Proyecto: **ADR Retrieval System**
Centro: **Universidad de la Habana**

Ejecutores:
Rolando Sánchez Ramos C-311
David Manuel García C-311
Andry Rosquet Rodríguez C-311

Date: 11 de diciembre de 2022.

Resumen. En este documento se pretende hablar sobre el proyecto de Sistemas de Recuperación de Información donde se implementó un buscador que puede utilizar entre 3 modelos especializados en esta labor. Además de ello se trabajo con técnicas de retroalimentación y de evaluación de rendimiento mediante métricas.

Palabras claves: SRI, recuperación, información, modelos, retroalimentación, métricas

1. Introducción

Con este informe se tratará de brindar una explicación relacionada al proyecto ADR-Retrieval-System perteneciente a la asignatura System-Recovery-Information. Se abordarán los modelos implementados y algunas otras particularidades. Además se hará un análisis objetivo del producto obtenido y se explicará su funcionamiento.

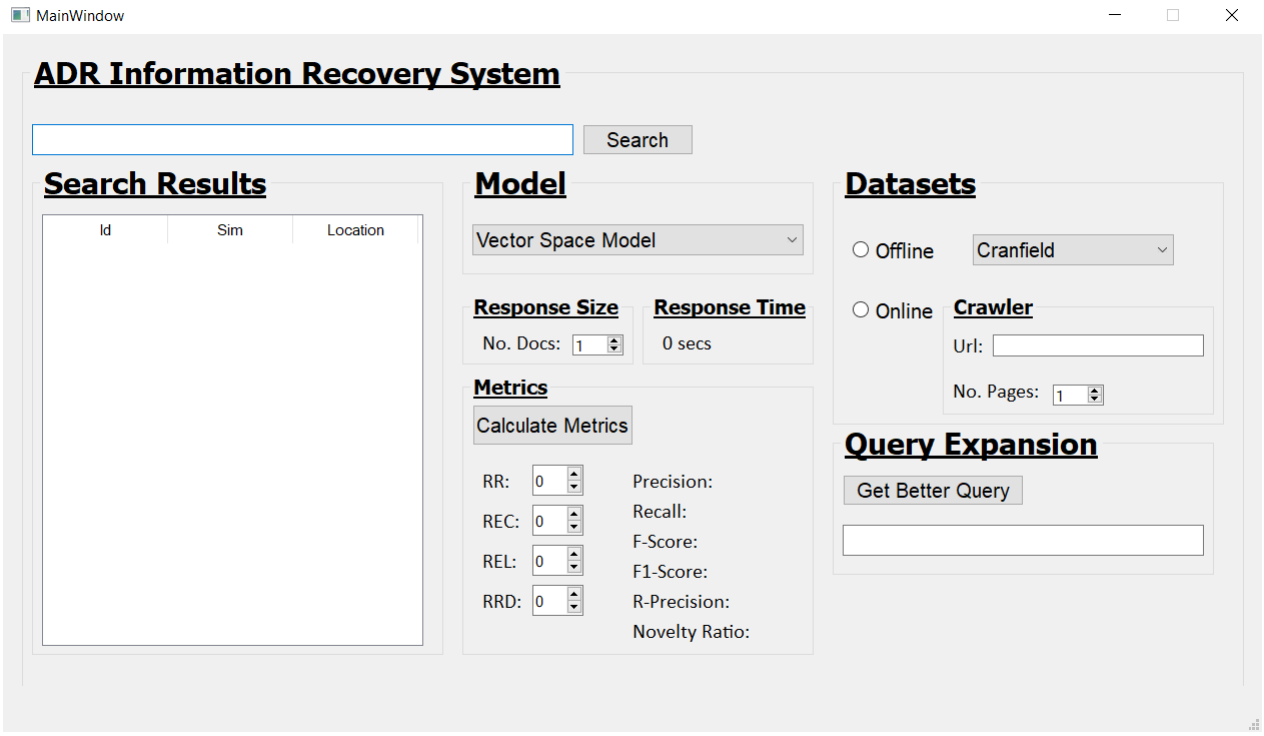
2. Desarrollo

2.1. Modo de Ejecución:

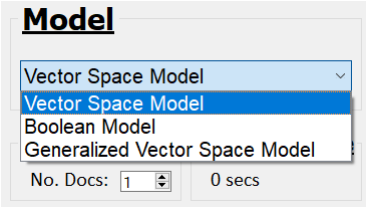
Para echar andar el buscador solamente es necesario ejecutar el main.py en el directorio inicial del proyecto haciendo simplemente *python main.py*.

2.2. Funcionamniento y ejemplos:

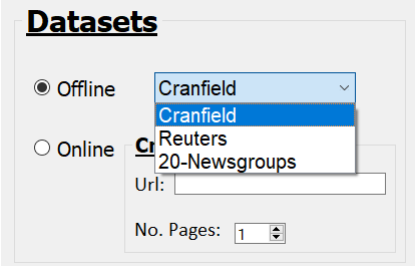
A continuación vemos una imagen de la pantalla general del buscador:



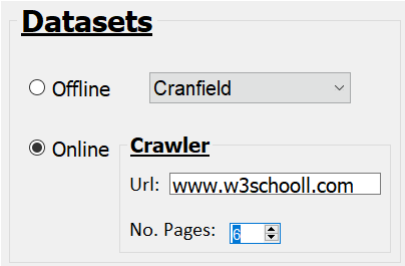
Observar el apartado Model, donde es posible escoger el modelo que desea utilizar para las búsquedas.



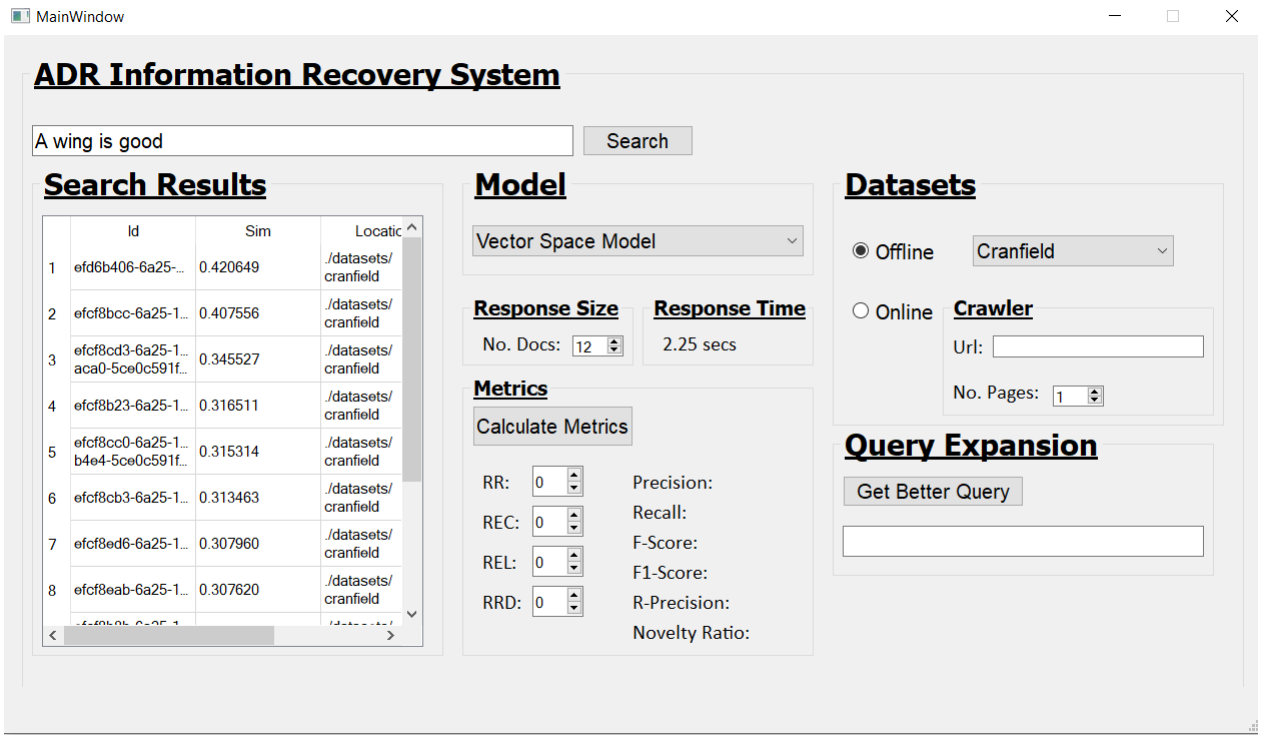
También tendremos el selector del dataset a utilizar.



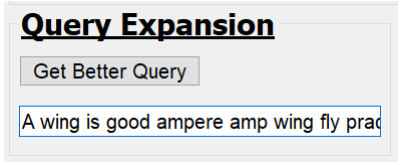
Podremos seleccionar entre una búsqueda local o utilizar alguna url y el funcionamiento de Crawler.



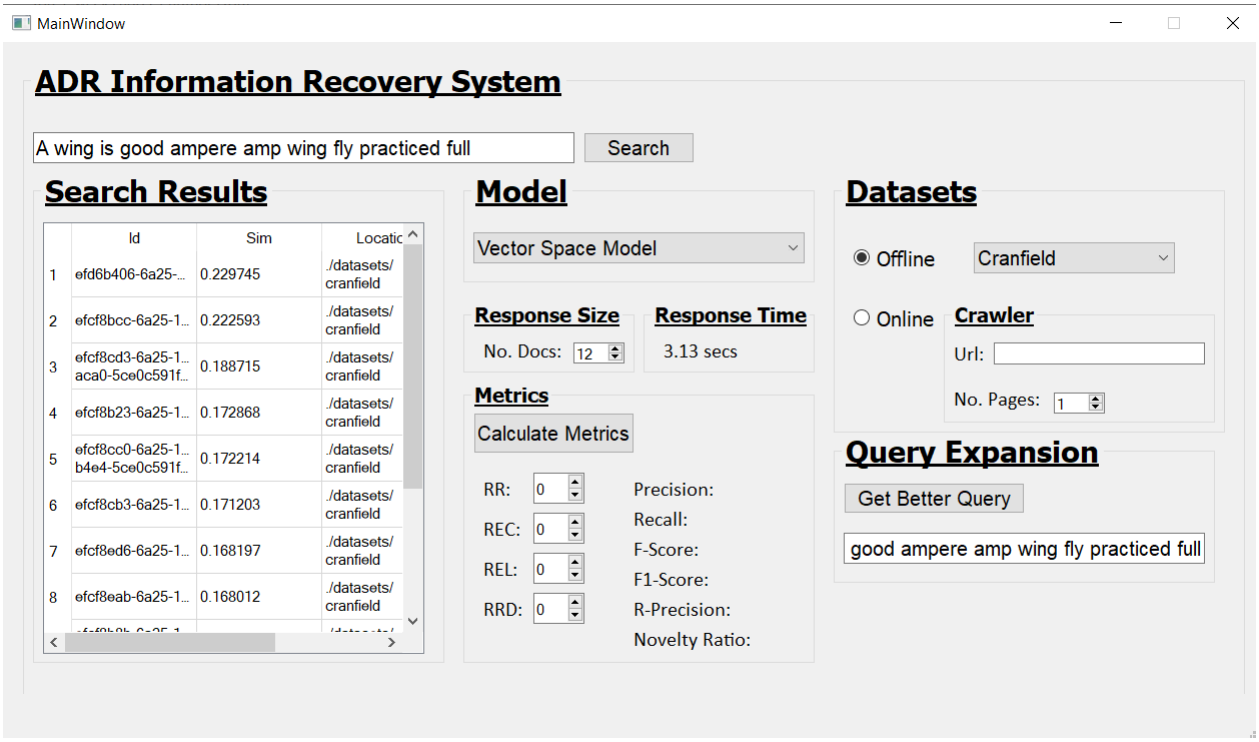
Se podrá establecer una cantidad máxima de documentos deseada en la respuesta de nuestro buscador en Response Size y a su derecha se mostrará el tiempo de respuesta a la consulta realizada. Las consultas podrán escribirse en el campo superior izquierdo y luego clickeando en el botón Search, como a continuación:



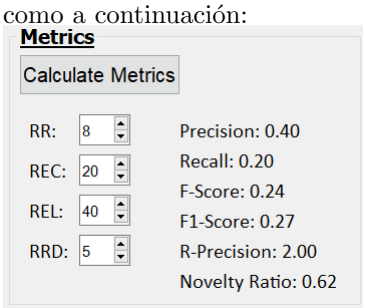
Se podrá obtener una consulta expandida en el apartado Query Exansion.



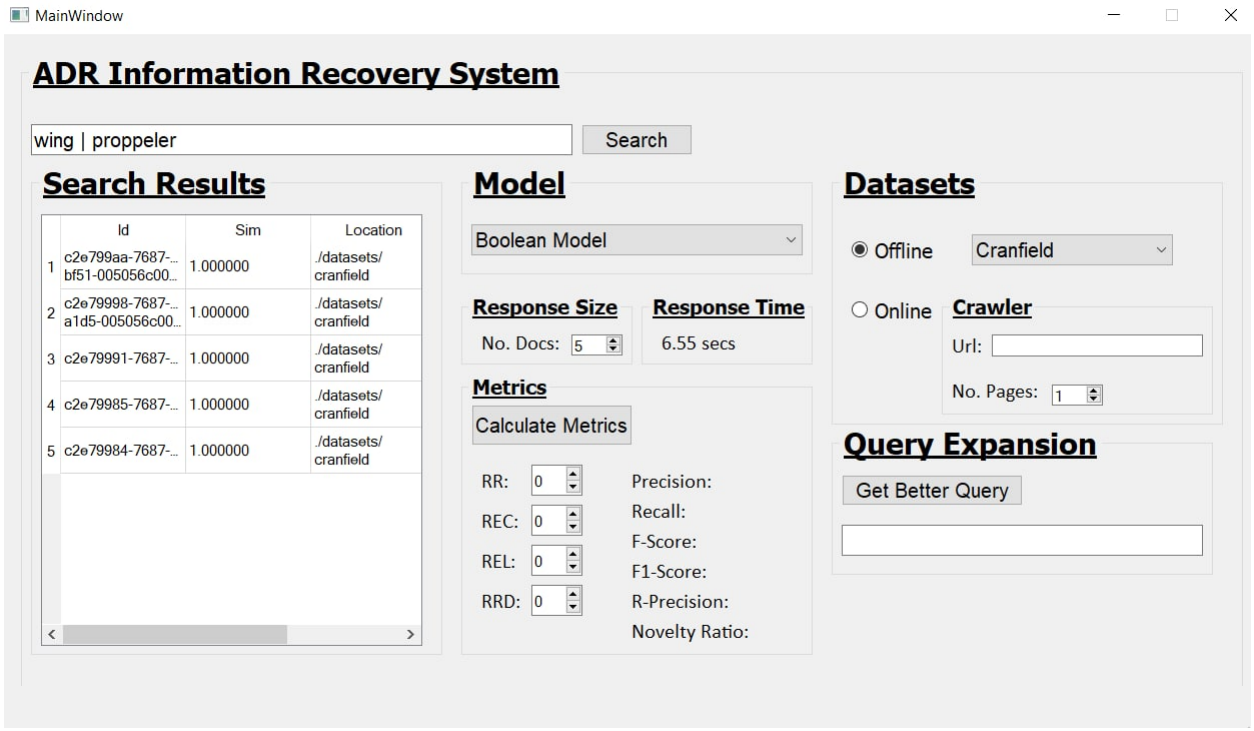
Esta puede utilizarse en el buscador y lo esperado es que mejore los resultados de la consulta no exandida.



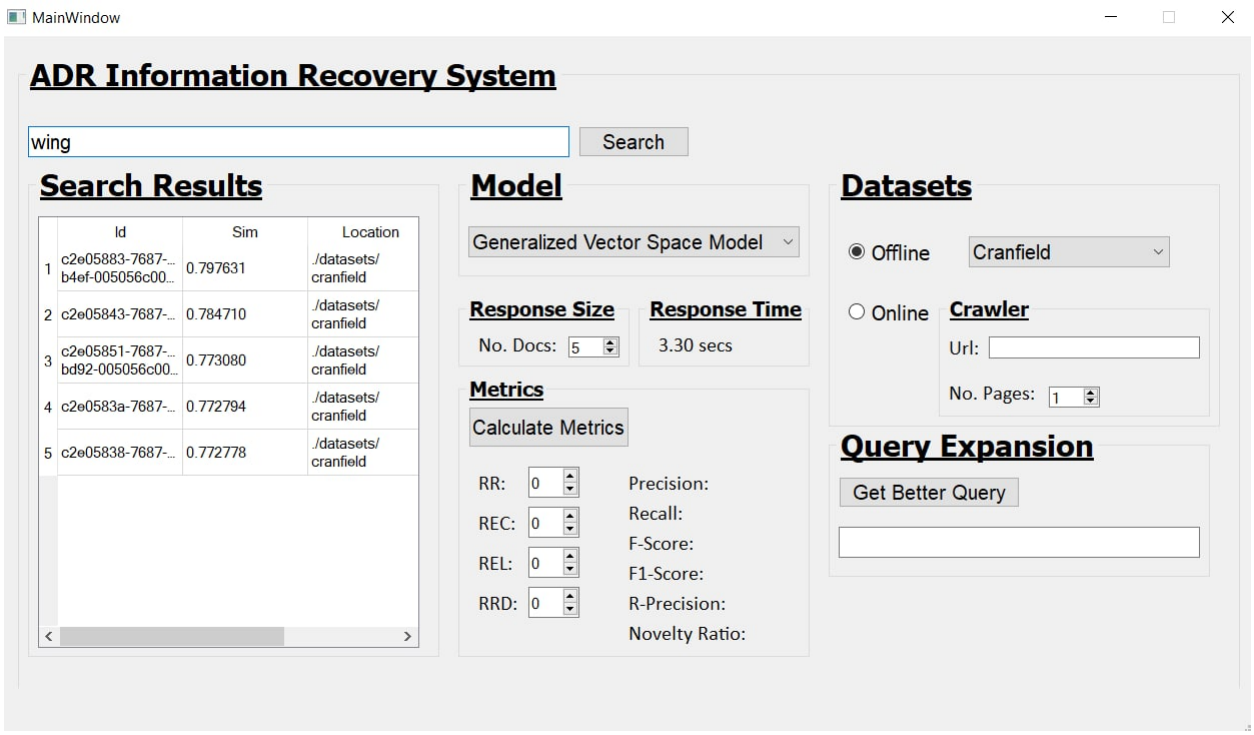
Por último también podremos hacer nuestro propio cálculo de métricas analizando la relevancia de los documentos obtenidos con la consulta y cuantos en la colección lo eran. Esto mediante las métricas como a continuación:



A continuación un par de ejemplos, el primero un ejemplo de búsqueda con el modelo booleano y el segundo con el vectorial generalizado.



Búsqueda con el modelo booleano.

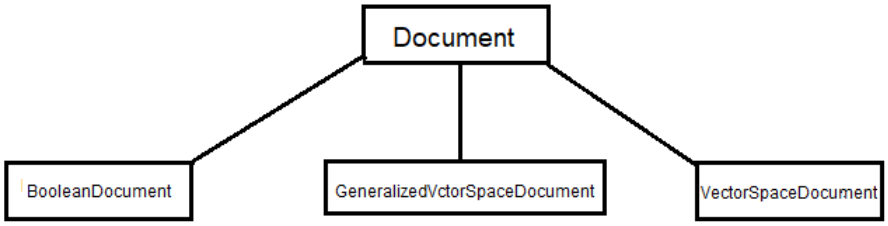


Búsqueda con Vectorial Generalizado.

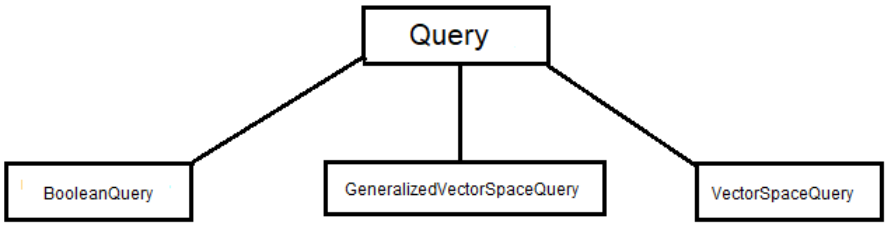
2.3. Modelos y otras características:

Los modelos implementados para nuestro sistema de recuperación de información son el Booelano, Vectorial y Vectorial Generalizado (el resumen de estos modelos se presentará en el informe definitivo). La implementación de estos se encuentra en el directorio llamado retrieval_models, cada uno de ellos se encuentra en una carpeta con su propio nombre. Cada modelo se encuentra dividido en tres archivos cuyos formatos son (NOMBRE será el específico de cada modelo):

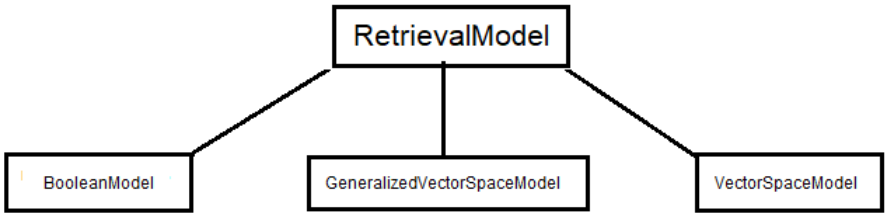
- NOMBRE_document: define una clase NOMBREDocument la cual se encarga de crear el vector del documento deseado con un vocabulario ya definido. Para cada modelo el vector se forma de manera específica, ejemplo: el booleano crea un vector binario pero los vectoriales forman uno basado en la frecuencia de cada término.



- NOMBRE_query: define una clase NOMBREQuery que mediantes sus métodos y propiedades específicas le otorga un formato a la consulta para que sea posible aplicarle la función de similitud con un documento.



- NOMBRE_model: define una clase NOMBREModel la cual mediante sus métodos internos permite calcular la función de similitud entre un documento y la consulta. Recordar que en función del modelo, la función de similitud y operaciones necesarias serán o no diferentes.



Abordemos otros elementos importantes del proyecto, estos los encontramos en el directorio utils. En este se llevaron a cabo implementaciones como:

- El algoritmo de Rocchio que sirve para aplicarle retroalimentación a los modelos vectoriales y otorgarle importancia a la información brindada por unos documentos u otros. Este se encuentra implementado en vector_feedback.py donde además de el método rocchio_algorithm, se implementó retroalimentación básica para modelos vectoriales en el método classical_vector_feedback.
- El archivo query_expansion, como dice su nombre, contiene la implementación de expansión de consultas; utilizada para obtener consultas más específicas y de la cual se esperan reultados más correctos o precisos.
- También se calculan las métricas basadas en la opinión de un experto, entre estas presición, recobrado, medida-f, r-presición y otras. Se pueden encontrar en metrics.py.

2.4. ¿Mejoras?

El proyecto no es perfecto y por tanto es propicio a mejoras. Es fácil notar que la interfaz del usuario no es la más simple ni la más vistosa, tomando como referencia otros buscadores famosos, por tanto este sería un buen aspecto a tener en cuenta. Notar que la ejecución de las consultas con modelos como el vectorial generalizado tardan un tiempo inadecuado, mayormente en la primera ejecución. Lo anterior responde a que es un modelo de recuperación de información bastante costoso y susceptible a ordenadores con pocas capacidades. Teniendo en cuenta todo lo anterior la ejecución de búsquedas con cualquier modelo resulta bastante costoso por tanto es un aspecto mejorable.

3. Resultados y Conclusiones

Los 3 modelos funcionan tan lento como amplio el espacio de búsqueda, en especial el Vectorial Generalizado. Se les realizaron pruebas con diferentes colecciones de docuementos, cada una arrojó resultados diferentes. Notamos que en espacios de búsqueda más reducidos los modelos obtienen resultados mucho mejores, ejemplo, para la colección de cranfield con 1400 documentos los modelos vectoriales tienen una precisión, entre 0.02 y 1.1, dependiendo de la calidad de la consulta y la cantidad de documentos relevantes que esta posea. El recobrado se comporta un poco mejor de manera general (oscila entre 0.04 y 1.4) para la mayor parte de las consultas, también depende en gran medida de la cantidad k de documentos que se designe recuperar. El modelo booleano no ha podido ser probado correctamente debido al formato de las consultas, esto provoca que encuentre muchos documentos que contienen los términos y quizás no era un documento seleccionado como relevante. Para las colecciones vaswani y trec-covid que poseen un gran número de documentos los modelos brindan peores resultados debido al gran espacio de búsqueda y a la gran cantidad de términos, el vectorial generalizado solo fue testeado 1 vez con estas colecciones debido al tiempo que tardó, en solo procesar los vectores de correlación entre términos para cada uno de ellos (ki). A modo de conclusión los modelos son muy mejorables basándonos en los experimentos realizados con dichas colecciones. Sin embargo en pruebas individuales realizads por el equipo los modelos no se comportan tan erróneos. Notamos que el Vectorial Generalizado es bastante probable mejore los resultados del vectorial simple pero es muy difícil comprobarlo debido al tiempo que tarda en ejecutar todos los cáculos necesarios. El modelo booleano encuentra resultados muy poderosos, sin embargo desprecia algunos parciales que son relevantes pero quizás no contiene todos los términos precisados en la consulta, además está sujeto a la correcta especificación de as necesidades en la consulta. El proyecto brinda una perspectiva de lo importante que son los modelos de búsqueda y el trabajo constante en mejorarlos con la misión de obtener resutados más precisos y más rápidos.