Moogle!

Fabián Alejandro Almeida Martínez

C112

Curso 2023-2024

Para la realización de este proyecto llamado Moogle!, que consiste en un buscador de texto inteligente, tuve q organizar el trabajo de forma progresiva y eficiente, trazando objetivos q ir cumpliendo paso a paso hasta tenerlo completo. Dicho esto solo tenía que buscar por dónde empezar y, dado q es un procesador de texto, pues lo más inteligente seria cargar y guardar los textos que queremos procesar.

Para esta primera tarea utilice diccionarios como estructura de datos ya que, dada su versatilidad y funcionamiento facilitarían el trabajo, pero antes cargue y guarde la ruta de los archivos, retire los signos de puntuación innecesarios y normalice los textos.

Luego cree dos diccionarios, uno para guardar cuantas veces se repite cada palabra en cada texto y otro las veces q se repite cada palabra en general. Para esto utilice una tupla conformada por ambos diccionarios.

Con esto ya tendría las piezas para implementar el modelo vectorial.

Mis primeros pasos en la construcción del modelo vectorial fueron la implementación del TF (Term Frecuency) o frecuencia normalizada que consiste en determinar la frecuencia de aparición de una palabra en un texto y dividirla por la frecuencia de la palabra más frecuente en dicho texto y el IDF (Inverse Document Frecuency) dada por el logaritmo del resultado de dividir la cantidad de documentos que estamos procesando entre la cantidad de documentos q aparece la palabra deseada.

Luego de esto procedemos a calcular el peso de cada palabra, para esto procedemos a calcular el producto del TF y el IDF, aplicando un suavizado para disminuir la diferencia entre el peso de los términos de poca frecuencia.

Una vez hecho esto, también implementé un método para calcular el peso de la query, esto lo hice aplicando el mismo procedimiento que para el resto de palabras de los documentos.

Solo quedaría entonces la implementación de un método que registre el score de cada documento, para esto, hallé el score dada la formula que me permite hallar la similitud entre la query y los documentos y luego, registre los datos en un diccionario en el cual a cada documento se le asigna su score.

Otra de las tareas a realizar sería la de otorgar al usuario una sugerencia lo mas similar posible a su query que pueda ser encontrada en el documento, para esto usamos la distancia de Levenshtein, que nos permite determinar la similitud entre palabras o un conjunto de ellas.

Posteriormente a esto procedí a implementar un método que seria llamado desde el Moogle Server q serviría para q el programa cargara todos los datos (TF, IDF, Peso, etc…) una vez se abriera el proyecto y los mantuviera guardados, también procedí a implementar el snippet.

El último paso sería cargarlo y acomodarlo todo, ordenar los resultados de mayor a menor score y mostrar la sugerencia y el snippet.

PD: También han sido implementadas algunas operaciones con matrices las cuales no han sido utilizadas en el proyecto.