Proyecto de Programación I.Facultad de Matemática y Computación - Universidad de La Habana.Cursos 2021, 2022

Edward

July 21, 2023

Moogle! es una aplicación web cuyo propósito es buscar inteligentemente un texto en un conjunto de documentos.Está desarrollada con tecnología .NET Core 6.0, específicamente usando Blazor como "framework" web para la interfaz gráfica, y en el lenguaje C Sharp.
La aplicación está dividida en dos componentes fundamentales:

- "MoogleServer" es un servidor web que renderiza la interfaz gráfica y sirve los resultados.
- "MoogleEngine" es una biblioteca de clases donde está implementada toda la lógica del programa

La búsqueda se realiza lo más inteligente posible, por ese motivo no nos limitamos a los documentos donde aparece exactamente la frase introducida por el usuario, sino que también se realiza una búsqueda por parecido entre las palabras.

El proyecto por defecto tiene tres clases; una clase Moogle donde voy a implementar lo necesario para que el programa busque y dos clases que me ayudan a organizar esos resultados de la búsqueda .La primera es searchitem para caracterizar a un resultado de la búsqueda la misma tiene tres propiedades ,el title snippet score esta última es que tanto se parece a lo que estoy buscando y es lo que utilizo para ordenar los resultados en relación a la concordancia que tienen o la similitud que tienen con la query , el snippet es un fragmento que voy a devolver junto con ese resultado mostrando que ese resultado que devuelve tiene realmente que ver con mi query, el title es el título del libro que estoy devolviendo. Para este proyecto implemente las siguentes clases comenzando por la clase Book para guardar todo lo referente a la información necesaria para la búsqueda de un libro o sea todo lo que se necesita sacar de

un libro, clase Library será mi biblioteca para guardar la información de varios libros para tener más información creando así un conjunto de documentos más organizados, clase toolsBox que contiene métodos útiles que se usarán más adelante para poder realizar la búsqueda estos códigos son generalmente recurrente para usarlos varias veces llamándolos desde cualquier otro lugar desde cualquier clase, SearchEngine este sería el motor de busqueda para construir la biblioteca para poder tener toda la información almacenada aquí se realizan todos los procesos para poder realizar la búsqueda es donde también esta implementado el modelo vectorial utilizado para poder dar los resultados de acuerdo a la búsqueda y en el método search se construyen los resultados de búsqueda, método que llama a todos los demás métodos ya sea de las demás clases y de la propia clase en sí para poder construir el resultado que queremos mandar, devolviendo un serchresult que en la clase Moogle simplemente tenemos que generar una instancia de la clase SearchEngine iniciarla y preguntarle a esa instancia por los resultados de la búsqueda, a través del método Search.

El modelo vectorial se basa en lo siguiente:
Toma la consulta, que consiste en un conjunto de palabras y la convierte en el vector $\vec{q} = (w_1, w_2, \ldots, w_n)$ donde cada componente se refiere al peso de una palabra de la consulta Luego, usando la mismas palabras de la consulta se construyen los vectores $\vec{d_j} = (w_{1j}, w_{2j}, \ldots, w_{nj})$ con $j = 1; 2; \ldots; D$ siendo D la cantidad de documentos sobre los que se hará la búsqueda.
Los valores de las componentes de cada vector se calculan según las fórmulas siguientes:

$$w_i = (a + (1-a) * \frac{TF_i}{maxTF}) * \log_{10}(\frac{D}{D_i}) \tag{1}$$

$$w_{ji} = \frac{TF_{ij}}{maxTF_j} * \log_{10}(\frac{D}{D_i}) \tag{2}$$

Donde TF_i es la cantidad de veces que se repite la palabra i en la consulta, $maxTF$ es la cantidad de veces que se repite la palabra que más se repite en la consulta, a es un valor de amoriguamiento que se en cuentra en el intervalo $(0; 1)$; se suele escoger un valor entre 0.5; y D_i es la cantidad de documentos que contienen a la palabra i. TF_{ij} es la cantidad de veces que se repite la palabra i en el documento j y $maxTF_j$ es la cantidad de veces que se repite la palabra que más se repite en el documento j. Cada uno de estos valores pueden ser obtenidos de la clase Library ya que son datos que se computan directamente de los documentos, y estos datos se computan a la hora de cargar todos los documentos para la aplicación.

De la asignatura de Álgebra conocemos que:

$$\vec{q} * \vec{d_j} = \sum_{i=1}^{n} w_i * w_{ji} \tag{3}$$

$$\vec{q} * \vec{d_j} = |\vec{q}| * |\vec{d_j}| * \cos \angle(\vec{q}, \vec{d_j}) \tag{4}$$

$$|\vec{q}| = \sqrt{\sum_{i=1}^{n} w_i^2} \tag{5}$$

$$|\vec{d_j}| = \sqrt{\sum_{i=1}^{n} w_{ji}^2} \tag{6}$$

Sustituyendo (3), (5), (6) en (4) se obtiene:

$$\sum_{i=1}^{n} w_i * w_{ji} = \cos \angle(\vec{q}, \vec{d_j}) * \sqrt{(\sum_{i=1}^{n} w_i^2) * (\sum_{i=1}^{n} w_{ji}^2)} \tag{7}$$

y despejando $\cos \angle(\vec{q}, \vec{d_j})$ en (7) obtenemos:

$$\cos \angle(\vec{q}, \vec{d_j}) = \frac{\sum_{i=1}^{n} w_i * w_{ji}}{\sqrt{(\sum_{i=1}^{n} w_i^2) * (\sum_{i=1}^{n} w_{ji}^2)}} \tag{8}$$

Donde $\cos \angle(\vec{q}, \vec{d_j})$ es la similitud entre la consulta $\vec{q}$ y el documento $\vec{d_j}$

Para una mayor eficiencia de la aplicación, cada dato es almacenado en una matriz de la siguiente forma; una matriz con el tf de cada palabra de la consulta por cada documento,

$$\begin{pmatrix} \frac{TF_{11}}{maxTF_1} & \frac{TF_{12}}{maxTF_1} & \dots & \frac{TF_{1n}}{maxTF_1} \\ \frac{TF_{21}}{maxTF_2} & \frac{TF_{22}}{maxTF_2} & \dots & \frac{TF_{2n}}{maxTF_2} \\ \dots & \dots & \dots & \dots \\ \frac{TF_{m1}}{maxTF_m} & \frac{TF_{m2}}{maxTF_m} & \dots & \frac{TF_{nm}}{maxTF_m} \end{pmatrix}$$

y un vector con el idf de cada palabra de la consulta en el conjunto de documentos $\vec{p} = (p_1, p_2, \dots, p_n)$ donde se cumple que $p_i = \log_{10}(\frac{D}{D_i})$; de esta forma, aplicar los operadores se reduce a alterar los valores de las componentes del vector $\vec{p}$ o las componentes de la matriz presentada.
Luego, cada componente del vector $\vec{p}$ se multiplica por la columna correspondiente, obteniendo así la matriz

$$W = \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \dots & \dots & \dots & \dots \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{pmatrix}$$

Luego, haciendo

$$\vec{q} = \begin{pmatrix} w_1 \\ w_2 \\ \dots \\ w_n \end{pmatrix}$$

si efectuamos el producto $W * \vec{q} = \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \dots & \dots & \dots & \dots \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{pmatrix} * \begin{pmatrix} w_1 \\ w_2 \\ \dots \\ w_n \end{pmatrix}$

se obtiene $\theta = \begin{pmatrix} \sum_{i=1}^{n} w_i * w_{1i} \\ \sum_{i=1}^{n} w_i * w_{2i} \\ \dots \\ \sum_{i=1}^{n} w_i * w_{mi} \end{pmatrix} = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \dots \\ \theta_m \end{pmatrix}$

Sea
$\mathrm{p}_j = \sqrt{(\sum_{i=1}^{n} w_i^2) * (\sum_{i=1}^{n} w_{ji}^2)}$

si hacemos $\theta' = \begin{pmatrix} \frac{\theta_1}{p_1} \\ \frac{\theta_2}{p_2} \\ \dots \\ \frac{\theta_m}{p_m} \end{pmatrix}$

entonces θ' será el vector con las similitudes de cada documento con la consulta después de haber aplicado los operadores de búsqueda a toda la consulta.