Diseño, Implementación, Evolución y Análisis de un Sistema de Recuperación de Información

Reinaldo Barrera Travieso - C511

Resumen: En el presente artículo, se describe la implementación de un Sistema de Recuperación de Información basado en el Modelo de Recuperación de Información Vectorial. Se expone el diseño del sistema según cada etapa de la recuperación de información y varias de sus funcionalidades son explicadas. Además, se describen las herramientas empleadas para la programación y la estructura de nuestro proyecto. Se explica la utilización de 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 objetivas. Posteriormente, un análisis crítico de las ventajas y desventajas del sistema desarrollado es realizado. Por último, se realizan propuestas de trabajo futuro con el objetivo de mejorar nuestra propuesta.

1 Introducción

El significado del término recuperación de información puede ser muy amplio. Simplemente sacar una tarjeta de crédito de su billetera para que pueda ingresar el número de la tarjeta es una forma de recuperación de información. Sin embargo, como campo de estudio académico, la recuperación de información podría definirse así:

> La recuperación de información (RI) consiste en encontrar material (generalmente documentos) de una naturaleza no estructurada (generalmente texto) que satisface una necesidad de información dentro de grandes colecciones (generalmente almacenadas en computadoras).

Tal como se define de esta manera, la recuperación de información solía ser una actividad en la que solo participaban unas pocas personas: bibliotecarios de referencia, asistentes legales y similares buscadores profesionales. Ahora el mundo ha cambiado y cientos de millones de personas se involucran en la recuperación de información todos los días cuando utilizan un motor de búsqueda web o buscan en su correo electrónico. La recuperación de información se está convirtiendo rápidamente en la forma dominante de acceso a la información, superando las bases de datos tradicionales. La IR también puede cubrir otros tipos de problemas de datos e información

más allá de los especificados en la definición básica anterior. El término "datos no estructurados"se refiere a datos que no tienen una estructura clara, semánticamente abierta y fácil de usar para una computadora. Es lo opuesto a los datos estructurados, cuyo ejemplo canónico es una base de datos relacional, del tipo que las empresas suelen utilizar para mantener inventarios de productos y registros de personal. RI también se utiliza para facilitar la búsqueda "semiestructurada", como encontrar un documento donde el título contiene Java y el cuerpo contiene subprocesos.

El campo de la recuperación de información también cubre el apoyo a los usuarios en la exploración o el filtrado de colecciones de documentos o en el procesamiento posterior de un conjunto de documentos recuperados. Dado un conjunto de documentos, la agrupación es la tarea de crear una buena agrupación de los documentos en función de su contenido. Es similar a organizar los libros en una estantería de acuerdo con su tema. Dado un conjunto de temas, necesidades de información permanente u otras categorías (como la idoneidad de los textos para diferentes grupos de edad), la clasificación es

la tarea de decidir a qué clase(s), si corresponde, pertenece cada uno de un conjunto de documentos. A menudo se aborda clasificando manualmente algunos documentos y luego esperando poder clasificar los nuevos documentos automáticamente.

En el presente trabajo se describe el diseño implementación, evolución y análisis de un Sistema de Recuperación de Información. Para ello, se describen todas las etapas del proceso de recuperación de información. Es decir, desde el procesamiento de la consulta hecha por un usuario, a la representación de los documentos y la consulta, el funcionamiento del motor de búsqueda y la obtención de los resultados.

El resto del artículo está organizado de la siguiente manera. La sección 2 describe el diseño completo del sistema según cada etapa de la recuperación de información. La sección 3 presenta las herramientas empleadas para la programación y aspectos más importantes del código. Posteriormente, en la sección 4 se expone la evaluación del sistema empleando distintas métricas y 4 colecciones de prueba distintas. En la sección 5 se realiza un análisis crítico de las ventajas y desventajas del sistema desarrollado. Luego, recomendaciones para trabajos futuros que mejoren la propuesta son expuestos en la sección 6 y en la sección 7 se presentan las conclusiones y un resumen de las características del sistema.

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. Opcionalmente, el usuario puede participar en la retroalimentación del sistema para mejorar los resultados obtenidos. Además, se implementa la expansión de consultas y el agrupamiento de documentos con el objetivo de mejorar el recobrado del sistema. Finalmente, funcionalidades de sistemas de recomendación son usadas con el objetivo de incorporar la recomendación de documentos a nuestro sistema de recuperación de información.A continuación, cada una de estas funcionalidades son explicada más exten- sivamente.

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:

- **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.
- **Minúsculas:** Las mayúsculas/minúsculas no aportan información relevante, así que todo el texto es representado en minúsculas.
- **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.
- **Eliminación de stopwords:** los stopwords son las palabras que no proveen información útil para la clasificación de un texto.

2.2 Representación de la consulta y los documentos

El siguiente paso, después del preprocesamiento textual, es la construcción de índices. Un término indexado es una palabra cuya semántica ayuda a definir el tema principal de un documento, brindando un resumen de su contenido. Generalmente se usan como términos indexados las palabras contenidas en el documento.

La recuperación de información basada en términos indexados tiene como base fundamental la idea de que la semántica de documentos y las necesidades informativas de los usuarios pueden ser expresadas a través de un conjunto de términos indexados.

Existen diferentes métodos de representación de documentos. Uno de los más empleados son las matrices de asociación. En ellas se representan en una dimensión, ya sea filas o columnas, los documentos y en la otra los términos

o palabras presentes en el vocabulario resultado del preprocesamiento textual de los documentos en el corpus. Cada celda de esta matriz indica la presencia de un término en un documento. Esta estructura de datos permite realizar operaciones de una manera relativamente eficiente, sin embargo, impone grandes costos de almacenamiento, pues por cada documento hay un vector con todos los términos del vocabulario y es probable que la mayor parte de ellos no esté contenida en todos los documentos. Esto origina que la matriz sea esparcida, es decir, la mayoría de los valores son cero.

Otra de las estructuras de datos más empleada en la actualidad son los índices invertidos. Cada término apunta a una lista de los documentos en donde está incluido. De esta manera se hace un uso más eficiente del espacio de almacenamiento, sin embargo, se dificulta el uso eficiente de dichos índices. Como los modelos clásicos de recuperación de información consideran que cada documento está descrito por un conjunto de palabras clave (términos indexados) se hace necesario la búsqueda eficiente de los términos dentro de un documento, razón por la cual los índices invertidos no fue la estructura utilizada para la implementación en nuestro sistema. En su lugar, se realizó una versión de las matrices de asociación usando una técnica muy usada cuando se trabajan con matrices esparcidas: la matriz se representa como una lista de diccionarios donde cada elemento de la lista serían las filas, las llaves de los diccionarios las columnas de la matriz y su valor el valor que ocuparía esa posición. Solo se almacenan los valores distintos de cero. Por lo tanto, de no existir un valor en el diccionario este se asume que es cero. Luego, nosotros tenemos listas de documentos en los que cada uno de ellos contiene un diccionario con los términos que están presentes en el documento como llave y la frecuencia de ellos como valor.

La estructura utilizada contiene además más información. Se realiza un mapeo entre términos y enteros, haciéndole corresponder a cada término del con- junto de documentos un valor entero y único que lo identifica. Se guarda además más información útil para las siguientes etapas de recuperación de información, como la cantidad de documentos en los que está presente un término.

2.3 Modelo de Recuperación de Información

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

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.

El modelo 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.

¿¿¿ Por otra parte se implemento un modelo Booleano menos complejo donde el mayor peso recae en librerías de python, pero que más adelantes no ayudaría a comparar los resultados obtenidos en los diferentes escenarios.¿¿¿¿

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:

$$w_{i,j} = tf_{i,j} \times idf_i$$

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:

$$tf_{i,j} = \frac{freq_{i,j}}{max_l freq_{l,j}}$$

donde el máximo (término l) 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:

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

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.

En el caso del cálculo de pesos para la consulta se introduce la frecuencia de los términos con una constante de suavizado (a). Esta permite amortiguar la variación en los pesos de términos que ocurren poco, para evitar, por ejemplo, grandes saltos entre la frecuencia de un término que aparece una vez a otro que aparece dos veces. Los valores más usados son 0.4 y 0.5, en nuestro sistema usamos el término de 0.4. 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:

$$w_{i,q} = \left(a + (1-a)\frac{freq_{i,q}}{max_l freq_{l,q}}\right) \times log\frac{N}{n_i}$$

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:

$$sim(d_j, q) = \frac{\vec{d_j} \cdot \vec{q}}{|\vec{d_j}| \cdot |\vec{q}|} = \frac{\sum_{i=1}^{n} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{n} w_{iq}^2}}$$

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. En nuestro sistema se usó el umbral 0.3, para garantizar la recuperación de la mayor cantidad de documentos.

¿¿¿¿Por parte del modelo booleano solo nos interesa saber que su función de peso está definida por:¿¿¿¿

$$sim(d_j, q) = \begin{cases} 1 & Si\ \exists \vec{q_{cc}} : (\vec{q_{cc}} \in \vec{q_{fnd}}) \wedge (\forall_{k_i}, g_i(\vec{d_j}) = g_i(\vec{q_{cc}})) \\ 0 & En\ otro\ caso \end{cases}$$

3 Retroalimentación

A veces para los usuarios es díficil plantear las consultas de modo que expresen sus necesidades informativas. Además, muchas veces los SRI no logran dar respuesta a la necesidad de información del usuario. Es por esto que una idea interesante podría involucrar al usuario en un proceso de retroalimentación que permita obtener mejores resultados. El algoritmo clásico de retroalimentación, qué es además el usado por nuestro sistema consiste en la realización de los siguientes pasos:

1 El usuario plantea la consulta.

2 El sistema devuelve un conjunto de documentos.

3 El usuario selecciona de estos documentos los que considera relevantes o no.

4 El sistema obtiene una mejor representación de las necesidades del usuario usando esta información.

5 Se regresa al paso 2.

Existen algoritmos específicos de retroalimentación para cada uno de los MRI clásicos en dependencia de las características de cada uno de ellos. La retroalimentación en el MRI Vectorial considera que los vectores de los documentos relevantes (a una consulta) son similares y que tienen una diferencia notable con los vectores de los documentos relevantes. Por lo tanto, la idea principal del algoritmo de retroalimentación consiste en encontrar un vector consulta $\vec{q}$ que maximice la similitud de los documentos relevantes mientras minimice a similitud con los documentos no relevantes.

Asumiendo que se conocen C_r el conjunto de documentos relevantes y C_nr el conjunto de documentos no relevantes la función del algoritmo de retroalimentación es encontrar el vector $\vec{q}$ opt que representa el vector promedio del conjunto de documentos relevantes y no relevantes respecto a la consulta expresado como:

$$\vec{q_{opt}} = max[sim(\vec{q}, C_r) - sim(\vec{q}, C_{nr})]$$

Utilizando el cálculo del coseno del ángulo entre los vectores de la consulta y los documentos como medida de similitud, resulta:

$$\vec{q_{opt}} = \frac{1}{|C_r|} \sum_{\vec{d_j} \in C_r} \vec{d_j} - \frac{1}{|C_{nr}|} \sum_{\vec{d_j} \in C_nr} \vec{d_j}$$

En un SRI real, se tiene una consulta y solo se conoce parcialmente el conjunto de los documentos relevantes y no relevantes. El algoritmo de Rocchio ofrece una alternativa para el cálculo de esta consulta ideal a partir del conocimiento de algunos documentos relevantes y no relevantes. Además, considera la influencia de la consulta original y otorga un peso a cada componente de la fórmula. De esta manera podemos ajustar a conveniencia la importancia de cada uno. Dado D_r y D_{nr} el conjunto de documentos relevantes y no relevantes conocidos respectivamente y α, β, γ los pesos establecidos para cada término en la consulta el algoritmo de Rocchio propone la siguiente fórmula:

$\vec{q_m} = \alpha\vec{q} + \frac{\beta}{|C_r|} \sum_{\vec{d_j} \in C_r} \vec{d_j} - \frac{1}{|C_{nr}|} \sum_{\vec{d_j} \in C_{nr}} \vec{d_j}$

Los valores de cada constante dependen del contexto de aplicación. Si tenemos muchos documentos, β y γ pueden tener valores grandes. Se puede asumir que es más importante la información obtenida de los documentos relevantes que de los no relevantes, por lo que β (retroalimentación positiva)

puede ser mayor que γ (retroalimentación negativa). Los valores usados en nuestro sistema son $\alpha = 1$, $\beta = 0,75$ y $\gamma = 0,15$ por ser valores comúnmente usados.

En el sistema implementado los vectores guardados en el algoritmo de Rocchio, tanto de la consulta como de los documentos ($\vec{q}$, d_j), son representados por la frecuencia de sus términos.

Una de las desventajas de los algoritmos de retroalimentación es que, en la práctica, pocos usuarios participan en ella. Por lo tanto, existen otros métodos de retroalimentación que omiten la intervención del usuario. En nuestro sistema implementamos además la pseudo-retroalimentación para estos casos. Con este método se seleccionan como documentos relevantes los k primeros documentos del ranking dado como resultado del sistema y luego se aplica la retro- alimentación. En nuestra propuesta el valor tentativo de k es 10, sin embargo, probablemente una búsqueda más refinada de este parámetro sea necesaria.

Otra de las limitaciones de la retroalimentación es que las consultas que se generan son muy grandes e inificientes para los algoritmos de recuperación. Por esto, una reducción de las dimensiones de este vector es realizada, solo cogiendo los valores que sean mayores de $\epsilon = 0,05$. Este es otro parámetro del sistema que necesita ser mejor determinado.

3.1 Expansión de consultas

Otra variante para lograr la retroalimentación del Sistema de Información es adecuar mejor la consulta al conjunto de documentos relevantes esperados. La expansión de consultas es una variante muy atractiva que permite mejorar las respuestas de los modelos y que además, se puede usar conjuntamente con la retroalimentación en otros modelos. Una de sus principales ventajas es que no requiere la intervención del usuario. En nuestro sistema, es usada conjuntamente con la retroalimentación ofreciendo alternativas de la consulta original al usuario.

Los métodos de expansión de consultas pueden ser clasificados atendiendo a si la información que emplean es local o externa al modelo

- Análisis global
    - Utilizan fuentes externas como tesauros u otras bases de conocimientos.

- Es la forma más común de expansión

- Análisis local

  - Analizan los documentos en la colección

Con el objetivo de mejorar el funcionamiento de nuestro sistema usamos el análisis global como fuente de información de la expansión de consultas. Para esto, usamos Wordnet como tesauro, que dado una palabra nos da información concerniente a dicha palabra: sinónimos, antónimos y otras relaciones semánticas más avanzadas como hipernónimos, hiponónimos, etc. Para la implementación de nuestro sistema consideramos la sustitución por sinónimos y hipernónimos. En semántica lingüística, se denomina hiperónimo a aquel término general que puede ser utilizado para referirse a la realidad nombrada por un término más específico. Por ejemplo, ser vivo es hiperónimo para los términos planta y animal.

Nuestro sistema combina el uso de tesauros con la ponderación de términos. Se considera **idf** que es una medida inter-documentos para determinar aquellos términos que están poco presentes en el corpus, y de esos términos se intenta hacer una sustitución (ya sea por sinónimos o por hipernónimos). Para esto se determina un umbral, que tentativamente es $log\frac{N}{20}$, donde N es la cantidad total de documentos del corpus. Los términos que tengan un idf mayor q el umbral determinado quiere decir que aparece en menos de 20 documentos, y son los elegidos para ser posiblemente sustituidos. Luego de que estos términos sean determinados se procede a la sustitución de estos por sinónimos o hipernónimos cuyo idf sea mayor que el umbral anteriormente determinado.

Además, se realizó la implementación de un corrector ortográfico. Para esto se cuenta con una lista de palabras en inglés (que cuenta con 236736 palabras). Para cada una de los términos de la consulta se calcula la distancia de edición de Levenshtein con respecto a las palabras en inglés. La distancia de edición es el número de caracteres que necesitan ser sustituidos, insertados o eliminados para transformar una palabra s1 en s2. La palabra en inglés que tenga la mínima distancia de edición con cada término de la consulta es la palabra escrita correctamente. Este término correctamente escrito es sustituido en la consulta original.

3.2 Agrupamiento de documentos

Los algoritmos de agrupamiento de documentos obtienen una división de un conjunto de documentos, de modo que los documentos similares estén en el mismo grupo (clúster), mientras que los documentos disimilares estén en grupos distintos.

Existen varias maneras de representar los documentos, a través de vectores binarios, vectores reales (tf $\times$ idf) o Word Embeddings. En nuestro algoritmo los documentos fueron representados a partir de vectores reales, calculando tf $\times$ idf , debido a la simplicidad de esta representación, ampliamente usada en nuestro sistema. A su vez, existen varias medidas de similitud, la usada para los vectores reales es el producto escalar y la distancia euclidiana entre los vectores. La última fue la elegida para nuestro sistema.

Según el tipo de estructura impuesta sobre los datos los algoritmos de clustering se clasifican como jerárquicos y particionados. Los algoritmos de agru- pamiento particionados obtienen una partición simple de N documentos en un conjunto de K grupos. El algoritmo usado fue K-means, pertenece a este último grupo. Fue elegido debido a su simplicidad, su eficiencia y su amplio uso. La definición general de los algoritmos de agrupamiento es: Dado un conjunto de documentos $D = d_1, d_2, ..., d_N$, la cantidad de grupos K deseados y una función objetivo que evalúa la calidad del agrupamiento encontrar una asignación $P : D \rightarrow w_1, w_2, ..., w_k$ que minimice la función objetivo, tal que, ningún w_i esté vacio.

En el caso de K-means se selecciona un representante de cada grupo, que es el vector promedio de todos los pertenecientes al grupo. Luego, tiene como función objetivo el cuadrado de la distancia Euclidiana media de los vectores del grupo.

$$RSS_k = \sum_{\vec{x} \in w_k} |\vec{x} - \vec{u}(w_k)|^2$$

$$RSS = \sum_{k=1}^{k} RSS_k$$

Es necesario notar, que para la realización de nuestro sistema no implementamos desde cero el algoritmo de K-means, si no que usamos una implementación existente, en concreto, la desarrollada en sklearn.

Clustering tiene diferentes aplicaciones en recuperación de información, la usada por nosotros explota la hipótesis de la agrupación para mejorar los resultados de búsqueda, basándose en una agrupación de toda la colección. Identifica- mos un conjunto inicial de documentos que coinciden con la consulta, pero luego agregamos otros documentos de los mismos clusters de los documentos retornados, incluso si tienen poca similitud con la consulta. Por ejemplo, si la consulta es carro y se toman varios documentos de carro de un cluster de documentos de automóvil, entonces podemos agregar documentos de este grupo que usen términos distintos a carro (automóvil, vehículo, etc.). Esto puede aumentar el recobrado, ya que un grupo de documentos con una gran similitud mutua suele ser relevante en su conjunto.

En concreto, retornamos para el usuario 10 documentos que estén en el mismo cluster que el primero document retornado. Estos documentos son puestos después de los 10 primeros documentos devueltos por nuestro MRI. Notar que para esto es necesario guardar todos los documentos que están en todos clusters.

Recomendación de documentos

Existe una amplia clase de aplicaciones web que implican predecir las respuestas de los usuarios a las opciones. Esta facilidad se denomina sistema de recomendación.

La aplicación usada de los sistemas de recomendación en nuestro sistema de recuperación de información fue la recomendación de documentos. De esta manera, devolvemos documentos que pueden resultar interesantes a nuestros usuarios basados en las consultas realizadas por este. Se almacenan como documentos en los que el usuario muestra interés el primero que es retornado en la búsqueda de una consulta y aquellos que el usuario marca como relevantes en la retroalimentación.

Para el desarrollo de un sistema de recomendación es necesario la creación de un perfil de ítem para cada ítem. El perfil de ítem es un conjunto de caracterísi- cas que matemáticamente es interpretado como un vector de características. Cada componente del vector representa una característica y puede tener solo valores binarios o reales. En nuestro caso, no es inmediatamente aparente las características que pueden ser utilizadas para representar a nuestros ítems, que son los documentos. La medida tf - idf de cada palabra en el documento tantas veces usada y computada por nuestro sistema es el

usado para representarlos. Además, se definen los perfiles de usuario que es un conjunto de características que representan los intereses del usuario. En nuestro sistema de recomen- dación, que es sencillo y es ejecutado localmente, no contamos con un concepto para un usuario. Se considera que el usuario es único, al no contar con un sistema para la autentificación que permita que existan varios.

Teniendo en cuenta que los ratings de todos los ítems son realizados por el mismo usuario, se estima el rating por:¿¿¿

$$\widehat{r}_{xi} = \frac{\sum_{j \in N(i,x)} S_{ij} \times r_{xj}}{\sum_{j \in N(i,x)}}$$

La componente que falta por definir en la fórmula es entonces la medida de similitud S_ij de los ítems i y j. Dados los ratings de dos documents A y B denominados r_A y r_B respectivamente, entonces:

Similitud de Jaccard: El problema principal de esta medida es que ignora los valores de los ratings, es una medida binaria:

$$sim(A, B) = \frac{|A \cup B|}{|A \cap B|}$$

Modelo Booleano

Python permite recuperar información de un modelo boolenao, este fue incorporado al proyecto para aplicar la capacidad del mismo. Con respecto al diseño de este mismo, en el se mantienen la mayoría de estructuras definidas en el modelo vectorial, donde aquí los pesos se toman como vectores binarios.

En este caso una consulta es traducida a un vector de conjunciones, lo cual trae muy malo resultados, pues obliga a recuperar solo los docuemntos donde existe la presencia de todo los términos de la consulta en el documento.

Para este se realiza una sola retroalimentación, y lo que hace es guardar la misma consulta pero ahora como una disyunción de términos, los cuales en algunos casos dieron mejores resultados que el modelo vectorial.

4 Implementación

El sistema fue implentado en Python 3.8 sobre Ubuntu 20.04. Las principales bibliotecas utilizadas y sus usos fueron:

- 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. Además se usa como corrector ortográfico en el uso de la distancia de edición de Levenshtein y su lista de palabras de inglés. A su vez, se usa la implementación de WordNet de nltk, que es usado como tesauro para la expansión de consultas. Para el total funcionamiento del sistema además de la instalación de nltk es necesario descargar datos adicionales, nltk data.

- gensim: Se usa el Diccionary localizado en gensim.corpora que encapsula el mapeo entre las palabras normalizadas y sus ids en entero. Es la estructura utilizada para realizar el indexing de los corpus. Se inicializa con una lista de documentos cuyo texto representado por tokens (listas de palabras). Tiene un conjunto de atributos y métodos, como doc2bow, que dado el índice de un documento devuelve una lista de tuplas, una componente representa el índice de un token y la otra la frecuencia de dicho token en el documento. Solo se devuelven los tokens que tienen al menos una ocurrencia en el texto. Por otro lado, también es usado en el MRI implementado el atributo dfs que retorna cuantos documentos contienen un token en específico.

- sklearn: Es un módulo para aprendizaje de máquinas para Python. Su implementación del algoritmo de K-means es utilizado

- whoosh: Fue la herramienta con la que se creó el modelo booleano

- django: Visual.

Estructura del proyecto

- doc/: carpeta donde se encuentra la documentación de este proyecto (este archivo).

- resources/: carpeta donde se encuentra los recursos usados y generados por el sistema.

- data/: Se guardan los corpus utilizados por el sistema.

- src/: Se encuentra todo el código escrito en Python del sistema de recuperación de información desarrollado.

4.1 Ejecución del sistema

Existen 3 formas para ejecutar el sistema, dos desde consola y una visual. De las dos de consola una de forma iterativa y otra como escenario de evaluación que está definido en test.py.

- **main.py**: Para esta vía solo se necesita ejecutar en consola el comando python3 main.py. Una vez cargado se selecciona el modelo y el corpus a usar y de forma repetida se solicita una consulta para continuar.
- **test.py**: Al igual que en el main aqui se ejecuta con el comando python3 test.py, se introduce el corpus y el modelo y la app realiza los test automaticamente.
- **manage.py**: Este es la parte visual del proyecto, para ejecutarlo puede ser por python3 manage.py runserver o ./manage.py runserver. Una vez cargado el servidor se abre el navegador en l direccion 127.0.0.1:8000/index

Todo esto se debe hacer desde el directorio src. En el caso del visual se emplea django y solo se monta un modelo y un corpus, si el usario desea otro, debe modificar una línea del código. Por otra parte cada estructura almacena información dentro del directorio resource, si se desea eliminar la retroalimentación o los rating ya realizados se debe eliminar los directorios o los ficheros contenidos en ../resource/ratings/ y ../resource/queries. Los demás directorios se pueden conservar en dependencia del usario. No se recomienda eliminar los archivos relacionados con los cluster ya que estas operaciones son muy costosas.

5 Evaluación 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 y 365 consultas.

- LISA: Sus iniciales significan Resúmenes de bibliotecas y ciencias de la información (del inglés Library and Information Science Abstracts). Contiene 5872 documentos y 35 consultas.
- CISI: Contiene 1460 documentos y 112 consultas.
- NPL: Contiene 11429 documentos y 93 consultas.

5.1 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:

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

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.

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

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.

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

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

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

Otras medidas de evaluación relacionadas con las anteriormente expuestas es la R-Precisión, que se define como la precisión en la pocisión R del ranking de documentos relevantes a una consulta dada para la cual existen R documentos relevantes. De manera análoga se puede aplicar este criterio al recobrado y la medida F.

5.2 Resultados obtenidos

A pesar de que todas estas medidas fueron implementadas en nuestro sistema, para medir su evaluación fueron consideradas 3: la precisión, el recobrado y la medida f1. En las siguientes tablas se puede ver un resumen los resultados obtenidos para dos corridas de los test, donde la primera fue directa, y la segunda con retroalimentación, también se hicieron dos corrida de cada test pero con modelos diferentes.

Corpus	Precision	Recobrado	F1
Cran	0.265	0.265	0.265
Cisi	0.183	0.129	0.144
Lisa	0.156	0.156	0.156
Npl	0.164	0.152	0.157

Table 1: Modelo vectorial

Corpus	Precision	Recobrado	F1
Cran	0.266	0.265	0.266
Cisi	0.194	0.132	0.148
Lisa	0.182	0.182	0.182
Npl	0.185	0.173	0.177

Table 2: Modelo vectorial con retroalimentación

Corpus	Precision	Recobrado	F1
Cran	0.012	0.012	0.0121
Cisi	0.006	0.0007	0.001
Lisa	0.0	0.0	0.0
Npl	0.005	0.001	0.002

Table 3: Modelo booleano

Corpus	Precision	Recobrado	F1
Cran	0.372	0.370	0.371
Cisi	0.169	0.092	0.110
Lisa	0.157	0.157	0.157
Npl	0.180	0.151	0.161

Table 4: Modelo booleano con retroalimentaciń

¿¿¿ Aspectos importantes que se deben destacar:

- Los resultados fueron bastante uniforme, pero con valores bajo.
- El corpus de Cran obtuvo los mejores resultados en todo los escenarios
- Los resultados de la retroalimentación mejora a el de las primeras consultas
- En el caso de del modelo booleano con disyunciones dio mejores resultados que con conjunciones, y para el caso del corpus cran, mejoró mucho más los resultados que el modelo vectorial.

6 Conclusiones

En el presente artículo, se describió la implementación de un Sistema de Recuperación de Información basado en el Modelo de Recuperación de In-

formación 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 Vectorial y Booleano, cómo se realiza la retroalimentación en el sistema, la expansión de consultas y el agrupamien- to de documentos. Las herramientas empleadas para la programación y la estructura del proyecto es explicado. Además, se describe como se utiliza el sistema. Luego, se realiza una evaluación del sistema en distintas colecciones de prueba empleando distintas métricas estudiadas en clase. Un análisis crítico de las ventajas y desventajas del sistema desarrollado es realizado y por último se identifican direcciones de trabajo futuro.