COSC 320 – 001
Analysis of Algorithms
2022/2023 Winter Term 2

Project Topic Number: 1

Project Proposal (Group 32)

Keyword Replacement in Corpus

Group Lead: Anitej Isaac Sharma

Group Members:

Anitej Isaac Sharma

Abdirahman Hajj Salad

Yuki Isomura

Abstract

This is a project proposal for Keyword Replacement (in Corpus) Algorithm, outlining the problem description, real-world examples, edge cases, expected complexities, dataset collection, choice of programming language(s), task separation and responsibilities, and unexpected cases/difficulties.

Problem Description

The complexity of the human brain tends to aim for efficiency. Efficiency can be referred to in various contexts but in terms of a corpus, it means conveying ideas and perspectives in the simplest and least time-consuming manner. Therefore, there exist millions of documents that include one or more abbreviations, which can be ambiguous or confusing, especially for non-native speakers or people with limited reading skills.

For people with cognitive or learning disabilities, abbreviations may not be comprehensible. Obstacles like these can create communication barriers and limit access to information. Furthermore, using abbreviations in the corporate world, for example, could display a lack of professionalism which can lead to undermining one's character in various contexts. Our goal is to ease such tensions by creating a keyword replacement tool to convert abbreviations to their original forms/phrases.

Real-World Examples

Peer Note-Taking.

The Disability Resource Center (DRC) accommodates students with disabilities and assigns them a peer in each of their courses who attends classes and creates legible notes for them. These notes are then analyzed for relevance and correctness. The analyzing process replaces all the abbreviations (if there are any) with their corresponding phrases, after which, the notes are then provided to the students being accommodated by DRC.

Web Search, Sites, and Services

People often use abbreviations in their search input, however, popular search engines, such as Google, automatically convert these abbreviations to corresponding phrases to deliver accurate results. Many search engines have downloadable extensions for their browsers that automatically convert abbreviations on different web pages to their original phrase/text. They also have built-in tools for converting abbreviations to corresponding phrases in their web services, such as Google Docs.

Medical Transcription

In the field of medical science, abbreviations are often used when documenting information and simplifying complex or scientific words/sentences. In such cases, a keyword replacement tool is used to convert the shorthands to their original forms for better understanding and communication.

Edge Cases

I. Since our focus is to work with the commonly used abbreviations, we may not have all existing keywords in our database as abbreviations aren't generic.

II. A keyword could have multiple meanings. For example, the keyword *AAA* could stand for *Administration, Authorization, and Authentication*, or it could also mean, *American Automobile Association* or *Anti-Aircraft Artillery*.

III. The input could be faulty. For example, in a text document given as an input, if an abbreviation and a word(s) are not separated by a space then it might not be recognized as an

abbreviation. Or, in some cases, if two or more words are not separated by a space, a keyword in our database could be similar to the words glued together, which could be recognized as an abbreviation and converted to a phrase that could be totally out of context.

IV. A recognized abbreviation is not an abbreviation. For example, we all know the word *CPU* as Central Processing Unit, right? Yes, but maybe *CPU* in a text document isn't referring to the brain of a computer but rather a company named *CPU*.

Expected Complexities

Naïve Algorithm A (Brute-Force Search) → $O(n^2 \bullet m)$
Refined Algorithm B (HashMap Implementation) → $O(n \bullet m)$
Note: n is the number of words in a document and m is the number of documents.

Dataset Collection

Our plan is to web scrape data from websites with a list of abbreviations and their full forms, like Webopedia, Wikipedia, GitHub, etc. There are other accurate sources for collecting datasets, such as Kaggle, which we will be using as well. We, then, store the web-scraped abbreviations and corresponding phrases in a text file to create a collective dataset to be used for training the keyword replacement algorithm/model.

Choice of Programming Language(s)

Python for data extraction (web scraping).
Java for modeling and implementing the Keyword replacement algorithm, and generating run-time plots.

Task Separation and Responsibilities

Data Extraction (Anitej, Hajj)	**Problem Formulation** (Hajj, Yuki)	**Pseudocode** (Anitej, Yuki)
Analysis (Anitej, Yuki, Hajj)	**Algorithm Modelling** (Anitej, Hajj, Yuki)	**Test Cases** (Anitej, Yuki)
Run-time plots (Hajj)	**Implementations** (Yuki, Hajj)	**Video** (Anitej)

Timelines

1. Problem Formulation, pseudo code, analysis (proof of correctness and running time of both the algorithms), and Description of the chosen data structures and rationales [By Feb 10]
2. Implementation of the first algorithm, test cases, and run-time plots [By Feb 27]
3. Implementation of the second algorithm, test cases, and run-time plots [By Mar 20]
4. Shoot a video [By Apr 10]

Unexpected Cases/Difficulties

- ➔ Input Format Uncertainty (ex. tweets can be in different languages)
- ➔ Typographical Errors (ex. multiple periods at the end of the sentence or misspelled words)