatched, terminate the for-loop in 8.
10. After terminating the for-loop in 8, if matched, store the keyword and terminate the for-loop in 7.
11. After terminating the for-loop in 7, if any keyword is stored, overwrite the word and store it in the words array created in 4. Make the first letter of the word uppercase if needed. Terminate the for-loop in 5.
12. After terminating the for-loop in 5, combine words array by space and overwrite the data.
13. After terminating the for-loop in 2, write data in the csv file in a new file. Processing completed here.

Result

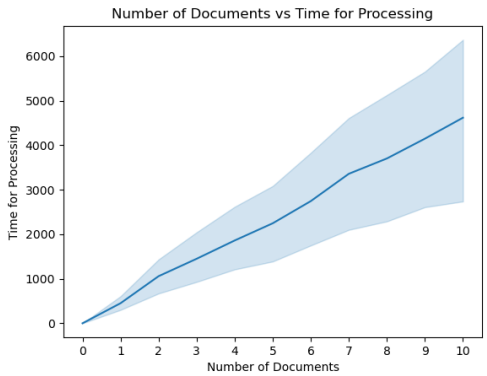
Plot1:

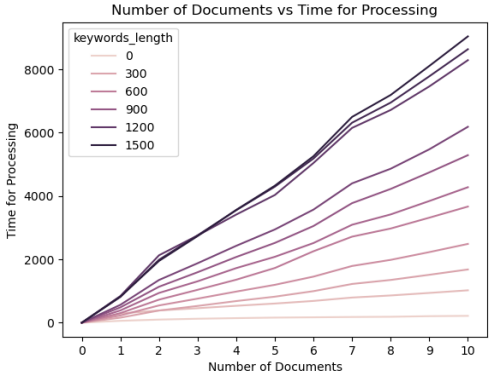




The above plot is the number of keywords versus the running time of the processing. As the plot indicates, the number of keywords and running time are in a proportional relationship. It can also be said that it is O(n) where n is the number of keywords. This observation is consistent as the analysis in the previous milestone which stated that O(d*w*n*m) where d is the number of documents, w is the number of words within each document, n is the number of keywords, and m is the length of the current word/keyword. As it is multiplied the observation of proportional relationship justified the analysis.

Plot2:





The above plots are the number of documents versus the running time of the processing. As the plot indicates, the number of documents and running time are in a proportional relationship. Same as the plot1, it is O(d) and justifies our analysis.

Unexpected Cases/Difficulties

There are some unexpected cases and difficulties due to the uncertainty in input and limitation of the coding.

Input uncertainty

- ➔ There are a limited number of keywords stored in abbreviations in the list. If uncovered abbreviations show up, the program is not able to replace it with its formal phrase.
- ➔ Processing the abbreviations that hold multiple meanings may collapse the context.

Limitation

- ➔ When trimming the special characters attached to the word, it only checks the first and last letter of the String. Therefore if more than 2 special characters are attached to the word, the word is not properly trimmed and it results in failure of matching.
- ➔ Some of the abbreviations have special characters at the beginning or end of the keyword (e.g. w/ for with). However, since the program trims the String to deal with the word at the special cases such as end of the sentence, it is not possible to check these abbreviations with current implementations.

Task Separation and Responsibilities

Anitej Isaac Sharma Preprocessing of Dataset	**Yuki Isomura** Preprocessing of Dataset Dataset Description Implementation and Testing of Algorithm and Description Unexpected Cases/Difficulties	**Abdirahman Hajj Salad**