Recuperación de Información utilizando Secuencias Frecuentes Maximales

Por

Javier Vázquez Cuchillo

Tesis sometida como requisito parcial para obtener el grado de **Maestría en Ciencias** en el área de **Ciencias Computacionales** en el Instituto Nacional de Astrofísica, Óptica y Electrónica.

Supervisada por:

DR. JOSÉ FRANCISCO MARTÍNEZ TRINIDAD

Investigador titular del INAOE

DR. JESÚS ARIEL CARRASCO OCHOA

Investigador titular del INAOE

Tonantzintla, Puebla.
2008

Abstract

The main objective of Information Retrieval (IR) methods is to solve a user's query that expresses an information need, by retrieving a set of documents, belonging to a collection, which contain information related to the query. In all IR methods, it is necessary to use a special representation for documents and queries, commonly through words using a vector model. Word vectors are not the only way to represent the documents; also other representations based on n-gram (n consecutive words obtained from the documents) have been used. Both representations have the disadvantage of generating a large number of terms for identifying documents, and also the word representation lost the word sequential order. To solve some of these drawbacks, in this research we propose some methods for IR, which use Maximal Frequent Sequences (MFS's) -by document- to represent the documents. A MFS by document is a sequence of words that frequently appears in the document, and it is not contained in any other frequent sequence within the same document. The results show that, when the query is a small set of words, the use of MFS's by document in the proposed IR methods does not have good results compared against a method based on word representation (LUCENE), because the documents used to evaluate the IR methods were very small, and therefore the number of MFS’s in each document also was very small, moreover, some documents could not be represented because they did not contain any MFS.

Additionally, we propose an IR method based on MFS’s by document where the query is a complete document. Using this method, good results were obtained.

Resumen

El objetivo principal de los métodos de Recuperación de Información (RI) es resolver la consulta de un usuario que expresa una necesidad de información, recuperando un conjunto de documentos pertenecientes a una colección, que contienen la información relacionada a dicha consulta. En todos los métodos de RI es necesario utilizar una forma de representación para los documentos y las consultas, comúnmente mediante palabras utilizando un modelo vectorial. El uso de palabras no es la única manera de representar a los documentos, también se han usado otras formas basadas en *n*-gramas (*n* palabras consecutivas obtenidas de los documentos). Ambas representaciones tienen la desventaja de generar un gran número de términos para identificar a los documentos, y en el caso de la representación por palabras además se pierde el orden secuencial. Para resolver las desventajas anteriores, en este trabajo de investigación se proponen métodos de RI que utilizan las Secuencias Frecuentes Maximales (SFM's) -por documento- para representar los documentos. Una SFM por documento es una secuencia de palabras que no está contenida en alguna otra secuencia frecuente dentro del mismo documento. Los resultados muestran que el uso de SFM's por documento en los métodos de RI propuestos, donde la consulta es un conjunto pequeño de palabras, no tienen buenos resultados comparándolos con un método que utiliza la representación basada en palabras, ya que los documentos utilizados para evaluar los métodos de RI son muy pequeños, lo cual provocó que el número de SFM's en cada documento fuera reducido.

Adicionalmente, se propone un método de RI basado en SFM por documento donde la consulta es un documento completo. Utilizando este método se obtuvieron buenos resultados en la tarea de recuperación de documentos.

Dedicatoria

Dedico este trabajo de tesis a mis padres
agradeciendo todo el amor y apoyo
que siempre me han dado.

Agradecimientos

Agradezco a mis asesores, Dr. Jesús Ariel Carrasco Ochoa y Dr. José Francisco Martínez Trinidad, por el apoyo recibido para la realización de esta tesis.

A mis padres, Odilón Vázquez y Gregoria Cuchillo, por su amor y apoyo incondicional, por impulsarme siempre a seguir adelante ante cualquier adversidad. A mis hermanos Odilón y Mary, por su comprensión, cariño, apoyo y consejos que siempre me han ofrecido. Por ser la mejor familia que puedo tener y que siempre ha creído en mí.

A Erika Amaro, por su amor, paciencia, apoyo, por ser una gran novia y acompañarme en todo momento.

A José Alberto Contreras por ser un gran amigo y apoyarme en los momentos difíciles.

A mis amigos, Joel Rea, Eduardo Antero, Oswaldo Marín, Julio Alberto Ramos, Luz del Carmen Nieva, Germán Cuaya, Patricia Orta y Luís Suárez, de quienes he aprendido y crecido con su amistad, los quiero mucho.

A mis amigos y compañeros de la maestría, agradezco su apoyo y amistad.

De manera especial quiero agradecer a las personas que de alguna manera me apoyaron para salir adelante en esta etapa de mi vida:

A mis sinodales, Dr. Aurelio López y López, Dr. Luís Villaseñor Pineda y al Dr. Manuel Montes y Gómez, por su paciencia y aportaciones para mejorar este trabajo.

A la Dra. Claudia Feregrino Uribe por su ayuda y apoyo.

Al Dr. Manuel Martín Ortiz que siempre me ha impulsado en mis estudios y a superarme constantemente.

A toda la academia de Ciencias Computacionales agradezco su apoyo y comprensión para culminar este trabajo de tesis.

Finalmente agradezco:

Al Instituto Nacional de Astrofísica, Óptica y Electrónica por su apoyo durante la realización de mis estudios de maestría.

Al Consejo Nacional de Ciencia y Tecnología (CONACYT) de México, por el apoyo económico proporcionado para mis estudios de maestría, bajo el número de beca 201878.

ÍNDICE

Índice de figuras

Índice de Tablas

Capítulo 1. INTRODUCCIÓN

1.1. Problemática

En la actualidad con el auge de la era informática, la cantidad de información en formato digital, estructurada y no estructurada, ha crecido considerablemente. La información estructurada, en su mayoría se encuentra almacenada en bases de datos, sin embargo, gran parte de la información es no estructurada, por ejemplo documentos de texto, que son la manera más habitual de almacenar la información.

Este incremento en la cantidad de información ha provocado un aumento notable en el interés sobre la investigación en el área de recuperación de información (RI), con el claro objetivo de mejorar los métodos de RI que tratan de solucionar los problemas de representación, almacenamiento, búsqueda y recuperación de documentos [1].

Un método de RI esencialmente se encarga de resolver una consulta de un usuario mediante un conjunto de documentos, pertenecientes a una colección, que contienen la información relacionada a dicha consulta. Un método de RI no retorna respuestas a preguntas explícitas, como por ejemplo ¿Quién fue el presidente de México en 1910?, sino que informa de la existencia y localización de documentos que podrían contener la información que el usuario necesita, estos documentos son llamados relevantes.

Un método de RI consta de 3 procesos básicos:

1. La representación de los documentos de la colección y la consulta.
2. La búsqueda y recuperación de los documentos a partir de la consulta.
3. La evaluación de los documentos obtenidos para determinar si son o no relevantes a la consulta.

Los métodos de RI tratan de solucionar diferentes problemas, entre ellos, encontrar una forma de representación de los documentos de una colección. Esta tarea está relacionada con elegir los términos que mejor representen a los documentos. Actualmente las computadoras pueden permitir representar un documento utilizando su contenido completo, a lo que suele llamarse representación de texto completo [2]. Pero a pesar de ser la forma más completa de representación, esto implica un conjunto de términos muy excesivo para grandes colecciones de documentos, que se refleja en un alto costo computacional. Por lo que se trata de encontrar un equilibrio entre el número de términos utilizados para representar el contenido de los documentos y su costo computacional. Una idea que se ha utilizado para elegir y disminuir los términos representativos de los documentos es que no todas las palabras de un documento poseen la misma utilidad para describir su contenido, por lo cual algunas no deberían tomarse en consideración. Adverbios, artículos, preposiciones y conjunciones suelen ser algunas de estas palabras.

Además, al utilizar las palabras como términos se pierde el orden secuencial en el que aparecen en los documentos, lo cual es importante, pues en una consulta se puede colocar un conjunto de palabras como “León

come cabra” para encontrar los documentos que contengan esa frase en ese orden, y no todos los documentos que contengan sólo alguna de esas palabras o estén en otro orden.

Otra idea también utilizada es representar a los documentos mediante términos llamados *n*-gramas [3], que son secuencias de *n* palabras ó caracteres consecutivos. Esta representación da la posibilidad de tener términos representativos de los documentos formados desde 1 hasta *n* palabras ó caracteres, pero tiene la desventaja de que sólo se pueden utilizar para representar a los documentos con *n*-gramas cuyo tamaño máximo sea *n*. Lo cual limita a no tener *n*-gramas de mayor tamaño para representar los documentos. Esto afecta al realizar una consulta que contenga un mayor número de palabras que los *n*-gramas, pues no se puede comparar directamente dicha consulta con los *n*-gramas, por lo que se tiene que idear alguna manera para hacer la comparación.

Tanto la representación de los documentos con palabras como con *n*-gramas tienen la desventaja de limitar a la consulta que se hace en un método de RI, pues por un lado utilizando una representación de palabras se pierde el orden secuencial de una consulta, i.e. el orden de las palabras no importa. Por otro lado, los *n*-gramas sólo permiten manejar consultas de tamaño *n*, para que puedan ser comparadas directamente. Ambos enfoques de representación generan un conjunto de términos muy grande. Por estas razones se necesitan encontrar formas de representación enfocadas a solucionar estos problemas en los métodos de RI.

Una forma de representación de los documentos que se puede usar en los métodos de RI para solucionar los problemas anteriores es utilizar secuencias de palabras que se repiten como mínimo cierto número de veces en un documento y que no están contenidas dentro de otra secuencia de

palabras que se repita dentro del mismo documento, a este tipo de secuencias se les conoce como Secuencias Frecuentes Maximales (SFM's) por documento. Utilizando SFM's el conjunto de términos que representarán a los documentos será más pequeño en comparación a si se usaran palabras o *n*-gramas.

1.2. Objetivos

Los objetivos de esta tesis son los siguientes:

Objetivo General

Proponer métodos de recuperación de información basados en secuencias frecuentes maximales por documento para la representación de documentos y consultas, evaluando la utilidad de este tipo de representación para la recuperación de información.

Objetivos particulares

- Utilizar SFM's por documento para la representación de los documentos y de las consultas.
- Adaptar métodos de Recuperación de Información que permitan el uso de representaciones basadas en SFM's.
- Proponer un método de recuperación de información para el caso donde la consulta sea un documento.
- Analizar y comparar los resultados de los métodos de recuperación propuestos contra otros métodos existentes.

1.3. Organización de la tesis

Este trabajo de tesis se encuentra dividido en cinco capítulos. En el capítulo uno se describe la problemática y se definen los objetivos. En el capítulo dos se dan los conceptos fundamentales de RI, además se describen algunos trabajos relacionados. En el capítulo tres se presentan los métodos propuestos en este trabajo de investigación. En el capítulo cuatro se describen los resultados obtenidos con los métodos de recuperación propuestos y se comparan con otros métodos de RI. En el capítulo cinco se dan las conclusiones y las perspectivas de este trabajo de tesis.

Capítulo 2. MARCO TEÓRICO

En este capítulo se definen los conceptos generales de la Recuperación de Información y otros conceptos importantes para este trabajo de investigación.

2.1 Recuperación de Información (RI)

La Recuperación de Información "trata la representación, el almacenamiento, la organización y el acceso a los objetos de información[1]" [1]. El objetivo de la RI es que dada una consulta de un usuario, éste obtenga un conjunto de documentos extraídos de una colección, los cuales satisfagan la solicitud de información deseada. Un esquema clásico de RI se muestra en la figura 2.1, donde un usuario tiene una necesidad de información (consulta), la introduce a un sistema de RI, el cual realiza una búsqueda en un conjunto de documentos almacenados, previamente organizados, proporcionando al usuario todos aquellos documentos que satisfagan la necesidad de información expresada a través de la consulta.

[1] Objetos de información son los documentos de texto.

Para la creación de un sistema de RI como el que se muestra en la figura 2.1, se necesitan realizar las siguientes tareas, las cuales se explican con más detalle en las siguientes secciones:

- Pre-procesamiento de los documentos de la colección y de la consulta.
- Indexación.
    - Selección y extracción de los términos índice.
    - Generación del índice.
- Representación de los documentos y la consulta basándose en un modelo de RI.
- Búsqueda y recuperación de documentos.

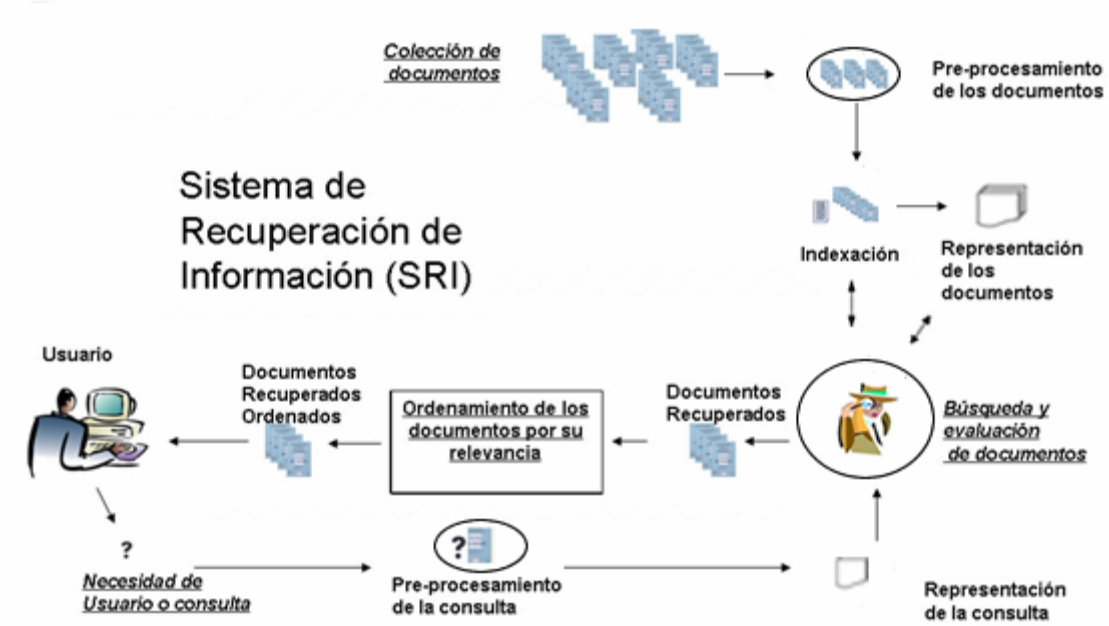

Figura 2.1. Esquema general de un sistema de Recuperación de Información.

2.1.1 Pre-procesamiento de los documentos de la colección y la consulta.

El pre-procesamiento de los documentos de la colección y la consulta, a pesar de ser dos procesos independientes como lo muestra la figura 2.1, en ambos se realizan algunas o todas las tareas siguientes:

1. Análisis del texto, que sirve para convertir un flujo de caracteres en un flujo de palabras separadas por algún signo que generalmente es un espacio en blanco [1]. En este proceso se realiza una limpieza del texto eliminando acentos, comas, puntos y otros signos de puntuación no funcionales para representar un documento. También los números son eliminados, a pesar de que representan información en una consulta como “guerras entre 1920 y 1950”. Esta eliminación de los números es debido a que tienen poco significado por sí solos y pueden encontrarse en una gran cantidad de documentos que no tengan relación a una consulta dada. También se convierten todas las palabras a mayúsculas o minúsculas, esto con el fin de uniformizar el texto de los documentos.

2. Eliminación de *stop-words* o palabras vacías, tales como las preposiciones, artículos, conjunciones, entre algunas otras palabras que aparecen en la mayoría o en todos los documentos, y por lo tanto no son útiles para diferenciar entre un documento y otro. Los beneficios de realizar esta eliminación de *stop-words* es que reduce el número de términos que identifican a los documentos. A pesar de este beneficio, la eliminación de *stop-words* podría reducir el número de documentos relacionados a una

consulta dada, por ejemplo para la consulta ”just do it” solamente seria útil la palabra “just”, lo cual reduciría la cantidad de documentos relacionados a la consulta.

3. Lematización o stemming [2]. Esto se refiere a reducir a una base léxica un conjunto de palabras similares, por ejemplo, computing, computer, computation, etc, que podrían reducirse a su forma común “comput”. La lematización permite reducir el conjunto de términos que identifican a los documentos. Una desventaja de la lematización es que no se aplica de la misma manera a cualquier lenguaje pues las bases léxicas son distintas de un idioma a otro.

Una vez realizado el pre-procesamiento de los documentos de una colección se sigue con la siguiente tarea de los métodos de RI, que es la indexación.

2.1.2 Indexación

De manera similar a los índices que encontramos en un libro, donde se indica en qué página aparece una palabra o conjunto de palabras, en un sistema de RI, los índices permiten localizar en qué documentos de la colección se encuentra una palabra o conjunto de palabras.

Así, la finalidad de la indexación es estructurar la representación de los documentos en un índice de tal manera que permita recuperarlos rápidamente para hacer la comparación con las consultas hechas por el usuario.

Para este fin, se deben realizar las siguientes tareas:

- Selección y extracción de los términos índice.
- Generación del índice.

2.1.2.1 Selección y extracción de los términos índice

Esta tarea consiste en elegir los términos que representarán a los documentos para posteriormente extraerlos de los documentos. A estos términos se les denominarán términos índice, pues formarán parte de la estructura de datos, llamada índice.

Una manera fácil de determinar los términos índice, sin tener que realizar una eliminación de palabras en el pre-procesamiento, es utilizando todas las palabras en el texto de cada documento. Esta forma de representación se le conoce como representación de texto completo que a pesar de ser una forma muy completa de representación, implica un conjunto de términos índice muy numeroso para grandes colecciones de documentos, lo cual lleva a buscar otras alternativas.

Estas otras alternativas hacen uso del pre-procesamiento, donde se eliminan palabras o se haga lematización, como se describió anteriormente, con lo cual tendremos un conjunto mucho más pequeño de palabras representativas de los documentos. Una vez obtenido este conjunto de palabras, la manera más común de elegir los términos índice es utilizando todas estas palabras resultantes, para posteriormente extraerlas y generar el índice.

Otra manera de elegir los términos índice es utilizando grupos de palabras. Para esto se han utilizado los *n*-gramas. Un *n*-grama es una secuencia de *n*-palabras (o *n*-caracteres) consecutivos. Para esto se extraen todas las secuencias de palabras de tamaño *n, n-1,..,* hasta las de tamaño 1, que aparezcan en los documentos de la colección, y estos *n*-gramas servirán para generar el índice. Una de las desventajas de usar los *n*-gramas es que limita el tamaño de los grupos de palabras que sirven como términos índice.

En este trabajo de investigación se propone un nuevo enfoque de representación para los documentos utilizando las SFM's por documento, siendo estas representaciones los términos índice. Otro enfoque usando SFM's fue propuesto por Doucet [6], el cual lleva a cabo la representación de los documentos extrayendo las SFM's por colección. En el enfoque por colección una secuencia de palabras es frecuente en la colección si al menos β documentos la contienen, donde β es un umbral dado. Una secuencia frecuente por colección es maximal si no existe otra secuencia frecuente en la colección que la contenga [8]. En el enfoque por documento propuesto, una secuencia de palabras en un documento es frecuente si aparece al menos β veces en el documento que se está analizando [6]. Una secuencia frecuente por documento es maximal si no existe otra secuencia frecuente en el mismo documento que la contenga.

2.1.2.2 Generación del índice

Una vez seleccionados y extraídos los términos que representaran a los documentos (términos índice), se genera el índice. Como se mencionó, en un sistema de RI los índices permiten localizar en cuáles documentos de

la colección se encuentra una palabra, *n*-grama o SFM (o combinación [6]) dependiendo la forma seleccionada para representar a los documentos.

Para la creación de un índice existen distintos métodos, los cuales permiten localizar rápidamente un término dentro de un documento de texto y acelerar las búsquedas, por ejemplo: archivo invertido [1], archivo de firma [10] y mapa de bits [11]. El método más común y más usado es el de archivo invertido debido a su sencillez, pues consiste básicamente en generar un listado con los términos extraídos de los documentos, relacionando a cada término con los documentos donde aparece. A continuación se muestra un ejemplo de un archivo invertido.

Supóngase que se ha elegido una representación de los documentos mediante palabras y se tienen los siguientes 3 documentos:

1. La vida en el planeta tierra es hermosa vida.
2. La vida se terminará por un meteoro.
3. El meteoro que cayó en el planeta Júpiter es un meteoro grande.

Eliminando las *stop-words* el archivo invertido que se genera se muestra en la tabla 2.1.

Como se muestra en la tabla 2.1, con esta forma de indexar podemos encontrar los documentos que contienen a algún término específico, por ejemplo, el término “vida” aparece en los documentos 1 y 2. Los términos encontrados dentro de un índice se denominan términos índice [10]. Estos términos índice pueden estar conformados de una palabra o varias palabras (como los *n*-gramas o SFM’s [6]), o inclusive conjunto de caracteres [3-5].

Número	Palabra	Documentos donde aparece la palabra
1	Vida	1-2
2	Planeta	1-3
3	Tierra	1
4	Hermosa	1
5	Terminará	2
6	Meteoro	2-3
7	Cayó	3
8	Júpiter	3
9	Grande	1

Tabla 2.1. Ejemplo de archivo invertido.

2.1.3 Representación de los documentos y la consulta

Un aspecto fundamental dentro de los métodos de RI, es cómo se realiza la comparación de similitud entre los documentos y la consulta, para esto se han definido diferentes modelos de RI, los cuales toman como base la forma de representación elegida (los términos índice elegidos) para identificar a los documentos y las consultas. Una definición formal de un modelo de RI, se encuentra en [1] que lo define como una cuadrupla $[D,Q,F,R(q_i,d_j)]$ con:

- D es un conjunto que contiene las representaciones para los documentos en la colección.

- *Q* es un conjunto que contiene las representaciones para las necesidades de información. Tales representaciones son llamadas consultas.
- *F* es un marco para modelar las representaciones de los documentos, consultas y sus relaciones.
- $R(q_i,d_j)$ es una función de ranking que asocia un número real con una consulta y un documento. El ranking define el orden en que el documento satisface la consulta.

Entre los modelos de RI más utilizados están el modelo Booleano y el modelo vectorial, los cuales se explican a continuación.

2.1.3.1 Modelo Booleano

En el modelo Booleano [11] los documentos se representan mediante vectores, de tamaño igual al número de términos diferentes que aparecen en el índice que se generó a partir de la colección de documentos, donde cada elemento del vector representa la aparición de un término del índice dentro de un documento, (1 si aparece y 0 si no aparece). Utilizando el ejemplo a partir del cual se generó el índice de la tabla 2.1, la representación Booleana de los documentos sería como se ve en la tabla 2.2.

DOCUMENTO	VIDA	PLANETA	TIERRA	HERMOSA	TERMINARA	METEORO	CAYO	JUPITER	GRANDE
1	1	1	1	1	0	0	0	0	0
2	1	0	0	0	1	1	0	0	0
3	0	1	0	0	0	1	1	1	1

Tabla 2.2. Matriz Booleana de documentos.

Se observa que cada renglón de la matriz corresponde al vector Booleano que representa a un documento.

Las consultas se construyen como expresiones Booleanas a través de los operadores AND, OR y NOT, por ejemplo la consulta “VIDA AND NOT (LUNES)”.

La búsqueda de los documentos se realiza tomando los vectores de la matriz Booleana de documentos en forma vertical para los términos de la consulta que se encuentran en el índice, para el ejemplo “VIDA AND NOT (LUNES)” serían los vectores de los términos “VIDA” y “LUNES”, con lo que se obtendrían los vectores de la tabla 2.3

VECTOR DEL TERMINO	DOCUMENTO 1	DOCUMENTO 2	DOCUMENTO 3
VIDA	1	1	0
LUNES	0	0	1

Tabla 2.3. Vectores Booleanos de los términos VIDA y LUNES, extraídos de la matriz.

Debido a que en la consulta dice NOT (LUNES), se toma el complemento del vector LUNES, generando el vector de la tabla 2.4.

VECTOR DEL TERMINO	DOCUMENTO 1	DOCUMENTO 2	DOCUMENTO 3
LUNES	1	1	0

Tabla 2.4. Vector complemento del término LUNES.

Una vez obtenidos el vector del término VIDA y el vector complemento del término LUNES se realiza el AND lógico de estos vectores. Lo que da el vector resultante de la tabla 2.5.

	DOCUMENTO 1	DOCUMENTO 2	DOCUMENTO 3
VECTOR RESULTANTE	1	1	0

Tabla 2.5. Vector resultante de la operación lógica 110 AND 110.

Esto indica que los documentos 1 y 2 son relevantes a la consulta debido a que aparece un 1 en dichas posiciones del vector resultante.

Finalmente, se puede decir que la búsqueda y recuperación de los documentos relevantes está basada en un criterio de decisión Booleano, donde un documento es relevante si en el vector resultante tiene un 1, en otro caso será no relevante.

La ventaja principal de utilizar este modelo de RI, es que es muy sencillo de entender, pues su funcionamiento está basado en operaciones Booleanas. Entre las desventajas más importantes es que la relevancia de un documento es considerada un aspecto puramente Booleano, relevante o no relevante, lo cual impide tener un ordenamiento de los documentos que tome en cuenta un grado de importancia entre la consulta y cada documento. Otra desventaja es que es difícil para un usuario expresar una consulta en una expresión Booleana, pues son combinaciones de operadores lógicos (and, or y not). Otro problema se encuentra en el poco control que hay sobre el tamaño de la salida producida por una consulta, esto ocasiona que se pueda obtener una gran cantidad de documentos como respuesta. También en este modelo los pesos de los términos, son considerados de igual importancia, no existe una manera de decir que un término es más importante que otro.

Debido a que en este trabajo se planteó la idea de utilizar consultas que fueran frases formadas de una o varias palabras, se hace disfuncional el hecho de expresar una consulta en términos de operadores lógicos. Además este modelo no permite tener un conjunto respuesta de documentos ordenados de acuerdo a un orden de relevancia. Por ende se optó por utilizar el modelo vectorial que a continuación se describe.

2.1.3.2 Modelo Vectorial

El modelo vectorial [1] representa los documentos y la consulta mediante vectores, donde la longitud de cada vector está dada por el número de términos índice en la colección. El valor de cada elemento del vector $p_i(t_j)$ representa el peso del término *j* en el documento *i.*

El peso de un término en un documento puede estar dado de diferentes formas, por ejemplo:

$$p_i(t_j) = \begin{cases} 1 & si\ \ t_j\ está\ en\ el\ documento\ i \\ 0 & otro\ caso \end{cases} \qquad (1)$$

$$p_i(t_j) = tf_{i,j} \qquad (2)$$

donde:

$tf_{i,j}$ = frecuencia del término *j* en el documento *i*.

$$p_i(t_j) = idf_j = \log\left[td \big/ d_j \right] \qquad (3)$$

donde:

idf_j = Frecuencia de documento inversa [2].

td = Total de documentos en la colección.

d_j = Número de documentos que contienen el término *j*

Otra opción es:

$$p_i(t_j) = tf_{i,j} \cdot idf_j \qquad (4)$$

En este modelo, la comparación de un documento con la consulta se realiza mediante una función de similitud que utiliza operaciones vectoriales. La función más utilizada es la función del coseno que se define como:

$$\cos(d_1, q) = \frac{\sum_{i=1}^{m}((d_1(t_i)) \bullet (q(t_i)))}{\left(\sum_{i=1}^{m}(d_1(t_i))^2\right)\left(\sum_{i=1}^{m}(q(t_i))^2\right)} \qquad (5)$$

donde:

$d_1(t_i)$ = peso que tiene el término *i* del documento d_1.

$q(t_i)$ = peso que tiene el término *i* de la consulta *q*.

m = número de términos diferentes en el índice de la colección completa.

[2] Es la proporción inversa de la cantidad de documentos en la colección que contienen un término. Mientras más documentos contengan a un término su idf será menor. Por otro lado, mientras menos documentos contengan a un término su idf será mayor.

Tomando nuevamente el ejemplo con el cual generamos el índice de la tabla 2.1 y suponiendo que hemos elegido un pesado basado en *(4)* obtendríamos una matriz de documentos-términos como la que se muestra en la tabla 2.6. Calculando el pesado sólo en los términos índice que aparezcan en el documento, poniendo un 0 en caso contrario. Por ejemplo, para calcular el pesado del vector del documento 1, para el término *VIDA* se multiplica la frecuencia del término en el documento, por la frecuencia inversa del mismo término en el documento, esto es: 2 * LOG 10(3/2)= 0.35.

DOCUMENTO	VIDA	PLANETA	TIERRA	HERMOSA	TERMINARA	METEORO	CAYO	JUPITER	GRANDE
1	0.35	0.18	0.47	0.47	0	0	0	0	0
2	0.18	0	0	0	0.47	0.18	0	0	0
3	0	0.18	0	0	0	0.35	0.47	0.47	0.47

Tabla 2.6. Matriz de documentos-términos con pesado tf-idf.

De la misma forma como se obtienen los pesados de los términos en los vectores de cada documento, se genera el vector consulta, esta consulta es un conjunto de palabras. Así por ejemplo si se tiene la consulta “VIDA HERMOSA METEORO” el vector consulta sería el que se muestra en la tabla 2.8.

DOCUMENTO	VIDA	PLANETA	TIERRA	HERMOSA	TERMINARA	METEORO	CAYO	JUPITER	GRANDE
Consulta	0.18	0	0	0.47	0	0.18	0	0	0

Tabla 2.8. Vector consulta

Ahora, para encontrar la similitud entre un documento con respecto a la consulta se realiza lo siguiente:

- Se buscan en el índice los términos de la consulta, cuyo peso sea distinto de 0, para obtener los documentos que tienen relación con la consulta. Para el ejemplo de la tabla 2.8 serían los términos VIDA, HERMOSA y METEORO.
- Se extraen de la matriz de documentos-términos (Tabla 2.6) los vectores de cada documento relacionado a la consulta, para el ejemplo serían los vectores de los documentos 1,2 y 3.
- Para cada uno de estos vectores junto con el vector consulta se aplica la función de similitud coseno, con lo que se encuentra el valor de parecido entre cada documento y la consulta.
- Finalmente, lo anterior proporciona un listado de valores de similitud que se ordena de manera descendente con respecto a su valor de parecido, generando el conjunto de documentos respuesta para la consulta dada. Para el ejemplo esta lista se muestra en la Tabla 2.9.

Documento	Valor de parecido
1	0.29
2	0.06
3	0.06

Tabla 2.9. Documentos respuesta a la consulta “VIDA HERMOSA METEORO” ordenados de acuerdo a su valor de parecido.

La ventaja de este modelo es la obtención de una lista ordenada de documentos que satisfacen la consulta, con lo cual es posible controlar el número de documentos dados como respuesta a una consulta, ya sea limitando este número o estableciendo un umbral de similitud. Además permite tener pesos distintos en los términos de los vectores, de acuerdo a la importancia de cada uno de ellos. También permite tener como consulta una frase de palabras con cualquier tamaño, sin problemas para el usuario al

expresar dicha consulta. Por estas ventajas en este trabajo se usó este modelo de RI, utilizando como términos índice las SFM's por documento.

2.1.4 Búsqueda y recuperación de documentos

La búsqueda y recuperación de los documentos relacionados a una consulta va ligada al modelo de RI que se use en el método de RI. Pues es ahí donde se define la función de comparación, que proporciona un valor de parecido entre cada documento y una consulta dada, con lo cual se puede generar un ordenamiento de los documentos con respecto a esos valores. Esta lista ordenada en grado de relevancia de los documentos es la respuesta que se le da al usuario, para una determinada consulta.

2.2 Evaluación de un sistema de RI

La evaluación de un sistema de RI [1] se realiza utilizando una colección de prueba. Esta colección consiste en un conjunto de documentos y un conjunto de consultas. Para cada una de las consultas varios especialistas han seleccionado los documentos relevantes de la colección, estos documentos serán los documentos relevantes para la consulta dada. Así en la evaluación de un sistema de RI se compara para cada una de las consultas de la colección de prueba, los documentos que el sistema ha obtenido, es decir, los documentos recuperados, y los documentos marcados como relevantes para esa consulta.

Para evaluar una respuesta para una consulta y al mismo tiempo permitir comparar diferentes sistemas de RI se han definido diversos criterios de evaluación de los mismos, entre los cuales los más utilizados son la

precisión y el recuerdo. Para una consulta, por precisión se entiende que porcentaje de los documentos recuperados son relevantes. Por recuerdo se entiende que porcentaje de los documentos relevantes en la colección son recuperados. Estas medidas están definidas como sigue:

$$Precisión = DRR / DR \quad (6)$$

donde:

DRR= Número de documentos relevantes recuperados

DR = Número de documentos recuperados

$$Recuerdo = DRR / DRC \quad (7)$$

donde:

DRC = Número de documentos relevantes en la colección

La precisión mide la capacidad que tiene un sistema de RI para recuperar solamente los documentos relevantes, mientras que el recuerdo mide la capacidad que tiene el sistema para recuperar todos los documentos relevantes.

2.2.1 Gráficas recuerdo-precisión

Las medidas de precisión y recuerdo se relacionan en gráficas de recuerdo-precisión [1], las cuales permiten visualizar el comportamiento y calidad de los métodos de RI. Para crear estas gráficas se revisan los documentos recuperados, que son relevantes. Para explicar esto utilizaremos un ejemplo descrito en [7]. Supongamos una consulta, para la cual existen 16 documentos relevantes. Ahora utilizando el método de RI y utilizando la

consulta, se obtienen 20 documentos, ordenados de acuerdo al criterio del método de RI, como se muestran en la tabla 2.10, indicando con un asterisco los documentos recuperados que son relevantes.

Ejemplo de documentos recuperados para una consulta				
1*	5	9	13*	17
2	6	10	14*	18
3*	7*	11*	15	19*
4	8*	12	16	20

Tabla 2.10. Documentos recuperados para una consulta dada, donde se ven marcados aquellos que son relevantes para esa consulta.

Una vez que se tiene la respuesta del método, se analiza la precisión y el recuerdo para cada documento recuperado que es relevante. Empezando por el primer documento relevante que se ha recuperado, la precisión será 1 de 1 (1 relevante de 1 recuperado), i.e. del 100%. El recuerdo es 1 documento relevante de un total de 16, i.e. el 6.25%. El siguiente documento relevante ocupa la tercera posición, es decir, la precisión es de 66.67%(2 relevantes de 3 recuperados), y el recuerdo de 12.50%(2 relevantes de un total de 16). Para el tercer documento relevante que ocupa la séptima posición, la precisión es de 42,86% (3 relevantes de 7 recuperados) y el recuerdo de 18.75% (3 relevantes de un total de 16). Para el resto de documentos relevantes se obtiene la tabla 2.11. Por convención se toma que la precisión se hace cero cuando ya no quedan documentos relevantes en los recuperados. La gráfica obtenida de los valores de la tabla 2.11, es la que se muestra en la figura 2.2.

Recuerdo (%)	Precisión (%)
6.25	100.00
12.50	66.67
18.75	42.86
25.00	50.00
31.25	45.45
37.50	46.15
43.75	50.00
50.00	42.11
56.25	0
62.50	0
68.75	0
65.00	0
81.25	0
87.50	0
93.75	0
100.00	0

Tabla 2.11. Valores resultantes de precisión y recuerdo para una consulta.

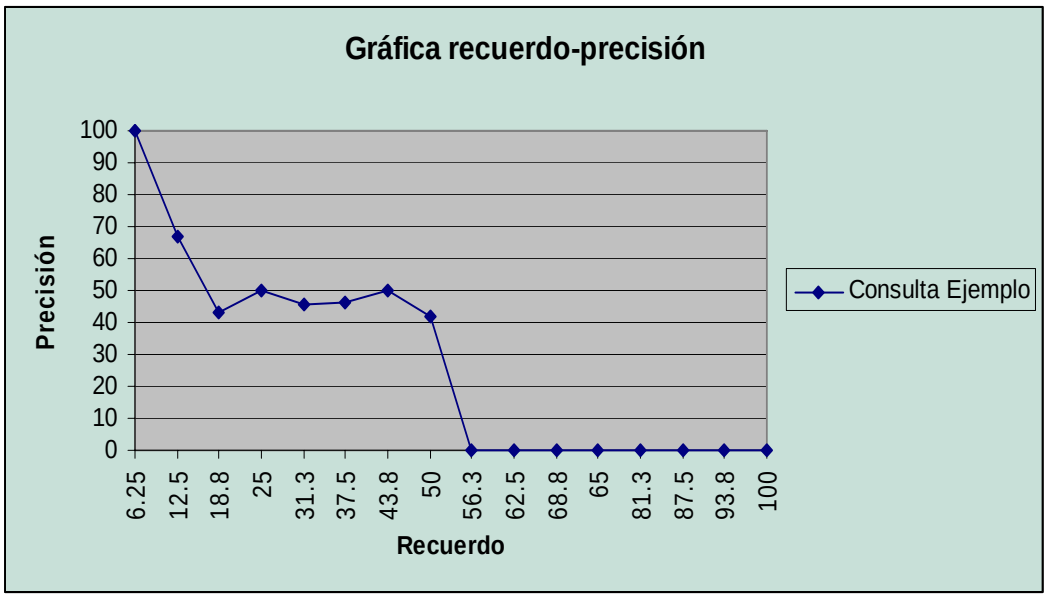

Figura 2.2. Gráfica recuerdo-precisión

El número de puntos en una gráfica recuerdo-precisión como la que se muestra en la figura 2.2 viene dada por la cantidad de documentos

relevantes en la colección para una consulta dada. La consulta tiene 16 documentos relevantes por lo tanto existen 16 puntos. Existen documentos relevantes no recuperados los cuales se identifican con los puntos que tienen precisión cero.

Un punto en la gráfica recuerdo-precisión es mejor mientras el valor de precisión se aproxime más a 100, pues esto indica que los documentos relevantes a la consulta se recuperan en las primeras posiciones del conjunto de documentos respuesta.

2.2.2 Gráficas recuerdo- precisión interpoladas a 11 puntos

Normalmente se toma interpolación de la precisión para 11 puntos estándar de recuerdo en los niveles 0%,10%,…100% [7]. La precisión para cada uno de estos niveles se calcula como el máximo valor de precisión entre ese valor y el siguiente. De manera formal se define en [7] como sigue, sea n_j *con* $j \in \{0,1,2,\ldots,10\}$*, el nivel estándar j-esimo de recuerdo, entonces, el valor de precisión interpolada para ese nivel es dado como:*

$$P(n_j) = \max_{n_j \le n \le n_{j-1}} P(n) \qquad \text{Siendo } P(n) \text{ la precisión.} \qquad (8)$$

Calcularemos el valor de precisión interpolada siguiendo el ejemplo de la sección 2.2.1. Se calcula primero el valor para el nivel 100% de recuerdo, obteniendo un valor de precisión de 0%. Para el nivel de recuerdo del 90% se toma el máximo entre el 90% y el 100% de recuerdo. En el ejemplo son los valores del 93.75% y del 100%. El valor máximo es 0, que es tomado para el nivel estándar 90% de recuerdo.

Para los niveles de recuerdo del 80%, 70% y 60% se obtiene una precisión interpolada de 0. Para el nivel 50% se tienen que tomar los valores entre 50% y 60%. En este caso tenemos los valores del 50%(42.11), 56.25(0) y 60%(0), de estos el máximo es 42.11. Para el nivel 40% se toman los valores entre 40% y 50%. Tenemos el valor del 43.75% (50) y el valor 50% (42.11). El máximo es el 50. Para el nivel 30% se toman los valores entre 30% y el 40%. Se tiene el valor del 31.25 (45.45), 37.50 (46.15) y el 40% (50). El máximo es 50. Y así sucesivamente hasta llegar al nivel 0 donde se toman los valores entre el 0% y el 10%. Se tiene el valor de 6.25 (100) y el 10% (66.67). El máximo es 100. La tabla 2.12 es la que se obtendría de la interpolación y su gráfica de recuerdo-precisión interpolada sería la que se muestra en la figura 2.3.

Recuerdo (%)	Precisión (%)
0	100
10	66.67
20	50
30	50
40	50
50	42.11
60	0
70	0
80	0
90	0
100	0

Tabla 2.12. Valores recuerdo-precisión interpolada.

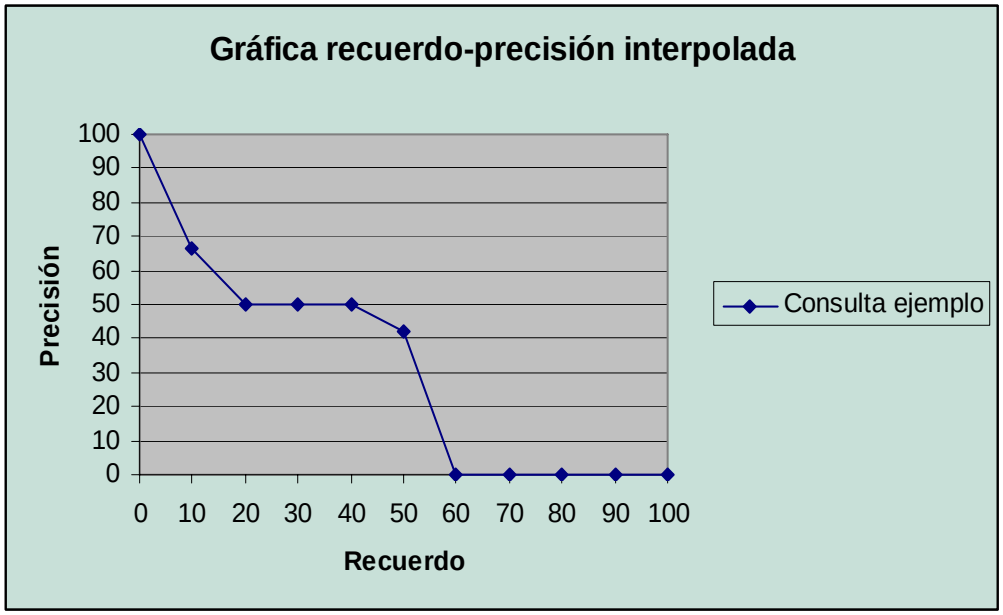

Figura 2.3. Gráfica recuerdo-precisión interpolada

En este trabajo de tesis se utilizará para mostrar el rendimiento de los métodos propuestos las gráficas de recuerdo-precisión interpolada a 11 puntos.

2.3 Trabajo relacionado

Existen una gran variedad de trabajos relacionados al área de recuperación de información, en los cuales se utilizan distintas maneras de representar a los documentos, usando palabras, *n*-gramas e inclusive SFM's. A continuación se describen algunos métodos de RI mostrando la utilidad que han tenido estas formas de representación.

En [4] se presenta un método de RI denominado TELLTALE que utiliza *n*-gramas como términos índice para representar a los documentos. Estos *n*-gramas son formados por secuencias de *n* caracteres consecutivos. El *n* utilizado para extraer los *n*-gramas fue 5, pues se trataba de que los *n*-gramas pudieran tener un significado como las palabras. El método TELLTALE puede indexar texto en cualquier lenguaje siendo una de las características más sobresalientes de este método. El uso de *n*-gramas de tamaño 5 disminuyó el tamaño de los índices que con palabras era mayor. El método tiene la capacidad de soportar errores en el texto, por realizar la comparación usando los *n*-gramas. En el método TELLTALE no se eliminó ningún tipo de palabra (como son las *stop-words*), ni se realizó lematización. El método TELLTALE representa a cada documento por un vector de *n*-gramas. La medida de similitud usada fue la función coseno. En conclusión, según los resultados experimentales, los métodos basados en palabras no fueron mejores que los resultados del método TELLTALE basado en los *n*-gramas de tamaño 5. El método TELLTALE tiende a tener un gran número de términos índices al usar solamente los *n*-gramas de tamaño 5, lo cual no permitió indexar grandes volúmenes de texto. A pesar de ello, es un método que ofrece independencia del lenguaje y es tolerante a errores.

En [3] se enfoca el problema de Recuperación de Información al procesamiento de documentos en lenguaje chino. La representación de los documentos y las consultas está basada en la combinación de palabras y *n*-gramas. Mencionan que la ventaja de usar *n*-gramas es que no se requiere un conocimiento lingüístico, ésta es la principal razón de su uso con el lenguaje chino y otros lenguajes asiáticos. En este trabajo se usan uni-gramas, bi-gramas y palabras, como términos índice. La representación de los documentos fue vectorial y el pesado de los términos índices de los

vectores fue calculado mediante la frecuencia en el documento y la frecuencia inversa en el documento. La función de similaridad utilizada para comparar consultas con documentos fue basada en la función coseno. No realizan ninguna eliminación de palabras, ni realizan lematización. Las pruebas de este método de RI estuvieron basadas en el corpus chino de TREC [12]. La combinación de los uni-gramas, bi-gramas y las palabras incrementaban el número de términos que conforman el índice, lo cual genera un aumento de espacio en disco y en procesamiento. Inclusive usando solamente los bi-gramas se generaba un índice muy grande. Por lo cual tomando en cuenta los experimentos es mejor usar las palabras en combinación con los uni-gramas de caracteres. Una desventaja importante de este trabajo es que el sistema está limitado a idiomas asiáticos como el chino, japones, entre otros. El método no se compara con otros trabajos relacionados, simplemente reporta los resultados obtenidos.

En [5] se aborda la Recuperación de Información usando un modelo vectorial de *n*-gramas para representar a los documentos y las consultas. Se utilizó como medida de similaridad la función coseno. Este sistema usó 4-gramas compuestos de caracteres en lugar de palabras, permitiendo la inclusión de espacios. Para el pesado de cada 4-grama se usó la medida de frecuencia de documento inversa (idf). La consulta igual que los documentos estuvo representada por 4-gramas. La evaluación se realizó con un programa denominado trec-eval provisto por NIST [13], que produce reportes de precisión y recuerdo. Se usaron conjuntos de datos y consultas TREC. Como el sistema descrito en [3] este método tolera errores en documentos y consultas. El método no requiere un pre-procesamiento como la lematización o eliminación de *stop-words*. El método es independiente del lenguaje. De acuerdo a los experimentos realizados, este método basado en 4-gramas es mejor que métodos similares basados en *n*-gramas. El método tiene una

limitante que es la gran cantidad de 4-gramas que se obtienen, lo cual generan un uso excesivo de espacio en disco y afecta el rendimiento del método. El método no es útil para uso interactivo real, para lo cual se plantea para trabajo futuro solucionar esta problemática.

En [6] se propone el uso de SFM's por colección para RI, representando los documentos mediante palabras y SFM's por colección. El peso de cada término es obtenido mediante la frecuencia inversa. No se eliminan las palabras vacías de los documentos, ni de las consultas. Se usó la colección INEX, que se compone de 12,107 artículos cientificos escritos en ingles de revistas de la IEEE, combinadas con un conjunto de consultas y sus correspondientes evaluaciones. Se extrajeron 328,289 SFM's por colección. Se usaron medidas de precisión y recuerdo para evaluar la calidad de la respuesta. Los resultados obtenidos al utilizar las SFM's por colección mejoran significativamente niveles de recuerdo. Por lo tanto las SFM's por colección deberían ser más útiles en el caso de necesidades de información exhaustiva. La precisión usando las SFM's por colección no es muy buena, pues usando un esquema de palabras se obtiene mejor precisión. El uso solamente de las SFM's por colección en dicho trabajo no tuvo buenos resultados, por lo que los autores propusieron una forma de representación mezclando las palabras con las SFM's por colección, mejorando el rendimiento del método propuesto.

En [14] se presenta un método denominado LUCENE en el cual se representan a los documentos y a las consultas mediante vectores de palabras. Permite eliminar las palabras vacías y realizar cualquier tipo de pre-procesamiento en los documentos. Permite indexar cualquier tipo de documento de texto mediante la técnica de archivo invertido, además esta indexación es de forma incremental i.e. que podemos agregar un nuevo

documento al índice sin necesidad de indexar nuevamente toda la colección. El pesado que utiliza LUCENE está basado en tf-idf. La comparación de cada uno de los documentos con la consulta se realiza utilizando la función coseno. Puede permitir consultas formadas de palabras unidas con conectores lógicos (AND, OR, NOT), ó un conjunto de palabras consecutivas. La información detallada de la estructura interna de este sistema la podemos encontrar en [14]. LUCENE tiene una buena precisión con respecto a otros métodos que usan representación basada en palabras.

Como se describió, en los trabajos previos se han realizado diferentes métodos de RI, basando la representación de los documentos en palabras, *n*-gramas y SFM's por colección. Al utilizar palabras se pierde el orden secuencial de las palabras y representa una alta dimensionalidad de los vectores. Por otro lado, el usar *n*-gramas a pesar de que permite tener cierto orden secuencial de palabras, la generación de los *n*-gramas provoca también tener una gran dimensionalidad de los vectores. Además, se han usado SFM's para la representación de los documentos, pero estas han sido extraídas a nivel colección, las cuales proporcionan más información acerca de lo que trata la colección y no de lo que trata un documento en especifico de la colección. Por estas razónes, en este trabajo se propone utilizar las SFM's por documento, pues éstas proporcionan información acerca de lo que trata el documento, al ser secuencias de palabras que se repiten como mínimo un par de veces. Otra ventaja al utilizar SFM's por documento es que se tiene una menor cantidad de términos índice, con respecto al uso de palabras o *n*-gramas, con lo cual la representación vectorial es más pequeña para cada documento. Además, usar las SFM's por documento permite conservar el orden secuencial de las palabras, lo cual es importante en las consultas que se realizan en los métodos de RI.

Capítulo 3. Métodos propuestos

En este capítulo se describen a detalle los métodos propuestos en esta tesis, los cuales utilizan el modelo vectorial y una representación de los documentos basada en las SFM's por documento. Se utilizan las SFM's por documento debido a que pueden describir el texto de los documentos conservando el orden secuencial de las palabras, además de que se genera una representación más pequeña para los documentos de una colección, en comparación a la representación usando palabras ó *n*-gramas de tal manera que la dimensionalidad de los vectores representativos para cada documento es menor.

En este trabajo de tesis se proponen tres métodos de RI, los cuales basan su representación de los documentos en las SFM's por documento. El primer método, el cual se denominó **MSFM1**, utiliza dos enfoques para llevar a cabo el pesado de los documentos: Booleano ó tf-idf y dos enfoques para realizar el pesado de la consulta: Booleano ó de intersección. El segundo método denominado **MSFM2_PAL** usa las palabras de las SFM's (por documento) para representar a los documentos, utilizando dos enfoques para el pesado: Booleano y tf-idf. Tanto MSFM1 como MSFM2_PAL utilizan como consulta un conjunto pequeño de palabras por lo que se propone un tercer método que se denominó **MSFM3,** el cual utiliza como consulta un

documento completo. A continuación se describen con detalle cada uno de los métodos.

3.1 Método MSFM1

Este método de RI está basado en el modelo vectorial y utiliza para representar el contenido de los documentos, las SFM´s por documento. Consiste en obtener las SFM's (por documento) de los documentos de una colección, usándolas para construir los vectores de cada documento. El tamaño de cada vector es igual al número de SFM's por documento distintas encontradas en la colección. Cada elemento de los vectores contiene un peso binario relacionado a si existe o no la SFM en cada documento. Luego, basándose en esta representación para poder realizar las búsquedas de los documentos relevantes a cada consulta, se genera una representación similar de cada consulta, utilizando un pesado Booleano y de intersección, como se explicará más adelante,

El esquema general que se usó para la creación de este método de RI se muestra en la figura 3.1. En las siguientes secciones se explica a detalle en qué consiste el método de recuperación con sus dos formas de pesado.

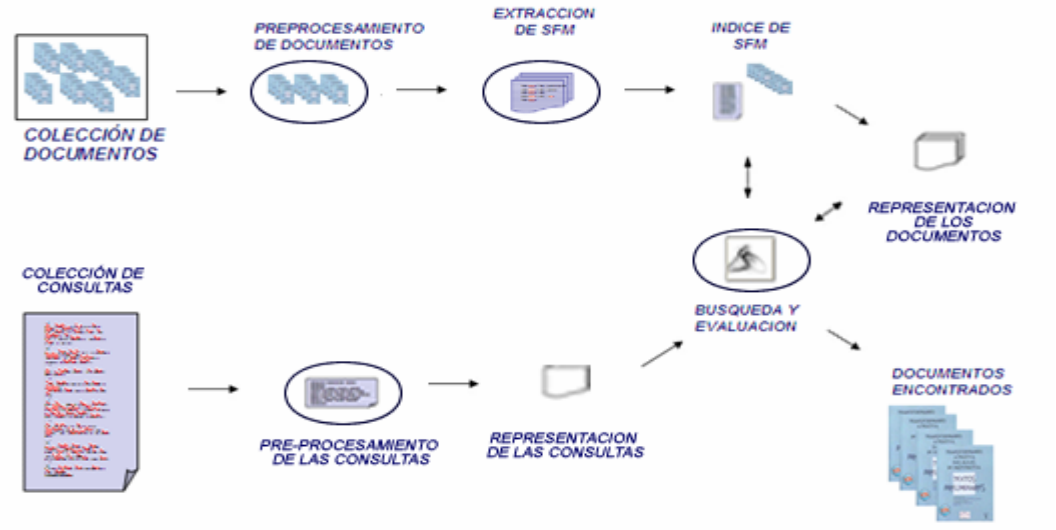

Figura 3.1. Esquema general del método de RI basado en las SFM´s por documento con pesado Booleano y pesado de intersección: MSFM1.

3.1.1 Pre-procesamiento de los documentos de la colección y de la consulta

El pre-procesamiento que se aplicó a los documentos de las colecciones y las consultas correspondientes fue el siguiente:

1. Eliminar signos de puntuación.

2. Convertir a mayúsculas el texto de los documentos y la consulta, con la finalidad de uniformizar el texto.

3. Eliminar *stop-words*. Las *stop-words* que se eliminaron se basaron en conjunciones, artículos, preposiciones y otras palabras[3]. Se decidió realizar la eliminación de las *stop-words*

[3] Ver Apéndice A.

debido a que son palabras que se repiten en la mayoría de los documentos, además de que se realizaron experimentos incluyéndolas y no se ganó precisión. También eliminar las *stop-words* permitió tener un índice más pequeño, con lo cual se aumentó la rapidez de la búsqueda.

Para ejemplificar el pre-procesamiento, en la figura 3.2a se muestra una colección de 3 documentos, los cuales se estarán utilizando a lo largo de este capítulo para explicar los métodos propuestos. El resultado del pre-procesamiento de los 3 documentos se muestra en la figura 3.2.b.

a. Colección de 3 documentos	**b. Documentos pre-procesados**
DOC.1--- The IBM about DSD technical and information market secure, the traditional systems library IBM mechanized features an computer systems the IBM session received, compares reviews is center new market secure. DOC.2--- The IBM systems is a information in market secure need and help record, buy the ibm libraires and network had memory market secure between center, one mouse recors and computer. DOC.3--- The systems in Mexico are systems good central.	1=IBM,DSD,TECHNICAL,INFORMATION,MARKET,SECURE,TRADITIONAL,SYSTEMS,LYBRARY,IBM,MECHANIZED,FEATURES,COMPUTER,SYSTEMS,IBM,SESSION,RECEIVED,COMPARES,REVIEW,CENTER,NEW,MARKET,SECURE 2=IBM,SYSTEMS,INFORMATION,MARKET,SECURE,HELP,RECORD,BUY,IBM,LIBRARIES,NETWORK,MEMORY,MARKET,SECURE,CENTER,MOUSE,RECORD,COMPUTER 3=SYSTEMS,MEXICO,SYSTEMS,CENTRAL

Figura 3.2. Ejemplo de documentos pre-procesados

3.1.2 Indexación para MSFM1

Para realizar la indexación en el método MSFM1, se obtienen los términos índice representativos de los documentos de cada colección, y

posteriormente se construye el índice, que servirá para acelerar la búsqueda de los documentos.

3.1.2.1 Extracción de términos índice

Para representar los documentos es necesario elegir los términos índice que los identificarán, en este método se utilizan las SFM's por documento, las cuales se definen como:

Una secuencia se considera que es frecuente si aparece al menos β veces en un documento, donde β es el umbral de frecuencia dado [8]. Una secuencia frecuente es considerada maximal si no es subsecuencia de otra secuencia frecuente, en el mismo documento, denominándose SFM por documento.

El utilizar SFM's es una manera de reducir el número de secuencias frecuentes encontradas en un documento. En este trabajo se decide utilizar SFM's por documento debido a que se necesitan palabras o conjunto de palabras que identifiquen a cada documento de la colección. Para esto, basados en la definición de SFM's por documento sabemos que dichas SFM's se repiten cierto número de veces como mínimo dentro de un documento por lo que se les puede considerar descriptivas, debido a que generalmente cualquier documento repite cierto vocabulario para desarrollar el tema del que trata. El usar SFM's por documento se aplica bien dentro del contexto RI, pues se necesita encontrar aquellos documentos que se relacionen a determinada consulta, y una manera de identificar esos documentos es usando las SFM's por documento que se repiten en cada documento como mínimo cierta cantidad de veces.

Es importante señalar que se optó por las SFM's por documento, en lugar de las SFM's por colección debido a que estas SFM's describen más lo que trata una colección de documentos, y no específicamente de lo que trata cada uno de los documentos que conforman dicha colección.

Para extraer las SFM's por documento se utiliza el algoritmo de García [9] con $\beta=2$, que toma como entrada uno o más documentos en el siguiente formato[4]:

ND = palabra1,palabra2,…,palabraN

donde

ND es el número de documento
N es el número total de palabras que tiene el documento.

La salida del algoritmo de García es dada de la siguiente manera:

{Numero de documento}
SFM's of size=[Tamaño de la SFM's]
[Frecuencia en el documento] SFM

Tomando como entrada la colección de 3 documentos pre-procesados de la figura 3.2.b, el algoritmo de García da como salida las SFM's por documento que se muestra en la figura 3.3.

[4] Se eligió el umbral β=2, porque es con el que se produce el mayor número de SFM's para representar a los documentos reduciendo la cantidad de documentos sin representar por alguna SFM.

Salida de las SFM's utilizando el algoritmo de García

```
{1}=========================================
SFM's of size=[1]----------------------------------------
[3] IBM
[2] SYSTEMS
SFM's of size=[2]----------------------------------------
[2]MARKET,SECURE

{2}=========================================
SFM's of size=[1]----------------------------------------
[2] IBM
[2] RECORD
SFM's of size=[2]----------------------------------------
[2]MARKET,SECURE

{3}=========================================
SFM's of size=[1]----------------------------------------
[2] SYSTEMS
```

Figura 3.3. Extracción de las SFM's por documento para la colección de 3 documentos de la figura 3.2.

En la tabla 3.1 se muestran las SFM's por documento para el ejemplo, junto con su frecuencia en cada documento.

DOCUMENTO	SFM	FRECUENCIA
1	IBM	3
1	SYSTEMS	2
1	MARKET,SECURE	2
2	IBM	2
2	RECORD	2
2	MARKET,SECURE	2
3	SYSTEMS	2

Tabla 3.1. SFM's por documento, con su respectiva frecuencia en cada documento.

3.1.2.2 Generación del índice

Una vez extraídos los términos índice (las SFM's por documento) de los documentos de la colección, se construye el índice del sistema que está

basado en la técnica de archivo invertido y el cual tiene la siguiente estructura:

$$NSFM \mid SFM \mid DT \mid FT \mid DOC_1 \mid FD_1 \mid FDN_1 \mid IDF_1 \mid FI_1 \mid \ldots \mid DOC_{DT} \mid FD_{DT} \mid FDN_{DT} \mid IDF_{DT} \mid FI_{DT}$$

donde:

NSFM= Número de SFM.

SFM= Secuencia Frecuente Maximal.

DT= Documentos totales que contienen a la SFM.

FT = Frecuencia de la SFM en todos los documentos.

DOC_j=Número de documento j donde aparece la SFM.

FD_j=Frecuencia en el documento j de la SFM.

FDN_j= FD_j normalizada.

IDF_j= Frecuencia Inversa en el documento j de la SFM.

FI_j= Producto de FDN_j*IDF_j.

Este índice contiene todas las SFM's por documento distintas encontradas en los documentos de una colección. Esta estructura permite tener toda la información necesaria para generar la representación vectorial de los documentos en la colección y de la consulta, como se explica en la siguiente sección. Además de que permite acelerar el proceso de búsqueda de documentos.

Para ejemplificar la estructura del índice se toman las SFM's por documento de la colección de 3 documentos que se muestra en la figura 3.3 y se construye el índice que se muestra en la tabla 3.2, en la cual se observa que la SFM "IBM", aparece en dos documentos (DT=2), los cuales son DOC 1 y DOC 2. En el caso de la SFM "RECORD" solamente aparece en un documento (DT=1), DOC 2, por lo que únicamente se obtienen los datos

relacionados a ese documento, a diferencia de las otras SFM's, que aparecen en dos documentos.

NSFM	SFM	DT	FT	DOC	FD	FDN	IDF	FI	DOC	FD	FDN	IDF	FI
1	IBM	2	5	1	3	1	0.18	0.18	2	2	1	0.18	0.18
2	MARKET,SECURE	2	4	1	2	0.67	0.18	0.12	2	2	1	0.18	0.18
3	RECORD	1	2	2	2	1	0.48	0.48					
4	SYSTEMS	2	4	1	2	0.67	0.18	0.12	3	2	1	0.18	0.18

Tabla 3.2. Índice de SFM's por documento de una colección de tres documentos.

3.1.3 Representación de los documentos y la consulta

Una vez extraídos los términos representativos de cada documento y generado el índice, se decidió utilizar el modelo vectorial para representar a los documentos y a la consulta.

3.1.3.1 Representación de los documentos

Para construir la representación vectorial de los documentos de la colección se utiliza el índice de SFM's por documento a partir del cual se genera un vector para cada documento de la colección, donde cada elemento del vector está asociado a una SFM encontrada en el índice. De esta manera cada documento es representado por un vector de tamaño NT, donde NT es el número de SFM's distintas encontradas en el índice. Por lo tanto, la colección de documentos es representada por una matriz NTxND donde ND es el número de documentos en la colección como se ve en la figura 3.4.

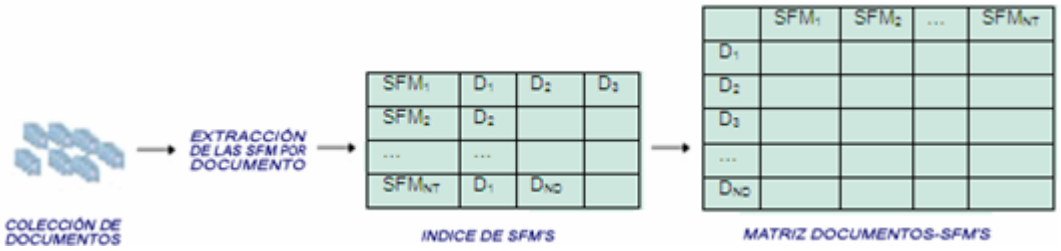

Figura 3.4. Matriz de documentos-SFM's generada a partir del índice de SFM por documento.

El pesado de cada elemento en los vectores de los documentos se realizó utilizando un pesado Booleano ó un pesado tf-idf, i.e. si la SFM se encuentra en el documento se coloca un 1, ó su pesado tf-idf, en el elemento del vector correspondiente a la SFM, en caso contrario un 0.

Para el índice de la tabla 3.2, la matriz de documentos-SFM's que se obtiene utilizando un pesado Booleano se muestra en la tabla 3.3. Usando un pesado tf-idf los valores de 1 cambian a un valor entre 0 y 1.

DOCUMENTO	IBM	MARKET,SECURE	RECORD	SYSTEMS
1	1	1	0	1
2	1	1	1	0
3	0	0	0	1

Tabla 3.3. Matriz de documentos-SFM's resultante del índice de la figura 3.2.

3.1.3.2 Representación de la consulta

La representación de la consulta en este método de RI es realizada mediante dos enfoques, el primero basado en un pesado Booleano y el segundo basado en la intersección entre la consulta y las SFM's. El pesado

Booleano es un pesado simple que no toma en cuenta el grado de parecido de la SFM con la consulta. Por esto se planteó usar un pesado basado en la intersección de la consulta y las SFM's de los documentos de la colección, obteniendo un grado de similitud entre cada documento de la colección y la consulta.

3.1.3.2.1 Representación de la consulta usando un pesado Booleano.

Para representar a la consulta mediante un vector, que es un conjunto de palabras, se compara cada SFM del índice con las palabras de la consulta, si existe alguna palabra de la consulta en la SFM se coloca un 1 en el elemento del vector correspondiente a esa SFM. Si no existe ninguna palabra que concuerde se coloca un 0. De esta manera se obtiene una consulta de tamaño igual al número de términos en el índice, es decir, el tamaño del vector para la consulta es del mismo tamaño que el de los vectores de documentos.

Por ejemplo para la consulta "THE SYSTEMS IBM ARE SECURE", después de realizar el pre-procesamiento se tendría "SYSTEMS,IBM,SECURE", con lo cual se genera el vector consulta que se muestra en la tabla 3.4.

CONSULTA	IBM	MARKET,SECURE	RECORD	SYSTEMS
SYSTEMS,IBM,SECURE	1	1	0	1

Tabla 3.4. Vector generado para la consulta "THE SYSTEMS IBM ARE SECURE con pesado Booleano.

3.1.3.2.2 Representación de la consulta usando un pesado de intersección.

Para representar la consulta, que es un conjunto de palabras, mediante un vector de consulta denominado *VC*, se realiza lo siguiente:

Para cada SFM-i del índice:

$$VC[i] = \frac{|SFM_i \cap CONSULTA|}{|SFM_i|}$$

Esto significa que cada SFM del índice se intersecta con la consulta, obteniendo un valor que indica cuantas palabras coinciden, dividiendo este valor entre el número de palabras que contiene la SFM para su normalización. El valor resultante se asigna al elemento *i* del vector consulta.

Una vez analizadas todas las SFM's del índice se obtiene el vector consulta de tamaño igual a los vectores de los documentos.

Para la consulta "THE SYSTEMS IBM ARE SECURE" después de realizar el pre-procesamiento se tendría "SYSTEMS,IBM,SECURE" con lo cual, siguiendo el procedimiento descrito, se genera el vector de la tabla 3.5. Los elementos del vector van ordenados de acuerdo al índice de la tabla 3.2.

CONSULTA	1 IBM	2 MARKET,SECURE	3 RECORD	4 SYSTEMS
SYSTEMS,IBM,SECURE	1/1=1	½=0.5	0/1=0	1/1=1

Tabla 3.5. Vector generado para la consulta "THE SYSTEMS IBM ARE SECURE con pesado basado en intersección.

Por ejemplo, para obtener el valor del elemento “MARKET,SECURE” que se muestra en la consulta de la tabla 3.5. Se tiene que esta secuencia se intersecta en una sola palabra con la consulta, y como la secuencia tiene longitud dos, entonces el valor obtenido es ½ = 0.5.

3.1.4 Búsqueda y recuperación de documentos.

La búsqueda y recuperación de los documentos en este trabajo está basada en el modelo vectorial. Utilizando alguno de los enfoques de pesado (Booleano ó tf-idf) para la representación vectorial de los documentos y algún enfoque de pesado (Booleano o intersección) para la consulta, se siguen los siguientes pasos teniendo como entrada un vector consulta (los cuales se ejemplifican teniendo un pesado Booleano tanto en la representación vectorial de los documentos (tabla 3.3) como en la consulta (tabla 3.4)):

1. Para cada elemento del vector consulta que tenga un valor distinto de 0, se busca la SFM correspondiente a ese elemento en el índice de SFM’s (por documento) para extraer los documentos relacionados a cada término en el índice, generando un listado de documentos *LD*. Para la consulta de la tabla 3.4 el listado *LD* que resulta es: (1, 3, 1, 2, 1, 2).

2. Este listado *LD* puede contener documentos repetidos por lo cual se realiza una depuración que consiste en lo siguiente:

I. Se extraen todos los documentos distintos que estén en *LD* y se guardan en *LDR*, junto con el número de veces que se repite cada documento en *LD*, de esta manera se obtienen parejas

ordenadas *(D,C)* donde *D* es el documento y *C* es el número de veces que se repite este documento en *LD*. Del listado *LD* (1, 3, 1, 2, 1, 2) se obtiene el listado *LDR* = [(1,3), (3,1), (2,2)].

II. Se ordenan los documentos en *LDR* descendentemente, de acuerdo al número de veces que se repite cada documento en *LD*, para obtener una lista ordenada de documentos distintos. Al ordenar *LDR* en el ejemplo se tiene: [(1,3), (2,2), (3,1)].

3. Una vez encontrados los documentos que se relacionan a la consulta, para cada documento almacenado en *LDR* se realiza lo siguiente:

I. Se busca en la matriz de documentos-SFM's, el vector correspondiente al documento.
II. Para el vector documento obtenido junto con el vector consulta se aplica la función de similitud coseno, con lo que se obtiene el parecido del documento con la consulta y se almacena en un listado de similitud *VS* como par ordenado (*D, P*) donde *D* es el documento y *P* es el parecido.

Para el listado *LDR* del ejemplo, se obtienen los vectores de los documentos 1, 2 y 3, con lo que al aplicar la función de similitud coseno se genera el listado *VS* [(1,3), (2,2), (3,1)].

4. Se ordena *VS* descendentemente con respecto al parecido (*P*), para generar el listado de documentos respuesta resultante. En este caso, al ordenar el listado *VS* queda sin cambio, es decir, [(1,3), (2,2), (3,1)], pues ya está ordenado.

Este proceso de búsqueda y recuperación obtiene diferentes resultados dependiendo de la forma de pesado elegida para representar a los documentos y la consulta.

3.2 Método MSFM2_PAL

MSFM2_PAL utiliza para representar el contenido de los documentos, las palabras de las SFM's por documento y está basado en el modelo vectorial utilizando dos formas de pesado, el pesado Booleano y el pesado tf-idf. Esta idea surge para resolver la cuestión de qué sucedería si se usaran sólo las palabras contenidas en las SFM's. Cabe señalar que el uso de las palabras de las SFM's puede comportarse de la misma manera que si sólo se utilizan las palabras que se repiten como mínimo un cierto número de veces en un documento.

El esquema general que se usó para la creación de este método de recuperación se muestra en la figura 3.2.1. Esencialmente se basa en la representación de los documentos obteniendo las SFM's (por documento) de los documentos de una colección y posteriormente extrayendo las palabras que contienen las SFM's, usándolas para construir los vectores de cada documento. El tamaño de cada vector es igual al número de palabras distintas encontradas en las SFM's por documento. Cada elemento de los vectores contiene un peso binario ó un peso basado en tf-idf, relacionado a si existe o no la SFM en cada documento. Luego, basándose en esta representación de los documentos se genera una representación similar de cada consulta, utilizando un pesado Booleano ó tf-idf como se explica en la sección 3.1.3.2, para realizar las búsquedas de los documentos relevantes a cada consulta.

Figura 3.2.1. Esquema general del método de Recuperación de Información basado en las palabras de las SFM por documento.

Primeramente se realizó un pre-procesamiento de los documentos de la colección y las consultas, como se explica en la sección 3.1.1. Posterior a esto se realizó la tarea de indexación.

3.2.1. Indexación para MSFM2_PAL

En este método de RI se utilizan dos índices, un índice de SFM's y un índice de palabras. El primero utilizado para encontrar los documentos relacionados a cada SFM y el segundo para encontrar las SFM's relacionadas a cada palabra. Para generar estos dos índices se obtienen dos tipos de términos índice, que se explican a continuación.

3.2.1.1 Extracción de términos índice

En MSFM2-PAL, para representar a los documentos se utilizan las SFM's por documento como términos índice obtenidos como se explica en la sección 3.1.2. Luego a partir de las SFM's por documento obtenidas se extraen las palabras que aparecen en las SFM's por documento.

En este enfoque se definen dos tipos de índice, utilizando las SFM's por documento, en un índice, y las palabras de las SFM's en otro índice.

Tomando las SFM's por documento que se muestran en la figura 3.3 se obtienen las palabras que serán utilizadas como términos índice como se muestra en la tabla 3.6, donde también se visualiza la SFM a la que pertenece y su frecuencia.

SFM	PALABRA	FRECUENCIA
IBM	IBM	1
SYSTEMS	SYSTEMS	1
MARKET,SECURE	MARKET	1
MARKET,SECURE	SECURE	1
RECORD	RECORD	1

Tabla 3.6. SFM's por documentos, con su respectiva frecuencia en cada documento.

En la sección siguiente se explica a detalle, cómo se usan estos términos índice.

3.2.1.2 Generación de índices

En este método de RI basado en las palabras de las SFM's, se utilizan dos tipos de términos índice, las SFM's por documento y las palabras de las SFM's.

El índice de SFM's se genera como se explica en la sección 3.1.2.2.

El índice de SFM's se utiliza para generar el índice utilizando las palabras de las SFM's aplicando la técnica de archivo invertido y el cual tiene la estructura siguiente:

$NPAL \mid PAL \mid ST \mid FT \mid SFM_1 \mid FS_1 \mid FSN_1 \mid IDF_1 \mid FI_1 \mid \ldots \mid SFM_{ST} \mid FS_{ST} \mid FSN_{ST} \mid IDF_{ST} \mid FI_{ST}$

donde:

NPAL= Número de palabra.

PAL= Palabra

ST= SFM's totales que contienen a la palabra.

FT = Frecuencia de la palabra en todas las SFM.

SFM_j=Número de SFM j donde aparece la palabra.

FS_j=Frecuencia en la SFM j de la palabra

FSN_j= FS_j normalizada.

IDF_j= Frecuencia Inversa en el documento j de la palabra

FI_j= Producto de FDN_j*IDF_j.

Este índice contiene todas las palabras distintas encontradas en las SFM's de la colección. Para ejemplificar la estructura del índice se toma el índice con las SFM's por documento de la tabla 3.2, para generar el índice de palabras de la tabla 3.7.

NPAL	PAL	ST	FT	SFM	FS	FSN	IDF	FI
1	IBM	1	1	**1**	1	1	0.60	0.60
2	MARKET	1	1	**2**	1	1	0.60	0.60
3	RECORD	1	1	**3**	1	1	0.60	0.60
4	SECURE	1	1	**2**	1	1	0.60	0.60
5	SYSTEMS	1	1	**4**	1	1	0.60	0.60

Tabla 3.7. Índice de palabras de las SFM's por documento para el ejemplo de la tabla 3.2.

En la figura 3.2.2 se visualiza cómo se generan los dos tipos de índice, así como la matriz que se genera a partir del índice de palabras. El índice de SFM's sirve para encontrar los documentos que se relacionan a una SFM. El índice de palabras sirve para encontrar las SFM's que se relacionan a cada palabra.

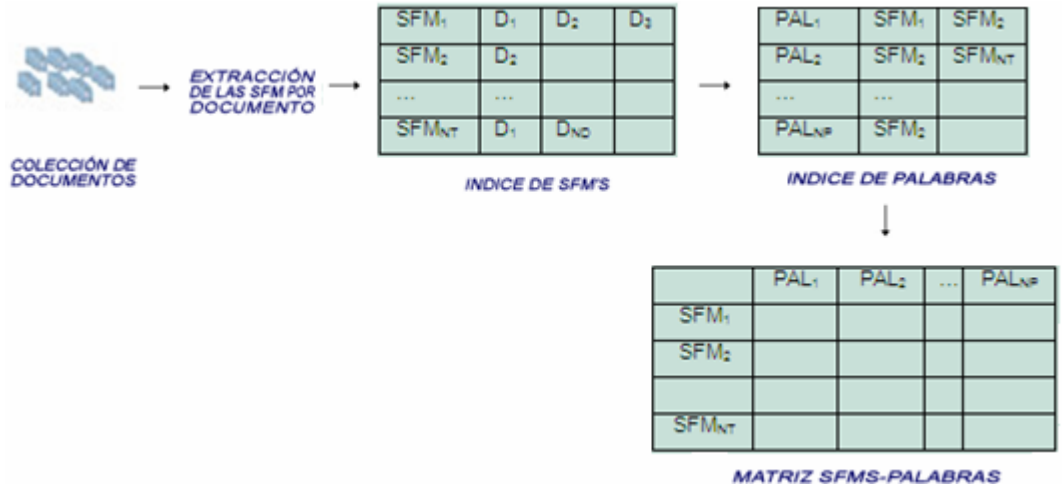

Figura 3.2.2. Matriz de SFM's-PALABRAS generada a partir del índice de palabras.

3.2.2 Representación de los documentos y la consulta

La representación de los documentos y las consultas en el método de RI basado en las palabras de las SFM's, MSFM2_PAL, se basó en el modelo vectorial. El proceso para llevar a cabo esta representación se describe a continuación.

3.2.2.1 Representación de los documentos

Los documentos en este método son representados con el índice de SFM's por documento, como se ejemplifica en la tabla 3.2, donde a cada SFM le corresponde uno o más documentos. Este índice sirve para buscar la SFM que está relacionada a la consulta y obtener los documentos que contienen a esa SFM.

Por otro lado, para poder comparar con la consulta, se realiza la representación vectorial de las SFM's por documento de la colección. Cada SFM se representa por un vector, en el cual cada elemento del vector es asociado al término correspondiente encontrado en el índice de palabras. De esta manera cada SFM es representada con un vector de tamaño *NP*, donde *NP* es el número de palabras distintas encontradas en el índice de palabras. Por lo tanto el conjunto de SFM's es representado por una matriz *NPxNT*, donde *NT* es el número de SFM's en el índice de SFM's como se ve en la figura 3.2.2.

El pesado de cada término en los vectores de las SFM's, para este método, se realizó utilizando dos enfoques, el pesado Booleano y el pesado tf-idf.

En el enfoque Booleano, si la palabra se encuentra en la SFM se coloca un 1 en el elemento del vector correspondiente a la palabra, en caso contrario un 0. Para el índice de la tabla 3.7 la matriz de SFM's-palabras que se obtiene es la que se muestra en la tabla 3.8.

SFM	IBM	MARKET	RECORD	SECURE	SYSTEMS
IBM	1	0	0	0	0
MARKET,SECURE	0	1	0	1	0
RECORD	0	0	1	0	0
SYSTEMS	0	0	0	0	1

Tabla 3.8. Matriz de SFM's-palabras resultante del índice de palabras de la tabla 3.7.

En el enfoque tf-idf, si la palabra del índice de palabras se encuentra en la SFM se obtiene su tf-idf con respecto a la SFM. Una vez obtenido esté valor se coloca en el elemento del vector correspondiente a esa palabra.

Para el índice de la tabla 3.7 la matriz de SFM's-palabras que se obtiene es la que se muestra en la tabla 3.9.

SFM	IBM	MARKET	RECORD	SECURE	SYSTEMS
IBM	0.60	0	0	0	0
MARKET,SECURE	0	0.60	0	0.60	0
RECORD	0	0	0.60	0	0
SYSTEMS	0	0	0	0	0.60

Tabla 3.9. Matriz de SFM's-palabras con pesado tf-idf resultante del índice de palabras de la tabla 3.7.

3.2.2.2 Representación de la consulta

Para representar la consulta, que es un conjunto de palabras, mediante un vector, también se utilizan dos enfoques en el pesado, el enfoque Booleano y el enfoque tf-idf.

En el enfoque Booleano para representar a la consulta se realiza lo siguiente:

1. Se genera un vector del tamaño de términos que tenga el índice de palabras (NP).
2. Se busca cada palabra de la consulta en el índice de palabras, si la palabra existe se coloca un 1 en el elemento correspondiente a esa palabra en el vector consulta, en otro caso un 0.

Una vez realizado esto se tendrá el vector consulta.

Por ejemplo para la consulta “THE SYSTEMS IBM ARE SECURE”, aplicando un pesado Booleano se genera el vector consulta que se muestra en la tabla 3.10, que es de tamaño igual al número de palabras en el índice de la tabla 3.7.

CONSULTA	IBM	MARKET	RECORD	SECURE	SYSTEMS
SYSTEMS,IBM,SECURE	1	0	0	1	1

Tabla 3.10. Vector de tamaño NP, generado para la consulta “THE SYSTEMS IBM ARE SECURE con pesado Booleano.

En el enfoque tf-idf para representar a la consulta se realiza lo siguiente:

1. Se genera un vector del tamaño de términos que tenga el índice de palabras, rellenado con 0.
2. Se busca cada palabra de la consulta en el índice de palabras, si la palabra existe se obtiene su idf del índice y se calcula su tf con respecto a la consulta. Una vez obtenidos el tf y el idf se multiplican y se coloca el valor resultante en el elemento correspondiente a esa palabra en el vector consulta.

Una vez realizado esto se tendrá el vector consulta.

Para la consulta de ejemplo “THE SYSTEMS IBM ARE SECURE”, después de realizar el pre-procesamiento se tendría “SYSTEMS,IBM,SECURE”, con lo cual se genera el vector consulta que se muestra en la tabla 3.11.

CONSULTA	IBM	MARKET	RECORD	SECURE	SYSTEMS
SYSTEMS,IBM,SECURE	0.06	0	0	0.06	0.06

Tabla 3.11. Vector de tamaño NP con pesado tf-idf, generado para la consulta “THE SYSTEMS IBM ARE SECURE.

3.2.3. Búsqueda y recuperación de documentos.

La búsqueda y recuperación de los documentos en el método de RI usando las palabras de las SFM’s, se basó en el modelo vectorial.

Utilizando alguno de los enfoques de pesado (Booleano ó tf-idf), para la representación vectorial de los documentos y la consulta, se siguen los siguientes pasos, teniendo como entrada un vector consulta [el cual se ejemplifica teniendo un pesado Booleano tanto en la representación vectorial

de las SFM's (tabla 3.8) como en la consulta (tabla 3.10), y usando el índice de SFM's por documento de la tabla 3.2)]:

1. Para cada elemento del vector consulta que tenga un valor distinto de 0, se busca la palabra correspondiente a ese término en el índice de palabras para extraer las SFM's relacionadas a cada término generando un listado *LS*. Por ejemplo, para la consulta de la tabla 3.10 el listado LS que resulta es (1, 2, 4).

2. Este listado *LS* puede contener SFM's repetidas por lo cual se realiza una depuración que consiste en lo siguiente:

I. Extraer todas las SFM's distintas que estén en *LS* y guardarlas en otra lista *LSR,* junto con el número de veces que se repite cada documento en *LS,* de esta manera se obtienen parejas ordenadas (*S,C*) donde *S* es la SFM y *C* es el número de veces que se repite esa SFM en *LS*. Del listado *LS* (1, 2, 4) de la consulta de ejemplo se obtiene LSR = [(1,1), (2,1), (4,1)].

II. Se ordenan las SFM's descendentemente, de acuerdo al número de veces que se repite cada SFM's en *LS*, para obtener una lista ordenada de SFM's distintas en *LSR*. Al ordenar *LSR* del ejemplo se tiene el vector [(1,1), (2,1), (4,1)].

3. Una vez encontradas las SFM's que se relacionan a la consulta, se realiza para cada documento almacenado en *LSR* lo siguiente:

I. Se busca en la matriz de SFM's-palabras, el vector correspondiente a la SFM.

II. Para el vector de la SFM obtenido junto con el vector consulta se aplica la función de similitud coseno, con lo que se obtiene el parecido de la SFM con la consulta y se almacena en el listado *VRS* como par ordenado (*S, P*) donde *S* es la SFM y *P* es el parecido.

Para el listado *LSR* del ejemplo, se obtienen los vectores de las SFM's 1, 2 y 4 correspondientes a las SFM's "IBM","MARKET SECURE" y "SYSTEMS" con lo que al aplicar la función de similitud coseno se genera el listado VRS = [(1,1), (2,1), (4,1)].

4. El conjunto de pares ordenados *VRS* se ordena descendentemente con respecto al parecido (*P*), para generar el listado de SFM's resultante. El listado *VRS* ordenado se mantiene sin cambio, es decir [(1,1), (2,1), (4,1)], ya que el parecido en este ejemplo es igual en todos los casos.

5. Utilizando la lista *VRS* ordenada que contiene las SFM's relacionadas a la consulta, se busca cada una de éstas en el índice de SFM's para extraer los documentos en los que se encuentra cada una de ellas, almacenándolos en la lista de documentos *DRS*. Por ejemplo, Para el listado *VRS* [(1,1), (2,1), (4,1)] se obtendrían los documentos (1, 2, 1, 2, 1, 3) en la lista *DRS*.

6. Este listado *DRS* puede contener documentos repetidos, por lo cual se realiza una depuración que consiste en lo siguiente:

I. Se extraen todos los documentos distintos que estén en *DRS* y se guardan en otra lista de documentos *DFRS*, junto con el número de veces que se repite cada documento en *DRS*, de esta manera se tienen parejas ordenadas (*D,C*) donde *D* es el número de documento y *C* es el número de veces que se repite ese documento en *DRS*. Del listado *DRS* (1, 2, 1, 2, 1, 3) de la consulta de ejemplo, se obtiene *DFRS* = [(1,3), (2,2), (3,1)].

II. Se ordenan los documentos descendentemente en *DFRS* de acuerdo a *C,* para obtener una lista ordenada de documentos distintos en *DFRS*. Al ordenar *DFRS* del ejemplo se tiene el vector [(1,3), (2,2), (3,1)], con lo cual se obtienen los documentos 1, 2, 3 como respuesta a la consulta.

3.3 Método MSFM3

El método de RI MSFM3, permite usar como consulta un documento completo y buscar los documentos relacionados al documento-consulta. Este método de RI utiliza para representar los documentos y la consulta a las SFM's por documento y se basa en el modelo vectorial utilizando un pesado tf-idf o Booleano. Un aspecto importante al permitir que la consulta sea un documento es que éste se puede representar mediante las SFM's por documento que contiene, y son estas SFM's las que se convierten en la consulta sustituyendo a las palabras de una consulta tradicional.

El pre-procesamiento de los documentos, la extracción de los términos índice y la generación del índice, se realizan como se explicó en las

secciones 3.1.1, 3.1.2.1 y 3.1.2.2 respectivamente. El esquema general que se sigue en este método de recuperación se muestra en la figura 3.3.1.

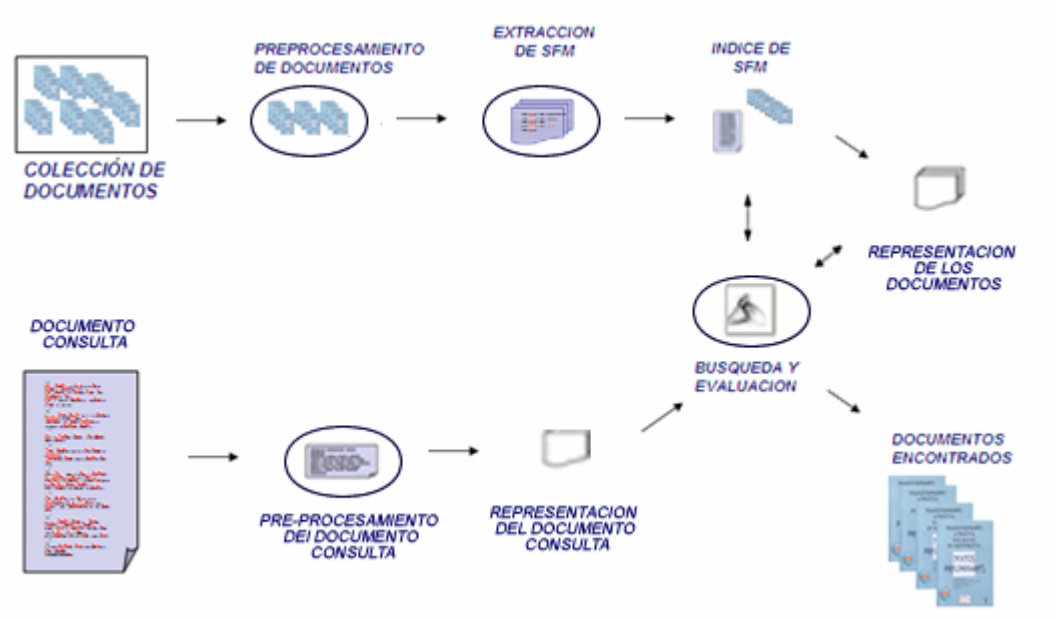

Figura 3.3.1. Esquema general del método de Recuperación de Información basado en las SFM por documento donde la consulta es un documento.

3.3.1 Representación de los documentos y la consulta

La representación de los documentos y las consultas en el método de RI donde la consulta es un documento, se basó en el modelo vectorial.

3.3.1.1. Representación de los documentos

La representación de los documentos se realiza como se explica en la sección 3.1.3.1, excepto en la forma como se obtiene el pesado que en este caso se utiliza un pesado Booleano o tf-idf.

Para realizar la representación vectorial de los documentos de la colección se utiliza el índice de SFM's por documento a partir del cual se genera para cada documento de la colección un vector, donde cada elemento del vector está asociado a una SFM encontrada en el índice. De esta manera cada documento es representado por un vector de tamaño *NT*, donde *NT* es el número de SFM's distintas encontradas en el índice de SFM's. Por lo tanto la colección de documentos se representa por una matriz *NTxND*, donde *ND* es el número de documentos en la colección, como se observa en la figura 3.3.2.

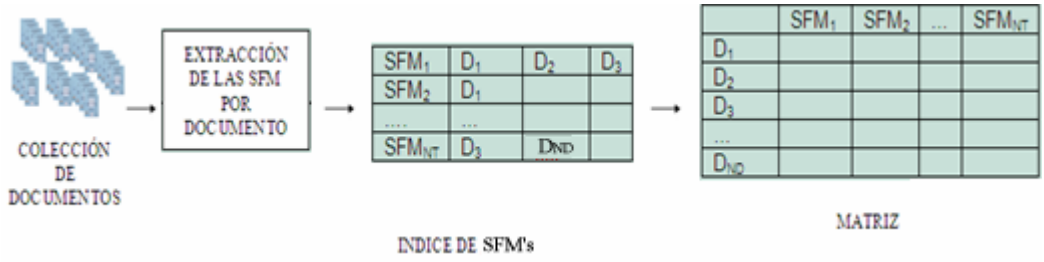

Figura 3.3.2. Matriz de documentos-SFM generada a partir del índice de SFM por documento.

Para calcular el pesado de cada uno de los elementos de un vector de documentos se realizó utilizando dos enfoques de pesado: tf-idf y Booleano.

En el enfoque tf-idf, si la SFM del índice se encuentra en el documento se calcula su tf-idf, y se coloca el valor resultante en el elemento del vector correspondiente a la SFM.

Para el índice de la tabla 3.2 la matriz de documentos-SFM's que se obtiene se muestra en la tabla 3.3.1.

DOCUMENTO	IBM	MARKET,SECURE	RECORD	SYSTEMS
1	0.18	0.12	0	0.12
2	0.18	0.18	0.48	0
3	0	0	0	0.18

Tabla 3.3.1. Matriz de documentos-SFM's resultante del índice de la tabla 3.3.1.

En el enfoque Booleano, si la SFM del índice se encuentra en el documento se coloca un 1 en el elemento del vector correspondiente a la SFM, en caso contrario un 0.

Para el índice de la tabla 3.2, la matriz de documentos-SFM's que se obtiene se muestra en la tabla 3.3.2.

DOCUMENTO	IBM	MARKET,SECURE	RECORD	SYSTEMS
1	1	1	0	1
2	1	1	1	0
3	0	0	0	1

Tabla 3.3.2. Matriz de documentos-SFM's resultante del índice de la tabla 3.3.2.

3.3.1.2. Representación del documento consulta

Para representar al documento-consulta, que es un conjunto de SFM's por documento, extraídas del mismo documento, se usan también dos enfoques de pesado: tf-idf y Booleano.

En el enfoque tf-idf se compara cada SFM del índice con las SFM's de la consulta, si existe alguna SFM de la consulta en las SFM's del índice se calcula su tf-idf, y el valor resultante se coloca en el elemento del vector consulta correspondiente a esa SFM. De esta manera se tiene un vector consulta de tamaño igual a los vectores de los documentos.

Por ejemplo supóngase que se tienen el documento-consulta que se muestra en la figura 3.3.3, al cual se le aplicó el pre-procesamiento descrito en la sección 3.1.1 obteniendo las SFM's por documento que contiene, las cuales son las que se muestran en la tabla 3.3.3.

IBM systems information market secure help buy IBM, libraries, network memory market secure, center mouse record computer.

Figura 3.3.3. Documento consulta.

SFM	FRECUENCIA
IBM	2
MARKET, SECURE	2

Tabla 3.3.3. SFM's por documento del documento consulta de la figura 3.3.3.

A partir de estas SFM's del documento-consulta se genera el vector consulta que se muestra en la tabla 3.3.4.

DOCUMENTO	IBM	MARKET,SECURE	RECORD	SYSTEMS
CONSULTA	0.18	0.18	0	0

Tabla 3.3.4. Vector del documento consulta usando pesado tf-idf.

En el enfoque Booleano se compara cada SFM del índice con las SFM's de de la consulta, si existe alguna SFM de la consulta en las SFM's del índice se coloca un 1 en el elemento del vector consulta correspondiente a esa SFM, en caso contrario se coloca un 0. De esta manera se obtiene un vector consulta de tamaño igual a los vectores de los documentos.

Retomando el ejemplo de la figura 3.3.3 cuyas SFM's son las de la tabla 3.3.3 se obtiene el vector consulta de la tabla 3.3.5.

DOCUMENTO	IBM	MARKET,SECURE	RECORD	SYSTEMS
CONSULTA	1	1	0	0

Tabla 3.3.5. Vector del documento consulta usando pesado Booleano.

3.3.2 Búsqueda y recuperación de documentos.

La búsqueda y recuperación de los documentos en el método que utiliza un documento como consulta está basado en el modelo vectorial y se realiza de manera similar a como se explica en la sección 3.2.3.

Utilizando alguno de los enfoques de pesado (Booleano ó tf-idf), para la representación vectorial de los documentos y la consulta, se siguen los siguientes pasos, teniendo como entrada un vector consulta [el cual se ejemplifica teniendo un pesado Booleano tanto en la representación vectorial de la consulta (tabla 3.3.5) como de los documentos (tabla 3.3.2)]:

1. Para cada elemento del vector consulta que tenga un valor distinto de 0, se busca la SFM correspondiente a ese término en el índice de SFM's por documento para extraer los documentos relacionados a cada término generando un listado de documentos *LD*. Para la consulta de la tabla 3.3.3 el listado *LD* que resulta es (1, 2, 1, 2).

2. Este listado *LD* puede contener documentos repetidos por lo cual se realiza una depuración que consiste en lo siguiente:

I. Se extraen todos los documentos distintos que estén en *LD* y se guardan en *LDR*, junto con el número de veces que se repite cada documento en *LD*, de esta manera se obtienen las parejas ordenadas (*D,C*) donde *D* es el documento y *C* es el número de veces que se repite este documento en *LD*. Del listado *LD*= (1, 2, 1, 2) de la consulta de ejemplo, *LDR* = [(1,2), (2,2)].

II. Se ordenan los documentos descendentemente, de acuerdo al número de veces que se repite cada documento en *LD*, para obtener una lista ordenada de documentos distintos en *LDR.* Al ordenar *LDR* del ejemplo obtenemos [(1,2), (2,2)].

3. Una vez encontrados los documentos que se relacionan a la consulta, se realiza para cada documento almacenado en *LDR* lo siguiente:

I. Se busca en la matriz de documentos-SFM's, el vector correspondiente al documento.

II. Para el vector documento obtenido junto con el vector consulta se aplica la función de similitud coseno, con lo que se obtiene el parecido del documento con la consulta y se almacena en el listado *VR* como par ordenado (*D, P*) donde *D* es el documento y *P* es el parecido.

Para el listado *LDR* de ejemplo, se obtienen los vectores de los documentos 1, 2, con lo que al aplicar la función de similitud coseno se genera el listado *VR* = [(1,0.06202), (2, 0.06202)].

4. El conjunto de pares ordenados *VR* se ordena descendentemente con respecto al parecido (*P*), para generar el listado de documentos respuesta resultante. Para el listado *VR* del ejemplo al ordenar queda sin cambio, es decir: [(1,0.06202), (2, 0.06202)], con lo cual la respuesta a la consulta son los documentos 1 y 2.

Capítulo 4. Experimentación y resultados

En este capítulo se describen las colecciones utilizadas para la experimentación, así como los resultados obtenidos con los métodos de RI propuestos.

4.1. Descripción de las colecciones de documentos y sistema de comparación -LUCENE-

Las colecciones de documentos usadas para probar los métodos propuestos MSFM1 y MSFM2_PAL (descritos en las secciones 3.1 y 3.2), donde la consulta es un conjunto pequeño de palabras, fueron las colecciones ADI, MED y CRAN. Estas colecciones han sido usadas para evaluar métodos de RI a través de los años y proveen un buen conjunto de consultas para pruebas preliminares de los métodos de RI. En la tabla 4.1.1 se muestran algunas características de estas colecciones. Se eligieron dichas colecciones debido a que proporcionan la información de cuáles documentos son relevantes para cada consulta dada, lo cual es importante para calcular el recuerdo y la precisión.

COLECCIÓN	Tema	# Documentos	# Consultas	Tamaño en Mb
ADI	Información de Ciencia	82	35	.04
MED	Medicina	1033	30	1.1
CRAN	Aeronáutica	1400	225	1.6

Tabla 4.1.1. Colecciones de prueba para Recuperación de Información.

Para probar el método MSFM3 que acepta como consulta un documento, se creó una colección de documentos con temas relacionados a computación. Las características generales de la colección se muestran en la tabla 4.1.2.

COLECCIÓN	Tema	# Documentos	#Documentos Consultas	Tamaño en Mb
C33	Computación	33	8	4.5

Tabla 4.1.2. Colecciones de prueba para Recuperación de Información.

Los temas de la colección C33 se eligieron de manera arbitraria sólo con la condición de que no tuvieran una relación muy estrecha con los otros temas que se encontraban en la misma colección. Los temas de la colección se muestran en la tabla 4.1.3 así como la cantidad de documentos que contienen en la colección.

TEMA	#Documentos
3DMAX	2
AWK	2
CISCO	3
CMMI	4
FUZZY	8
OLAP	5
PATTERN RECOGNITION	1
PHP	3
RUP	5

Tabla 4.1.3. Temas de la colección C33.

Para la consulta se utilizaron ocho documentos-consulta ajenos a la colección, los cuales fueron seleccionados arbitrariamente, tomando en cuenta que trataran alguno de los temas de la colección, de tal manera que al obtener los documentos respuesta del método de RI resultara fácil reconocer si eran relevantes o no. De esta manera los temas de los que trataron los documentos-consulta se muestran en la tabla 4.1.4. Mostrando también el número de documentos relevantes para cada tema en la colección C33.

DOCUMENTO CONSULTA	TEMA DEL DOCUMENTO CONSULTA	# DOCUMENTOS RELEVANTES EN LA COLECCIÓN C33
1	AWK	2
2	OLAP	5
3	CMMI	4
4	RUP	5
5	CISCO	3
6	OLAP	5
7	FUZZY	8
8	AWK	2

Tabla 4.1.4. Temas de los documentos consulta para la colección C33.

Para realizar una comparación de los métodos propuestos, MSFM1 y MSFM2_PAL, se utilizó el método LUCENE, el cual es un método que usa una representación basada en palabras. LUCENE toma como base el modelo vectorial y aplica un pesado basado en tf-idf, razones por las cuales se decidió utilizar dicho método.

4.2 Extracción de las SFM's por documento de las colecciones

A cada una de las colecciones de las tablas 4.1.1 y 4.1.2 se aplicó un pre-procesamiento como se describe en la sección 3.1.1 en el cual se eliminaron las *stop-words* de los documentos que se muestran en el apéndice A. Se decidieron eliminar estas *stop-words*, con base en el análisis de 5 bases de datos de *stop-words* distintas incluidas algunas de éstas en colecciones para pruebas de RI, decidiendo eliminar todas las *stop-words* que aparecieran en las 5 bases. También existía la posibilidad de hacer una eliminación en base al idf de cada palabra, quitando las que tuvieran un idf muy bajo, pero se optó por la primera opción.

Una vez realizado el pre-procesamiento, para cada colección se extrajeron las SFM's por documento usando el algoritmo de García [9], con un umbral de 2 para representar a los documentos. Se eligió el umbral β=2, porque es con el que se produce el mayor número de SFM's para representar a los documentos reduciendo la cantidad de documentos sin representar por alguna SFM. Además, se obtuvieron las palabras que forman las SFM's para ser utilizadas en el método. El número de SFM's por documento y el número de palabras de las SFM's para cada colección se muestran en la tabla 4.2.1.

La cantidad de SFM's y el número de palabras pertenecientes a las SFM's es mucho menor que la cantidad de palabras distintas encontradas en cada colección. Esto provoca que la representación vectorial de los documentos sea más pequeña, lo cual ayuda para tener respuestas más rápidas en los métodos de RI.

COLECCIÓN	# SFM's por documento (umbral 2)	# Palabras de las SFM sin repetición	#Palabras sin repetición sin stop-words	#Palabras sin repetición con stop-words	#Palabras con repetición sin stop-words	#Palabras con repetición con stop-words
ADI	213	241	1323	1524	2893	5229
MED	5289	3941	12968	13377	83041	155379
CRAN	5844	2715	8725	9126	129894	245462
C33	6928	4587	11019	11490	75952	136360

Tabla 4.2.1. Cantidad de SFM's por documento y palabras de las SFM en las colecciones.

4.3 Resultados de los métodos de RI propuestos y análisis.

Para probar los métodos de RI propuestos, MSFM1 y MSFM2_PAL, donde la consulta es un conjunto pequeño de palabras, se procesaron todas las consultas de cada una de las colecciones de prueba. Para cada consulta, en cada colección, se generaron los valores de interpolación de precisión para 11 puntos estándar de recuerdo, 0%,10%,…100%, como se explica en la sección 2.2.2. Posteriormente, tomando todos los valores de precisión obtenidos en cada punto estándar de recuerdo, se obtuvo la precisión promedio en cada punto de recuerdo para todas las consultas en la colección, por ejemplo para el punto de recuerdo 0%, se promediaron las precisiones de todas las consultas, de una colección, en dicho punto y así sucesivamente hasta llegar al punto 100%. Estos valores muestran el comportamiento general del método de RI para cada colección.

Los métodos MSFM1 y MSFM2_PAL se probaron con diferentes enfoques en el pesado de los documentos y la consulta. La tabla 4.3.1 muestra el significado para cada enfoque.

VARIANTE	SIGNIFICADO
BINCONS_BINDOCS	Pesado Booleano en la consulta y en los documentos
BINCONS_TFIDFDOCS	Pesado Booleano en la consulta y pesado tf-idf en los documentos
BIN_SFMPORCOL_DOC	Pesado Booleano utilizando las SFM's por colección y por documentos
INTERCONS_BINDOCS	Pesado de Intersección en consulta y Booleano en documentos
INTERCONS_BINDOCS_PORCOL_DOC	Pesado de intersección utilizando las SFM's por colección y por documento
PAL_BINARIO	Pesado Booleano utilizando las palabras de las SFM
PAL_TFIDF	Pesado tf-idf utilizando las palabras de las SFM
TF-IDF	Pesado tf-idf en la consulta y en los documentos.
BINARIO	Pesado Booleano en la consulta y en los documentos.

Tabla 4.3.1. Variantes de pesado para los métodos propuestos y su significado.

Para los experimentos se utilizaron las colecciones de la tabla 4.1.1 y se aplicaron en su totalidad las consultas de cada colección para los métodos MSFM1 y MSFM2_PAL con sus respectivas variantes. Los resultados que se presentan muestran las precisiones promedio a 10 puntos estándar de recuerdo.

Para la colección ADI los resultados obtenidos al aplicar los métodos con sus respectivas variantes se muestran en la tabla 4.3.2 donde se observa claramente que el método que tuvo mejores valores de precisión promedio fue LUCENE. Ninguno de nuestros métodos pudo superarlo, siendo el método que mejor se comportó MSFM1_BIN_SFMPORCOL_DOC, aunque con un rendimiento similar al de MSFM1_INTERCONS_BINDOCS.

MÉTODOS CON SU VARIANTE	PRECISIONES DE CADA MÉTODO EN LOS VALORES ESTANDAR DE RECUERDO										
MSFM1_BINCONS_BINDOCS	44.19	43.62	36.44	34.25	25.28	20.05	14.75	5.21	4.37	3.33	3.33
MSFM1_BINCONS_TFIDFDOCS	45.36	44.77	38.79	36.07	29.54	21.48	12.26	5.89	4.53	3.44	3.44
MSFM1_BIN_SFMPORCOL_DOC	46.12	45.55	40.86	38.87	28.22	22.99	16.17	6.94	6.11	4.63	4.63
MSFM1_INTERCONS_BINDOCS	45.60	45.60	41.39	33.78	28.06	23.01	14.55	5.56	4.99	3.59	3.59
MSFM1_INTERCONS_BINDOCS_PORCOL_DOC	37.39	36.07	34.59	31.76	23.90	19.94	12.31	7.56	7.03	4.87	4.87
MSFM2_PAL_BINARIO	43.14	42.29	38.46	36.05	23.17	18.65	16.56	11.05	8.63	7.21	7.21
MSFM2_PAL_TFIDF	44.64	44.07	38.93	36.32	25.46	20.88	16.32	9.03	6.84	4.00	4.00
LUCENE	52.78	52.65	52.32	47.01	40.10	37.87	30.21	21.22	18.24	12.41	12.41
VALORES ESTÁNDAR DE RECUERDO	**0**	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**

Tabla 4.3.2. Resultados para la colección ADI utilizando los métodos MSFM1 Y MSFM2_PAL con sus variantes.

La figura 4.3.1 muestra la gráfica de los resultados de la tabla 4.3.2.

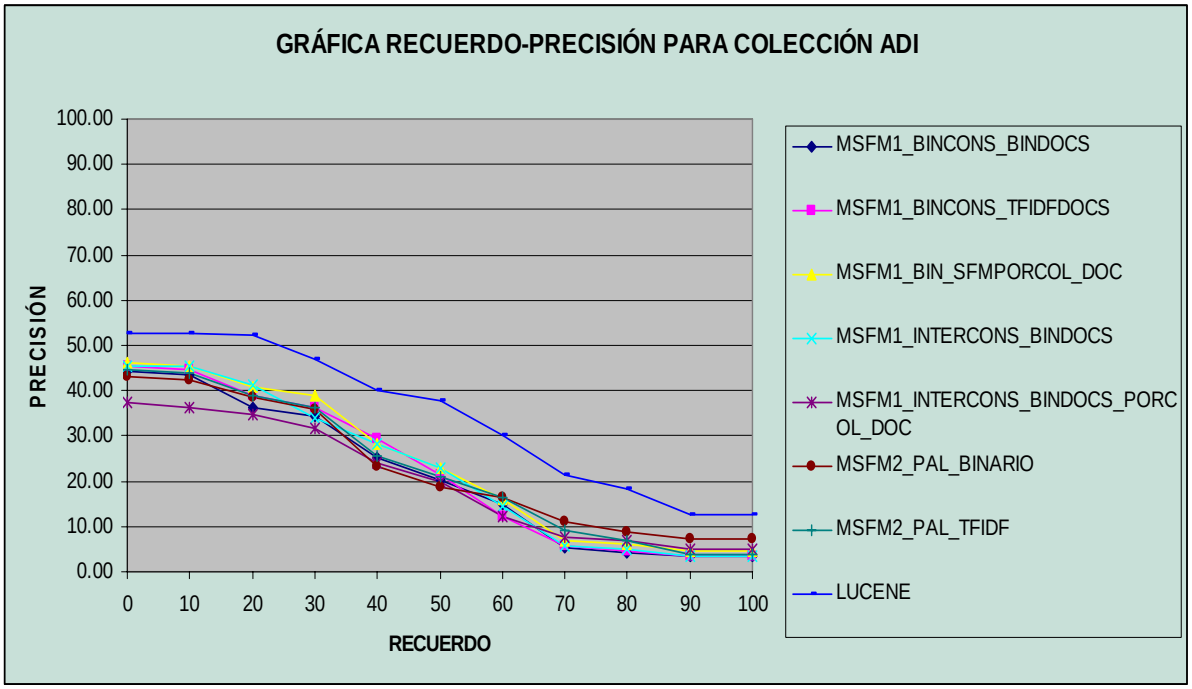

Figura 4.3.1. Gráfica recuerdo-precisión para la colección ADI que muestra los resultados de la tabla 4.3.2.

Se observa a partir de la figura 4.3.1 que en general los valores de precisión son bajos, inclusive con el método LUCENE que fue el más alto, esto probablemente es debido a que los documentos de la colección ADI son pequeños, por lo que se tenía un conjunto de términos índice más pequeño, lo cual afectaba a las consulta pues tenía menos palabras para hacer la comparación. Se observa que ninguna de las variantes de nuestros métodos tuvo un comportamiento similar al de LUCENE.

Para la colección MED los resultados obtenidos al aplicar los métodos con sus respectivas variantes se muestran en la tabla 4.3.3. Siendo LUCENE el que tuvo valores de precisión promedio mayores con respecto a los métodos propuestos utilizando SFM's por documento. En la figura 4.3.2 se muestran gráficamente estos resultados.

METODOS CON SU VARIANTE	PRECISIONES DE CADA MÉTODO EN LOS VALORES ESTANDAR DE RECUERDO										
MSFM1_BINCONS_BINDOCS	74.6	61.6	53.9	42	35.4	32.4	24.2	15.2	11.2	4.03	1.36
MSFM1_BINCONS_TFIDFDOCS	45.4	44.8	38.8	36.1	29.5	21.5	12.3	5.89	4.53	3.44	3.44
MSFM1_INTERCONS_BINDOCS	82.7	68.7	59.7	44	36.7	32.7	27.4	18.3	12	3.91	1.44
MSFM2_PAL_BINARIO	60.1	44.3	37.4	32.4	26.4	21.5	16.5	12.6	8.07	4.91	3.44
MSFM2_PAL_TFIDF	50.6	43.2	39.6	32.9	27.8	20.1	16.4	12.3	9.05	4.19	1.99
LUCENE	91.9	79.5	74.2	68.4	60.9	49.2	44.1	37.6	29.7	18.2	5.97
VALORES ESTANDAR DE RECUERDO	0	10	20	30	40	50	60	70	80	90	100

Tabla 4.3.3. Resultados para la colección MED utilizando los métodos MSFM1 Y MSFM2_PAL con sus variantes.

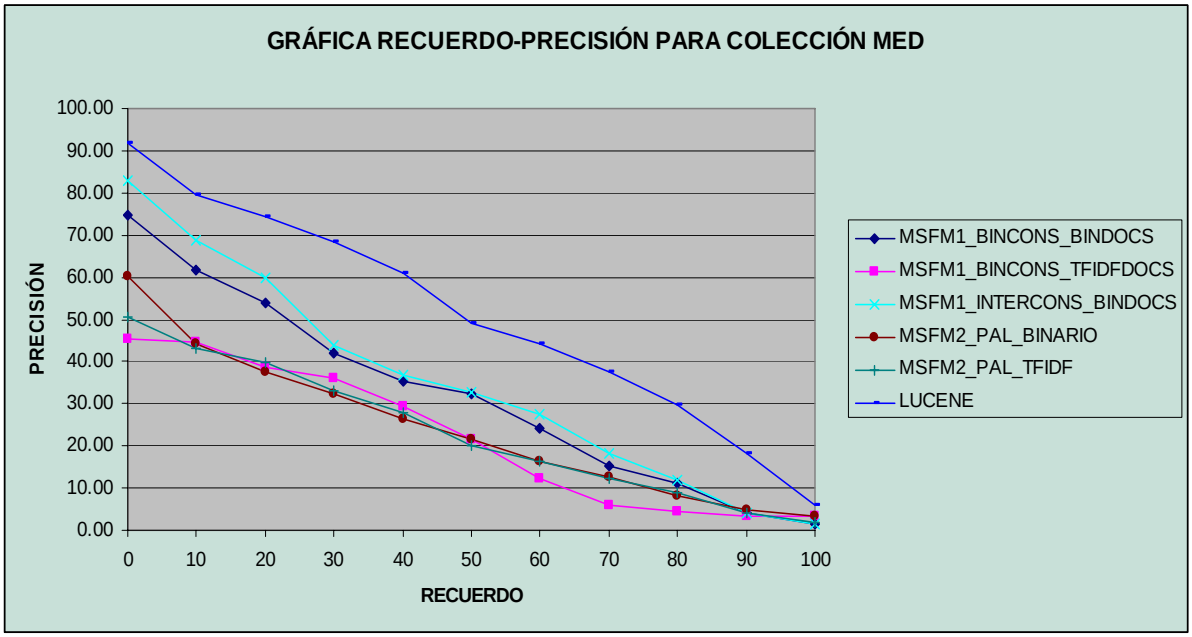

Figura 4.3.2. Gráfica recuerdo-precisión para la colección MED que muestra los resultados de la tabla 4.3.3.

Para la colección CRAN los resultados obtenidos al aplicar los métodos con sus respectivas variantes se muestran en la tabla 4.3.4. Nuevamente LUCENE es el método que tiene valores de precisión promedio mayores con respecto a los métodos propuestos. En la figura 4.3.3 se muestran gráficamente los resultados de la tabla 4.3.4.

METODOS CON SU VARIANTE	PRECISIONES DE CADA MÉTODO EN LOS VALORES ESTANDAR DE RECUERDO										
MSFM1_BINCONS_BINDOCS	33.5	30.4	23.4	17.7	12.6	10.5	8.49	5.65	4.39	2.96	2.24
MSFM1_BINCONS_TFIDFDOCS	26.9	24.1	18.2	14.5	9.91	8.22	6.76	4.79	3.94	2.55	1.88
MSFM1_INTERCONS_BINDOCS	42.1	37.5	26.5	19.5	13.1	10.4	7.82	5.38	4.35	2.7	2.13
MSFM2_PAL_BINARIO	27.1	25.4	20	14.7	11.2	9.49	7.12	4.51	3.42	2.24	1.56
MSFM2_PAL_TFIDF	28.2	25.2	17.4	13	9.4	8.14	6.08	4.08	3.24	2.16	1.76
LUCENE	83	78.2	72.3	54.1	44.8	40.1	26.9	15.5	7.63	4.21	3.49
VALORES ESTANDAR DE RECUERDO	**0**	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**

Tabla 4.3.4. Resultados para la colección CRAN utilizando los métodos MSFM1 Y MSFM2_PAL con sus variantes.

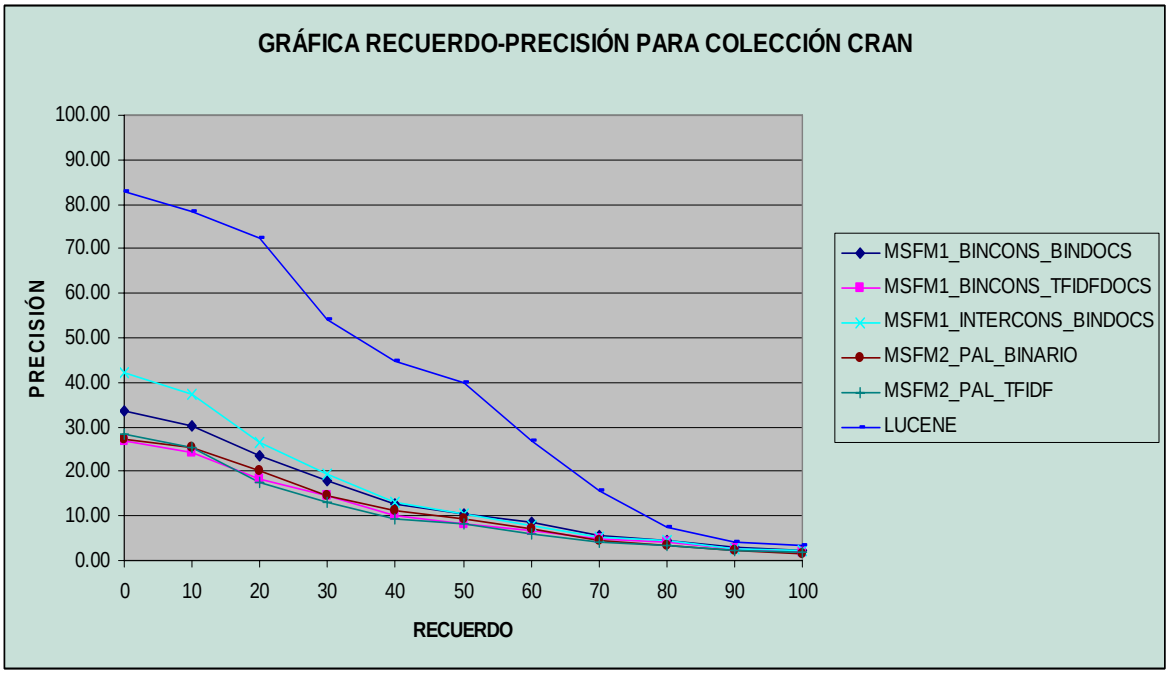

Figura 4.3.3. Gráfica recuerdo-precisión para la colección CRAN que muestra los resultados de la tabla 4.3.3.

Comparando las gráficas de las tres colecciones, ADI, MED y CRAN se observa que para los métodos propuestos, los mejores resultados se obtuvieron con la colección MED, esto fue debido a que los documentos de esta colección fueron un poco más grandes que los que contenían las otras colecciones. Por lo cual se generó un conjunto de términos índice más amplio para comparar las consultas.

A partir de las gráficas se deduce que con colecciones pequeñas los métodos de RI que utilizan una representación basada en palabras tienen un mejor rendimiento que los métodos utilizando SFM's por documento, sea cual sea su variante. Es importante señalar que con documentos de mayor tamaño, los resultados utilizando una representación con SFM's mejoran, como se mostró en la gráfica 4.3.2 donde los documentos de la colección tienen un tamaño mayor.

Para probar el método MSFM3 descrito en la sección 3.3, se usó la colección C33, esto con la finalidad de probar el comportamiento del método con documentos más grandes en los cuales se pudieran extraer más SFM's por documento. Además de que las consultas para este método también son documentos más grandes. En este caso los documentos tienen al menos una pagina de texto (alrededor de 300 palabras aproximadamente). Los resultados de este método de RI se muestran en la tabla 4.3.3, cuya gráfica resultante se presenta en la figura 4.3.4.

METODOS CON SU VARIANTE	PRECISIONES DE CADA MÉTODO EN LOS VALORES ESTANDAR DE RECUERDO										
MSFM3_TF-IDF	82.1	82.1	82.1	63	52.6	50	37	32	29	23	23
MSFM3_BINARIO	80.4	80.4	80.4	58.5	45.4	45.4	37	35	30	22	22
VALORES ESTANDAR DE RECUERDO	**0**	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**

Tabla 4.3.3. Resultados para la colección C33 utilizando los métodos MSFM1 Y MSFM2_PAL con sus variantes.

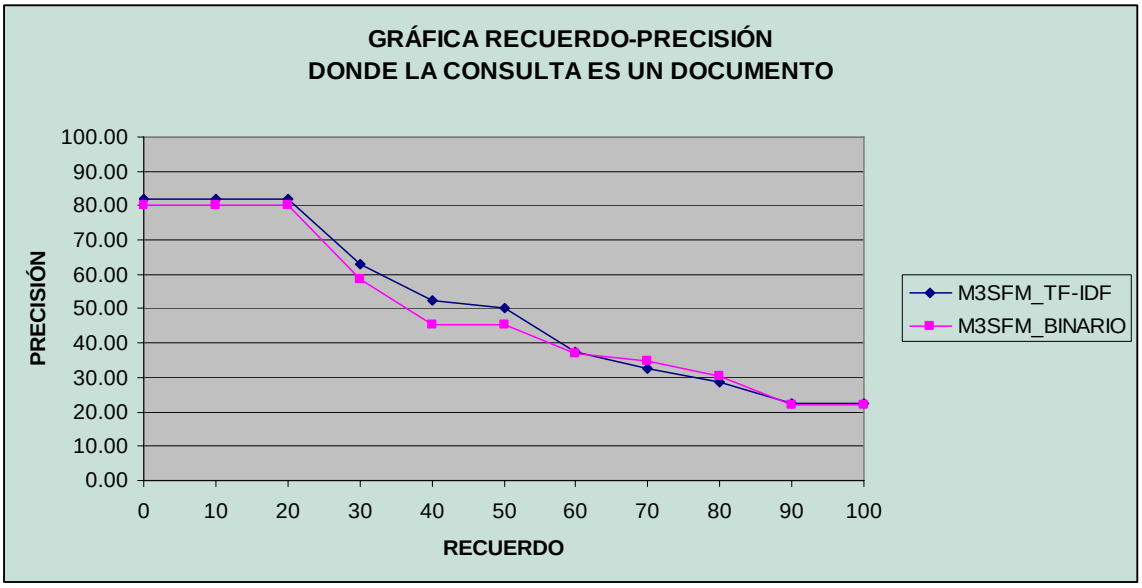

Figura 4.3.4. Gráfica recuerdo-precisión para la colección C33 que muestra los resultados de la tabla 4.3.3.

A partir de la figura 4.3.4 se observa que existen valores altos de precisión utilizando las dos variantes de pesado, aunque el usó del pesado tf-idf obtuvo mejores resultados.

Con el método MSFM3 se obtienen mejores resultados al tener como consulta un documento completo, del cual se extraen las SFM’s por documento y generando un vector cuyas entradas de los elementos del vector corresponden a estas SFM’s. Esto tiene la ventaja de poder comparar estas SFM’s de manera directa con las SFM’s obtenidas dentro de la colección de documentos.

Capítulo 5. Conclusiones y trabajo futuro

5.1 Conclusiones

Los resultados obtenidos muestran que los métodos MSFM1 y MSFM2_PAL, utilizando una representación basada en las SFM's por documento, en cualquiera de sus variaciones de pesado, no superaron al método LUCENE que utiliza una representación basada en palabras. Esta diferencia en los resultados puede ser debida a que en algunos casos, existen documentos que por ser tan pequeños, no están representados por ninguna SFM's por documento, como sucedió con algunos documentos de las colección ADI y CRAN. Otras de las causas podría ser que algunas consultas de las colecciones no tenían ninguna palabra que estuviera contenida dentro de las SFM's por documento, que contenía la colección, por lo cual no se pudieron obtener documentos relevantes.

Una de las finalidades de este trabajo fue mostrar qué tan útiles son las SFM's por documento en los métodos de RI. En este sentido, se observó que con los métodos propuestos MSFM1 y MSFM2-PAL, se tuvo un bajo rendimiento comparándolos contra un método que utiliza una representación basada en palabras como lo es LUCENE. Aunque se podría mejorar el rendimiento de los métodos propuestos claramente si se complementara la

representación basada en SFM's utilizada con una representación basada en palabras.

Una aportación de este trabajo es un método que utiliza secuencias frecuentes maximales por documento para Recuperación de Información a partir de consultas dadas como un documento completo. En este método los documentos de la colección son representados mediante SFM's por documento siguiendo el modelo vectorial, usando un pesado tf-idf y Booleano. Siendo el pesado tf-idf el que tuvo un mejor rendimiento, y en cuyos resultados se refleja que el uso de documentos grandes como consulta mejora la efectividad, pues se pueden comparar directamente las SFM's del documento-consulta con las SFM's por documento de la colección. Este método MSFM3 en una aplicación real puede ser utilizado para buscar aquellos documentos de una colección que se encuentren relacionados a algún documento-consulta (artículo, noticia, etc), basándose en que los documentos relacionados contendrán SFM's por documento similares a las que contiene el documento-consulta, al ser expresiones comunes y repetitivas para transmitir las ideas que se tratan en los temas de los documentos de la colección y la consulta.

5.2 Trabajo Futuro

A partir de los resultados obtenidos en esté trabajo, se plantea como trabajo futuro modificar la forma de pesado del método donde la consulta es un documento, utilizando la intersección de las SFM's por documento de la consulta con las de la colección, con el objetivo de mejorar la precisión, ya que como mostraron los resultados obtenidos, para los métodos MSFM1 y

MSFM2_PAL el pesado de la consulta basada en la intersección tuvo un mejor rendimiento que las demás variaciones.

Otro trabajo futuro es definir un método de RI donde la consulta sea un documento que utilice una forma de representación híbrida basada en la combinación de las palabras de los documentos y las SFM’s por documento, de manera que permita mejorar la precisión y el recuerdo.

Apéndice A

En este apéndice se muestran las stop-words eliminadas de los documentos de la colección y de las consultas.

about	becoming	enough	his	more	out	that've	unlike	who
above	been	etc	how	moreover	over	the	unlikely	who'd
according	before	even	however	most	overall	their	until	who'll
across	beforehand	ever	i'd	mostly	own	them	up	who's
actually	begin	every	i'll	much	per	themselves	upon	whoever
after	beginning	everyone	i'm	must	perhaps	then	us	whole
afterwards	behind	everything	i've	my	rather	thence	very	whom
again	being	everywhere	if	myself	recent	there	was	whomever
against	below	except	in	namely	recently	there'd	wasn't	whose
all	beside	few	indeed	neither	same	there'll	we	why
almost	besides	for	instead	never	seem	there're	we'd	will
alone	between	former	into	nevertheless	seemed	there've	we'll	with
along	beyond	formerly	is	next	seeming	thereafter	we're	within
already	both	found	isn't	nobody	seems	thereby	we've	without
also	but	from	it	none	several	therefore	well	won't
although	by	further	it's	nonetheless	she	therein	were	would
always	can	had	its	noone	she'd	thereupon	weren't	wouldn't
among	can't	has	itself	nor	she'll	these	what	yet
amongst	cannot	hasn't	last	not	she's	they	what'll	you
an	caption	have	later	nothing	should	they'd	what's	you'd
and	could	haven't	latter	nowhere	shouldn't	they'll	what've	you'll
another	couldn't	he	latterly	of	since	they're	whatever	you're
any	did	he'd	less	off	so	they've	when	you've
anyhow	didn't	he'll	let	often	some	this	whence	your
anyone	do	he's	let's	on	somehow	those	whenever	yours
anything	does	hence	like	once	someone	though	where	yourself
anywhere	doesn't	her	likely	one	something	through	where's	yourselves
are	don't	here	ltd	one's	sometime	throughout	whereafter	
aren't	down	here's	made	only	sometimes	thru	whereas	
around	during	hereafter	make	onto	somewhere	thus	whereby	
as	each	hereby	makes	or	still	to	wherein	
at	eg	herein	many	other	such	together	whereupon	
be	either	hereupon	maybe	others	taking	too	wherever	
became	else	hers	me	otherwise	than	toward	whether	
because	elsewhere	herself	meantime	our	that	towards	which	
become	end	him	meanwhile	ours	that'll	under	while	
becomes	ending	himself	might	ourselves	that's	unless	whither	

Tabla A1. Stop-words.

Bibliografía

[1] Baeza Y. R., Ribeiro N.B. *"Modern Information Retrieval".* Ed. Pearson Addison Wesley, ACM Press New York. 1999.

[2] Leloup C. *"Motores de búsqueda e indexación".* Ed. Gestion 2000 S.A., Barcelona, 1998.

[3] Nie J. Y. "On the use of words and N-grams for Chinese Information Retrieval". IRAL-2000, Fifth International Workshop on Information Retrieval with Asian Languages. Institute of System Enginnering, pp.141–148. Hong Kong, China. 2000.

[4] Miller E., Shen D., Liu J., and Nicholas C. "Performance and Scalability of a Large Scale N-gram Based Information Retrieval System". Journal of Digital Information, Vol.1, No. 5, pp. 1-25. Jan. 2000.

[5] Canvar W. B. "Using An N-gram Based Document Representation With A Vector Processing Retrieval Model". In Proceedings of the Third Text Retrieval Conference (TREC-3). pp. 269-278. 1994.

[6] Doucet A. and Ahonen-Myka M. "Non Contiguous Word Sequences for Information Retrieval". In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Workshop on Multiword Expressions: Integrating Processing, Barcelona, Spain, July 21-26. pp. 88-95. 2004.

[7] Zazo R. A., Figuerola P. C., Alonso B. J., Goméz D. R. "Informe Técnico: Recuperación de Información utilizando el modelo vectorial". Participación en el taller CLEF-2001. Departamento de Informática y Automática Universidad de Salamanca. Mayo 2002.

[8] Ahonen M. H. "Finding All Maximal Frequent Sequences in Text". In Proceedings of the 16th International Conference on Machine Learning ICML-99, Workshop on Machine Learning in Text Data

Analysis, Stefan Institute, Eds. D. Mladenic and M Grobelnik. Slovenia. pp. 11-17. 1999.

[9] García R. A. "Desarrollo de algoritmos para el descubrimiento de patrones secuenciales maximales". Tesis Doctoral, Instituto Nacional de Astrofísica Óptica y Electrónica, INAOE, Puebla, México. 2006.

[10] Grossman A. D., Ophir F. "*Information Retrieval Algorithms and Heuristic*". Press Kluwer Academic Publisher. 1998.

[11] Kowalski G. "*Information Retrieval Systems Theory and Implementation*". Press Kluwer Academic Publisher. 1997.

[12] Text Retrieval Conference, en sitio web: http://trec.nist.gov/

[13] National Institute of Standards and Technology, en sitio web: http://www.nist.gov/

[14] Gospodnetic O., Hatcher E. "LUCENE in Action, A guide to the Java search enguine". Manning Publications Co. 2005. Software en sitio web: http://lucene.apache.org/

[15] Harding S.M., Croft W. B. and Weir C. "Probabilistic Retrieval of OCR Degraded Text Using N-Grams". Proceedings of the 1st European Conference on Research and Advanced Technology for Digital Libraries. Pisa, Italy. pp. 345-359.1997.

[16] Stensmo M. "A Scalable and Efficient Probabilistic Information Retrieval and Text Mining System". Proceedings of the International Conference on Artificial Neural Networks. Lecture Notes In Computer Science, Vol. 2415. pp. 643-648. 2002.

[17] Wu J., Tanioka H, Wang S, Pan D, Yamamoto K. and Wang Z. "An Improved VSM Based Information Retrieval System and Fuzzy Query Expansion". Lecture Notes in Computer Science, Springer Berlin / Heidelberg, pp. 537-546. 2005

[18] Blair D.C. "*Language and representation in information retrieval*". Univ. of Michigan, Elsevier North-Holland, Inc, 1990.