CalPolyPomona

Assignment 1

1. Web Crawling
2. Tokenization & Stemming
3. Text Analysis

1

1. Web Crawling

A. Create Web Crawler
- **Input**: The crawler should support 1 or more seed URLs, and 0 or more domain restrictions (e.g. only cpp.edu)
- For each URL
  - Check if the crawled page is in a desired language (see below)
  - Download the complete textual page content (including all html tags) into a folder called "**repository"**
  - Add the page URL and the number of outlinks from that URL to a file called "**report.csv".**
  - Crawl all outlinks
- **Output**: "repository" folder and "report.csv"

B. Run crawler
- 3 different domains, each in a different language
- Minimum 50 pages per crawl
- At the end of each crawl, your program should have (for each domain/language):
  - Downloaded the content from all crawled pages into the "repository" folder
  - Generated one "**report.csv"** file, which contains the following data for each downloaded page:
    - URL, number of outlinks

2

2. Tokenization and Stemming

A. Tokenization
- **Input**: all individual crawled documents
- Remove all code fragments (i.e. html, css, javascript)
- All remaining words/sequences of alphanumeric characters are retained
- **Output:** tokenized documents

B. Stemming
- **Input:** tokenized documents
- Apply any stemmer of your choice to all tokenized documents
- **Output:** stemmed documents

3

3. Text Analysis

A. Zipf's law Analysis
i. Calculate stem frequencies and ranks
ii. Plot stem frequencies and stem ranks
iii. Check if crawled content follows Zipf's law (i.e. compare with typical Zipf's distribution)

B. Heap's Law Analysis
i. Plot vocabulary growth by collection growth, e.g. as shown in examples in class

4

Submission

1. **Code for parts 1 & 2**
2. **Output folder** (see next slide)
3. **Report** (in PDF - 5 page max), containing:
   - The main system components for 1A, 2A, 2B
   - Word frequency/rank plots for your 3 different crawls (i.e. Zipf's law plots), including a discussion on whether the 3 word distributions follow Zipf's law
   - Heap's law plots for your 3 different crawls, including a discussion on whether the word distributions follow Heap's law
   - Any challenges faced during the development
   - Discussion about each team member's contribution
   - *Note*: Plots and Screenshots can also be added to the appendix, and do not count towards page limit

5

Submission format

One .zip file: **GroupX.zip**

- **Code (folder)**
  - CodePart1 (folder)
  - CodePart2 (folder)
- **Output (folder)**
  - Report1.csv (rows of URL, #outlinks)
  - Report2.csv (rows of URL, #outlinks)
  - Report3.csv (rows of URL, #outlinks)
  - Words1.csv (50 most frequent stems for crawl 1)
  - Words2.csv (50 most frequent stems for crawl 2)
  - Words3.csv (50 most frequent stems for crawl 3)
- **Report.pdf**

6

Deadlines

- Pick group (8 groups of 5, 3 groups of 4) and domains:
  - **Wednesday, February 19th, 2025, end of day**
  - https://docs.google.com/spreadsheets/d/1rYbRtmx0j66Ke_FQTQiPqF3L2RVBaFk2-ApC9sGXO1E/edit#gid=0
- Submit report and code:
  - **Wednesday, March 5th, 2025, end of day**
  - **Canvas**

Example Resources

Reading from and Writing to a URLConnection
https://docs.oracle.com/javase/tutorial/networking/urls/readingWriting.html

HTML parser (e.g. for link extraction)
https://jsoup.org

Language detection
https://github.com/detectlanguage/detectlanguage-java
https://github.com/pemistahl/lingua-py

Stemming
https://www.nltk.org/howto/stem.html