Hedwig

Proyecto Final de Sistema de Recuperación de Información

Marié del Valle Reyes
Roxana Peña Mendieta
Josué Rodríguez Ramírez

Facultad de Matemática y Computación, Universidad de La Habana, Cuba

Resumen La recuperación de información se define como el modo para encontrar materiales, usualmente documentos, que pertenecen a una gran colección y cuya naturaleza es no-estructurada; estos documentos satisfacen una necesidad de información expresada mediante una consulta al Sistema de Recuperación de Información (SRI). En la búsqueda de dichos documentos relevantes, participan modelos de recuperación de información. Estos se diferencian en la representación para los documentos de la colección, la representación para las consultas y en el algoritmo o proceso que permite discernir los documentos relevantes de los irrelevantes en relación a cada consulta. En este artículo, se realiza una descripción del diseño, implementación, análisis y evaluación de un SRI, realizando una comparación entre los modelos booleano, vectorial e indexación de semántica latente (LSI, *por siglas en inglés*).

Palabras claves SRI · booleano · vectorial · LSI · cranfield · scifact · precisión · recobrado

Introducción

Históricamente, el hombre ha necesitado de medios sobre los cuales representar todo acerca del mundo que lo rodea y de reflejar, de cierta forma, su evolución. La escritura ha sido el mecanismo "tradicional" y fundamental que soporta su conocimiento en el tiempo.

Esta misma evolución ha facilitado la existencia de diferentes medios de representación de la escritura, llegando hasta nuestros días donde la información se representa digitalmente y es posible su almacenamiento y su distribución masiva en forma simple y rápida, a través de redes de computadoras. La digitalización abrió nuevos horizontes en las formas en que el hombre puede tratar con la información que produce.

La recuperación de información (IR, *por siglas en inglés*) en informática y ciencias de la información es el proceso de obtener recursos del sistema de información que son relevantes para una necesidad de información, a partir de una colección de esos recursos.

El objetivo de este artículo es realizar un análisis profundo del Sistema de Recuperación de Información (SRI) desarrollado: **Hedwig**. Para ello, se empezará

con una descripción del diseño e implementación de las colecciones utilizadas y de los distintos enfoques de modelación que se utilizaron: el modelo booleano, el modelo vectorial y el modelo de indexación de semántica latente (LSI, *por siglas en inglés*). Se explicarán todas las etapas del proceso de recuperación de información, desde el procesamiento de la consulta hecha por un usuario, la representación de los documentos y la consulta, el funcionamiento del motor de búsqueda y la obtención de los documentos relevantes. Luego, se aplicarán medidas de evaluación objetivas y subjetivas a cada modelo implementado, en las diferentes colecciones utilizadas, y se realizará una comparación crítica de los resultados obtenidos.

1. Diseño e implementación

En esta sección, se describirá el flujo del proceso de recuperación de información que se desarrolla en el SRI Hedwig. Se comenzará con la interfaz de usuario, donde se realizan las consultas y se seleccionan las opciones de modelado a utilizar, hasta la obtención de resultados que responden a dicha consulta.

1.1. Interfaz de Usuario

La interfaz de usuario permite que éste especifique la consulta mediante una expresión escrita en un lenguaje preestablecido y, además, sirve para mostrar las respuestas retornadas por el sistema.

En la Figura 1, se muestra la primera vista de la interfaz visual del SRI Hedwig. Para realizar una consulta, primero se selecciona la colección y el modelo de recuperación de información a utilizar en el botón `Options` que aparece en la esquina inferior derecha de la barra de búsqueda (Ver Fig. 2). La consulta deseada se escribe en dicha barra de búsqueda y luego se presiona el botón con un símbolo de lupa al extremo derecho de la barra.

Los documentos relevantes a la consultan aparecerán en otra vista (Ver Fig 3), ordenados descendientemente según el valor de relevancia. Cada documento estará caracterizado por el valor de relevancia, el número de identificación en la colección, el título y el autor. Al presionar el documento, se mostrará una vista con el título, autor y cuerpo del documento, como se muestra en la Figura 4.

1.2. Colecciones

Las colecciones consisten en un conjunto de documentos utilizados para responder las consultas del usuario.

En este proyecto se utilizaron las colecciones **cranfield** y **scifact**.

Para la implementación de las colecciones, se creó una clase abstracta `Collection`, de la cual heredan las clases `Crafield` y `Scifact`, que representan las colecciones anteriormente mencionadas. También se implementó una clase `Document`, que representa un documento de la colección y cuyos atributos son: número de identificación en la colección, título, autor y cuerpo del documento.

Entre los métodos que contiene la clase `Collection` se encuentran:

Figura 1. Primera vista de la interfaz visual del SRI.

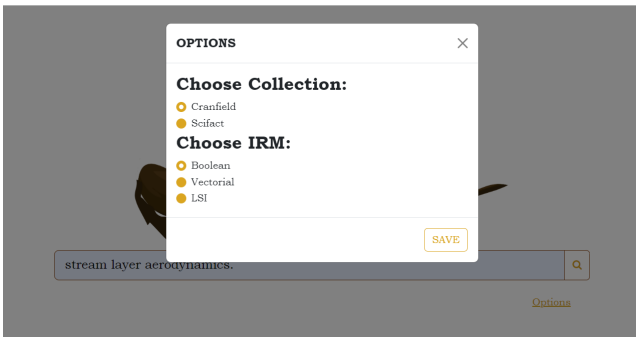

Figura 2. Vista de la ventana que aparece al presionar botón `Options`.

- `parse_documents`: Lee los documentos de un archivo, los convierte en instancias de objeto `Document` y los guarda en un diccionario. Además, crea un diccionario de términos indexados y una lista que contiene la frecuencia del término que más se repite en cada documento. Las llaves del diccionario de términos indexados son los términos, y los valores son listas, del tamaño del total de documentos de la colección adicionado en 1. Las posiciones desde 0 hasta `(total_documentos_colección - 1)`, tienen la frecuencia del término en esos documentos. La última posición de la lista representa la cantidad de documentos en los que aparece el término. Utilizando el paquete `pickle`, se guardan los dos diccionarios y la lista en una carpeta predeterminada con el nombre de la colección, que se encuentra en el archivo **data** del proyecto.
- `retrieve_documents`: Recibe un diccionario cuyas llaves son los números de identificación de los documentos en la colección y los valores son la relevancia de los documentos al aplicarles cierto modelo. Devuelve un diccionario con

Figura 3. Vista de la interfaz de usuario con los resultados relevantes a la consulta.

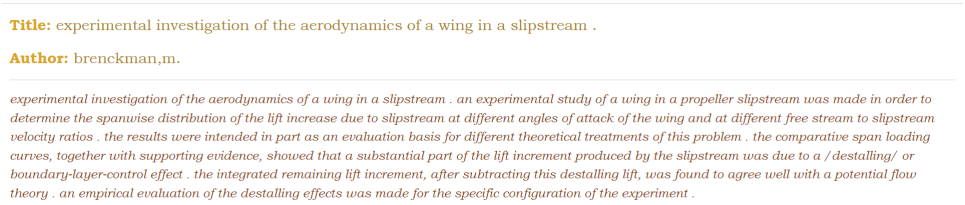

Figura 4. Vista de ejemplo de documento.

las instancias `Document`, correspondientes a esos números de identificación y los valores de relevancia.

- `number_of_documents`: Retorna la cantidad total de documentos utilizados de la colección.
- `parse_querys`: Lee las consultas de un archivo y las guarda en un diccionario. Las llaves del diccionario son el número que identifica la consulta y los valores son el texto de la consulta.
- `parse_relevant_documents`: Lee un archivo con los documentos relevantes y guarda en un diccionario la lista de documentos relevantes a una consulta dada.

Cranfield

La colección Cranfield cuenta con un total de 1400 documentos que son resúmenes científicos y 7352 términos. Viene con un conjunto de consultas y los documentos relevantes para cada consulta.

En la Figura 5 se muestra una gráfica que representa la cantidad de términos por documentos. Se puede apreciar que la mayoría de los documentos, tienen

entre 0 y 2000 términos, con máximos de 4000. El promedio de términos por documentos es 1029.

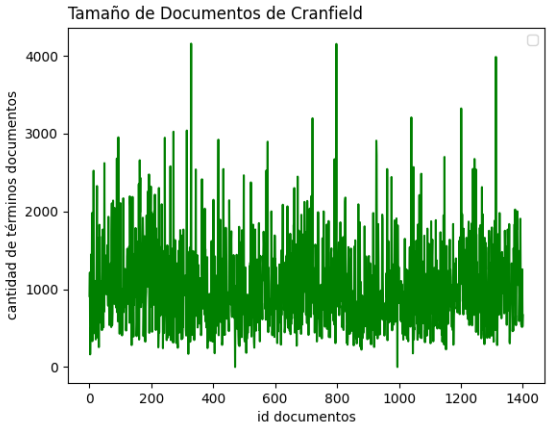

Figura 5. Gráfica que muestra el tamaño de los documentos de la colección Cranfield.

Scifact

La colección Scifact tiene un total de 5183 documentos, pero por temas de costo espacial y temporal relativamente altas, se decidió utilizar solo 2000 documentos. La cantidad total de términos de los documentos seleccionados es 20896, casi tres veces la cantidad total de términos de Cranfield. Esta colección tambien cuenta con un conjunto de consultas y los documentos relevantes asociados a cada consulta.

El tamaño de los documentos está entre 0 y 5000 términos (Ver Fig. 6), y el promedio es aproximadamente 1250 términos por documentos.

1.3. Modelos

Para recuperar documentos relevantes de manera efectiva mediante estrategias de IR, los documentos se transforman normalmente en una representación adecuada. Cada estrategia de recuperación incorpora un modelo específico para sus propósitos de representación de documentos.

Los Modelos de Recuperación de Información (IRM, *por sus siglas en inglés*) son un cuádruplo (D, Q, F, R) donde:

- D es un conjunto de representaciones lógicas de los documentos de la colección.
- Q es un conjunto con las consultas.
- F es un framework para modelar las representaciones de los documentos, consultas y sus relaciones.

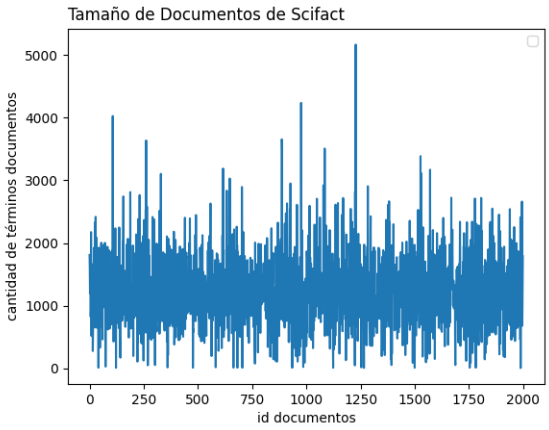

Figura 6. Gráfica que muestra el tamaño de los documentos de la colección Scifact.

R es una función de ranking que asocia un número real a un tupla formada por una consulta y una representación de un documento.

En el SRI desarrollado se implementaron tres IRM: modelo booleano, modelo vectorial y modelo de indexación de semántica latente.

En la implementación de los modelos, se creó una clase abstracta `IRM` de la cual heredan las clases `BooleanModel`, `VectorialModel` que representan los modelos booleano y vectorial, respectivamente. La clase `LSIModel`, representa el modelo indexación de semántica latente, que es un derivado del modelo vectorial, por lo tanto la clase hereda de `VectorialModel`.

Además del constructor de la clase, dichas clases tienen un método `search()`, cuyos parámetros son una cadena que representa la consulta y una instancia de un objeto `Collection`.

Modelo Booleano

El modelo booleano es un modelo clásico de recuperación de información. Está basado en teoría de conjuntos y álgebra booleana. La recuperación se basa en si los documentos contienen o no los términos de consulta.

En la representación de los documentos, se utilizó el diccionario de términos indexados que contruye el método `parse_documents()` de la clase `Collection`. Como se explicó anteriormente, los valores de este diccionario son listas con la frecuencia del témino en los distintos documentos de la colección. En el modelo booleano, los pesos de los términos indexados son binarios, pero con el objetivo de ganar en eficiencia, se decidió utilizar dicho diccionario; para ello, se consideró que si el valor de la lista de frecuencias era mayor que 0, el término estaba en el documento y por tanto su peso sería 1, en otro caso, el término no estaba en el documento y su peso sería 0.

Para procesar la consulta, se utilizó una modificación del algoritmo Shunting Yard para expresiones lógicas. Este algoritmo retorna una lista del tamaño de la cantidad de documentos de la colección. Si en alguna posición de la lista hay un 1, entonces el documento que representa esa posición es relevante, en caso contrario, es irrelevante.

En esta implementación se tuvieron en cuenta algunas consideraciones:

- Si la consulta no tiene operadores lógicos en su texto, entonces se considera que cada término está separado por el operador lógico **and**.
- Si la consulta tiene operadores lógicos en su texto, pero la expresión lógica que forma no es válida, entonces se considera que cada término está separado por el operador lógico **or**.

Las modificaciones se realizaron con el objetivo de que siempre halla una respuesta a una consulta, sin importar si la consulta constituye una expresión lógica o no.

Modelo Vectorial

El modelo vectorial, también es un modelo clásico de recuperación de la información. Se caracteriza por dar un ranking de los documentos de acuerdo con su grado de similitud a la consulta.

Para la representación de los documentos, se construyó una matriz término-documento, cuyos elementos son el peso w_{ij} del término t_i en el documento d_j,

$$w_{ij} = tf_{ij} * idf_i. \tag{1}$$

Esto se logró auxiliándose del diccionario de términos indexados, mencionado anteriormente, para hallar las matrices con el valor tf y el valor idf de los términos en los documentos y luego multiplicarlas.

Las consultas se representaron como un vector del tamaño de la cantidad de documentos de la colección, cuyos elementos son el valor $w_{i,q}$,

$$w_{i,q} = (a + (i - a)\frac{freq_{i,q}}{max_l freq_{l,q}}) * \log \frac{N}{n_i} \tag{2}$$

donde $freq_{i,q}$ es la frecuencia del término t_i en el texto de la consulta q, el término a es de suavizado, N es la cantidad de documentos de la colección y n_i es la cantidad de documentos en los que aparece el término t_i. El valor de a por defecto es 0.5.

Para hallar el ranking de los documentos, se utiliza la similitud del coseno del ángulo formado entre los vectores columna de la matriz término-documento y el vector consulta. Se consideran documentos relevantes aquellos cuyo valor de similitud a la consulta sean mayores que 0.1.

Modelo Indexación de Semántica Latente

El modelo indexación de semántica latente (LSI, *por sus siglas en inglés*) es un modelo alternativo del modelo vectorial clásico, pero que supera los problemas

de coincidencia lexicográfica de los términos indexados al ubicar dichos términos en un contexto de semántica común.

Al igual que en el modelo vectorial, se halla la matriz término-documento C, que contiene los pesos de los términos en los documentos. Este modelo propone descomponer la matriz C en tres componentes, usando la descomposición de valores singulares (SVD, *por siglas en inglés*), de la siguiente forma :

$$C = U\Sigma V^T \tag{3}$$

La matriz U es la matriz de los vectores propios derivada de una correlación término-a-término dada por la matriz CC^T. La matriz V^T es la matriz derivada de la transpuesta de la matriz documento-a-documento dada por C^TC. La matriz Σ es una matriz diagonal rxr de valores singulares donde $r = min(M, N)$ es el rango de C .

Luego, se construye una aproximación de bajo rango C_k de la matriz de término-documento C, para un valor de k[1] que es mucho más pequeño que el rango original de C y que representa la cantidad de conceptos a los que se redujo la búsqueda. Entonces, se convierte cada fila/columna (que corresponde a un término/documento) en un espacio k-dimensional; este espacio se define por los principales k vectores propios de CC^T y C^TC. El resultado es el siguiente:

$$C_k = U_k\Sigma_k V_k^T \tag{4}$$

El vector consulta se halla como se explicó en la sección anterior, y se convierte en su representación en el espacio k-dimensional, quedando,

$$\overrightarrow{q_k} = \Sigma_k^{-1} U_k^T \overrightarrow{q}. \tag{5}$$

El ranking de los documentos se obtiene hallando la similitud del coseno del ángulo entre los vectores columna de la matriz V_k^T y el vector $\overrightarrow{q_k}$. Los documentos cuyo valor sea mayor que 0.1 son los que se consideran relevantes.

1.4. Requerimientos para la ejecución

Framework y lenguaje

El proyecto se realizó con el framework Django y el lenguaje de programación Python versión 3.11. Para el diseño de la interfaz visual se utilizó el framework Bootstrap.

Obtención de datos necesarios

En el archivo **ir_app** de la carpeta principal del proyecto, se encuentra una carpeta llamada **data**, donde se encuentran todos los archivos necesarios para la ejecución del programa.

En caso que la carpeta **data** esté vacía, se deben ejecutar el método `parse()` del archivo **main.py** en la carpeta principal. A dicho método se debe pasar

[1] El valor de k por defecto es 200.

una instancia del objeto que hereda de la clase `Collection`, ya sea `Cranfield` o `Scifact` y el nombre del archivo que contiene los documentos. Esto se debe realizar para cada colección que se quiera utilizar.

Ejecución del programa

Para ejecutar el programa, primero se debe verificar en el archivo **main.py**, que en el diccionario `collection` las direcciones de las instancias de las clases herederas de `Collection` correspondan con las carpetas donde se encuentran los archivos de las colecciones.

Luego, se escribe en la terminal, en el directorio donde se encuentra el archivo `manage.py`,

```
python manage.py runserver
```

y se abre el enlace `http://127.0.0.1:8000/`.

2. Evaluación

La evaluación de un sistema de recuperación de información es el proceso de evaluar qué tan bien un sistema satisface las necesidades de información de sus usuarios. En general, la medición considera una colección de documentos a buscar y una consulta de búsqueda. Entre las métricas de evaluación objetivas tradicionales se encuentran la precisión y el recobrado.

Para la evaluación del SRI desarrollado, se implementó la clase `Eval`, que tiene los métodos:

- `precision`: calcula la precisión, que es una fracción de los documentos recuperados que son relevantes.
- `recall`: calcula el recobrado, que es una fracción de los documentos relevantes que fueron recuperados.

Por cada colección utilizada, se tomaron las consultas de prueba y se utilizaron para evaluar el SRI a partir de las medidas precisión y recobrado.

2.1. Precisión

La Figuras 7,8 y 9 muestran las gráficas de la precisión en los modelos implementados en la colección Cranfield.

La precisión en el modelo booleano varía entre 0 y 0.14, con una gran cantidad de consultas que tienen precisión 0 (Ver Fig. 7).

En el modelo vectorial, se observaron los peores resultados en la precisión, con valores que varían entre 0 y 0.06. Sin embargo, el modelo LSI, es el que tiene mejores valores de precisión y menos valores con 0.

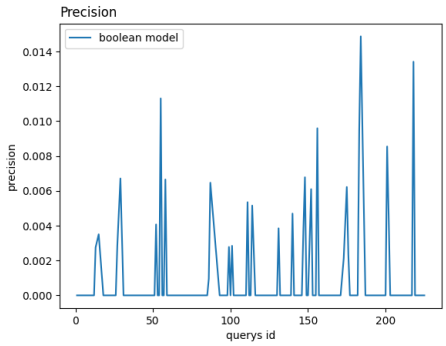

Figura 7. Gráfica que muestra la precisión del modelo booleano en la colección Cranfield.

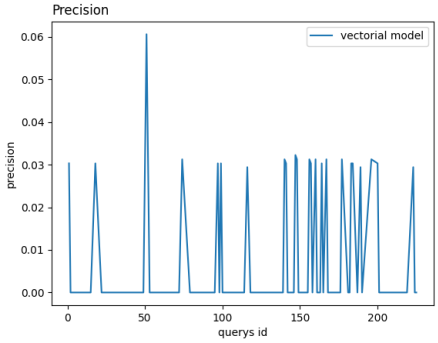

Figura 8. Gráfica que muestra la precisión del modelo vectorial en la colección Cranfield.

2.2. Recobrado

En las Figuras 10, 11 y 12, se muestran unas gráficas con la representación de los valores de recobrado para los diferentes modelos implementados.

3. Ventajas y Desventajas

3.1. Ventajas

Las ventajas del SRI desarrollado son:

- Es extensible.
- Se otorga facilidad al usuario en la formulación de consultas. Presenta un guía de búsqueda y visualización del resultado.

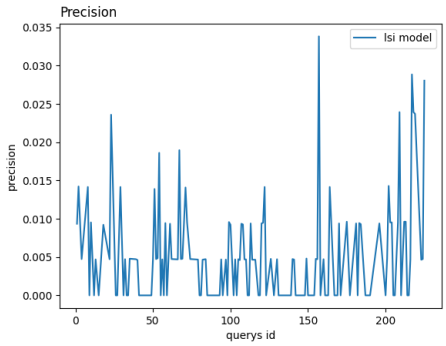

Figura 9. Gráfica que muestra la precisión del modelo lsi en la colección Cranfield.

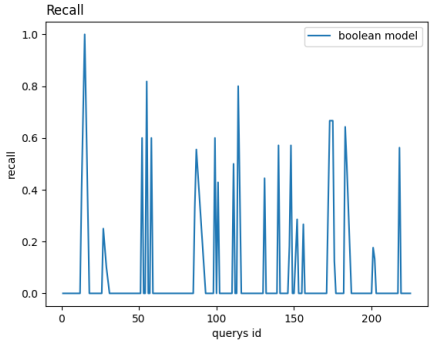

Figura 10. Gráfica que muestra del recobrado del modelo booleano en la colección Cranfield.

- Poco tiempo de respuesta: El intervalo de tiempo entre la especificación de la consulta y la representación de los resultados es mínimo.
- Formato de presentación amigable: Los resultados devueltos de la consulta son de fácil empleo.

3.2. Desventajas

Entre las desventajas del SRI desarrollado se encuentran:

- Con colecciones muy grandes las matriz termino-documento tiene gran dimension. Por tanto, no hay fuerza de computo suficiente para realizar la descomposicion de la matriz con SVD.
- Dificultades con la presición y el recobrado:Los valores obtenidos no son muy satifactorios. Están mas cerca de 0 que de 1.
- Es necesario el uso de colecciones pequeñas debido a el modelo LSI.

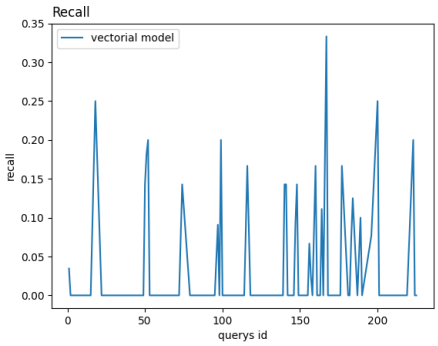

Figura 11. Gráfica que muestra del recobrado del modelo vectorial en la colección Cranfield.

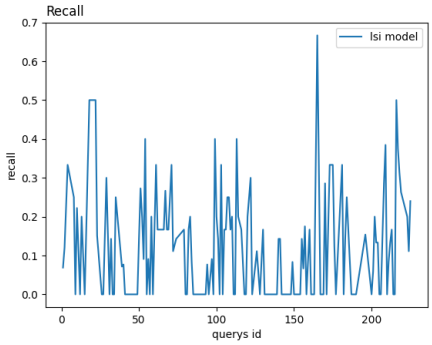

Figura 12. Gráfica que muestra del recobrado del modelo lsi en la colección Cranfield.

4. Recomendaciones

Se recomienda:

- Probar el SRI con otras colecciones, tal vez con algunas cuyos textos sean más largos.
- Aumentar el número de MRI implementados.
- Utilizar alguna estructura de datos más eficiente a la hora de parsear los documentos.
- Evaluar con otras medidas de evaluación.

5. Conclusiones

En conclusión, los sistemas de recuperación de la información son herramientas fundamentales para la búsqueda y localización de documentos relevantes dentro de grandes bases de datos. En este trabajo, se implementaron tres modelos

diferentes para la recuperación de la información: el modelo booleano, el modelo vectorial y el modelo de indexación semántica latente (LSI). Cada uno de estos modelos tiene sus propias fortalezas y debilidades, y es importante evaluar cuál es el más adecuado para una tarea específica.

El modelo booleano es una herramienta simple y eficiente para la búsqueda de documentos que cumplan con ciertas condiciones específicas, utilizando operadores lógicos como AND, OR y NOT. Sin embargo, este modelo no tiene en cuenta la similitud semántica entre los términos de la consulta y los documentos, lo que puede limitar su capacidad para recuperar documentos relevantes.

El modelo vectorial, por otro lado, utiliza técnicas de procesamiento del lenguaje natural y medidas de similitud para asignar un peso a cada término de la consulta y los documentos, y luego utiliza estos pesos para calcular la similitud entre la consulta y cada documento. Esto permite una mayor precisión en la recuperación de documentos relevantes, aunque puede ser menos eficiente en términos de tiempo de procesamiento.

Por último, el modelo LSI utiliza técnicas de reducción de dimensionalidad para representar a los documentos y las consultas en un espacio latente de menor dimensión, donde se pueden calcular medidas de similitud más precisas. Esto permite una mayor precisión en la recuperación de documentos relevantes, aunque también puede requerir más tiempo de procesamiento y puede ser más complejo de implementar.

En general, la elección del modelo adecuado para una tarea específica de recuperación de la información dependerá de las necesidades y requisitos de la aplicación, incluyendo el tamaño y la complejidad de la base de datos, el tiempo de procesamiento disponible y la precisión requerida en la recuperación de documentos relevantes.