Recuperación y acceso

a la información

GRADO EN INGENIERÍA INFORMÁTICA

Motor de búsqueda

Realizado por:

Rodrigo Borges (Grupo 80-100317579)

(100317579@alumnos.uc3m.es)

David del Rey García (Grupo 80-100315243)

(100315243@alumnos.uc3m.es)

Tabla de contenido

INDICE DE TABLAS

INDICE DE ILUSTRACIONES

Mejoras implementadas

Guardado en base de datos

Uno de los principales inconvenientes a la hora de implementar un motor de búsqueda, es la gestión de la memoria. Los datos que hay que utilizar son demasiado pesados, por lo que es muy poco recomendable mantenerlos todos en memoria, puesto que ralentizaría todo el sistema, pudiendo llegar incluso a dar fallo por salirse del "heap" (zona de memoria reservada para los datos).

Por ello decidimos guardar los datos utilizados en una base de datos.

Elección de la base de datos.

Hemos barajado dos posibilidades para la base de datos, MySQL y MongoDB.

En este caso MySQL no es una herramienta adecuada, puesto que el volumen de datos que queremos manejar es muy grande y se consulta varias veces la base de datos al realizar búsquedas, por lo que el tiempo necesario para guardar los datos en la BD y en consultarlos es demasiado alto, al tener que insertar y consultar fila a fila en la base de datos.

Por otro lado, MongoDB almacena documentos, por lo que es mucho más rápido a la hora de insertar grandes volúmenes de datos. Y para consultarlos podemos trabajar con los documentos directamente, sin tener que acceder a la BD por cada fila que queramos consultar, con lo que ahorramos mucho tiempo.

Por estas razones, hemos elegido la base de datos MongoDB para el proyecto.

Preparación de la base de datos.

Antes de comenzar a insertar los datos realizamos el diseño que queríamos implementar.

Para el proyecto decidimos utilizar una base de datos con 3 colecciones.

Diccionario

La primera colección "**diccionario**" está compuesta por múltiples documentos, cada uno de ellos hace referencia a un documento HTML de los utilizados para crear el diccionario.

Los documentos HTML tienen la siguiente estructura:

ID	"nombre del html"
Palabra 1	Valor TF
Palabra 2	Valor TF
Palabra 3	Valor TF
Palabra X	Valor TF

1 HTML

Siendo el ID el nombre del documento original HTML, para poder identificar con qué documento estamos trabajando.

Palabra X es la clave que utilizamos para guardar los valores TF de todas las palabras del documento.

Valor TF es el valor de la palabra del documento.

Con esta estructura creamos el documento en memoria rellenando cada palabra con su valor TF y cuando está completo lo introducimos en la colección, de forma que solo hay 1 acceso a BD por documento HTML.

IDF

La segunda colección "**idfColl**" contiene un único documento llamado IDF.

El documento IDF tiene la siguiente estructura:

ID	IDF
Palabra 1	Valor de IDF
Palabra 2	Valor de IDF
Palabra X	Valor de IDF

2 IDF

El documento tiene como ID el valor IDF puesto que representa estos valores para las palabras. Siendo cada Palabra X una palabra de todas las recogidas en los anteriores documentos HTML, y su correspondiente valor IDF.

1 HTML/IDF

Consultas

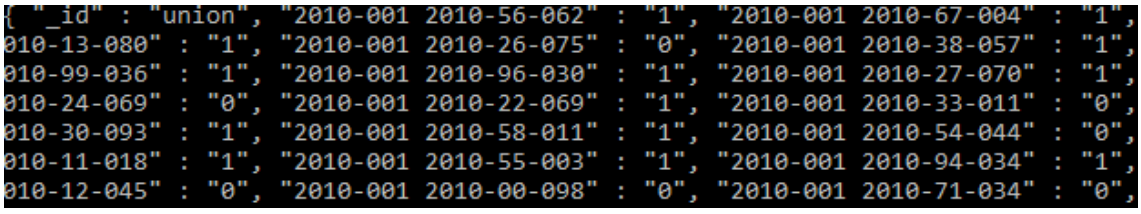
La tercera colección "**consultas**" contiene dos documentos.

El documento "union", que hace referencia al documento original "union.trel" con la siguiente estructura

ID	union
2010-001 2010-26-075	0
2010-007 2010-54-054	1
2010-007 2010-68-054	0
IDConsulta+ + IDDocumento	relevancia

3 UNION

De esta manera mantenemos en la base de datos el documento union que nos aporta la relevancia de los documentos sobre las consultas.



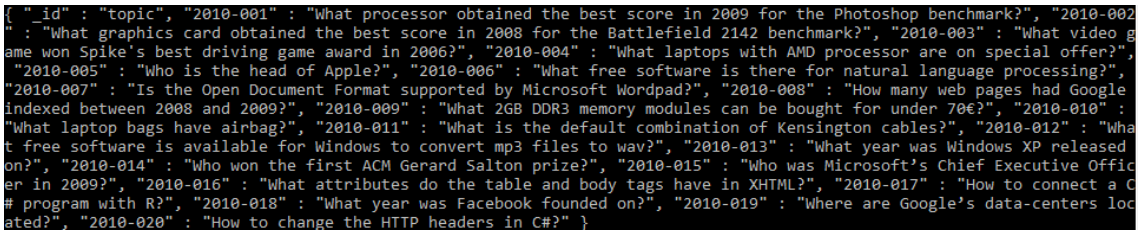
2 UNION

El documento “topics” hace referencia al original “topics.xml” con la siguiente estructura

ID	topics
2010-001	What processor obtained the best score in 2009 for the Photoshop benchmark?
2010-002	What graphics card obtained the best score in 2008 for the Battlefield 2142 benchmark?
IDConsulta	Consulta

4 TOPICS

De esta forma mantenemos en la base de datos todas las consultas que se van a realizar con su ID.



3 TOPICS

Funcionamiento

Tras diseñar e implementar la base de datos quedaba utilizarla.

La hemos utilizado para guardar los datos antes mencionados y para consultar al realizar búsquedas.

Guardado de datos: Para guardar los datos hemos utilizado la clase CreadorDiccionario, que se encarga de limpiar los HTML, formatearlos y generar los documentos que se irán guardando en la BD. A su vez va creando y actualizando el documento IDF que se guarda tras leer todos los HTML (de forma que solo se introduce 1 vez en toda la creación). También hemos utilizado la clase LectorMetricas para guardar los ficheros union y topics.

Consulta de datos: el diseño elegido permite realizar una consulta eficiente de los datos, de forma que solo es necesario mantener en memoria los documentos que se quieran utilizar, es por ello que, a la hora de consultar datos, se accede a la BD, se recupera el documento que se quiera consultar (en función del ID) y tras utilizarlo se elimina para que deje de ocupar memoria. Con esto reducimos la cantidad de memoria utilizada al no cargar el diccionario completamente y ganamos en tiempo puesto que tarda menos en recuperar 1 archivo pequeño que todos los archivos HTML almacenados en la BD.

Normalización del lenguaje natural

Para normalizar el lenguaje natural de los documentos y las consultas hemos utilizado el stemmer de lucene.

Con el uso de esta técnica esperamos mejorar la recuperación al convertir palabras similares a una forma canónica (raíz de la palabra), de este modo bebería aumentar el matching entre palabras con ligeras variaciones.

Vamos a ver un pequeño ejemplo de funcionamiento:

```
What video game won Spike best driving game award in 2006
What video game won Spike best drive game award in 2006
```

4 EJEMPLO STEMMER 1

Podemos ver como después de aplicar el stemmer a la primera cadena de texto la palabra driving ha sido sustituida por drive.

```
What attributes do the table and body tags have in XHTML
What attribut do the tabl and bodi tag have in XHTML
```

5 EJEMPLO STEMMER 2

En este otro ejemplo vemos como se han modificado las siguientes palabras:

- attributes – attribut
- table – tabl
- body – bodi
- tags - tag

Aplicando este método tanto a los documentos como a las consultas conseguimos aumentar la probabilidad de coincidencia y mejorar la recuperación de documentos. Además, reducimos el espacio utilizado para almacenar los documentos, pero tiene un pequeño inconveniente y es que aumenta un poco el tiempo de procesamiento.

Expansión de consulta

Como mejora para el proyecto hemos incluido la expansión de consulta. Para ello hemos utilizado Lucene Wordnet, que nos proporciona las funcionalidades para expandir las consultas por medio de los sinónimos.

Hemos incluido el diccionario de sinónimos en el proyecto con el nombre “wn_s.pl”.

Para expandir la consulta hemos añadido los sinónimos que nos ofrecía Wordnet a la consulta original, y la hemos tratado de la misma forma que los HTML que forman el diccionario. De manera que las palabras del diccionario y de la consulta mantienen el mismo formato.

Funcionamiento del programa

El programa tiene un menú de inicio que permite al usuario realizar diferentes acciones.

Creación y actualización

La primera es la creación/ actualización de la base de datos. Esta opción tarda alrededor de 25 minutos, puesto que carga todos los HTML, el idf, el union, y el topics en la base de datos. Todos estos documentos ya se encuentran en el proyecto por lo que no es necesario tocar el código fuente.

Ejecución consultas

Esta opción ejecuta todas las consultas que se encuentren almacenadas en la base de datos.

Primero carga las consultas de la base de datos, la expande y la formatea.

Después, con las consultas originales y las expandidas, calculamos la relevancia de los documentos para cada consulta utilizando la función coseno TF-IDF. De todos los documentos nos quedamos con los 100 más relevantes para la evaluación del motor. Puesto que el cutoff más alto que utilizamos es de 100.

Para abordar el cálculo de las métricas hemos decidido dividir cada calculo en una clase distinta. Con este diseño desacoplamos la parte de la evaluación del buscador del motor de recuperación en sí.

Mantenemos una signatura muy parecida en para cada método en las clases de cálculo. Pasando siempre el documento con los archivos relevantes y un array con los nombres de los archivos recuperados, en caso de que la métrica a calcular requiera de un corte especifico o una relevancia mínima también se la pasamos por parámetro. Con este diseño podríamos adaptar muy fácilmente las métricas a otras relevancias mínimas o cortes.

Finalizar programa

Esta opción finaliza la ejecución de programa.

Métricas utilizadas

Recall

Recall mide el porcentaje de documentos relevantes que han sido recuperados. Es decir de todos los documentos relevantes, cuantos hemos recuperado. Cuanto más cerca de 1 este significa que hay menos silencio.

En cortes 5 y 10

Qué porcentaje de documentos relevantes hay entre los 5 ó 10 primeros documentos con respecto al total de documentos relevantes.

Precision

Precision mide el porcentaje de documentos recuperados que son relevantes. Es decir, cuantos documentos relevantes hemos recuperado entre todos los documentos recuperados. Cuanto más cerca de 1 este significa que hay menos ruido

En cortes 5 y 10

Qué porcentaje de documentos son relevantes en los 5 ó 10 primeros documentos recuperados.

F-valor

Esta medida agrupa precision y recall.

Reciprocal Rank

Esta métrica mide en qué posición se encuentra el primer documento relevante en la colección de documentos recuperados. Se calcula mediante la inversa de la posición del documento relevante. Cuanto más cerca de 1 este significa que recuperamos antes un documento relevante

Relevancia mínima 1

En qué posición se encuentra el primer documento que tenga al menos relevancia 1.

Relevancia mínima 2

En qué posición se encuentra el primer documento que tenga al menos relevancia 2.

Average Precision

Esta métrica mide en qué posición se encuentran los documentos relevantes en la colección de documentos recuperados. Cuanto más se acerque a 1 mejor responde el motor a la pregunta dada.

nDCG

Esta métrica mide la ganancia de conocimiento a través de los documentos relacionados. Es decir, si leyésemos todos los documentos recuperados responderíamos perfectamente a la pregunta dada. Cuanto antes se acerque la métrica a 1 mejor es el motor de recuperación.

Resultados

Una vez ejecutadas las consultas guardamos los resultados obtenidos en la carpeta resultados. Cada archivo tiene como nombre el identificador de la consulta asociada y el sufijo Exp si se trata de esa consulta expandida. Dentro del fichero se encuentra, en primer lugar, el texto de la consulta (o la consulta expandida), en segundo lugar, aparecen 100 filas, una fila por cada documento recuperado ordenados por cos(IDF). En cada una de estas filas se puede ver el nombre del documento recuperado y a su lado la relevancia (que puede ser 2, 1, 0, -1 o espacio en blanco), hemos incluido esta información en el fichero de resultados para tener una referencia con la que comprobar si las metricas estaban bien calculadas. Finalmente, en la parte baja del fichero aparecen todas las métricas solicitadas ordenadas de la siguiente manera:

Consulta

```
what processor obtained the best score in 2009 for the photoshop benchmark
```

6 CONSULTA

Estos son los diez primeros resultados a la consulta anterior.

```
2010-69-054 0
2010-12-045 0
2010-05-084 1
2010-40-061 0
2010-58-011 1
2010-33-011 0
2010-48-049 1
2010-43-060 1
2010-75-042
2010-21-023 0
```

7 RELEVANTES CONSULTA

Estas son las métricas para esta consulta.

```
Recall R@5  0,0323
Precision P@5   0,4000
F-valor 5   0,0597
Recall R@10 0,0645
Precision P@10  0,4000
F-valor 10  0,1111
ReciprocalRank rel 1     0,0250
ReciprocalRank rel 2     0,0135
AveragePrecision AP@100 0,5952
nDCG    [0,0000, 0,0000, 0,1362, 0,1230, 0,1909, 0,1785, 0,2249, 0,2638,
0,2518, 0,2414, 0,2321, 0,2595, 0,2509, 0,2430, 0,2655, 0,2862, 0,3054,
0,3232, 0,3398, 0,3554, 0,3473, 0,3616, 0,3750, 0,3878, 0,3801, 0,3729,
0,3845, 0,3776, 0,3885, 0,3989, 0,3923, 0,3859, 0,3798, 0,3740, 0,3834,
0,3778, 0,3723, 0,3671, 0,3758, 0,3842, 0,3792, 0,3872, 0,3950, 0,3901,
0,3976, 0,3928, 0,3881, 0,3836, 0,3907, 0,4088, 0,4043, 0,4108, 0,4064,
0,4021, 0,3979, 0,3938, 0,3898, 0,3959, 0,3920, 0,3882, 0,3940, 0,3903,
0,3997, 0,3997, 0,4091, 0,4091, 0,4091, 0,4183, 0,4183, 0,4275, 0,4366,
0,4366, 0,4366, 0,4547, 0,4547, 0,4547, 0,4547, 0,4637, 0,4637, 0,4637,
0,4725, 0,4814, 0,4814, 0,4814, 0,4814, 0,4814, 0,4814, 0,4814, 0,4901,
0,4901, 0,4901, 0,4901, 0,4901, 0,4901, 0,4901, 0,4901, 0,4901, 0,4901,
0,4901, 0,4985]
```

8 MÉTRICAS CONSULTA

Esta es la consulta anterior expandida.

```
what processor obtained the best score in 2009 for the photoshop benchmark  cpu
mainframe better outdo outflank scoop topper trump account grade grievance grudge hit
make mark nock scotch seduce tally inch indiana indium inward inwards
```

9 CONSULTA EXPANDIDA

Estos son los diez primeros resultados.

```
2010-12-045 0
2010-94-040
2010-05-084 1
2010-40-061 0
2010-43-060 1
2010-31-054 1
2010-33-011 0
2010-10-078
2010-56-068
2010-06-051 1
```

10 RELEVANTES CONSULTA EXPANDIDA

Estas son las métricas para la consulta expandida.

```
Recall R@5  0,0323
Precision P@5   0,4000
F-valor 5   0,0597
Recall R@10 0,0645
Precision P@10  0,4000
F-valor 10  0,1111
ReciprocalRank rel 1     0,0256
ReciprocalRank rel 2     0,0133
AveragePrecision AP@100 0,6757
nDCG    [0,0000, 0,0000, 0,1362, 0,1230, 0,1909, 0,2435, 0,2297, 0,2182,
0,2083, 0,2412, 0,2319, 0,2236, 0,2496, 0,2417, 0,2345, 0,2561, 0,2761,
0,2947, 0,3120, 0,3282, 0,3208, 0,3356, 0,3497, 0,3425, 0,3555, 0,3678,
0,3610, 0,3725, 0,3660, 0,3597, 0,3704, 0,3806, 0,3745, 0,3842, 0,3935,
0,3877, 0,3965, 0,4050, 0,4132, 0,4076, 0,4154, 0,4230, 0,4304, 0,4375,
0,4321, 0,4390, 0,4338, 0,4287, 0,4238, 0,4190, 0,4255, 0,4318, 0,4379,
0,4332, 0,4287, 0,4346, 0,4302, 0,4259, 0,4315, 0,4371, 0,4329, 0,4477,
0,4571, 0,4665, 0,4665, 0,4665, 0,4757, 0,4850, 0,4850, 0,4850, 0,4941,
0,4941, 0,5032, 0,5032, 0,5212, 0,5302, 0,5392, 0,5392, 0,5392, 0,5392,
0,5392, 0,5481, 0,5569, 0,5569, 0,5569, 0,5569, 0,5569, 0,5569, 0,5569,
0,5569, 0,5655, 0,5655, 0,5655, 0,5655, 0,5741, 0,5741, 0,5741, 0,5741,
0,5741, 0,5741]
```

11 MÉTRICAS CONSULTA EXPANDIDA

Hemos generado dos ficheros por consulta, uno con la consulta normal y otro con la consulta expandida, ambos con el formato descrito anteriormente, con el fin de facilitar la comparación de resultados.