Hapax Analysis

Student Name: Kıvanç Gördü

Subject: Grammar Theory (KOL/TEGRA)

Department: Department of Linguistics

School: Palacký University Olomouc

Date: 27.04.2025

Github Profile: https://github.com/kivanc57

Project Overview:

This project performs an analysis on text files in a specified directory, extracting **hapax legomena** (words that appear only once) and saving the results in an Excel file.

More specifically, the program reads all text files from a specified directory, processes the text by *tokenizing* it, optionally *lemmatizing* words, and calculates the *frequency* of each word. Later, it extracts words that only appear once (hapax legomena) and writes them to an Excel file. The entire project is built on the Go programming language which is an extremely fast and efficient language.

More details about the project can be found below on the Github repository:

https://github.com/kivanc57/hapax_analysis

Features:

- **Read text files** from a specified directory.
- **Tokenize text** and **lemmatize** (optional).
- **Count word frequency**, filtering out non-hapax words.
- **Save results** in an Excel file, listing hapax legomena.

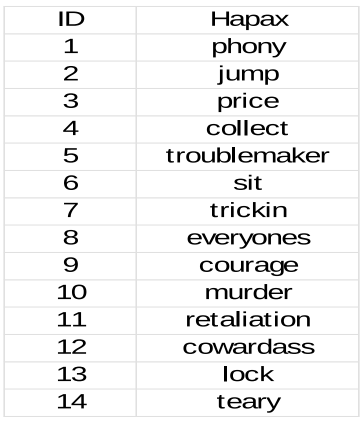
My Usage Case

I curated the scripts and utilities to realize the project. I start by reading text files exclusively in the given directory, byte by byte. I write the content of each text file to a **strings.Builder** to efficiently concatenate large files. I later assign this to a variable named corpus.

Subsequently, I use the **GetFreqMap** function, which is a key component. By adding *regex* functionality with the built-in regexp library, I compile the expression **[^a-z]+** immediately after converting the characters to lowercase. This expression makes it possible to replace everything that is not an alphabetic character in the given string. I also check the *lemmatize* option; if it is enabled, I lemmatize using the Lemmatize function from the golem library. For each token or lemma, I increment its count in the frequency map. The function then returns this map to the outer scope, to be passed to another function.

Ultimately, I pass this frequency map to the **WriteExcel** function, where I utilize the *excelize* library. First, I create the Excel file and define a style variable to format the ID header. I create the "ID" and "Hapax" headers. Then, for each word and its count in the frequency map, I check if the count is equal to 1 (the simplest way to identify hapaxes). Following this, I write each hapax and its corresponding ID (which I increment after every iteration where the condition is met), and I save the file as “hapax_lists.xlsx” in the project's root directory and this entire process is happening *within seconds*.

A small excerpt of the Excel sheet is attached below to illustrate the findings after processing some of 2Pac's songs.



ID	Hapax
1	phony
2	jump
3	price
4	collect
5	troublemaker
6	sit
7	trickin
8	everyones
9	courage
10	murder
11	retaliation
12	cowardass
13	lock
14	teary

Installation

1. Clone this repository.

2. Make sure Go is installed. If not, install Go from the official website: https://golang.org/dl/

3. Install the required Go packages:

   ` go get github.com/aaaton/golem/v4`

   ` go get github.com/xuri/excelize/v2`

4. Install the required dependencies for your project:

   ` go mod tidy`

Script Usage

1. Change the 'absDirPath' in 'main.go' to the path of your text files directory.

2. Run the main program:

   ` go run main.go`

3. After running the program, check the output file 'hapax_list.xlsx' for the **list of hapax legomena.**

License

This project is licensed under the GNU 3.0 License - see the GNU GENERAL PUBLIC LICENSE file for details.

Acknowledgments

- This project uses the Golem library for lemmatization.

- The output is saved using the excelize library.