Proyecto Final Sistemas de Recuperación de la Información

Alejandro Campos, Darian Dominguez

Facultad de Matemática y Computación
Universidad de la Habana
2021

Resumen

En el mundo de hoy se tiene acceso a mucha información. Sería muy complicado para una persona buscar, entre millones y millones de documentos, cuáles serían útiles y relevantes para su objetivo. Los sistemas de recuperación de la información facilitan este trabajo. Estos reciben una consulta por parte del usuario y procesan toda la colección de documentos, que luego de analizarlos y representarlos de una forma que el sistema entiende, devuelven en un corto período de tiempo aquellos archivos más relevantes a la petición del usuario.

I. Introducción

Realizar búsquedas de información es una actividad diaria en todos los ambientes que reviste especial importancia. En áreas de investigación es una actividad que puede insumir un tiempo importante entre operaciones que van desde: definir los criterios, iniciar la búsqueda, filtrar los resultados, encontrar la información y luego tenerla a disposición de una manera clasificada y ordenada. La fase de búsqueda se inicia sobre las expectativas planteadas por el investigador y finaliza cuando el mismo encuentra finalmente el o los documentos que poseen la información de interés. [1]

Varios documentos pueden responder a la consulta del usuario, por ello, los sistemas implementan un ranking de documentos ordenados según la relevancia a la petición del usuario, dependiendo del modelo sobre los que son implementados. Existen discímiles modelos para recuperar información, estos se diferencian en la manera de representar los documentos y la consulta, en dependencia de los términos que la conforman. A partir de esta forma de representación, se le asocia a cada documento un valor calculado mendiante una función de ranking, que varía en dependencia del modelo utilizado.

Este informe tiene como objetivo describir el diseño, implementación, evaluación y análisis de un Sistema de Recuperación de Información.

II. Diseño del sistema

Luego del estudio y análisis de los modelos estudiados en [2], se decidió usar el modelo vectorial para el desarrollo del sistema de recuperación de la información. En la presente sección se explicarán los motivos de esta selección, no sin antes explicar en qué consiste este modelo.

Brevemente, según este modelo, cada documento es representado mediante un vector de n elementos, siendo n igual al número de términos indizables que existen en la colección documental. Hay, pues, un vector para cada documento, y, en cada vector, un elemento para cada término o palabra susceptible de aparecer en el documento. Cada uno de esos elementos es cubierto u ocupado con un valor numérico. Si la palabra no está presente en el documento, ese valor es igual a 0. En caso contrario, ese valor es calculado teniendo en cuenta diversos factores, dado que una palabra dada puede ser más o menos significativa (tanto en general como, sobre todo, en ese documento en concreto); este valor se conoce con el nombre de peso del término en el documento. Las consultas son representadas también mediante un vector de las mismas características que las de los documentos. Esto permite calcular fácilmente una función de similaridad dada entre el vector de una consulta y los de cada uno de los documentos.[3]

El modelo vectorial es ampliamente usado en operaciones de recuperación de información, así como también en operaciones de categorización automática, filtrado de información, etc. Aunque tiene la desventaja de asumir independencia de términos, el cálculo del valor asociado a cada documento para realizar el ranking se determina por la cercanía de este con respecto a la consulta del usuario, definida por el coseno del ángulo entre el vector que representa al documento y el de la consulta. Podemos decir, además, que el esquema de ponderación de este modelo (tf-idf como se verá posteriormente) permite un mejor rendimiento en la recuperación de información.

III. Implementación del sistema

El sistema se ha implementado usando el lenguaje de programación python, cuyas bibliotecas ofrecen facilidades para el trabajo con el procesamiento textual. La implementación se encuentra en la carpeta **code**. En las siguientes subsecciones se explica el uso de cada uno de los módulos más importantes del sistema.

I. Procesamiento textual

En el módulo `text_processing.py` se realizan todas las fases del preprocesamiento textual, tokenización, normalización, lematización o reducción de las palabras a su raíz gramatical y eliminación de stopwords. Para esto se usó la librería *nltk*.

II. Representación de documentos y consultas

`representation.py` provee métodos para la representación de documentos y consultas. En este se calcula el tf de cada término en cada documento, así como idf de cada término y, a través de estos, se determina el peso de cada término i en el documento j (w_{ij}). Se precisan a continuación cada uno de los términos anteriores.

$$tf_{ij} = \frac{freq_{ij}}{max_i freq_{ij}}$$

donde el numerador corresponde a la cantidad de veces que se repite el término i en el documento j, y el denominador es la cantidad de veces que se repite el término que más se repite en el documento j. También se tiene que

$$idf_i = log\frac{N}{n_i}$$

donde N corresponde a la cantidad total de documentos de la colección y n_i a la cantidad de documentos en los que aparece el término

i. Finalmente, el peso de un término en el documento, viene dado por

$$w_{ij} = tf_{ij} * idf_i$$

. Como ya se dijo, los documentos son represantados como vectores, cuyas componentes vienen dadas por los pesos de los términos que aparecen en estos.[4]

Las consultas también son representadas como vectores, el peso de un término en la consulta q viene dado por

$$w_{iq} = (a + (1 - a)tf_{ij}) * idf_i$$

donde a usualmente toma valor 0.4 o 0.5 [4]. Notemos que si a = 0 el peso de la consulta es el mismo que el del documento.

III. Documentos

Los documentos deben encontrarse en la carpeta "docs" en el mismo directorio del código del proyecto. El sistema trabajará con la colección de archivos con extensión .txt localizados en dicha ruta. En `document_processing.py` se procesan todos los documentos de la colección. A sus textos se les realiza el procesamiento textual y se calcula el peso de cada término resultante en cada documento.

IV. Función de similitud

El módulo `sim.py` contiene el cálculo de la función de similitud, mediante la cual se realiza el ranking de los documentos. Acorde al modelo empleado, esta función se calcula como el coseno del ángulo comprendido entre el vector que representa cada uno de los documentos y la consulta.

$$sim(d_j, q) = \frac{\vec{d_j}\vec{q}}{|\vec{d_j}||\vec{q}|},$$

esto es

$$sim(dj, q) = \frac{\sum_{i=1}^{n} w_{ij} w_{iq}}{\sqrt{\sum_{i=1}^{n} w_{ij}^2} \sqrt{\sum_{i=1}^{n} w_{iq}^2}}$$

[4]

V. Procesamiento de consulta y devolución de resultados

Al texto de la consulta se le realiza un preprocesamiento, como se expuso en la sección correspondiente. Luego se le calcula el peso de cada término para formar el vector consulta, a partir del cual la función de similitud le asocia un valor a cada documento y devuelve el ranking de estos, a partir de un umbral que se define previamente. Esto se realiza en `query_processing.py`.

VI. Expansión de consulta

El sistema implementa la expansión de consulta, lo cual permite expandir las consultas ingresadas por el usuario a fin de ampliar el espectro de las búsquedas que se realicen. El módulo `query_expansion.py` implementa esta funcionalidad, donde por cada término t de la consulta, esta es expandida automáticamente con sinónimos y palaras relacionadas con t.

VII. Clustering

Los documentos se agrupan según su similitud de acuerdo al contenido que presentan. El módulo `clustering.py` utiliza el algoritmo k-means con este fin. De esta forma el usuario recibe la colección de documentos agrupados por tópicos, lo cual es de utilidad para explorar documentos similiares a los devueltos como resultado en la consulta, y explorar los documentos de aquellos temas que le resulten de interés.

VIII. Main

El flujo del sistema es controlado por el módulo `main.py`, por el momento, en esta primera versión que se presenta, la consulta debe insertarse donde indica los comentarios del código. Igualmente puede modificarse el umbral a partir del cual se devuelve el ranking de documentos.

IV. Modelo booleano

Con el fin de comparar el sistema desarrollado, se implementó otro modelo: el modelo booleano. Este modelo se basa en la teoría de conjuntos y el álgebra booleana para devolver los resultados [4]. En este modelo las consultas están diseñadas como expresiones booleanas que tienen una semántica precisa y, considera solo si los términos indexados están presentes o ausentes en un documento. La estrategia de recuperación se basa en un criterio de decisión binario.

En el módulo `boolean_model.py` se encuentra la implementación de este modelo. Luego del preprocesamiento del texto de los documentos, estos se expresan como vectores binarios atendiendo a la presencia o no de términos indexados. Posteriormente, despúes del preprocesamiento y expansión de la consulta, esta se convierte en una expresión booleana de tal forma que si el término t_q de la consulta está presente en los términos indexados, entonces la componente correspondiente al término es 1, en otro caso toma valor 0.

Con la representación booleana abordada anteriormente, la función de similitud en este modelo devuelve aquellos documentos que "se parezcan más" a la consulta, es decir, aquellos documentos que tengan más componentes en común con esta.

V. Interfaz de usuario. Manual de usuario

VI. Evaluación del sistema

Para evaluar un sistema es necesario utilizar colecciones de prueba, las cuales contienen los documentos y, además, información acerca de los documentos relevantes para un conjunto de consultas. Las pruebas utlizadas para evaluar el presente sistema son las colecciones de Cranfield y Medline, que se encuentran en **code/collections/**. El módulo `evaluation.py` se encarga de procesar los documentos y las consultas de cada una de las colecciones antes mencionadas, devolviendo las métricas estudiadas durante el curso de SRI, que nos permitirán evaluar el sistema implementado.

Es importante aclarar que, dado el tamaño que presentan las colecciones, el proceso de evaluación se demora considerablente, ya que procesa los documentos para dos modelos, además de un gran número de consultas.

I. Métricas utilizadas

En [5] se proponen varias métricas para evaluar un sistema de recuperación de la información y, para un mejor entendimiento de sus fórmulas se aclaran las siguientes notaciones:

1. **REL** conjunto de documentos relevantes
2. **REC** conjunto de documentos recuperados
3. **RR** conjunto de documentos relevantes recuperados
4. **NN** conjunto de documentos no relevantes no recuperados
5. **RI** conjunto de documentos recuperados irrelevantes
6. **NR** conjunto de documentos no recuperados relevantes

[5]

Las medidas usadas son las siguientes:

- Precisión.

$$P = \frac{|RR|}{|RR \cup RI|} = \frac{|RR|}{|REC|}$$

Esta medida evalúa la consulta de acuerdo a los documentos relevantes que se recuperan[5]. Despuées de tener los documentos que fueron recuperados en la consulta, se buscan cuáles son relevantes con ayuda de la información brindada por la colección, y teniendo esto, se aplica la fórmula. Este proceso se realizó para cada consulta de la colección y los resultados obtenidos fueron promediados.

- Recobrado.

$$R = \frac{|RR|}{|RR \cup RN|} = \frac{|RR|}{|REL|}$$

fracción de los documentos relevantes que fueron recuperados[5]. Con ayuda de la información que brinda la colección se buscan los documentos relevantes para la consulta, teniendo ya los documentos recuperados se va analizando de estos, cuáles son relevantes, e igualmente se aplica la fórmula. El proceso se realiza para cada consulta de la colección y los resultados obtenidos se promedian.

- Medida F.

$$F = \frac{(1+\beta^2)PR}{\beta^2 P + R} = \frac{(1+\beta^2)}{\frac{1}{P} + \frac{\beta^2}{R}}$$

enfatiza la precisión sobre el recobrado o viceversa[5]. Igualmente, el proceso se realiza para cada consulta de la colección y los resultados obtenidos se promedian.

- Medida F1.

$$F1 = \frac{2PR}{P+R} = \frac{2}{\frac{1}{P} + \frac{1}{R}}$$

armoniza precisión y recobrado teniendo en cuenta a ambos.[5] Para su cálculo se usó la precisión y el recobrado, previamente calculados, y se aplicó la fórmula. Igualmente, el proceso se realiza para cada consulta de la colección y los resultados obtenidos se promedian.

- R-Presición.

$$P_R = \frac{|RR|_R}{|RR \cup RI|_R} = \frac{|RR|_R}{|REC|_R}$$

es el ranking de documentos relevantes a una consulta para la cual existen R documentos relevantes[5]. Para su cálculo se seleccionaron los primeros R documentos recuperados y de estos se vieron cuáles son relevantes, de acuerdo a la información que brinda la colección y se aplica la fórmula. El proceso se realiza para cada consulta de la colección y los resultados obtenidos se promedian.

- Fallout

$$Fallout = \frac{|RI|}{|RI \cup NI|} = \frac{|RI|}{|I|}$$

tiene en cuenta la cantidad de documentos irrelevantes y el ranking [5]. Se puede calcular la cardinalidad del conjunto de los documentos recuperados irrelevantes con la diferencia de los documentos recuperados y de los recuperados relevantes. Por otro lado, con la diferencia del total de documentos y los documentos relevantes, se tiene el conjunto de los documentos irrelevantes; luego, se aplica la fórmula y se repite el proceso para cada consulta de la colección y los resultados se promedian.

II. Análisis de los resultados obtenidos

Para mostrar los resultados de la evaluación del método booleano y el vectorial en dos escenarios distintos, se debe abrir la consola en el directorio del proyecto y ejecutar `python evaluation.py` Esto ejecutará ambos modelos sobre las colecciones de Cranfield y Medline, mostrando el cálculo de las medidas de evalucación abordadas en la sección anterior.

VII. Conclusiones

VIII. Trabajo futuro

- Incorporar al sistema una interfaz de usuario agradable
- Añadir funcionalidades al sistema como expansión de consulta y clustering de documentos
- Poner dinámica la ruta de la colección de documentos
- Incorporar al informe evaluación del sistema a partir de las métricas estudiadas y analizar los resultados obtenidos con estas

- Incorporar al informe conclusiones y recomendaciones para el trabajo futuro con el sistema implementado.

IX. Bibliografía

[1] http://sedici.unlp.edu.ar/bitstream/handle/10915/43840/ Documento_completo.pdf?sequence=1&isAllowed=y

[2] Conferencias 2 y 3 SRI para la carrera de Ciencia de la Computación, Universidad de La Habana. Curso 2022

[3] https://bid.ub.edu/04figue2.htm

[4] Conferencia 2 SRI para la carrera de Ciencia de la Computación, Universidad de La Habana. Curso 2022

[5] Conferencia 4 SRI para la carrera de Ciencia de la Computación, Universidad de La Habana. Curso 2022

2