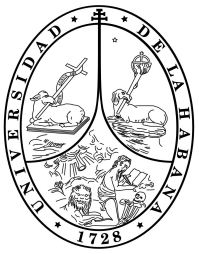

Informe Escrito de Sistema de Recuperación de Información

Thalia Blanco Figueras lia.blanco98@gmail.com
Grupo C512

Eziel C. Ramos Piñón ezielramos498@gmail.com
Grupo C511

Ariel Plasencia Díaz arielplasencia0@gmail.com
Grupo C512

Índice

1 Introducción **3**

2 Diseño del sistema **3**
2.1 Preprocesamiento de la consulta y los documentos . . . 3
2.2 Modelo de Recuperación de Información . . . 3
2.2.1 Modelo Booleano . . . 4
2.2.2 Modelo Vectorial . . . 4
2.3 Crawler . . . 5

3 Implementación **5**
3.1 Ejecución del sistema . . . 6

4 Evaluacion del sistema **6**
4.1 Medidas de evaluación . . . 6
4.2 Resultados obtenidos . . . 7

5 Recomendaciones **7**

6 Conclusiones **7**

1 Introducción

Los sistemas de recuperación de la información son sistemas sumamente importantes en el mundo de hoy, donde la gran mayoría de la información se encuentra digitalizada, inclasificada y mezclada. Estos tienen como objetivo, tal como su nombre lo dice, encontrar información en una amalgama de datos generalmente grande para satisfacer las necesidades del usuario que la busca. Los sistemas de la información deben ser precisos, y aportar una buena cantidad de respuestas de calidad, deben tener una buena comunicación con el usuario y ser relativamente fáciles de usar.

Para su construcción se pueden tomar y tener en cuenta diferentes modelos de recuperación, como parte principal a la que se adjuntan otras capas del sistema, en las que se toman decisiones de diseño igualmente importantes.

2 Diseño del sistema

El sistema desarrollado cuenta con varias funcionalidades. Primero, dado una consulta, es necesario realizar el preprocesamiento y la representación de la consulta y los documentos. Posteriormente, el Modelo de Recuperación de Información es utilizado para obtener una ordenación de los documentos relevantes. Además, con el objetivo de generar más información para el sistema se usaron técnicas de Crawling. A continuación, cada una de estas funcionalidades son explicada más extensivamente.

2.1 Preprocesamiento de la consulta y los documentos

Para el procesamiento de la consulta del usuario y de los documentos en el corpus es necesario representarlos de una manera lógica para ser procesados por la máquina. Para ello se realizan operaciones de procesamiento textual que permiten estructurarlos. Esto facilita la obtención de información necesaria para identificar los documentos relevantes. El siguiente preprocesamiento textual es usado para procesar la consulta y los documentos:

1. Eliminación de los signos de puntuación: Los signos de puntuación son eliminados, así como otros caracteres especiales que no se piensa que aporten información en la tarea de recuperación de información mediante expresiones regulares.
2. Minúsculas: Las mayúsculas/minúsculas no aportan información relevante, así que todo el texto es representado en minúsculas.
3. Tokenización: El texto es convertido en una secuencia de tokens, que es una cadena de caracteres que tiene un significado coherente en el idioma usado.
4. Eliminación de stopwords: los stopwords son las palabras que no proveen información útil para la clasificación de un texto.
5. Stemming: Es un método para reducir una palabra a su raíz o (en inglés) a un stem. Hay algunos algoritmos de stemming que ayudan en sistemas de recuperación de información, en nuestro sistema se el algoritmo de Porter.

2.2 Modelo de Recuperación de Información

Un modelo de recuperación de información (MRI) es un cuádruplo $[D, Q, F, R(q_j, d_j)]$:

- D es un conjunto de representaciones lógicas de los documentos de la colección.
- Q es un conjunto compuesto por representaciones lógicas de las necesidades del usuario. Estas representaciones son denominadas consultas.

- F es un framework para modelar las representaciones de los documentos, consultas y sus relaciones.
- R es una función de ranking que asocia un número real con una consulta $q_j \in Q$ y una representación de documento $d_j \in D$. La evaluación de esta función establece un cierto orden entre los documentos de acuerdo a la consulta.

En la literatura se identifican tres modelos clásicos de recuperación de información: el modelo booleano, el vectorial y el probabilístico.

2.2.1 Modelo Booleano

Como uno de los modelos utilizados se encuentra el MRI Booleano por su facilidad a la hora de su implementación. Esta basado en la teoría de conjuntos y el álgebra booleana, solo considera si los términos indexados están presentes o no en un documento, los pesos de los términos indexados son binarios, es decir, $w_{i,j} \in 0, 1$ y las consultas están formadas por términos relacionados por tres conectores: *not*, *and* y *or*.

La similiud entre un documento $\overrightarrow{d_j}$ y la consulta q se define como:

$$sim(d_j, q) = \left\{ \begin{array}{ll} 1, & si\ \exists \overrightarrow{q_{cc}} : (\overrightarrow{q_{cc}} \in \overrightarrow{q_{fnd}}) \wedge (\forall k_i, g_i(\overrightarrow{d_j}) = g_i(\overrightarrow{q_{cc}})) \\ 0, & en\ otro\ caso \end{array} \right\} \quad (1)$$

2.2.2 Modelo Vectorial

Uno de los modelos de recuperación de información usado en este sistema es el MRI Vectorial ya que este es simple, rápido, y en algunos casos, brinda mejores resultados en la recuperación de información que el resto de los MRI clásicos. Sin embargo, sabemos que esto no significa que sea el mejor modelo; no existe un modelo general que dé solución a todos los problemas.

De manera general, todos los modelos de recuperación de información tienen alguna noción de peso. Esta definición siempre está relacionada con la importancia de un término en un documento. Cada documento va a tener asociado un vector de términos indexados con la información del peso de cada uno obtenida a partir de una función. Es común a todos los modelos que si un término no aparece en un documento su peso sea cero.

En el modelo vectorial, el peso de un término t_i en el documento d_j está dado por:

$$w_{i,j} = tf_{i,j} \times idf_i \quad (2)$$

Sea $freq_{i,j}$ la frecuencia del término t_i en el documento d_j. Entonces la frecuencia normalizada $f_{i,j}$ del término t_i en el documento d_j ($tf_{i,j}$) se calcula como:

$$tf_{i,j} = \frac{freq_{i,j}}{max_i freq_{i,j}} \quad (3)$$

donde el máximo se calcula sobre los términos del documento d_j.

Por otro lado, sea N la cantidad total de documentos en el sistema y n_i la cantidad de documentos en los que aparece el término t_i. La frecuencia de ocurrencia de un término t_i dentro de todos los documentos de la colección idf_i (del inglés inverse document frequency) está dada por:

$$idf_i = log\left(\frac{N}{n_i}\right) \quad (4)$$

La medida $tf_{i,j}$ es una medida de similitud intra-documento. Se considera la frecuencia de cada término en un documento dividido entre la frecuencia máxima de ese mismo documento para normalizar esta medida. Esto se hace con el objetivo de evitar valores de frecuencia sesgados por la longitud del documento. Por otro lado, idf_i es una medida inter-documentos. Además de la medida de frecuencia de términos es importante analizar la frecuencia de ese término en todos los documentos de la colección. Un término que aparezca en pocos documentos de la colección tiene mayor valor frente a una consulta pues permite discriminar una mayor cantidad de documentos.

Sea además $freq_{i,q}$, la frecuencia del término t_i en el texto de consulta q, el peso de una consulta está dado por la siguiente fórmula:

$$w_{i,q} = \frac{freq_{i,q}}{max_i freq_{i,q}} \times \log\left(\frac{N}{n_i}\right) \tag{5}$$

Una vez definidos los pesos de los documentos y las consultas solo falta la función de ranking para completar la definición del modelo vectorial. Esta función permite tener una ordenación por relevancia de los documentos y se basa en la similitud entre la consulta y los documentos empleando el coseno del ángulo comprendido entre los vectores documentos d_j y la consulta q:

$$sim(d_j, q) = \frac{\overrightarrow{d_j} \cdot \overrightarrow{q}}{|\overrightarrow{d_j}| \cdot |\overrightarrow{q}|} \tag{6}$$

$$sim(d_j, q) = \frac{\sum_{i=1}^{n} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2} \times \sum_{i=1}^{n} w_{i,q}^2} \tag{7}$$

Luego de obtener este ranking, se ordenan los documentos de acuerdo a su medida de similitud de mayor a menor, ya que documentos más similares tienen una mayor similitud. Para la recuperación de los documentos, puede establecerse un umbral de similitud y recuperar los documentos cuyo grado de similitud sea mayor que este umbral.

2.3 Crawler

Un crawler es un agente del tipo bot que recorre recursivamente la World Wide Web bajo algún orden predeterminado, y que recopila información acerca de los documentos que encuentra y su estructura de vínculos.

Según sea el caso, un crawler puede ser programado para trabajar con un propósito general o sobre un dominio específico. En nuestro caso, el crawler implementado recoge información definida sobre el dominio de Wikipedia. Dado un n especificado recoge información del resumen de n artículos de Wikipedia siguiendo el siguiente algoritmo:

1. Mientras la cantidad de artículos extraídos sea menor que n
2. Si la lista de urls es cero, entonces se agrega un url random de Wikipedia (esto es posible mediante el link https://en.m.wikipedia.org/wiki/Special:Random)
3. Se extrae una url de la lista de urls.
4. El título del documento y el resumen del artículo de wikipedia es guardado y son extraídos todos los enlaces del documento que estén bajo el dominio de https://en.m.wikipedia.org/wiki.
5. Añade a la lista de urls cada uno de los enlaces extraídos, siempre que no haya sido extraído antes.

La información extraída de los artículos de Wikipedia son almacenados en un diccionario donde la llave es el título del artículo y el valor es el contenido del resumen del mismo. Esta información es almacenada en un json para ser fácilmente parseada e indexada, con el objetivo de ser utilizada, en un futuro, como corpus adicional del sistema.

3 Implementación

El sistema fue implementado en Python 3.10.4. Las principales bibliotecas utilizadas y sus usos fueron:

- bs4: Para extraer datos de archivos HTML. Funciona con su analizador para proporcionar formas idiomáticas de navegar, buscar y modificar el análisis.
- nltk: Para el procesamiento de texto. Se usa para tokenizar, para realizar stemming, para la eliminación de los stopwords mediante la lista de stopwords presente en nltk.

- numpy: Para establecer operaciones matemáticas y estructuras de datos y que sus valores se calcules rápidamente.
- streamlit: Fue usado para la realización de la interfaz visual del proyecto.

El link de Github del proyecto, donde puede ser clonado es: https://github.com/ezielramos/information-retrieval-system.git

3.1 Ejecución del sistema

Para correr nuestro sistema se ejecutan los siguientes comandos desde una terminal abierta desde esta misma dirección.

```
cd src/
streamlit run main.py
```

Nuestro proyecto funciona con las dependencias requeridas en los sistemas operativos Windows y Linux.

4 Evaluacion del sistema

Con el objetivo de analizar el rendimiento de nuestro sistema este fue probado utilizando distintos corpus, que contenían varias consultas de prueba con los documentos que son relevantes. Algunas contenían además un ranking de relevancia entre estos documentos y otros no. De todas formas, este ranking no fue tomado en cuenta para la evaluación del sistema de recuperación de información.

Las colecciones de prueba usadas fueron:

- CRANFIELD: Contiene 1400 documentos.
- CISI: Contiene 1460 documentos.

4.1 Medidas de evaluación

Para describir las medidas de evaluación implementadas comencemos con las siguientes definiciones:

- RR: conjunto de documentos recuperados relevantes.
- RI: conjunto de documentos recuperados irrelevantes.
- NR: conjunto de documentos no recuperados relevantes.
- NI: conjunto de documentos no recuperados irrelevantes.

La precisión es una de las medidas fundamentales. La precisión tiende a decrecer cuando la cantidad de documentos recuperados aumenta. Calcula lo fracción de los documentos recuperados que son relevantes:

$$P = \frac{|RR|}{|RR \cup RI|} \tag{8}$$

Por otro lado, tenemos el Recobrado, que es fundamental en los procesos de recuperación de información. En contraposición a la precisión el recobrado aumenta a medida que incorporamos más documentos a la respuesta, pues es cada vez más probable que los elementos del conjunto de documentos relevantes estén contenidos en nuestra respuesta. Esto hace que siempre sea posible tener el mayor valor de recobrado, 1. Esto se lograría devolviendo la colección completa aunque, lógicamente, no es una solución factible. Representa la fracción de los documentos relevantes que fueron recuperados.

$$R = \frac{|RR|}{|RR \cup NR|} \tag{9}$$

Con el objetivo de lograr una compensación entre la precisión y el recobrado se define la medida F. Permite enfatizar la precisión sobre el recobrado y viceversa.

$$F = \frac{1 + \beta^2 PR}{\beta^2 P + R} = \frac{1 + \beta^2}{\frac{1}{P} + \frac{\beta^2}{R}} \tag{10}$$

La medida F1 es un caso particular de la medida F en la que la precisión y el recobrado tienen igual importancia.

$$F_1 = \frac{2PR}{P + R} = \frac{2}{\frac{1}{P} + \frac{1}{R}} \tag{11}$$

4.2 Resultados obtenidos

Con el objetivo de medir la evaluación del sistema fueron consideradas las tres medidas mencionadas con anterioridad: la precisión, el recobrado y la medida F1.

Como en nuestras colecciones de prueba se tenían la cantidad de documentos relevantes para cada consulta y el resultado del Modelo de Recuperación de Información es un ranking entre los documentos relevantes (existe una ordenación entre ellos) se utilizaron medidas que trabajan con el ranking de los documentos. El promedio de cada una de estas mediciones de cada consulta individual es el calculado para evaluar el rendimiento del sistema.

Corpus	Precisión	Recobrado	Medida F1
CRAN	0.0304	0.1703	0.0495
CISI	0.028	0.0432	0.0275

Cuadro 1: Promedio de las primeros 50 documentos de la lista ranking

Corpus	Precisión	Recobrado	Medida F1
CRAN	0.0209	0.2149	0.037
CISI	0.0287	0.0797	0.036

Cuadro 2: Promedio de las primeros 100 documentos de la lista ranking

5 Recomendaciones

Recomendamos la inclusión de retroalimentación mediante el Algoritmo de Rocchio, la expansión de consultas y la implementación de algún algoritmo de agrupamiento. También podemos integrar los documentos crawleados como un corpus adicional y la implementación del modelo clásico probabilístico.

6 Conclusiones

En el presente artículo, se describió la implementación de un Sistema de Recuperación de Información basado en el Modelo de Recuperación de Información Booleano y Vectorial. Se expone el diseño del sistema según cada etapa de la recuperación de información. Varias de sus funcionalidades son explicadas: cómo se realiza el preprocesamiento y la representación de la consulta y los documentos, las principales características y cómo es implementado el MRI Booleano y Vectorial. A su vez, se expone

como son usadas técnicas de Crawling para generar más información para el sistema. Las herramientas empleadas para la programación son explicadas. Además, se describe como se utiliza nuestro sistema y los aspectos más importantes de la interfaz visual del mismo. Luego, se realiza una evaluación del sistema en distintas colecciones de prueba empleando distintas métricas estudiadas en clase.