Project Documentation

Table of Contents

1. Notes

- **I have injected the values statically through Application.prorperties. I made it that way instead of dynamic assignment since i focused more on the functionality So to use the app make sure to assign the paths in Applications. Properties and that the pool of files only contain .txt files**

- I have created this application based on the assumptions that were given. There is a different implementation to this project (using a Tasklet instead of a Step to read the files in bulk).

- Also, I have picked this algorithm based on its simplicity and efficiency, since a certain accuracy rate wasn't specified.

- I didn't want to add more details and complexity to the project to make it simple.

I have used Java 17 with Dependencies

- Lombok
- Spring Batch Starter (Spring Batch 5.2.2)
- H2 database

2. Introduction of the Algorithm Used

In this project, the Weighted Jaccard Similarity algorithm is utilized to measure the similarity between two files.
Every file gets translated into a map of words and their number of occurrences.

Then, it calculates the sum of the minimum and the sum of the maximum frequencies for all words, and computes the score as follows:

Score = (Sum of Minimums) / (Sum of Maximums)

Example:
Text 1
“ hello this is me “

Text 2
“ hello hello this is me away”

Set 1
(hello, 1), (this, 1), (is, 1), (me, 1)

Set 2
(hello, 2), (this, 1), (is, 1), (me, 1), (away, 1)

All Combined Words with Frequencies:
- hello → (1, 2)
- this → (1, 1)
- is → (1, 1)
- me → (1, 1)
- away → (0, 1)

Sum of Minimums = 1 + 1 + 1 + 1 + 0 = 4

Sum of Maximums = 2 + 1 + 1 + 1 + 1 = 6

Final Score = 4 / 6 = 66%

3. Project Structure

This project uses Spring Batch to process files one by one
I have used a Job with only a single Step. It goes as follows there are three passes provided (the given file path, the pool of files (directory path), the output file path). The given file would be loaded and read once while Spring batch would be responsible of going through every file in the directory one by one read it and then process the matching score against the given file then the score is then written into the output file. I have made a data model (File record) to hold the values of the data used throughout the whole process.
Read file → Process the matching score → Write it into a file

Project Files Overview:

- **FileItemReader**: Reads the directory and process1es each file one by one, passing them to FileItemProcessor.
- **FileItemProcessor**: Transforms each file into a word frequency map, calculates the similarity score, and sends it to ScoreItemWriter.
- **ScoreItemWriter:** Writes each calculated similarity score (in append mode) to an output file.
- **FileRecord:** Holds the file attributes throughout the entire processing step.
- **TargetFileLoader**: Loads the target file once and pre-computes its map for repeated comparison.
- **MapService**: Normalizes strings by converting them to lowercase, filtering, and converting to a word-frequency map.
- **CompareService:** Applies the Weighted Jaccard Similarity algorithm to calculate the similarity score.
- **Config**: Contains Spring Batch configuration for the Job and Step.
- **TaskApplication**: The main class that launches the Spring Batch job.