Clustering de Documentos Web basado en una Matriz de Términos Frecuentes por Oraciones de Documentos, FP-Growth y Bisecting k-means

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz

Director: Ph.D. (c) MSc Carlos Alberto Cobos Lozada

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Sistemas
Grupo de I+D en Tecnologías de la Información
Gestión de la Información – Búsqueda Web
Popayán, Abril de 2011

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

Agradecimientos

A Dios, por ser nuestro creador, por permitir nuestra formación como profesionales y por su amor incondicional y guía en todos los caminos de nuestras vidas.

A nuestras familias, que sin esperar nada a cambio han acompañado cada momento de nuestras vidas, con su esfuerzo y aliento para que lleguemos a nuestras metas propuestas.

Al Ph.D. (c). Carlos Alberto Cobos Lozada por su dedicación, por su tiempo, apoyo y su enorme conocimiento para guiarnos en este reto.

A nuestros compañeros, amigos y educadores, quienes nos acompañaron en este proceso, por su ánimo, colaboración y palabras de aliento.

Para finalizar, nuestros agradecimientos a la Universidad del Cauca institución que nos forjó como personas, brindándonos la oportunidad a través del programa de Ingeniería de Sistemas de realizar nuestros estudios de pregrado.

Tabla de Contenido

LISTA DE FIGURAS

LISTA DE TABLAS

Presentación

El objetivo primordial de un sistema de recuperación de información consiste en identificar los documentos que son útiles a un usuario en particular. Un documento es relevante para un usuario cada vez que éste considere el documento útil a sus necesidades especificas de información, de lo contrario el documento es considerado como no relevante. Existen muchos factores complejos los cuales gobiernan la relación de relevancia entre un documento y una consulta de usuario. Se puede decir que hoy en día es prácticamente imposible diseñar un sistema que realice predicciones exactas acerca de la relación de relevancia mencionada (Wong and Butz 2000)**.**

Cuando se maneja un reducido número de documentos, un archivo invertido es suficiente para almacenar la información que representa el contenido del documento. Sin embargo, dado el número de documentos que se encuentran en la Web, esto no es suficiente. Los motores de búsqueda deben responder a las consultas usando un sistema distribuido mediante el cual envía las consultas a numerosos procesos, que a su vez inspeccionan sus propios archivos invertidos. Usando estos archivos estructurados, un motor de búsqueda debería encontrar rápidamente aquellas páginas Web donde las palabras claves de dichas páginas concuerden con la búsqueda solicitada por el usuario, y así realizar el cálculo adecuado por cada página encontrada, indicando el grado de similitud con la solicitud realizada (Savoy 2000).

En la actualidad existen varios sistemas de recuperación de información en la web, que son muy usados a nivel mundial, entre ellos: Google, Yahoo!, Bing y Ask, los cuales son llamados buscadores web; pero también existen meta buscadores que se especializan en usar más de un buscador web tradicional para entregar al usuario un conjunto de documentos más completo (extenso) y

en muchos casos más apropiado a las necesidades especificas de búsqueda del usuario, ya que aplican procesos adicionales de personalización y filtrado de los documentos.

Además de los meta buscadores, existen los meta buscadores que agrupan documentos web, o motores de agrupación web (web clustering engines), que además de usar como fuentes de información los motores de búsqueda web tradicional, presentan los resultados de una búsqueda (los documentos), de una forma distinta, basada en el agrupamiento por afinidad o como se conoce en la comunidad informática como Clustering de Documentos Web. Algunos representantes de estos meta buscadores son: Carrot, Clusty e iBoogie.

El clustering es un proceso de formación de grupos de objetos similares dado un conjunto de entrada. Cuando se aplica clustering a los resultados de las búsquedas Web, el clustering puede ser percibido como una manera de organizar los resultados dentro de un número razonable de grupos temáticos. Por ejemplo, cuando se busca por “Clinton”, los resultados podrían representarse en grupos temáticos tales como: “Bill Clinton”, “Hillary Clinton”, “Clinton Country”, de esta manera los usuarios inexpertos, quienes tienen dificultad a la hora de formular la búsquedas, se ven beneficiados porque identifican la información que realmente les interesa; incluso los usuarios avanzados también pueden ser beneficiados con esta técnica (BrightPlanet).

Diversos algoritmos de clustering de documentos web han sido implementados por la comunidad científica, muchos utilizan el enfoque clásico de clustering, pero no existe una propuesta que cumpla a cabalidad con los requisitos necesarios del clustering de documentos Web, por tal razón en este trabajo de grado se presenta una propuesta para el clustering de documentos Web basado en una Matriz de Términos Frecuentes por Oraciones de Documentos, FP-Growth y Bisecting k-means.

En este documento se encuentran diferentes secciones que contienen la descripción de los conceptos teóricos además de la metodología utilizada para el desarrollo del proyecto. A continuación se describe de manera general el contenido de esta monografía y su organización.

Capítulo 1 – Introducción: Este capítulo presenta el planteamiento del problema que dio origen al proyecto, la justificación del mismo, los objetivos planteados y los resultados obtenidos durante el proceso de desarrollo del proyecto.

Capitulo 2 – Contexto Teórico: Este capítulo enmarca las bases teóricas utilizadas para el desarrollo del proyecto, se encuentran las definiciones de las funciones objetivo utilizadas, los modelos con los cuales se representan los documentos, medidas de similitud y el algoritmo Bisecting K-Means.

Capitulo 3 – El Algoritmo Propuesto: Se presenta en detalle el algoritmo de clustering de Documentos Web basado en una Matriz de Términos Frecuentes por Oraciones de Documentos, FP-Growth y Bisecting k-means. Se describe la etapa de pre-procesamiento, los parámetros y los pasos del algoritmo. Además se describe la construcción de las diferentes matrices de representación de documentos TDM, TFOD y COD, usadas como entrada del algoritmo de agrupación.

Capitulo 4 – Evaluación: En este capítulo se presenta la metodología utilizada para realizar la evaluación de algoritmo, análisis y resultados obtenidos.

Capitulo 5 – Conclusiones, recomendaciones y trabajo futuro: Aquí se describen las conclusiones generadas después de terminar el proyecto, además se presentan mejoras o elementos adicionales que se puedan incluir en un trabajo futuro para de esta forma dar continuidad al proyecto.

Capitulo 6 – Glosario y Bibliografía: Este capítulo contiene la bibliografía y documentación empleada en la realización del proyecto, incluye además el catálogo de palabras claves en el contexto del proyecto con su respectiva definición.

Capitulo 1: Introducción

1 Planteamiento del Problema

Actualmente, el acceso a la información Web se realiza principalmente mediante motores de búsqueda y directorios temáticos. Como ejemplo se tiene a Google, un buscador Web que funciona bien cuando se busca información como páginas personales o sitios correspondientes a instituciones, corporaciones o eventos, pero cuando se requiere encontrar información detallada sobre un tema en especifico, este modelo presenta algunas limitaciones, ya que la lista ordenada que devuelve no se presenta al usuario de una forma conceptualmente organizada y tampoco realiza la respectiva relación de información que se extrae de diferentes páginas encontradas. Por lo anterior, se han desarrollado otras estrategias de búsqueda, como por ejemplo los meta buscadores que agrupan documentos Web (o motores de agrupación Web), como Vivísimo (Li 2000), que además de presentar los documentos más importantes también presentan una jerarquía temática asociada a los resultados obtenidos en la búsqueda.

El objetivo del clustering de documentos Web, es agrupar en este caso documentos de forma tal que los resultados relacionados a un mismo tema sean muy semejantes entre sí, y que los documentos pertenecientes a temas diferentes estén aislados (o separados) de los demás grupos, es decir que cada grupo se diferencie de los demás.

Del conjunto de algoritmos que se encuentran en la actualidad para realizar clustering, algunos se inspiran en técnicas para definir reglas de asociación, los cuales reflejan relaciones entre los atributos presentes en los datos (BrightPlanet). Las reglas de asociación permiten la extracción de información

útil a partir de un gran volumen de información almacenada en las bases de datos transaccionales. La mayoría de los algoritmos para el clustering de documentos Web, parten del modelo espacio vectorial (MEV) como esquema de representación de los documentos. En este modelo, los documentos son tratados como un conjunto de términos (bolsa de palabras) y se representan como un vector donde cada término se trata de manera individual para ser manipulado y medido por su importancia. Los documentos largos quedan poco representados puesto que contienen valores que son poco comunes, lo que se conoce como sensibilidad semántica, es decir, que los documentos con contextos similares pero con diferente significado no son asociados (Casillas 2005). Dado que no se encuentra en la literatura ninguna investigación que se base en el MEV combinada con conceptos de las reglas de asociación en las oraciones de los documentos (como unidad mínima semántica), en este proyecto se definió la siguiente pregunta de investigación ¿Qué tan apropiado puede ser el clustering de documentos Web basado en la extracción de los términos frecuentes de una Matriz de términos por oraciones de documentos?

Para resolver esta pregunta, se parte del uso de un algoritmo tradicional para la extracción de los términos frecuentes (frequent term sets), en este caso FP-Growth por tener un mejor rendimiento sobre los otros algoritmos reportados en la literatura, normalmente variaciones de Apriori y GRI. Y para el algoritmo de clustering de documentos Web, propiamente dicho, se realizó una variación de Bisecting K-Means que define automáticamente el número de grupos que se deben generar.

2 Justificación

En los últimos años el Clustering de Documentos Web se ha convertido en un tema importante para la ciencia de la computación, y mas específicamente para el área de investigación en Recuperación de Información (RI), y aunque se han desarrollado muchas propuestas para este tema, hasta ahora no se ha creado una que satisfaga totalmente los requisitos necesarios para obtener una

búsqueda eficiente y óptima. Por tal razón este proyecto presenta una nueva propuesta de un Algoritmo de Clustering de Documentos Web basado en basado en una Matriz de Términos Frecuentes por Oraciones de Documentos, FP-Growth y Bisecting k-means, buscando superar algunas deficiencias que se presentan en otros algoritmos.

Una variante en esta investigación es que se utiliza como fuente de información una matriz de términos por oraciones de documentos, términos frecuentes por oraciones de documentos y conceptos frecuentes por oraciones de documentos y para la construcción de etiquetas representativas de los grupos obtenidos, se hizo uso del método de etiquetado por términos frecuentes estadísticamente más representativos.

Desde el punto de vista práctico este proyecto presenta al usuario una búsqueda de resultados más precisa, reduciendo el tiempo de consulta puesto que tiene la ventaja de encontrar los documentos más relevantes de manera más rápida que en los buscadores convencionales a saber Google, Yahoo! y Bing.

Como soporte tecnológico para el desarrollo de este proyecto se utilizó, Microsoft: Visual Studio 2008, SQLServer 2005, Microsoft Office 2007, herramientas que fueron seleccionadas de acuerdo a la experiencia en su manejo por parte del Grupo de investigación y desarrollo en Tecnologías de la Información (GTI) durante los últimos años, sobre todo en proyectos de trabajo de grado exitosos relacionados con Búsqueda Web. Además la disponibilidad del software gracias al convenio de la Universidad del Cauca con el programa MSDN Academic Alliance.

Además, en el transcurso del desarrollo del proyecto los autores aplicaron diferentes conocimientos adquiridos durante el proceso de formación como

ingenieros de sistemas, integrándolos con conocimientos nuevos que ayudaron a culminar el proyecto cumpliendo con los objetivos propuestos.

3 Objetivos

A continuación se muestran los objetivos del proyecto, conforme fueron aprobados por el Comité de Investigaciones de la Facultad de Ingeniería Electrónica y Telecomunicaciones (CI-FIET) en el documento de anteproyecto.

3.1 Objetivo General

Proponer un algoritmo para el clustering de documentos Web basado en una Matriz de Términos Frecuentes por Oraciones de Documentos, FP-Growth y una variación de Bisecting k-means, y evaluar sus resultados con medidas clásicas de recuperación de la información.

3.2 Objetivos Específicos

1. Obtener un survey[1] que refleje los más recientes (últimos diez años, 1999-2009) aportes de investigación en el diseño y uso de algoritmos de reglas de asociación en minería de datos.
2. Modelar un algoritmo de clustering de documentos Web para:
   - Incluir los conceptos que se usan en minería de datos para encontrar reglas de asociación (específicamente con el algoritmo FP-Growth) que genere una matriz de términos frecuentes por oraciones de los documentos (TF-OD) reduciendo así la alta dimensionalidad de los documentos de entrada y convirtiendo esta matriz en la fuente original del proceso de clustering.

[1] En ciencias de la computación, es un documento de investigación que presenta una visión general de los puntos de vista, modelos, estrategias y/o algoritmos usados por la comunidad científica para enfrentar y solucionar un problema o tema de investigación. Teniendo en cuenta la reglamentación vigente en cuanto al desarrollo de los trabajos de grado en la FIET y el esfuerzo que puede suponer el desarrollo de un survey, para este objetivo se tomarán un máximo de treinta (30) referencias.

- Definir automáticamente el número de grupos (k) a formar basado en una modificación del algoritmo bisecting k-means.
- Establecer nombres de los grupos (etiquetas) basado en los términos frecuentes estadísticamente más relevantes en cada grupo.
- Usar como matriz de entrada una compuesta de conceptos por oraciones de documentos, construida a partir de la matriz TF-OD y la ontología WordNet.

3. Definir la precisión[2], recuerdo[3] y medida F[4] del algoritmo propuesto usando conjuntos de datos sintéticos basados en Reuters-21578 y comparar los resultados del algoritmo cuando se toma como fuente de datos la matriz TF-IDF estándar del MEV.

4 Resultados Obtenidos

Los productos obtenidos al culminar este proyecto de investigación fueron los siguientes:

- Prototipo funcional del algoritmo propuesto para el Clustering de Documentos Web basado en una Matriz de Términos Frecuentes por Oraciones de Documentos, FP-Growth y Bisecting k-means.
- Survey de Reglas de Asociación que se anexa a la presente monografía.
- Artículo titulado “Clustering de Documentos Web basado en una Matriz de Términos Frecuentes por Oraciones de Documentos, FP-Growth y Bisecting k-means” en proceso de evaluación en una revista nacional indexada.
- Monografía de trabajo de Grado. Hace referencia al presente documento y a los anexos del mismo, donde se presenta la descripción detallada de la investigación.

---

[2] $\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$

[3] $\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$

[4] $F = \frac{2 \cdot \text{precision} \cdot \text{recall}}{(\text{precision} + \text{recall})}.$

Capitulo 2: Marco Teórico

5 Clustering

En pocos años la Web se ha convertido en una herramienta universal para todo tipo de actividades culturales, profesionales y comerciales. Los avances tecnológicos de los últimos años han provocado un aumento exponencial de la información disponible, lo que obliga al desarrollo de herramientas dedicadas a la gestión y Recuperación de Información (RI), entendida ésta como la selección de la información solicitada por un usuario de entre toda la información disponible. Esta labor es realizada por los llamados sistemas de recuperación de información (SRI), una clase de sistemas de información que tratan con bases de datos compuestas por documentos y los índices de la información en ellos contenidos, que procesan las consultas de los usuarios y entregan los documentos relevantes en un intervalo de tiempo apropiado (Fersini 2009).

La agrupación de documentos se basa en la hipótesis de que los documentos fuertemente asociados tienden a ser relevantes para una misma consulta, lo que permite agilizar el procesamiento documental al poder atender a las características de grupos de documentos en lugar de los documentos individuales. Con ello, al mismo tiempo que se logra una menor carga sobre el sistema, también se mejora la eficacia y eficiencia el presentar al usuario los resultados en grupos de documentos similares con la información solicitada (Fersini 2009).

Los motores que agrupan documentos web pueden utilizarse como complemento de los motores de búsqueda tradicional por las siguientes razones (Carpineto 2009):

- Rápida recuperación de subtemas: si los documentos que pertenecen al mismo subtema han sido colocados correctamente dentro del mismo cluster y el usuario es capaz de escoger el camino correcto en la lista de temas, cada documento puede ser accedido de manera logarítmica en vez de manera lineal, lo que hace más rápida la búsqueda.
- Exploración de temas: La jerarquía de los cluster proporciona una vista completa de alto nivel incluyendo términos para la reformulación de la consulta, lo que es particularmente útil para las búsquedas de información en dominios desconocidos o dinámicos.
- Los usuarios de buscadores web tradicionales, por lo general solo ven los primeros resultados de la página, pasando por alto mucha información. El Clustering ayuda a resumir el contenido de los resultados de la búsqueda en muchos resultados en una sola vista en los resultados de la primera página, el usuario puede revisar cientos de resultados altamente relevantes sin necesidad de descargarlos.

El Clustering es un campo bastante amplio que estudia los métodos generales para la agrupación de datos sin etiqueta, con aplicaciones en muchos campos. En general, el Clustering puede ser caracterizado como un proceso de descubrimiento de subconjuntos en la entrada; la entrada es un conjunto de resultados de la búsqueda obtenida en respuesta a una consulta realizada por un usuario, cada una descrita por una URL, un titulo y un snippet (un texto corto que resume el contexto en el que las palabras aparecen en la página de resultados) (Steinbach, Karypis et al. 2000).

El Clustering de documentos Web, se ha convertido en una técnica ampliamente utilizada para mejorar los resultados en los motores de búsqueda,

rastreo Web, organización no supervisada de documentos, recuperación de la información, entre otros (Mahdavi and Abolhassani 2009).

El objetivo de los motores de Clustering Web son efectivos en ciertos casos o en diferentes tipos de tareas, tales como consultas de navegación (este tipo de consulta es cuando el usuario necesita encontrar una dirección URL en particular) y consultas transaccionales (este tipo de consulta es cuando el usuario se interesa en la actividad Web). Los motores de búsqueda muestran al usuario una lista ordenada por relevancia, lo que hace que el usuario tenga que verificar dicha lista haciendo que el proceso se vuelva lento, es más, los motores de búsqueda se vuelven ineficientes a la hora de dar solución a consultas ambiguas y difíciles ya que los resultados se muestran en una lista que mezcla los resultados relevantes e irrelevantes (Carpineto 2009).

Los motores de Clustering Web agrupan y organizan los resultados, y los presenta al usuario por categorías, esto para facilitar la selección del resultado que mejor se adapte a la consulta (Carpineto 2009). Los motores de Clustering son una ayuda para los motores de búsqueda cuando estos fallan, algunos de los aspectos donde los motores de Clustering son de mucha ayuda son los siguientes (Carpineto 2009):

1. Rápida recuperación de subtemas: Cuando los clusters se forman correctamente de acuerdo al subtema y el usuario es capaz de escoger el camino correcto dentro de la lista de resultados desplegada, obtener la respuesta correcta va a ser más rápido, puesto que el acceso no es lineal es logarítmica.
2. Exploración de temas: La jerarquía de los clúster da una vista de alto nivel a toda la consulta, esto incluye los términos para reformular la consulta, lo cual es útil para la búsqueda de dominios desconocidos o dinámicos.

3. Reducir pasar por alto la información: Una vista típica de los motores de búsqueda muestra solamente la primera página de resultados, un motor de Clustering resume el contenido de muchos resultados de búsqueda en una sola vista o en la primera página de resultados. El usuario puede revisar cientos de resultados relevantes.

El proceso de Clustering puede se caracteriza por ser un proceso en el cual se descubren subconjuntos de datos de entrada, de tal manera que los objetos dentro de los cluster descubiertos son similares entre sí y diferentes a otros clusters, usualmente de acuerdo a alguna medida de similitud (Carpineto 2009)

Al hacer un estudio más conveniente del Clustering Web, se ha identificado dos etapas de esta técnica, una de estas etapas se conoce como pre-recuperación, la cual se basa en rasgos o características más frecuentes que haya en la colección, la otra etapa se conoce como post-recuperación esta etapa puede considerarse más efectiva y la razón es que esta se basa en las características de una consulta específica (Carpineto 2009).

En el Clustering de documentos se identifican dos parámetros, la *entrada y la salida*, la entrada un conjunto de resultados de búsquedas obtenidos como respuesta a una consulta realizada por un usuario, los cuales se describen en una dirección URL, un título, un resumen o snippet, la salida es un conjunto de grupos que representan la etiqueta lo más cercana y organizada posible en un conjunto de particiones planas. La naturaleza dinámica de los datos adicional al uso interactivo de las agrupaciones de documentos ha colocado nuevos requerimientos y cambios a la tecnología de Clustering, así: (Carpineto 2009)

1. Etiquetas significativas: Los algoritmos tradicionales usan el centroide del cluster como un cluster representativo, pero a la hora de ofrecer ayuda al usuario para alcanzar los elementos buscados es de poca

utilidad. El contenido del cada cluster debe estar bien indicado en la etiqueta del cluster.

2. Eficiencia Computacional: El clustering de documentos Web se hace en línea, debido a esto la respuesta se debe hacer en muy corto tiempo. El obtener los resultados de la búsqueda es el paso crítico en estos sistemas, ya que son consultados en buscadores Web tradicionales, pasan el envió de la consulta y la espera de los resultados a través de la red. La eficiencia del algoritmo de construcción del cluster es menos importante debido al reducido número de resultados de entrada.
3. Descripción corta de los datos de entrada: Debido a razones computacionales, los datos disponibles del algoritmo de clustering para los resultados de búsqueda se limitan usualmente a una URL, un título opcional, un resumen del contenido de un texto (snippet).
4. Número desconocido de clusters: En algunos métodos se requiere que el número de clusters a formar sea definido con anterioridad. Para el clustering de documentos Web es necesario que el algoritmo defina automáticamente este valor, puesto que las consultas de los usuarios hace que el número de grupos cambie.
5. Solapamiento de clusters: Un resultado puede estar asignado a muchos temas, se hace útil el solapamiento de clusters. Incluso, un grafo puede ser mas flexible que una estructura tipo árbol, ya que este último no permite fácilmente la recuperación para las malas decisiones.
6. Interfaz gráfica de Usuario (GUI): Para presentar la información se hace necesario tener en cuenta las interfaces basadas en Web, además se deben diseñar de tal forma que sean fáciles de usar y que sean computacionalmente eficientes.

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

6 Arquitectura y técnicas de motores de Clustering Web

Las implementaciones básicas de los motores de clustering Web se basan en cuatro componentes generales:

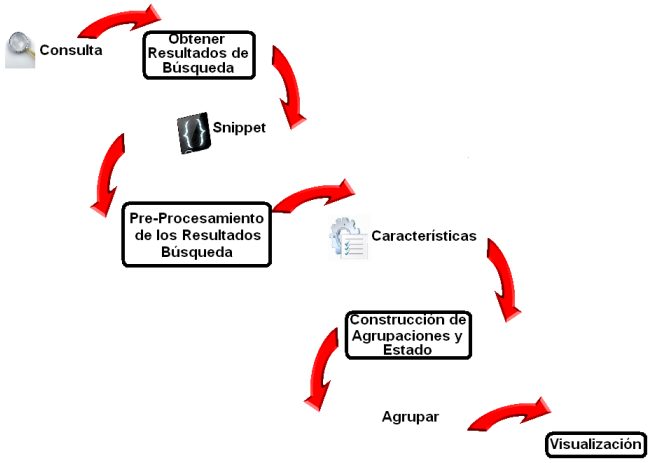


Figura 1. Componentes de un motor que agrupa documentos Web (Adaptado de (Carpineto 2009)).

6.1 Obtener Resultados de Búsqueda

Esta tarea se basa en proveer entradas para el resto del sistema, a partir de las especificaciones dadas por el usuario para la consulta, el componente debe entregar entre 50 y 500 resultados, cada uno de los cuales debe tener un título, un resumen (snippet), y una URL que direcciones al documento completo al que se refiere. Una fuente potencial de búsqueda de resultados para el componente de adquisición son los motores de búsqueda públicos: Yahoo!, Google, MSN Search. Una forma que comúnmente se usa para buscar resultados en los motores de búsqueda públicos son las API's, aunque estas tengan algunas limitaciones desde un punto de vista técnico y legal.

Search Engine	Protocol	Queries Per Day	Results Per Search	Terms of Service
Alexa	SOAP or REST	n/a	20	Paid service (per-query).
Gigablast	REST/XML	100	10	Noncommercial use only.
Google	SOAP	1 000	10	Unsupported as of December 5, 2006. Noncommercial use only.
Google CSE	REST/XML	n/a	20	Custom search over selected sites/ domains. Paid service if XML feed is required.
MSN Search	SOAP	10 000	50	Per application-ID query limit. Noncommercial use only.
Yahoo!	REST/XML	5 000	100	Per-IP query limit. No commercial restrictions (except Local Search).

Tabla 1. Motores de Búsqueda más populares y sus limitaciones Dic. 2007. Tomado de (Carpineto 2009).

Algunas API's, las cuales se presentan en la tabla 1, son las utilizadas para la recuperación de la información, aunque dichas técnicas pueden afectar el rendimiento general del sistema de clustering. Otra manera de obtener resultados desde un motor de búsqueda público se denomina HTML scraping. Esta técnica hace uso de expresiones regulares o también hace uso de detección de marcas para extraer títulos, resúmenes y URLs desde el flujo HTML. Debido a que jurídicamente el tratamiento automatizado de resultados está prohibido, esta técnica actualmente no es usada.

6.2 Pre-procesamiento de los resultados

En todos los sistemas de clustering hay un paso en común que es el pre-procesamiento de entrada. El objetivo de este paso es convertir el contenido de los resultados de la búsqueda en una secuencia de características usadas por el algoritmo de clustering. Una secuencia común hace un recorrido así: Identificación del lenguaje, tokenización, pre-procesamiento de lenguaje (Stemming: por lo regular se limita a la raíz de las palabras) y finalmente se selecciona los atributos o características. En la tokenización, en donde se descompone en forma de listas de cadenas (strings), los

resultados de la búsqueda se dividen en una secuencia de tokens (unidades) que generalmente representa una sola palabra, número, símbolo. En la lematización (stemming) donde se eliminan sufijos y prefijos de las frases para reducir las diferentes formas gramaticales para así llevarlas a su forma base.

En este paso se hace necesario obtener los atributos o características para cada resultado presente en los datos de entrada. En términos generales de minería de datos, los atributos son entidades atómicas las cuales pueden describir un objeto y representar sus más importantes características en un algoritmo.

6.3 Construcción de Agrupaciones y Etiquetado

El conjunto de resultados de la búsqueda, que se obtiene del pre-procesamiento son tomados como datos de entrada para el algoritmo de clustering, el cual es el que construye los grupos y las etiquetas. Debido a la variedad de algoritmos de clustering propuestos, existe una clasificación de estos de acuerdo a las características presentes en la estructura de los datos de salida, así:

6.3.1 Algoritmos centrados en datos

La representación común de este grupo consta de algoritmos de Clustering tales como jerárquicos, de optimización, de espectro y que son aplicados al problema de Clustering de búsqueda de resultados, a veces, modificados mínimamente para producir algunos tipos de representación textual por cada grupo seleccionado por el usuario final. Un ejemplo de este tipo de clúster es Scatter/Gather, fue desarrollado en 1992 y se conoce como uno de los predecesores de los sistemas de recuperación que surgieron después. Este algoritmo opera haciendo uso de un agrupamiento inicial, el cual se denomina dispersión; el proceso se realiza offline y posibilita la obtención inicial de documentos, luego, en tiempo de consulta el usuario escoge los clusters de su

interés y el sistema reagrupa dinámicamente la sub-colección de documentos (Carpineto 2009).

6.3.2 Algoritmos consientes de la descripción

El inconveniente esencial de los algoritmos centrados en datos es la creación de una descripción comprensible desde un modelo de representación de textos que no es realizado para este propósito. Los algoritmos consientes de la descripción tratan este problema y pretenden dar solución mejorando la construcción del clúster para que sea interpretado por los usuarios. Una forma de obtener ese resultado es usar algoritmos de Clustering "Monothetic". Estos algoritmos otorgan importancia a una sola característica del agrupamiento. El primer algoritmo que implementó esta idea fue Suffix Tree Clustering (STC) (Zamir and Etzioni 1999), y está centrado en las etiquetas y basado en las frases frecuentes de los documentos.

6.3.3 Algoritmos centrados en la descripción Céntrica

Los elementos de este grupo incorporan algoritmos que se han diseñado específicamente para Clustering de búsqueda de resultados y toma en cuenta dos calidades: la del Clustering y la de las descripciones (Carpineto 2009). Un ejemplo de este algoritmo es Lingo implementado en el framework de Carrot.

6.4 Visualización de las agrupaciones resultantes

Los sistemas generalmente muestran los resultados de manera ordenada, como por ejemplo, en carpetas organizadas jerárquicamente, mostrando en el más alto nivel las etiquetas, permitiendo así que el usuario tenga la facilidad de encontrar lo que necesite; algunos sistemas utilizan la metáfora de carpetas, ya que es la más familiar para los usuarios y porque no requiere entrenamiento, para desplegar sus resultados para que así el usuario con tan solo dar un click sobre la etiqueta se despliega los diferentes snippets de los documentos contenidos en dicha carpeta.

6.5 Eficiencia e Implementación

El primer propósito de un motor de Clustering web es reducir el esfuerzo necesario para encontrar información, un factor importante es la experiencia del usuario. Parte de esto radica en la velocidad con la que se entregan los datos al usuario. Algunos de los factores que contribuyen en la mejora de la eficiencia se pueden resumir en las siguientes técnicas: Adquisición de los resultados de búsqueda, pre-procesamiento y Clustering.

6.5.1 Adquisición de resultados de búsqueda

Sin importar de donde se obtiene los resultados de búsqueda, ya sea de un motor de búsqueda público o de los índices de un documento local, los datos de entrada para el Clustering toma gran parte del tiempo total del procesamiento. La razón más importante en el retraso de la solicitud es que muchos de los API's de los motores de búsqueda, admiten que los resultados para una solicitud tengan un número fijo, sin embargo existe un retraso que está alrededor de 1.6 segundos.

Motor de Búsqueda	Tiempo Promedio de Espera (seg)
API Google	5.85
API Yahoo!	2.12
API MSN	0.56
Lucene (snippets)	1.78

Tabla 2. Tiempo que toma 200 resultados en ser recolectados. (Adaptado de (Carpineto 2009))

6.5.2 Clustering

En esta fase se realiza el etiquetado y se crean las agrupaciones, esta tarea depende mucho del algoritmo de Clustering específico utilizado, la fase de Clustering puede contribuir enormemente en el tiempo total del procesamiento. La mayoría de los algoritmos de Clustering tienen componentes comunes, tales como tokenización y stemming, la eficiencia del Clustering es una función

principal en la complejidad computacional del elemento central de un algoritmo de agrupamiento particular. Por ejemplo, Lingo utiliza la mayoría de tiempo en el proceso de descomposición en valores singulares. Un problema mas es la extracción de frases frecuentes, donde la forma más eficaz es hacerlo por medio de matrices de sufijos.

Como resumen, se hace énfasis en que la eficiencia del Clustering Web es la suma de los tiempos requeridos para extraer los documentos de los motores de búsqueda, mas el tiempo de pre-procesamiento, mas el resultado del Clustering. En algunos algoritmos, como el caso de Lingo, el tiempo gastado en la construcción del cluster toma alrededor de entre el 10% y 30% del tiempo total de pre-procesamiento.

6.6 Algoritmos de Clustering

La técnica de Clustering de documentos se describe como la división de un conjunto de documentos dentro de un número específico de grupos. Los documentos que se encuentran dentro de un mismo grupo deben presentar un alto grado de similitud entre ellos, y poca similitud con los documentos que estén en otros grupos. Los métodos más comunes de Clustering tradicional son los métodos particionales, jerárquicos, basados en la densidad, entre otros (Mahdavi and Abolhassani 2009).

Desde el enfoque tradicional de Clustering en minería de datos o aprendizaje de máquina, existen diferentes algoritmos de Clustering dentro de los cuales se pueden destacar: algoritmos jerárquicos, algoritmos particionales, algoritmos jerárquicos y particionales (Mezcla). A continuación se presentan algunos de ellos.

6.6.1 Técnicas Jerárquicas

Las técnicas jerárquicas buscan obtener una secuencia de particiones anidadas o multinivel y con una estructura de árbol (Mahdavi and Abolhassani

2009). Cada nivel intermedio se puede ver como la combinación de dos clusters desde el nivel inmediatamente inferior (o la división de un cluster desde el nivel inmediatamente superior). El resultado de un algoritmo de Clustering jerárquico se puede representar gráficamente como un árbol, llamado dendograma. Este árbol muestra gráficamente el proceso de fusión y los clusters intermedios. La Figura 1 muestra la forma como cuatro puntos se pueden combinar en un solo cluster. Para el Clustering de documentos, el dendograma proporciona una taxonomía o un índice jerárquico (Steinbach, Karypis et al. 2000).

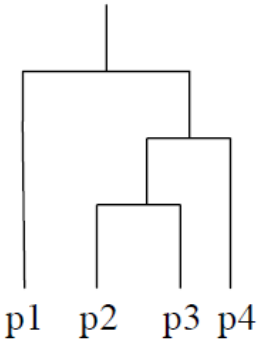


Figura 2. Dendograma generado por algoritmos de Clustering jerárquico tomado de (Steinbach, Karypis et al. 2000)

Hay dos enfoques básicos para generar un clustering jerárquico:

6.6.1.1 Aglomerativo

Comienza con los puntos como clusters individuales y en cada paso se combina el mas similar o el par más cercano de clusters. Esto requiere una definición de similitud o distancia del cluster.

6.6.1.2 Divisivo

Comienza con un cluster (todo incluido) y en cada paso se divide el cluster hasta llegar a un solo individuo (cluster sencillo). En este caso, se decide en cada paso que cluster de divide y como se realiza la división.

Algunos de los Algoritmos del Clustering jerárquico son: Single Link (SL), Average Link (AV), Complete Link (CL), Chameleon (D. Pascual, F. Pla et al.) y el más representativo UPGMA (Unweighted pair-group method with arithmetic mean)[5] que construye un árbol de conjuntos de secuencias basado en distancias entre pares de secuencias, estas secuencias pueden ser los nodos del árbol, donde la altura de cada nodo es la mitad de la distancia entre dos clusters (Hammouda 2001) . La principal ventaja de UPGMA es que es rápido y puede soportar muchas secuencias de palabras, otra ventaja es que genera árboles ultramétricos. Una distancia es ultramétrica si y solo si para cada dos de tres distancias son la mismas y más grandes que la tercera (Hammouda 2001). Las desventajas de UPGMA son: construye raíces de árboles, es decir, solamente un nodo hijo se convierte en la raíz del siguiente subárbol; todas las hojas se encuentra en el mismo nivel (Steinbach, Karypis et al. 2000).

Aunque se dice que los métodos jerárquicos obtienen el cluster con la mejor calidad, comúnmente no realizan la reubicación de los documentos, los cuales pueden lograr una clasificación insatisfactoria en los siguientes pasos del clustering (Mahdavi and Abolhassani 2009).

La ventaja del Clustering jerárquico es que es eficiente y conceptualmente simple, no se necesita conocer el número de clusters k (Oren and Oren 1998). Las desventajas del Clustering jerárquico son: Usualmente sufren por su inhabilidad para desarrollar ajustes una vez se haya fusionado o una vez haya comenzado la fusión o división del cluster (Fung, Wang et al. 2003), la inflexibilidad a veces reduce la eficiencia del Clustering, ya que no se puede corregir errores o decisiones erróneas que se han hecho antes de su ejecución.

[5] Método de grupo de pares no ponderado con media aritmética

6.6.2 Técnicas de clustering particional

En contraste con las técnicas jerárquicas, las técnicas de clustering particional crean una partición de un solo nivel (no anidado) de los datos. Si K es el numero deseado de clusters, entonces los algoritmos con enfoques particionales suelen encontrar todos los K clusters al mismo tiempo. Esto contrasta con los esquemas jerárquicos tradicionales, que dividen en dos grupos un cluster para obtener dos clusters o combinar dos clusters para obtener uno. Por supuesto, una aproximación jerárquica puede ser usada para generar partición plana de K clusters, y del mismo modo, la aplicación repetida de un esquema particional puede proporcionar un clustering jerárquico (Steinbach, Karypis et al. 2000).

Los algoritmos particionales han presentado buenos resultados a la hora de realizar Clustering en bases de datos grandes, con muchos documentos, esto se debe al relativo bajo consumo computacional que requieren, el algoritmo más conocido dentro de los métodos particionales es el algoritmo K-Means (Mahdavi and Abolhassani 2009).

Los algoritmos particionales clasifican los documentos en k grupos que cumplen con los siguientes requisitos: cada grupo debe contener al menos un documento y cada documento debe pertenecer a un grupo (MARTINEZ A. 2007). Este proceso particional se repite hasta que se alcanza el número establecido de cluster definidos por el usuario o hasta que los cluster no se puedan seguir dividiendo (Sasaki and Shinnou 2004).

Entre los algoritmos particionales se cuenta que en 1967 Mac Queen propone K-Means (D. Pascual, F. Pla et al.) mediante el cual se puede definir un conjunto de documentos a particionar, el número de grupos y un centroide por cada grupo para cada documento de la base de datos, se calcula la distancia a cada centroide y se determina el centroide más cercano (Perez, Henriques et

al. 2007). En 1974 Fuzzy C-Means (Dunn 1974) hace su aparición y en 1981 se mejora, con este algoritmo un documento puede pertenecer a dos o más cluster y rara vez pertenece a un solo cluster ya que se basa en conceptos de lógica difusa.

Las ventajas de los algoritmos particionales son: Simplicidad del algoritmo, velocidad que permite más recorridos en bases de datos grandes, baja complejidad y alta precisión y provee un cluster que satisface un criterio óptimo (Berry and Castellanos 2007). Las desventajas de los métodos particionales son: El rendimiento no tiene el mismo resultado con cada ejecución, dado que el cluster resultante depende de la asignación inicial y del número K, y un alto tiempo de cómputo (Berry and Castellanos 2007).

6.6.3 WordNet

En este sentido, WordNet (1985) ha sido una de las herramientas más utilizadas. WordNet es una herramienta basada en ontologías que agrupa las palabras en grupos de sinónimos llamados Synsets (Li, Chung et al. 2008), proporcionando definiciones cortas y generales y almacenando las relaciones semánticas entre estos conjuntos de sinónimos. Con lo anterior, un modelo de RI basada en MEV puede extenderse para usar el concepto de similitud semántica. Así mismo, los modelos de frases y de términos frecuentes pueden integrar las ontologías para resolver problemas relacionados con la semántica.

6.6.4 Modelo de representación de documentos

En 1999 con la aparición de Suffix Tree Clustering (STC) (Zamir and Etzioni 1999) se presenta el primer algoritmo que se basa en un Modelo de representación de documentos basado en frases, el cual define una frase frecuente como la secuencia de palabras frecuentes que aparece en el documento una cierta cantidad de veces (Li, Chung et al. 2008). Para calcular esa frecuencia se utiliza una medida de peso, este método tiene una

desventaja y es que no considera que un término pueda aparecer en varios documentos, para solucionar este problema se utiliza un valor de discriminación que tiene como base la tradicional medida de peso del término. El valor de discriminación significa que el término tiene la habilidad de identificarse dentro de un documento dependiendo del contenido (Xiang-Wei, Pi-Lian et al. 2005). La principal ventaja de este modelo es que las frases frecuentes describen mejor un documento en lugar de los términos individuales (Leouski and Croft 1996). La principal desventaja de este modelo es que las frases frecuentes son difíciles de encontrar y filtrar en un documento (J. Stefanowski. and Weiss. 2007).

Como evolución de STC, en la literatura aparecen cuatro (4) algoritmos muy importantes: SHOC, LINGO, CFWS y CFWMS.

6.6.4.1 SHOC

SHOC (Semantic Hierarchical OnLine Clustering) es una extensión del trabajo presentado en STC (Oren and Oren 1998), combina dos técnicas que son las palabras claves y el Clustering ortogonal (basado en LSI), logrando así clusters razonables y entendibles. Ventajas de SHOC: Multilenguaje, es decir no solamente es para ingles; las búsquedas se hacen más fácil y rápidamente gracias al algoritmo de Clustering. Mejora la calidad de los clusters. Es útil en la construcción de etiquetas de los clusters generados. SHOC no es apropiado para colecciones de documentos con alta dimensión ya que debido a que es online la eficiencia del algoritmo puede reducirse (Zhang and Dong 2004).

6.6.4.2 LINGO

En 2003 aparece Lingo, un novedoso algoritmo especialmente adecuado para resolver el problema de búsqueda de resultados con Clustering. A diferencia de la mayoría de los algoritmos, primero trata de descubrir los nombres descriptivos para los futuros clusters y solo entonces procede a asignar cada cluster con su respectivo documento. Este proceso invertido, comparado con

otros algoritmos de Clustering de búsqueda de resultados, permite que Lingo evite parcialmente la trampa de los clusters verbalmente inexplicables (Osiński 2003). Ventajas de Lingo (Osiński 2003): Las etiquetas de los cluster son legibles y las descripciones pueden ser fácilmente entendidas por los usuarios, las etiquetas de los cluster son diversas, permite el solapamiento de clusters (colocar el mismo documento en un gran número de clusters incrementa las posibilidades de que, viendo solo algunos grupos, el usuario será capaz de identificar todos los documentos relevantes), independencia del lenguaje (esto evita las frases o términos sin sentido, como las palabras vacías), una versión simple del algoritmo de Clustering se puede usar para todo tipo de consultas las cuales pueden resultar mucho más conveniente para los usuarios. Desventajas de Lingo (Osiński 2003): Al no definir una estructura jerárquica en los grupos puede resultar en algunos clusters demasiado grandes o generales, número fijo de clusters (el número de clusters resultantes es muy limitado, depende del número actual de temas presentes en la colección de entrada), el número de etiquetas producido depende del valor del umbral de cluster candidatos (si una colección que contiene muchos más temas que el número precalculado de etiquetas candidatas puede ocurrir que muchos de los documentos pueden ser clasificados en “otros temas”), LINGO se basa en la descomposición en valores singulares (SVD) que es un algoritmo costoso en tiempo, para LINGO un conjunto de documentos de más de 150 – 200 artículos puede tardar mas de 3 segundos para armar los clusters.

6.6.4.3 CFWS - CFWMS

En 2007 aparecen CFWS (Clustering Frequent Word Sequence) y CFWMS (Clustering Based On Frequent Word Mining Secuences) (Li, Chung et al. 2008). CFWS Solo utiliza las secuencias de palabras frecuentes las cuales pueden representar adecuadamente el contenido de un documento (O. Zamir., O. Etzioni. et al. 1997), por lo tanto la dimensión de los documentos es reducida drásticamente. CFWS no tiene en cuenta el orden de las palabras, sino el número de veces que se encuentra una frase dentro de un documento

(Li, Chung et al. 2008). CFWMS este algoritmo utiliza el significado de las palabras para referirse al concepto léxico que una palabra puede usar para expresar una idea, debido a que los significados son mejores que las formas de la palabra en términos de representación de los temas de los documentos. Este algoritmo utiliza el método de frases frecuentes y además de eso le adiciona el significado de la frase mediante sinónimos, hipónimos e hipernonimos basado en Wordnet. Ya que CFWMS hace uso de sinónimos se puede encontrar la misma frase en diferentes documentos permitiendo una mejor cobertura de estos mismos. Una ventaja de CFWMS es que hace uso de sinónimos se puede encontrar la misma frase en diferentes documentos permitiendo una mejor cobertura de estos mismos. Este algoritmo presenta una desventaja y es que se debe conocer el orden de las palabras y el numero k de clusters.

6.6.5 Reglas de Asociación

Partiendo del concepto básico de las reglas de asociación (Conjuntos de Ítems Frecuentes, y en el marco de recuperación de información como Términos Frecuentes) se propone otro modelo de representación de documentos para el Clustering de documentos web, los Conjuntos de Términos Frecuentes, que son conjuntos de términos que aparecen en mayor proporción (con mayor frecuencia) en la colección de documentos (Beil, Ester et al. 2007). En este modelo se representa el documento como una transacción en una base de datos (Fung, Wang et al. 2003). Las ventajas de este modelo son: Reduce de forma natural la alta dimensionalidad del documento (Fung, Wang et al. 2003), da una alta precisión en el Clustering, y es robusto aun cuando se aplica a un gran conjunto de documentos (Beil, Ester et al. 2007). La desventaja principal de este modelo es que si existen en la colección términos que no son frecuentes pero si determinantes en la creación de los grupos, estos términos no se tienen en cuenta, disminuyendo así la calidad de los resultados entregados a los usuarios.

6.6.6 Hibridos

En un intento de optimización de los resultados de Clustering se han hecho unas propuestas tales como las mezclas, como por ejemplo, el híbrido entre los métodos particionales y jerárquicos. En (Sasaki and Shinnou 2004) se propone (PDDP – Principal Direction Divisive Partitioning), este método construye un cluster jerárquico para los conjuntos de datos y emplea un Clustering particional basado en el análisis principal del componente. Este algoritmo tiene la ventaja de que acepta alta dimensionalidad de los datos, cosa que no tienen la mayoría de los algoritmos y comparado con otras técnicas, como ejemplo LSI, tiene la ventaja de ser un algoritmo de baja complejidad, pero tiene la desventaja de no seleccionar efectivamente el cluster con el mayor valor (Tasoulis and Tasoulis 2008).

6.6.7 Bisecting Kmeans

En el 2000 se da a conocer el algoritmo *Bisecting K-means* (Steinbach, Karypis et al. 2000), un enfoque superior al método K-means básico en términos de precisión y eficiencia. En este algoritmo, inicialmente la base de datos entera es tratada como un cluster. Basado en una regla, selecciona un grupo para dividirlo en dos usando el algoritmo k-means básico. Este paso de bisección se repite hasta que se obtiene un número deseado de clusters. El algoritmo Bisecting K-means presenta las siguientes desventajas: no da una descripción a cada cluster, no tiene pasos para reducir la alta dimensionalidad de los documentos de texto durante el agrupamiento y el usuario debe especificar el número deseado de grupos antes del proceso de agrupamiento (Li, Chung et al. 2008). Por otro lado, los resultados son muy superiores a los reportados por UPGMA y k-means (Steinbach, Karypis et al. 2000).

Primero se debe hacer una introducción del algoritmo K-Means tradicional. K-Means es un algoritmo particional que realiza Clustering en tiempo de complejidad lineal, O(n). El criterio que se usa para destacar la convergencia y

caracterizar bien un cluster se define como la suma del error cuadrado y se calcula con la fórmula (1).

$$E(A,C) = \sum_{i=1}^{K}\sum_{j=1}^{n}(a_{ij})D^2(c_i, d_j) \quad \textbf{(1)}$$

Donde, A es la matriz de asignación y C = $\{c_i\}$ es el conjunto de K centros de clusters. La matriz de asignación se obtiene asignando a cada punto de datos al centro de cluster asociado. Esta matriz incluye valores entre 0 y 1 (fuzzy sets) o de 0 y 1 (crisp sets, sin solapamiento).

K-Means es un algoritmo que optimiza y busca la mejor solución a partir de una solución inicial (Mahdavi, Chehreghani et al. 2008). K-Means tiene algunos inconvenientes, uno de ellos es la sensibilidad ante la selección de la partición inicial, de modo que normalmente converge a un óptimo local, lo cual en muchos casos es inaceptable.

Por otro lado, Bisceting K-means, es un algoritmo jerárquico basado en el algoritmo K-Means. El algoritmo comienza con un solo cluster de los datos y realiza los pasos que se muestra a continuación:

1. Elegir cluster a dividir.
2. Encontrar dos sub-cluster utilizando el algoritmo básico K-Means. (Paso de bisección)
3. Repetir el paso 2, el paso de bisección, y tomar la división que produce el Clustering con la mayor similitud general.
4. Repetir los pasos 1, 2 y 3 hasta que se encuentre el número deseado de clusters.

Pasos del Bisecting K-means (Tomado de (Capparelli 2004))

Este algoritmo tiende a producir clusters con un tamaño relativamente uniforme. Bisecting K-Means tiene un tiempo de complejidad lineal en relación con el número de documentos que se procesan. Si el número de clusters es grande y sí no se hace refinamiento, entonces el Bisecting K-Means es aún más eficiente que el algoritmo K-Means regular. (Steinbach, Karypis et al. 2000).

6.7 Modelo Espacio Vectorial y Clustering de Documentos

En el modelo espacio vectorial cada documento, *d*, es considerado como un vector *d* (conjunto de "palabras"). En su forma más simple, cada documento es representado por el vector $d_{tf} = (tf_1, tf_2, \ldots, tf_n)$, conocido como TF.

Donde tf_i es la frecuencia del i-esimo término en el documento. Normalmente las palabras muy comunes (palabras vacías o stop words) se desechan por completo y las diferentes formas de una palabra se reducen a una forma canónica (misma raíz, conocido como el proceso de stemming). Por último, para dar cuenta de los documentos de longitudes diferentes, cada vector documento es normalizado de modo que sea de una misma unidad de longitud. Adicionalmente, el vector que representa el documento puede contemplar el peso relativo del término en la colección, conocido como IDF.

La similitud entre dos documentos se debe si un algoritmo de Clustering se va a utilizar, o si se necesita encontrar la similitud entre un documento y una consulta de usuario, entre otros motivos. Hay diversas formas para calcular la similitud entre documentos, pero la más común en RI es la medida del coseno, ver fórmula (2).

$$Cosine(d1, d2) = (d1 \bullet d2) / ||d1|| \, ||d2|| \quad \textbf{(2)}$$

Donde • indica productor escalar y $||d||$ es la longitud del vector, o su norma.

Dado un conjunto S de documentos y su correspondiente representación vectorial, se define el centroide del vector *c*, como se muestra en la fórmula (3).

$$c = \frac{1}{|S|}\sum_{d \in S} d \quad \textbf{(3)}$$

Que es el vector obtenido al promediar los pesos de varios términos presentes en los documentos de S. De manera análoga a los documentos, la similitud entre dos centroides de vectores y entre un documento y un centroide de un vector se calculan utilizando la similtud del coseno. Por ejemplo, calcular el producto entre un documento y un centroide de un cluster es equivalente a calcular la similitud promedio entre ese documento y todos los documentos que integran el cluster de centroides representados.

6.8 Evaluación de Calidad de Agrupamiento

Para el Clustering, dos medidas que expresan la calidad de una agrupación o clusters son usadas. Un tipo de medida permite comparar los diferentes conjuntos de cluster sin hacer referencia al conocimiento externo y es llamada una medida de calidad interna. Otra forma de medir la calidad del cluster se denomina medida de similitud global y se basa en la similitud de un par de documentos en un cluster. El otro tipo de medidas permiten evaluar que tan bien el Clustering esta trabajando, mediante la comparación de grupos producidos por las técnicas de Clustering conocidas; este tipo de medida es llamada una medida de calidad externa. Una medida de calidad externa es la entropía, la cual provee una medida de “lo mejor” para los cluster no anidados o para los cluster de un nivel de Clustering jerárquico (Steinbach, Karypis et al. 2000).

6.8.1 Entropía

Es un tipo de medida externa, se utiliza como medida de calidad de clusters, haciendo la salvedad de que la mejor entropía se alcanza cuando cada grupo contiene exactamente un punto de datos, para cada cluster, se calcula en primera instancia la distribución de las clases de datos, por ejemplo, en un

cluster j se estima p_{ij} la "probabilidad" de que un miembro del cluster *j* pertenezca a la clase *i*. Por tanto usando esta distribución de clases, la entropía de cada cluster *j* se calcula usando la fórmula estándar (Steinbach, Karypis et al. 2000):

$$E_j = -\sum_i \ p_{ij} \log(p_{ij}) \quad (4)$$

Donde, p_{ij} se define: $\frac{n_r^i}{n_r}$ donde n_r es el tamaño del cluster y n_r^i es el número de documentos de la i-ésimo clase que fueron asignados al *r-ésimo* cluster (Baroni, Evert et al. 2008).

Donde la suma se toma sobre todas las clases. La entropía total para un conjunto de clusters se calcula como la suma de las entropías de cada cluster ponderado por el tamaño de cada cluster:

$$E_{CS} = \sum_{j=1}^{m} \frac{n_{j*E_j}}{n}$$

Donde n_j es el tamaño del cluster *j, m* es el número de clusters, y *n* es el número total de puntos de datos (Steinbach, Karypis et al. 2000).

La solución del cluster es prefecta si el cluster contiene solamente palabras de una sola clase, en ese caso la entropía del cluster es cero. Valores pequeños de entropía indica mejores soluciones de Clustering (Baroni, Evert et al. 2008).

6.8.2 Pureza

Es otra medida de calidad externa, usando las mismas definiciones matemáticas, la pureza de un cluster se define como (Baroni, Evert et al. 2008):

Pu(S_r)=$\frac{1}{n_r}\max\,(i)n_r^i$, Tomado de (Baroni, Evert et al. 2008).

La pureza da la fracción del tamaño total del cluster y la clase mas grande de palabras que representa a ese grupo. La pureza del Clustering solución es entonces la suma ponderada de las purezas de los clusters individuales (Baroni, Evert et al. 2008), así:

$pureza = \sum_{r=1}^{k} \frac{n_r}{n} Pu(s_r)$, tomado de (Baroni, Evert et al. 2008)

Grandes valores de pureza indica el mejor Clustering solución (Baroni, Evert et al. 2008).

6.8.3 Precisión

Es la fracción de documentos recuperados que son relevantes en la búsqueda:

Como primera medida se asume que hay un conjunto D de documentos, y hay un conjunto A de documentos que fueron retornados como resultado a una consulta de usuario.

$precision = \frac{|R_A|}{|A|}$ tomado de (Osiński 2003)

Donde, R_A es la intersección de R y A, se denota R al conjunto de todos los documentos en D que son relevantes para la consulta.

6.8.4 Recuerdo

En RI es la fracción de los documentos que son relevantes para la consulta y que son recuperados satisfactoriamente, teniendo en cuenta la anterior denotación se tiene:

$recall = \frac{|R_A|}{|R|}$, (Osiński 2003).

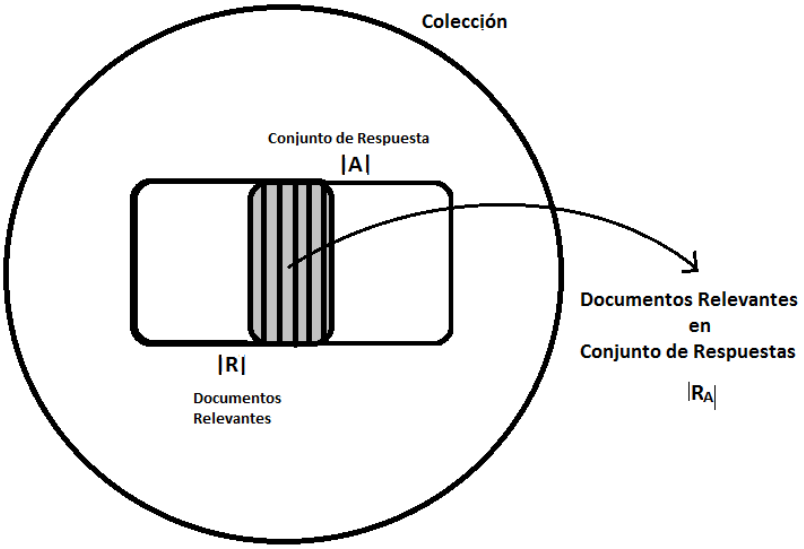


Figura 3. Precisión y Recuerdo en un ejemplo de petición de Información. (Adaptado de (Baeza-Yates and Ribeiro-Neto 1999)).

6.8.5 Medida F

Es una medida que combina la precisión y el recuerdo. Se trata cada cluster como si fuera el resultado de una consulta y cada clase como si fuera el conjunto deseado de documentos para una consulta. Luego se calcula el recuerdo (recall) y precisión de cada cluster para cada clase dada. Específicamente para el cluster j y la clase i.

Teniendo en cuenta que (Steinbach, Karypis et al. 2000):

$$precision = precision\ (i,j) = \frac{n_{ij}}{n_i}$$

$$recuerdo = recuerdo(i,j) = \frac{n_{ij}}{n_j}$$

Donde n_{ij} es el numero de miembros de la clase *i* en el cluster *j*, n_j es el número de miembros del cluster *j.* y n_i es el número de miembros de la clase *i (Steinbach, Karypis et al. 2000)*.

La medida F del cluster j y la clase i viene dada por:

$$F(i,j) = \frac{2}{\frac{1}{recuerdo(i,j)} + \frac{1}{precision(i,j)}}$$

tomado de (Steinbach, Karypis et al. 2000).

La F asume valores en el intervalo de [0,1]. Es 0 cuando no hay documentos relevantes y 1 cuando todos los documentos catalogados son relevantes. Es mas, la medida f toma un valor alto únicamente cuando recuerdo y precisión son altos. Si la medida f toma un valor alto también se puede asumir o se puede interpretar como un intento para alcanzar la mejor comprensión posible entre recuerdo y precisión (Baeza-Yates and Ribeiro-Neto 1999).

Para todo cluster jerárquico la medida F de cualquier clase es el valor maximo que se alcanza en cualquier nodo en el arbol y un valor total para la medida F se calcula tomando el promedio ponderado de todos los valores de la medida F de la siguiente manera.

$$F = \sum_i \frac{n_i}{n} \max\left(F(i,j)\right)$$

Donde max es tomado de los cluster en todos los niveles, y n es el número de documentos.

6.8.6 Similitud general

En ausencia de cualquier información externa, tales como etiquetas de clase, la cohesión de los cluster se puede utilizar como medida de similitud del cluster.

Un método para el cálculo de la cohesión de un cluster es el uso de la similitud ponderada de la similitud interna del cluster, teniendo en cuenta las fórmulas (2) y (3) se tiene:

$$\frac{1}{\left|S\right|^2}\sum_{\substack{d\in S\\ d\in S}}\cos(d',d)$$

Esto es la longitud del cuadrado del centroide del cluster, $\left\|c\right\|^2$.

7 Reglas de Asociación y Clustering de documentos web

El descubrimiento de las reglas de asociación es una tarea exitosa e importante de la minería de datos, tiene como objetivo descubrir todos los patrones frecuentes en las operaciones compuestas de atributos de datos o elementos. Los resultados son presentados en forma de reglas de primer orden entre los diferentes conjuntos de elementos, junto con las métricas como las probabilidades conjuntas y condicionales del antecedente y consecuente, para juzgar la importancia de la regla (Zaki 1999).

La tarea de minería de reglas de asociación se puede describir así: Sea L = {1,2, … , m} un conjunto de Ítems, y sea τ = {1,2, … , n} un conjunto de identificadores de transacción. La base de datos de entrada es una relación binaria $\delta \subseteq L \times \tau$ si un ítem i se presenta en una transacción t, se denota como (i, t) $\in \delta$. Generalmente la base de datos está dispuesta como un conjunto de transacciones, donde cada transacción contiene un conjunto de ítems. En la Figura 3 se muestra un ejemplo clásico de conjunto de ítems y transacciones.

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

BASE DE DATOS DE ITEMS

Jane Austen	Agatha Christie	Sir Arthur Conan Doyle	Mark Twain	P. G. Wodehouse
A	C	D	T	W

BASE DE DATOS

Transaccion	Items
1	A C T W
2	C D W
3	A C T W
4	A C D W
5	A C D T W
6	C D T

ITEMSETS FRECUENTES

SOPORTE MINIMO = 50%

Soporte	Itemsets
100% (6)	C
83% (5)	W, CW
67% (4)	A, D, T, AC, AW CD, CT, ACW
50% (3)	AT, DW, TW, ACT, ATW CDW, CTW, ACTW

Figura 4. Generación de Itemsets Frecuentes (Tomado de (Zaki 1999))

La Figura 4 se tiene un conjunto L = {A, C, D, T, W} y τ = {1, 2, 3, 4, 5, 6}; un conjunto $X \subseteq$ L es llamado *Itemset,* y un conjunto $Y \subseteq \tau$ es llamado un *tidset* (Conjunto de identificadores de transacción) (Zaki 1999). El soporte de un itemset *X* que pertenece a τ, el cual se denota como $\sigma(X)$, es la relación entre el número de transacciones (en τ) que contienen a *X* y el número total de transacciones en τ (Shen 1999) .

Una regla de asociación es una expresión $A \xrightarrow{\rho} B$, en donde A y B son itemsets, y $A \cap B = \phi$. El soporte de una regla viene dado por $\sigma(A \cup B)$ y la confianza de una regla de asociación se denota como $\rho = \frac{\sigma(AUB)}{\sigma(A)}$ y es la probabilidad de encontrar a B en una transacción dado que ya contiene a A. Una regla de asociación es confiable sí, su confianza ρ es más grande o igual a la mínima confianza especificada por el usuario (Zaki 1999).

La minería de reglas de asociación consiste en dos pasos generales que son:

- *Encontrar todos los itemsets frecuentes*: Este es un paso computacionalmente costoso, ya que se debe tener en cuenta todas las posibles combinaciones de las transacciones. Un Itemset es frecuente si su soporte es más grande o igual qué el mínimo soporte especificado por el usuario (Zaki 1999).

- *Generar reglas altamente confiables*: Este proceso es relativamente directo, y se hace para todos los itemsets frecuentes encontrados. Se hace necesario acceder a los soportes de los subconjuntos de la regla encontrada para generar reglas de ella. Para obtener las reglas se necesita examinar cada itemset frecuente y repetir el proceso de generación de la regla. Las reglas obtenidas serán aquellas reglas que cumplan con la mínima confianza establecida (Zaki 1999).

Teniendo en cuenta que los itemsets frecuentes han sido usados muy comúnmente en el proceso de encontrar reglas de asociación (Arunasalam, Chawla et al. 2009), es preciso mencionar que estas reglas tienen dos elementos importantes que son: el soporte y la confianza. Donde el soporte es la proporción de términos sobre documentos (Jiang-Chun, Jun-Yi et al. 2004) y la confianza puede interpretarse como un estimador de probabilidad, la probabilidad de encontrar la parte derecha de una regla condicionada a que también se encuentre la parte izquierda. La confianza determina cuando una regla es aplicable a un conjunto de datos (Arunasalam, Chawla et al. 2009).

Entre los algoritmos más conocidos para encontrar reglas de asociación se tiene Apriori y FP-Growth (Li 2008).

7.1 A priori

En este algoritmo al ingresar el valor del soporte para los itemsets se eliminan aquellos candidatos que tienen sub-conjuntos no frecuentes, esta optimización se puede realizar ya que la búsqueda que realiza el algoritmo BFS, es búsqueda en anchura por sus siglas en ingles (BFS - Breadth First Search), el cual intuitivamente inicia la búsqueda desde un elemento raíz y se exploran todos los vecinos de este nodo; así se hace para cada uno de los vecinos adyacentes y así hasta recorrer todos los elementos de la estructura asegura que el valor del soporte de todos los sub-conjuntos de un candidato sean conocidos con anterioridad (Hipp, Güntzer et al. 2000).

Apriori reduce el costo computacional de búsqueda de reglas de asociación, por lo que reduce el número de conjuntos, los cuales cumplen con la condición de tener un soporte menor o igual al soporte mínimo al generarse todas las reglas de asociación.

En el Clustering basado en Conjuntos de Términos Frecuentes (reglas de asociación). Los siguientes algoritmos son los más representativos: Frequent Term Based Text Clustering, Frequent Itemsets Based Hierarchical Clustering (FIHC), Hierarchical Frequent Term Based Clustering (HFTC), Frequent Term Set Based Clustering (FTSC) y Frequent Term Set Based Hierarchical Clustering (FTSHC).

Específicamente en el Clustering de documentos web basado en conceptos de reglas de asociación, en 2002 se propone FTC (Frequent Term Based Clustering) y HFTC (Hierarchical Frequent Term Based Clustering).

7.2 FTC

FTC funciona con un modelo bottom-up. Determina un cluster pequeño, empieza con un conjunto vacío para después seleccionar un elemento más del

conjunto de términos frecuentes hasta que la base de datos sea cubierta en su totalidad y todos sus conjuntos frecuentes sean escogidos de manera que se elimina elemento a elemento de la base de datos hasta dejarla vacía (Beil, Ester et al. 2007). Una ventaja del algoritmo FTC es que la etiqueta asignada al cluster viene dada por el conjunto frecuente de palabras que se repite más veces en los documentos de cada cluster (Li, Chung et al. 2008).

7.3 HTFC

El algoritmo HTFC se basa en el algoritmo FTC pero se aplica de una forma jerárquica empezando con un conjunto de términos vacíos en la raíz cubriendo toda la base de datos (Beil, Ester et al. 2007). Ventajas: Cubre todos los itemsets y eficiencia en la creación del cluster (Beil, Ester et al. 2007). Desventaja: Debido al resultado jerárquico en el primer nivel podrán existir muchos cluster y como resultado se tendrán documentos del mismo tipo distribuidos en diferentes ramas de la jerarquía y de ocurrir esto la eficiencia total del Clustering decrece (Fung, Wang et al. 2003).

7.4 FICH

En 2004 se propone FIHC (Fung, Wang et al. 2003) que realiza el Clustering basado en la idea de que hay itemsets frecuentes para cada cluster en el conjunto de documentos y que diferentes clusters comparten unos pocos itemsets frecuentes. Ventajas: Reducción de dimensionalidad, es decir se utiliza sólo los ítems frecuentes que aparecen en alguna fracción mínima de los documentos en el vector de documentos, lo que reduce drásticamente la dimensionalidad del conjunto de documento, lo que hace que se más eficiente y escalable. Clustering de alta precisión, es robusto incluso cuando se aplica a los grandes y complejos conjuntos de documentos. Número de clusters es un parámetro de entrada opcional. La desventaja de este algoritmo es que en una jerarquía plana o sea no tan profunda un tema padre puede contener muchos subtemas lo que hace que se dificulte para el usuario la localización de su objetivo empleando demasiado tiempo (Fung, Wang et al. 2003).

7.5 FTSC

Para el 2005 surge FTSC (Frequent Term Set Based Clustering) (Xiang-Wei, Pi-Lian et al. 2005) es un procedimiento down-top. Consiste en escoger elementos de los conjuntos de términos frecuentes y colocarlos dentro de un conjunto vacío, el algoritmo termina cuando todos los elementos que cumplan con una condición han sido escogidos. Las ventajas que presenta este algoritmo son: reduce la dimensionalidad de los datos de texto de manera eficiente, por lo que puede mejorar la tasa de precisión y velocidad del algoritmo de Clustering (Xiang-Wei, Pi-Lian et al. 2005). La principal desventaja es que este algoritmo se basa en todos los itemset frecuentes, los cuales heredan los problemas de la minería de itemset frecuentes en sí, esto es, el conjunto de itemset frecuentes es demasiado grande para un uso efectivo [43].

7.6 FTSHC

En 2005 se propone FTSHC (Frequent Term Set Based Hierarchical Clustering Algorithm) (Xiang-Wei, Pi-Lian et al. 2005), este algoritmo es una modificación del FTSC ya que el FTSC produce un solapamiento de los clusters sino se remueve los itemsets frecuentes de la base de datos, para solucionar este problema FTSHC utiliza dos niveles para el cluster en lugar de uno y los documentos no se eliminan de la base de datos. FTSC y FTSHC decrementan la dimensión de los vectores efectivamente y se obtiene un mejor cluster [27]. Este algoritmo al utilizar apriori reduce su capacidad, puesto que apriori reduce la velocidad al realizar muchas pasadas o recorridos en la base de datos.

En 2008 Monika Akbar propone un algoritmo de Clustering (Akbar 2008) el cual consiste en sacar las palabras claves de los documentos haciendo uso de WordNet para luego trasformar esas palabras clave en grafos y utilizando el algoritmo FP-Growth obtener los subgrafos frecuentes, se debe aclarar que en (Akbar 2008) se hace una modificación al algoritmo FP-Gowth para generar los subgrafos frecuentes. En esta propuesta se realiza el clustering final haciendo

uso de una técnica Jerárquica Aglomerativa, la cual presenta una desventaja y es que los objetos pueden ser agrupados incorrectamente de manera que el resultado debe ser revisado para asegurarse de que este bien hecho. Además esta técnica hace uso de varias medidas de distancia entre clusters, lo que genera diferentes resultados, que son difíciles de comparar y en especial de seleccionar el mejor.

Estudios de desempeño muestran, que FP-Growth es más rápido que Apriori, por las siguientes razones: No hay generación de candidatos ni prueba de candidatos (El número total de candidatos puede ser muy amplio y una transacción puede contener muchos candidatos, lo que castiga en gran medida el rendimiento de Apriori). Usa una estructura de datos compacta. Elimina escaneo repetido a la base de datos. La operación básica es contar y construir el árbol FP.

Teniendo en cuenta los trabajos anteriormente propuestos, se decide hacer un modelo que hace búsqueda de Conjuntos de Términos Frecuentes dentro de las oraciones de los documentos para así formar una Matriz de Términos Frecuentes por Oraciones de Documentos (TF-OD) usando FpGrowth y luego realizar el Clustering de los documentos con una versión modificada de Bisecting K-Means que determine en forma automática el número apropiado de grupos y asigne etiquetas apropiadas a los grupos, finalmente definir la precisión del algoritmo con el objetivo de que pueda ser comparado frente a otros del área.

Para realizar el clustering de documentos web es importante tener en cuenta algunos procedimientos, por ejemplo extracción de las características en los conjuntos de términos, para realizar este procedimiento se tiene en cuenta la diferenciación de documentos, para esto se utiliza un valor de discriminación, que en este caso pueden ser las palabras claves (keyWords) las cuales son muy útiles a la hora de encontrar el tema central de los documentos (Liu and

He 2005). En el presente proyecto, se utilizó el algoritmo FP-Growth para encontrar los itemsets frecuentes (términos que co-ocurren en los documentos) eficientemente.

8 WordNet

Recurso lingüístico ideado para uso automático. Incorpora información psicolingüística. Organizado en base a significados (thesaurus). WordNet divide el lexicón en cinco categorías: nombres, verbos, adjetivos, adverbios, partículas Evidentemente hay formas que pueden estar en más de una. Por ejemplo, close, puede ser nombre, verbo, adjetivo y adverbio (Beckwith, Fellbaum et al. 1993).

Una palabra se define como una asociación convencional entre un concepto lexicalizado y un lexema que desempeña un papel sintáctico, Otra definición de palabra, más matemática, y por tanto más adecuada para una representación formal puede servir para aclarar los conceptos:

palabra = <forma,significado>

La forma puede ser simple o múltiple (colocación). De manera que se separa el aspecto de la palabra (*Word form*) de lo que significa la palabra (*word meaning*). Según esta definición WordNet es un sistema de representación que combina las formas y los significados.

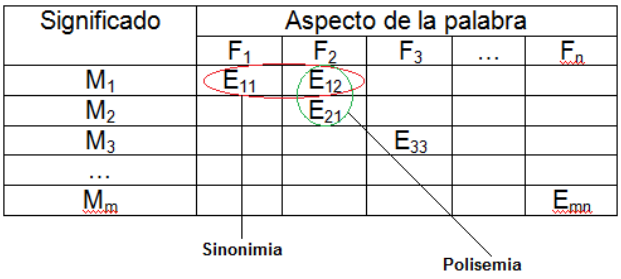


Figura 4. Adaptada de (Beckwith, Fellbaum et al. 1993)

WordNet está organizado en base a relaciones. Dado que los significados se representan mediante synsets, las relaciones semánticas se pueden representar mediante enlaces entre synsets.

Las relaciones más importantes contempladas en WordNet son: Sinonimia, antonimia, hiponimia/hiperonimia, holonimia/meronimia, morfológica.

Significado	Aspecto de la Palabra			
	Suelo	Piso	Territorio	Planta
M_1	E_{11}	E_{12}		
M_2	E_{21}		E_{23}	
M_3	E_{31}			
M_4	E_{41}			
M_5		E_{52}		E_{54}

Synsets (synonym sets):

M1={suelo, piso}
M2={suelo, territorio}
M3={suelo, (superficie inferior de algunas cosas; p.e.,de las vasijas.)}
M5={suelo, (modalidad de gimnasia artística)}
M4={piso, planta}

Figura 5. adaptada de (Beckwith, Fellbaum et al. 1993)

8.1 Sinonimia

La sinonimia es la relación mas importante de WordNet, dos expresiones son sinónimas en un contexto C si la sustitución de una por otra en dicho contexto no altera el significado. La definición de los synsets en términos de sustitución hace necesaria la separación en categorías sintácticas (nombres, verbos, adjetivos y adverbios) (Beckwith, Fellbaum et al. 1993).

8.2 Antonimia

La antonimia se refiere a los contrarios, por ejemplo rico/pobre. No siempre la negación de un concepto coincide con su antónimo, por ejemplo no ser rico no significa ser pobre (Beckwith, Fellbaum et al. 1993).

8.3 Hiponimia

Se denomina hiponimia la relación de inclusión de un significado respecto de otro. Así, el significado de perro está incluido en el de animal; tulipán en el de flor. Perro, gato, conejo, cabra, vaca son hipónimos de animal; tulipán, rosa, clavel, margarita, son hipónimos de flor (Beckwith, Fellbaum et al. 1993).

8.4 Hiperonimia

Hiperonimia es el fenómeno inverso a la hiponimia; animal es el hiperónimo de perro, gato...; color es hiperónimo de rojo, azul, verde, amarillo; árbol es hiperónimo de pino, roble, castaño (Beckwith, Fellbaum et al. 1993).

8.5 Meronimia

La meronimia es una relación semántica no-simétrica entre los significados de dos palabras dentro del mismo campo semántico. Se denomina merónimo a la palabra cuyo significado constituye una parte del significado total de otra palabra, denominada ésta holónimo. Por ejemplo: dedo es un merónimo de mano y mano es merónimo de brazo (Beckwith, Fellbaum et al. 1993).

8.6 Holonimia

Es una noción semántica que se opone a meronimia, del mismo modo en que se oponen el todo y la parte. Así, por ejemplo, bicicleta es un holónimo mientras que pedal, aro y manubrio son merónimos (Beckwith, Fellbaum et al. 1993).

8.7 Relaciones Morfológicas

Si la palabra árbol está registrada en WordNet es necesario que ante la palabra árboles sea capaz de analizarla morfológicamente para acceder a la información de su forma base (Beckwith, Fellbaum et al. 1993).

Capítulo 3: El Algoritmo Propuesto

En este capítulo se presenta en detalle la descripción del algoritmo propuesto, una técnica nueva para Clustering de Documentos Web basado en una Matriz de Términos Frecuentes por Oraciones de Documentos, FP-Growth y Bisecting k-means.

El nuevo algoritmo propuesto, denominado WDC-PTFBK, es un algoritmo que utiliza una matriz de Términos Frecuentes por Oraciones de Documentos en lugar de la clásica matriz de términos por documentos. Este algoritmo se basa en la extracción de oraciones de los documentos para realizar el proceso de búsqueda, basado en un segmentador de oraciones de documentos y FP-Groth para encontrar los términos frecuentes en dichas oraciones. Los sistemas de clustering inicialmente hacen uso de algoritmos de RI, los cuales convierten los documentos en una matriz de términos por documento (Osinski and Weiss 2005). Luego, haciendo uso de una modificación de Bisecting K-Means y el Criterio Bayesiano de la Información, se producen los grupos de los documentos y se generan las etiquetas de los grupos basado en los términos que estadísticamente representan más a cada agrupación o grupo.

WDC-PTFBK tiene una rutina principal que ejecuta los pasos:

```
1. INICIO
2.     Obtener Documentos // Esto se hace de los Buscadores Web o de la BD Local.
3.     SI (Obtener Documentos) entonces
4.         PARA Cada Documento en Documentos hacer
5.             Segmentar Documento en Oraciones.
6.             Adicionar Oraciones a Matriz de Oraciones por Documentos
7.             //Cada Documento tiene su ID y que cada oracion tiene un ID asociado al Documento que pertenece//
8.             Retornar MAtriz de Oraciones x Documentos (MOxD).
9.         Fin_PARA
10.        PARA cada Oracion en la Matriz MOxD hacer
11.            Eliminar Stop Words. // Palabras Vacias
12.            Realizar proceso de Stemming //Reducir palabras a su raiz o lema.
13.            Ejecutar Lucene para Obtener Matriz de Terminos por Oraciones de Documento (MTxOD)
14.            Retornar MTxOD.
15.        Fin_PARA
16.        Calcular Frecuencias Observadas para cada término en MTxOD
17.            contar cuantas veces se repite cada termino dentro de las oraciones de Documentos
18.        Llamado a Procedimiento FP-GROWTH
```

```
19.     INICIO
20.         Establecer Parametros Reglas de Asociación
21.         Minimo Soporte
22.         Minimo Confianza
23.         Crear MAtriz Terminos Frecuentes x Oraciones de Documento MTFxOD
24.         Extraer Patrones Frecuentes de MTxOD //Paso Exclusivo de FP-Growth
25.         PARA cada Patron Frecuente en Patrones Frecuentes hacer
26.             Comparar termino con cada Patron frecuente
27.             SI termino esta en Patrones Frecuentes entonces
28.                 adicionar Termino a MTFxOD
29.             FIN_SI
30.         Fin_PARA
31.         Retornar MTFxOD
32.     FIN_PROCEDIMIENTO
33.     Realizar Cluster con Bisecting K-means
34.     INICIO
35.         Establecer Parámetros
36.         Definir un numero K inicial.
37.         Establecer Criterio de Parada
38.         Hacer llamado a rutina K-Means
```

```
39.                 Retornar Clusters
40.             FIN_PROCEDIMIENTO
41.             Asignar etiquetas a los clusters
42.             Solapar los clusters.
43.     FIN_SI
44.     SINO Salir
45. FIN.
```

Figura 6. Pasos generales de WDC-PTFBK

9 Pre-procesamiento de Documentos

Inicialmente se realiza la separación del texto del documento en oraciones, para esto se utilizan signos de puntuación que aparecen en los escritos para marcar las pausas necesarias que le den el sentido y el significado adecuado, aunque no hay reglas fijas que nos den el uso correcto de estos signos. Para ésta tarea se hizo uso de una herramienta software llamada Segmenter la cuál facilita la extracción de las frases (oraciones) de forma adecuada. Segmenter se basa en una técnica llamada TextTiling (Hearst 1997), que consiste en subdividir textos en unidades de párrafo las cuales representan sub-temas o pasajes. El algoritmo Segmenter tiene como parámetro de entrada un archivo donde se encuentra los documentos, para así retornar un archivo de texto con las oraciones que contiene un documento. Estas oraciones se almacenan en una matriz denominada Matriz de Oraciones por documento (MOxD). Luego se construye la Matriz de Términos por Oraciones de Documento (MTxOD) utilizando Lucene.NET (http://lucene.apache.org). Esta fase incluye la remoción de las palabras vacias o stop words, aplicación del algoritmo de stemming de Porter, el cuál está disponible en el idioma Inglés, para. El siguiente paso es aplicar el algoritmo FP-Growth para construir la matriz de Términos Frecuentes por Oraciones de Documento (MTFxOD), todo esto para que la dimensionalidad de las colecciones disminuya y la semántica de la matriz aumente.

Para la agrupación de la entrada de datos al algoritmo de clustering de documentos web, se tuvo en cuenta varias herramientas de preprocesamiento de documentos y se determinó cual era la mejor para el contexto y las necesidades del presente proyecto. A continuación se presenta el estudio realizado, la herramienta seleccionada y la utilización de la misma. En la implementación de los mecanismos para la recuperación de información sobre una colección de documentos de texto es necesario conseguir una representación de los mismos, la cual debe tener en cuenta un conjunto de

criterios mediante los cuales se obtienen las frases y las relaciones entre estas. Toda implementación de un Sistema de Recuperación de Información (SRI) comienza con la tarea de pre-procesamiento de la colección. La colección se divide en frases y de estas salen los términos, esto debido a que no todas las frases que componen un documento son igualmente representativas de su contenido. Cuestiones como posición, la cantidad de ocurrencias o su función lingüística, entre otras, definen el grado de importancia de cada uno de los términos. El resultado es una representación de la colección, que es computacionalmente adecuada para los procesos siguientes y que, generalmente, se describe como Indexación de la Colección. El proceso de Indexación se divide principalmente en cinco operaciones de texto como se describe en (Andrade R. and Constain D. 2010)

1. *Análisis lexicográfico:* Se extraen las palabras y se normalizan, con el fin de tratar los dígitos, guiones, signos de puntuación, y el tipo de las letras.

2. *Eliminación de palabras vacías*: o de alta frecuencia en la colección, con el fin de filtrar palabras con valores de discriminación muy bajos para propósitos de recuperación.

3. *Stemming*: Se reducen las palabras morfológicamente parecidas a una base, con la finalidad de aumentar la eficiencia de un SRI, permitiendo la recuperación de documentos que contengan variaciones sintácticas de los términos de la consulta.

4. *Selección de términos a indexar*: Se extraen aquellas palabras simples o compuestas que mejor representan el contenido de los documentos.

5. *Construcción de estructuras de clasificación de términos*: tales como tesauros, o extracción de la estructura directamente representada en el texto, para permitir la expansión de la consulta original con los términos relacionados.

A continuación se presentan en detalle cada una de las fases.

9.1 Analisis lexicografico del texto

Es el proceso de convertir un flujo de caracteres (el texto de los documentos) en un flujo de palabras (las palabras candidatas a ser adoptadas como términos índice) o tokens (por lo que también recibe el nombre de tokenization). Esto implica detectar el comienzo y el fin de las palabras bajo diversas circunstancias (al inicio, en el medio o en el fin de una oración). Alternativamente, el analizador deber ser capaz de tratar con símbolos no alfabéticos como dígitos, caracteres especiales que componen las palabras, los cuales generalmente se reemplazan por el caracter base relacionado. Además, se debe normalizar y expandir siglas y tratar guiones como conectores de términos. Luego de realizar este tratamiento todo el texto es, generalmente, transformado a mayúsculas o minúsculas. Si el archivo a procesar posee marcas de formato también son removidas en esta etapa. Este es el caso de los mensajes de correo electrónico, las páginas HTML, etc. Sin embargo, se puede dar la situación que algún SRI requiera conservar información sobre algún atributo ligado a oraciones o términos para un uso posterior. Algunos atributos que comúnmente se conservan son los títulos de los documentos, autores, títulos y subtítulos de secciones, etc. Otro aspecto importante a tener en cuenta es el tratamiento de los números. Su inclusión en el índice depende del tipo de colección y de la importancia que tengan como posibles términos de búsqueda. Por ejemplo, en una colección con normativas, patentes o histórica (donde hay fechas) resulta importante conservar los términos numéricos. De no ser necesaria esta información, los números son fácilmente eliminados y no aparecen en el índice. Los guiones plantean otra decisión difícil para el analizador, debido a que hay palabras que incluyen guiones como una parte integral. El procedimiento más adecuado es adoptar una regla general y especificar excepciones caso por caso.

9.2 Eliminación de palabras vacías

Las palabras que son demasiado frecuentes entre los documentos en la colección no son buenos discriminantes. De hecho, una palabra que ocurre en el 80% de los documentos en la colección es poco útil para propósitos de recuperación. A tales palabras se les llama stopwords (palabras vacías). La eliminación de palabras vacías reduce considerablemente el tamaño de la estructura de indexación. Es común obtener una compresión en el tamaño de la estructura de indexación del 40% o más solo con la eliminación de las palabras vacías.

9.3 Stemming

Es una técnica de reducción que permite detectar variantes morfológicas de un mismo término y reemplazarlas por el término raíz o lema. Frecuentemente, el usuario cuando realiza una consulta especifica una palabra, pero solo una variante de esta palabra está presente en un documento relevante. Los plurales, los gerundios y los sufijos del tiempo pasado son ejemplos de las variaciones sintácticas que impiden una correspondencia perfecta entre la palabra de una consulta y una palabra respectiva en el documento. El stemming es bastante útil porque mejora el desempeño de la recuperación, puesto que reduce las variantes de una misma palabra raíz a un concepto común. Además, el stemming reduce el tamaño de la estructura de indización porque el número de términos índice distintos se reduce (Porter 1980).

9.4 Selección de términos a indexar

En esta fase se tiene por objetivo analizar cada documento a indexar con el fin de extraer el conjunto de palabras (simples o compuestas) que representen de mejor forma su contenido. Por regla general, las palabras de alta frecuencia se eliminan con la asistencia de un diccionario de palabras vacías y las palabras de baja frecuencia también se eliminan. Los términos que superan el proceso de filtrado componen el vocabulario de la colección.

9.5 Utilización de tesauros

El tesauro es una lista escogida de palabras importantes en un dominio del conocimiento, para cada palabra en esta lista, existe un conjunto de palabras relacionadas, comúnmente derivadas de relaciones de sinonimia y polisemia. Los propósitos principales de un tesauro son básicamente: proporcionar un vocabulario estándar (o sistema de referencia) para la indexación y la búsqueda; ayudar a los usuarios con la ubicación de términos para formular una consulta apropiada y proporcionar jerarquías clasificadas que permitan la ampliación o reducción de la consulta actual de acuerdo a las necesidades del usuario.

Para el pre-procesamiento de documentos se han implementado herramientas que ayudan a agilizar el desarrollo de un proyecto como el presente en el que se construye un algoritmo para el clustering de documentos web, ya que la fase de pre-procesamiento se puede adaptar de una de las herramientas que se presentan a continuación, esto hace posible la reutilización de código fuente ampliamente probado y usado por muchas aplicaciones garantizando un buen producto en esta fase, la cual precede la ejecución del algoritmo propuesto. A continuación se presenta el estudio que se realizó para la selección del algoritmo de pre-procesamiento.

10 Tecnología para el procesamiento de documentos

Para la Recuperación de la información hay varias bibliotecas de código que ofrecen funciones para los procesos de indexación y búsqueda de documentos, muchas de las cuales son Open Source, por lo tanto su código fuente está a disposición de la comunidad y puede ser reutilizado en cualquier aplicación. Entre las bibliotecas más destacadas para la recuperación de la información se encuentran Lemur, Xapian, Terrier y Lucene. A continuación se presenta cada una de ellas.

10.1 Lemur

Permite todas las etapas desde la indexación hasta la búsqueda de documentos. Lemur aporta una API implementada en C++ y está diseñada para trabajar en todos los sistemas operativos, permite la indexación incremental e indexa atributos de los documentos (Ramos H. and Hernandez 2008). Está licenciada bajo la licencia BSD (Berkeley Software Distribution) (Se puede obtener en la web del proyecto: http://www.lemurproject.org/). Proporciona indexadores capaces de leer archivos PDF, HTML y XML (Eckard and Chappelier 2007).

10.2 Xapian

Librería para crear motores de búsqueda, codificada en C++, Perl, Python, PHP, Java, C#, y Ruby. Tiene un API potente y adaptable, enfocado en la recuperación probabilística, que facilita los procesos de indexación y búsqueda (Ramos H. and Hernandez 2008). Está disponible bajo la licencia GPL (General Publical License) (*Sitio web de Xapian*: Disponible de: http://www.xapian.org). Xapian proporciona paquetes software pre-compilados principalmente para las distribuciones de Linux (Eckard and Chappelier 2007).

10.3 Terrier

Implementado en Java, permite indexar una colección de documentos de forma que se sabe cuántos documentos contienen un término determinado. También permite realizar búsquedas (Ramos H. and Hernandez 2008). Terrier ha sido utilizada para búsquedas en la web y búsquedas en empresas, también en aplicaciones de escritorio, en la intranet y en motores de búsqueda específicos, así como para el desarrollo y la evaluación de nuevas técnicas de recuperación textual. Está disponible bajo la licencia Mozilla Free License (*Sitio web de Terrier*: Disponible desde: http://ir.dcs.gla.ac.uk/terrier/).

10.4 Lucene

Lucene es una librería de funciones que permite tanto la indexación como la búsqueda de documentos. Creada con una metodología orientada a objetos e implementada completamente en java. Tiene versiones para otros lenguajes como Perl, Phyton, C#, Ruby y C++. Es multiplataforma, permite la indexación incremental, posee algoritmos de búsquedas confiables, realiza stemming, búsquedas por campos y permite la indexación de documentos con formato TXT, PDF, DOC, RTF, XML, PPT y HTML (Eckard and Chappelier 2007). Está disponible bajo la licencia Apache Software License (*Sitio web de Lucene*: Disponible desde: http://lucene.apache.org/).

10.5 Comparación de bibliotecas de funciones para la recuperación de la información

A continuación (ver Tabla 4- Resumida de (Ramos H. and Hernandez 2008), (Eckard and Chappelier 2007)) se presenta una comparación de las principales bibliotecas de RI, en la cual se tienen en cuenta las características más relevantes y un test de rendimiento (ver Tabla 5- Tomada de (Middleton and Baeza-Yates 2008)) realizado en el trabajo “A Comparison of Open Source Search Engines” (Middleton and Baeza-Yates 2008), en el que se prueban las librerías con varios datasets.

	LUCENE	**TERRIER**	**XAPIAN**	**LEMUR**
Multiplataforma	Si	No	Si	Si
Lenguaje de implementación	Java	Java	C++	C++
Soporte para otros lenguajes	Perl, Phyton, C#, Ruby, C++	No	Perl, Phyton, Php, Tcl, C#, Ruby	Java, C#
Archivos que indexa	Pdf, word, html, htm,	Html, pdf, Word, xls,	Pdf, word, html, htm, txt,	Pdf, word, html, ppt, txt,

	txt, xml, rtf, entre otros	ppt, txt	xml, rtf, entre otros	xml, rtf, entre otros
Stemming para varios idiomas	Si	Si	Si	Si
Búsqueda mientras actualize índice	Si	No	No	No
Indexación incremental	Si	No	No	Si
Modelo de representación	Espacio vectorial	Probabilístico	Probabilístico	Probabilístico
Búsquedas por cualquier campo	Si	No	No	No
Tecnologías OpenSource	Si	Si	Si	Si
Ultima Actualización	Versión: Lucene.Net 2.3.2 Fecha: 24/07/2009	Versión: Terrier 2.2.1 Fecha: 29/01/2009	Versión: Xapian 1.0.16 Fecha: 10/09/2009	Versión: Lemur 4.10.1 Fecha: 28/07/2009

Tabla 3. Comparación de librerías de RI (Adaptado de (Middleton and Baeza-Yates 2008))

	LUCENE	XAPIAN	TERRIER	LEMUR
Máximo en uso de CPU	99.4%	100.0%	99.5%	100.0%
Máximo en uso de RAM	20.0%	26.8%	58.1%	7.3%
Tamaño del índice (con respecto a la colección)	26%	104%	52%	63%

Tabla 4. Comparación del rendimiento de las librerías de RI (Adaptado de (Middleton and Baeza-Yates 2008))

Con base en el estudio realizado sobre las librerías más importantes para realizar indexación y búsqueda, se tomó la decisión de seleccionar Lucene en su implementación para C# llamada Lucene.net, porque en comparación con las otras librerías, posee mejores características: Es la única librería que utiliza como modelo de representación de los documentos el modelo de espacio vectorial, el cual es considerado el más apto para el desarrollo del algoritmo hibrido, tema central del proyecto; en cuanto al uso de CPU y el tamaño del índice creado, reduce la dimensionalidad de los documentos lo que influye en el desempeño del algoritmo de clustering; además Lucene está siendo utilizado por una gran cantidad de aplicaciones tanto de escritorio como aplicaciones web, una lista de estas aplicaciones se puede consultar en (Lucene-java Wiki, *Applications and web applications using Lucene*: Disponible en: http://wiki.apache.org/lucene-java/PoweredBy). A continuación se presenta en detalle la estructura de Lucene.net y se especifican las funciones que se reutilizaron para implementar la etapa de pre-procesamiento que antecede al algoritmo de clustering propuesto.

11 LUCENE.NET

Lucene permite a las aplicaciones tratar con las reglas de negocio específicas al dominio del problema mientras oculta la complejidad de la implementación de la indexación y la búsqueda, siendo un API sencilla de usar. Lucene puede verse como una capa sobre la que se construyen las aplicaciones, Lucene no se preocupa sobre la fuente de los datos, su formato, ni incluso sobre el idioma, mientras que quien haga uso de Lucene pueda convertir esto a texto. Esto significa que Lucene se puede utilizar para indexar y buscar datos almacenados en: páginas web en servidores web remotos, documentos almacenados en sistemas locales de archivos o en bases de datos locales, archivos de texto plano, documentos de Microsoft Word, archivos HTML o pdf, o cualquier otro formato del que se pueda extraer información textual (Hatcher and Gospodnetic 2005).

Las fases de pre-procesamiento de documentos que implementa Lucene son las típicas en esta área, las cuales se definieron previamente en la presentación del proceso de indexación, entre ellas se encuentran: análisis lexicográfico, eliminación de palabras vacías, stemming y utilización de tesauros, de las cuales se puede utilizar independientemente cualquiera de ellas según se requiera.

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

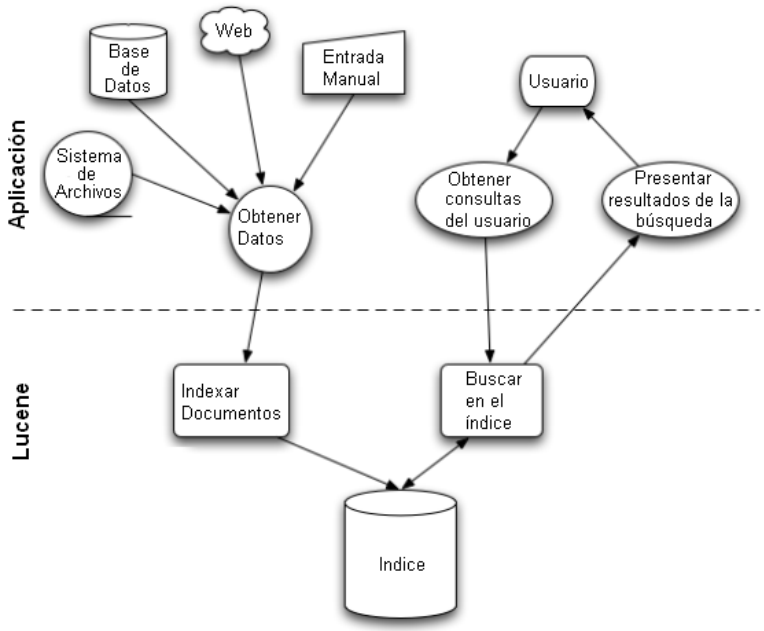


Figura 7. Integración típica de una aplicación con Lucene. (Adaptado de (Hatcher and Gospodnetic 2005))

11.1 Descripción de las principales clases de indexación

1. IndexWriter

Es el componente central del proceso de indexación. Esta clase crea un nuevo índice y adiciona documentos a un índice existente.

2. Directory: Esta clase representa la ubicación del índice de lucene. Es una clase abstracta que permite a sus subclases que almacenen el índice como mejor les parezca. Estas subclases son FSDirectory y RAMDirectory las cuales se definen a continuación:

3. FSDirectory: Subclase que mantiene una lista de archivos reales en el sistema de archivos.

4. RAMDirectoryK: Mantiene todos sus datos en memoria.

5. Analyzer: Antes del texto ser indexado, este pasa a través de un analizador. El analizador especificado en el contructor de la clase IndexWriter, se encarga de extraer los tokens (porción de texto que esta en muchos documentos) para indexarlos y eliminar el resto.

6. Document: Un documento representa una colección de datos. Los campos de un documento representan el documento o meta datos asociados con ese documento. Los meta datos tales como el autor, titulo, tema, fecha de modificación son indexados y almacenados por separado.

7. Field: Cada documento en un índice contiene uno o más campos de nombre, representado en una clase llamado Campo. Cada campo corresponde a una pieza de datos que es consultado o recuperado en el índice durante la búsqueda. Lucene ofrece cuatro tipos diferentes de campos, los cuales se pueden escoger:

8. Keyword: no es analizado, pero se almacena y se indexa en el índice. Este tipo es conveniente para los campos cuyo valor original se conserva en su totalidad, como las direcciones URL, rutas de sistemas de archivos, fechas, nombres de personas, números de seguridad social, números de teléfono.

9. UnIndexed: no es analizado ni indexado, pero su valor se almacena en el índice. Este tipo es conveniente para los campos que necesitan mostrar los resultados de búsqueda (por ejemplo una direccion URL, o una llave primaria de una base de datos), pero cuyos valores no se buscan directamente. Dado que el valor original de un campo de este tipo se almacena en el índice, este tipo no es adecuado para el almacenamiento de los campos con valores muy grandes si el tamaño del índice es importante.

10. UnStored: opuesto a UnIndexed. Este tipo de campo es analizado e indexado pero no almacenado en el índice. Es conveniente para indexar una gran cantidad de texto que no necesita ser recuperado en su forma original, por ejemplo: los cuerpos de las páginas web u otro tipo de documento de texto.

11. Text: es analizado e indexado. Esto implica que los campos de este tipo pueden ser buscados pero siendo cuidadosos con el tamaño del campo. Si los datos indexados son una cadena, se almacenan, pero si los datos vienen de un reader (no han sido tratados), no se almacenan. En la tabla 5 se presenta un resumen de las características de los diferentes campos, mostrando como los campos se crean.

Field method/type	Analyzed	Indexed	Stored	Example usage
Field.Keyword(String, String) Field.Keyword(String, Date)		Si	Si	Telefonos y numeros de seguridad social, URLs, nombres personales y fechas
Field.UnIndexed(String, String)	No	No	Si	Documentos de tipo (PDF, Html) si no se usa un criterio de busqueda
Field.UnStored(String, String)	Si	Si	No	Titulos de documentos y contenido
Field.Text(String, String)	Si	Si	Si	Titulos de documentos y contenido
Field.Text(String, Reader)	Si	Si	No	Titulos de documentos y contenido

Tabla 5. Visión General de los diferentes tipos de campos, sus caracteristicas y su uso. (Adaptado de (Hatcher and Gospodnetic 2005))

Con el fin de implementar la funcionalidad básica de búsqueda, se necesita estar familiarizado con un pequeño conjunto de clases de búsqueda de Lucene.

11.2 Descripción de las principales clases de búsqueda

La interfaz de búsqueda básica que proporciona Lucene es tan sencilla como el de la indexación. Solo unas pocas clases se necesitan para llevar a cabo la operación de búsqueda:

1. IndexSearcher: IndexSearcher es a la búsqueda como IndexWriter es a la indexación. Es una clase que abre un índice en un modo de solo lectura.

2. Term: Es la unidad básica para la búsqueda. Al igual que el objeto de campo, consiste en un par de par de elementos de cadena: el nombre del campo y el valor de ese campo.

3. Query: Lucene viene con una serie de subclases de consulta concretas: TermQuery, BooleanQuery, PhraseQuery, PrefixQuery, PhrasePrefixQuery, RangeQuery, FilteredQuery y SpanQuery. Query es la clase padre abstracta.

4. TermQuery: Es el tipo de Query básico soportado por Lucene y es uno de los tipos de Query primitivos. Es usado para encontrar documentos que contienen campos con valores específicos.

5. Hits: Es un simple contenedor de punteros para la clasificación de los resultados de búsqueda. Por razones de rendimiento, las instancias Hits no se cargan en el índice de todos los documentos que coinciden con una consulta, solo una pequeña parte de ellos a la vez.

12 ALGORITMO PARA CLUSTERING

12.1 Construcción de la Matriz Términos Por Oraciones de Documento

Como ya se ha hecho un proceso previo, que es la extracción de las oraciones que se encuentran dentro de un documento, y luego de tener los documentos pre-procesados se crea una matriz de términos frecuentes por oraciones de documento, ya que el modelo de representación tiene presente sólo los itemsets (conjunto de palabras) frecuentes de los documentos.

A continuación describimos los pasos realizados para crear la matriz de términos frecuentes por oraciones de documentos.

1. Crear índice: El índice se crea utilizando las fuentes de datos definidas como Reuter 21578, o la web, dependiendo del caso. Posteriormente se crea el índice “Crear el Índice” y se indexan los documentos “Indexar los documentos”.
2. Obtener los términos de todos los documentos: Luego “Consultar en el índice” de Lucene y se obtiene los términos de las oraciones de los documentos.
3. Obtener lista de Itemsets (FP-Growth): Los términos de todas las oraciones de los documentos consultados en el paso anterior se utilizan para obtener los itemsets frecuentes, que se obtienen después de invocar una función que extrae patrones frecuentes presentes en la lista de términos, esto se lleva a cabo utilizando el algoritmo FP-Growth, cuya implementación se tomo del código fuente disponible en Internet en CodePlex Open Source Community en la sección Frequent Pattern Miner.
4. Crear la Matriz de términos frecuentes por oraciones de Documento: Primero se crea la matriz con el registro de la frecuencia observada de cada término en cada oración por documento, luego se calcula la frecuencia más alta observada en cada documento y se multiplican para

obtener lo que se conoce como TF, luego se calcula la importancia relativa de cada uno de los términos de la colección o IDF como el Log ($N/ n_i +1$), donde N es el número de documentos y n_i es el número de documentos donde aparece el término i. La matriz de términos por oraciones de documentos en cada una de sus celdas corresponde al peso representado como TF-IDF, a diferencia que en esta matriz también se utilizan los itemsets o conjunto de términos frecuentes en las oraciones de los documentos.

12.2 Construcción de la matriz de CDM

Una matriz CDM es una matriz de Conceptos por Documento que se obtiene de la matriz de términos frecuentes por oraciones de documento, la cual busca que el Clustering sea más preciso ya que según la literatura las uniones significativas asocian varios términos lo cual lleva a realizar un Clustering mas compactos. La matriz CDM se construye asi:

1. Crear índice: Se utiliza una fuente de datos definida por ejemplo Reuter 21578 o la web. Luego se obtiene la matriz de oraciones por Documento. Como paso siguiente se indiza los documentos con los índices creados anteriormente, esto se hace con la herramienta Lucene.
2. Obtener los términos de todos los documentos: “Consulta el índice” de Lucene y así obtener los términos de las Oraciones de documentos.
3. Proceso WordNet: este proceso sigue los siguientes pasos:
   3.1. Lista de Conceptos Candidatos: Cada término que se encuentra en la lista de términos por oraciones de documento, que se obtiene en el paso anterior, se consulta en la ontología WordNet, donde por cada término se obtiene un conjunto de

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

synsets determinado por su modo sustantivo y verbo, cada synset se ordena por el número de frecuencia de uso.

La figura 6 muestra que cada synset se compone de:

- Lista de Hipónimos: Conjunto de significados inmersos en los significados del conjunto de Sinónimos.
- Lista de Sinónimos: Conjunto de significados que son sinónimos del término consultado.
- Profundidad D: Se calcula únicamente para el synset 1, el cual da la profundidad del concepto en su modo sustantivo o verbo.

El índice es la posición del término en la lista de términos por oraciones de documento.

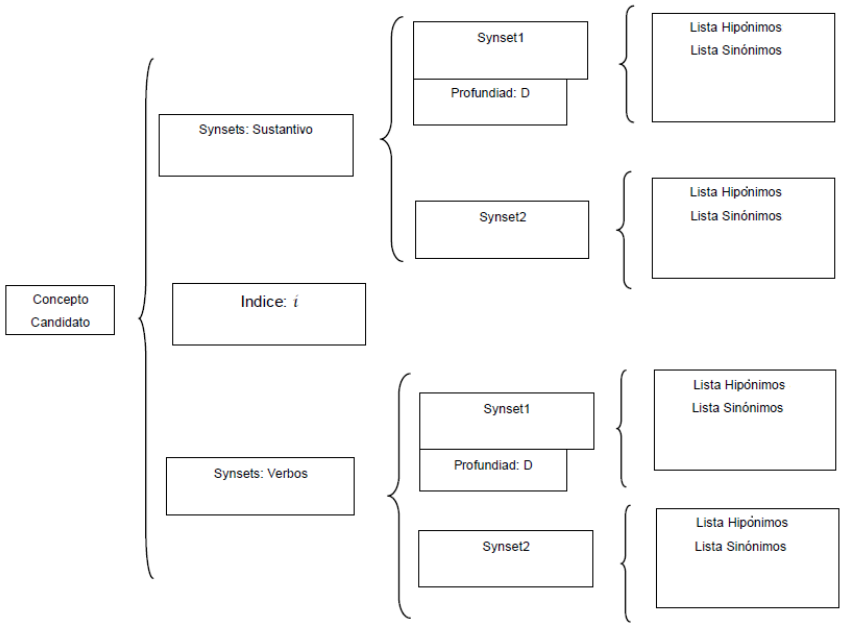


Figura 8. Elementos del Concepto Candidato (Tomado de (Mejía P. and Montealegre P. 2010))

Se debe hacer una aclaración y es que la lista de Hipónimos no se utiliza para ser agregada al synset para este proceso, debido a que puede producir conceptos demasiado restringidos y muy específicos de cada área temática.

Tal y como se puede ver en la Figura 9 con el término “car” como sustantivo, para su synset 1.

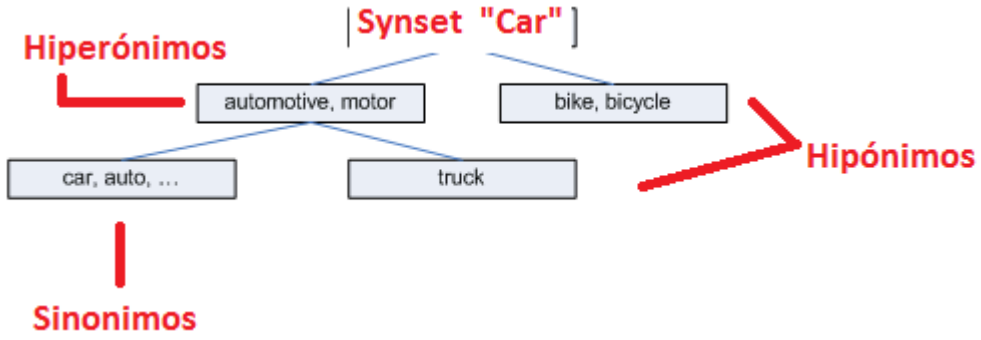


Figura 9. Componentes de un synset para la construcción de la matriz CMD.

1.2 Obtener Uniones Significativas (US): para realizar este paso se tiene en cuenta los siguientes elementos:

Unión Significativa: Asociación de varios synsets que comparten algún significado, para esto se procesa los synsets de cada concepto candidato. Cada US se compone de: *lista de hipónimos, lista de sinónimos, lista de hiperónimos, lista de índices*.

Synset: Los componentes de este elemento son: Lista de Sinónimos, Hiperónimos, índice correspondiente.

Creación de la lista de uniones significativas (ListUS): Esta Actividad lleva los siguientes pasos:

1. Ordenar lista de conceptos candidatos de mayor a menor profundidad.
2. Crear la lista de uniones significativas inicializada con el primer synset del primer concepto en la lista de conceptos candidatos.
3. Procesar los synset {lista de sinónimos, lista de hiperónimos} {indice}, de cada concepto candidato, esto se hace utilizando las reglas de asociación de conceptos.

4. Ordenar la lista de conceptos candidatos de mayor a menor de acuerdo a la profundidad del concepto como verbo y se asocian las uniones significativas como en el paso anterior.

4. La lista de uniones significativas se recorre obteniendo los índices de cada unión significativa, estos índices se utilizan para sumar las columnas correspondientes de la matriz de términos por oraciones de documento, proceso que se realiza antes de crear la matriz CMD.

La matriz de Conceptos Frecuentes por Oraciones de Documento matriz no se define, porque el Algoritmo FP-growth al ser utilizado con el soporte de 5% no obtiene conceptos, estos son nulos.

12.3 Llamado a la rutina de Bisecting K-Means

Se ejecuta el algoritmo Bisecting K-Means como nuestro algoritmo primario de clustering. Este algoritmo empieza con un simple cluster de todos los documentos. El desarrollo de este algoritmo se describirá en la siguiente sección.

12.4 Asignar etiquetas a los clusters

Este proceso se desarrolla utilizando el método de Términos Estadísticamente más representativos, que está contemplado en (Ralambondrainy 1995). Un grupo está representado por un concepto probabilístico, el cual es un conjunto de términos estadísticamente más representativos, el proceso es el siguiente:

1. **Inicializar los parámetros del algoritmo:** Se define el límite o umbral máximo de términos y el umbral de frecuencia mínima de términos. El umbral máximo simboliza el número máximo de términos que comprende una etiqueta en cada grupo. El umbral mínimo de términos

simboliza el porcentaje de la suma total de los términos que se pueden considerar como etiqueta de los grupos.

2. **Construcción de la etiqueta y el grupo "Otros":** Al final, si cualquiera de los grupos no corresponde con la etiqueta de algún grupo, este se asigna al grupo "Otros". Esto solo ocurre cuando la fuente es una matriz FTDM.
3. **Inducción de las Etiquetas Candidatas:** Teniendo en cuenta la matriz de términos frecuentes por oraciones de documento y la frecuencia de los términos que alcanzan el umbral de frecuencia mínima. Posteriormente los términos con las frecuencias más altas se seleccionan teniendo en cuenta que no excedan el umbral máximo de términos.
4. **Eliminar los términos repetidos:** Las etiquetas finales no deben tener términos repetidos, estos son eliminados.
5. **Mejora visual de etiquetas:** Cada término en la etiqueta se reemplaza por la mejor representación del mismo, teniendo en cuenta que si es un verbo, éste será reemplazado por el infinitivo del verbo.

4. Solapamiento de Clusters: Este proceso hace mención de que puede haber documentos que pertenecen a más de un grupo, esto es representativo desde el punto de vista del usuario en IR.

12.5 Rutina Bisecting K-Means

La rutina de Bisecting K-Means para encontrar K clusters sigue así:

1. Seleccionar el cluster a dividir
2. Encontrar 2 Sub-Clusters usando el K-Means Basico; este algoritmo se explica en (Andrade R. and Constain D. 2010). Este es un paso propio de el algoritmo Bisecting K-Means.
3. Se debe repetir el paso 2, para tantas repeticiones y se tiene en cuenta la división que produce el clustering con la más alta similitud.

4. Repetir pasos 1, 2 y 3 hasta alcanzar el número de clusters deseados.

Existen muchas y diferentes maneras de escoger el cluster a dividir. Por ejemplo, podemos escoger el cluster más grande en cada paso, el único con la menor similitud, o usar un criterio basado en los dos rasgos, tamaño y similitud (Steinbach, Karypis et al. 2000).

Si se nota detalladamente, el algoritmo Bisecting K-Means puede producir un cluster no anidado ó un cluster Jerárquico. Para el primero, Cluster no anidado, se puede refinar los clusters haciendo uso de el algoritmo K-Means básico, pero no se puede refinar los clusters anidados.

Para este algoritmo se definen ciertos criterios o parámetros de parada. El algoritmo k-Means por su parte se utiliza como estrategia para improvisar soluciones, el criterio BIC (Criterio de Información Bayesiano) o el índice DB (Davies-Bouldin) para encontrar el número de grupos automáticamente. En cuanto a estos índices se puede utilizar alguno de los dos. En esta investigación, el problema de optimizar se fundamenta en minimizar el criterio BIC o DB, llamado Función Fitness. El algoritmo Bisecting K-Means selecciona un punto aleatoriamente para realizar los cómputos necesarios en el centroide inicial, cada centroide corresponde a un documento diferente seleccionado aleatoriamente en la matriz TDM, en la matriz TFOD o en la matriz CDM (estrategia Forgy en el algoritmo K-Means). Luego divide la matriz obtenida en dos sub-clusters, luego toma cada uno de los sub-clusters y realiza los mismos pasos hasta que encuentre la condición de parada la cual para nuestro proyecto es el número de filas encontradas en cada paso.

12.6 Complejidad del Algoritmo

En el algoritmo WDC-PTFBK genera una matriz de oraciones por documento un proceso de *O(S*D)* donde S es el número de frases por documento, luego se generan los grupos, donde cada grupo generado hace uso de K-Means, el algoritmo K-Means es ejecutado y el valor de la función objetivo o fitness (índice BIC o DB) es calculado. La complejidad del algoritmo Bisecting K-Means es *O(n)*. Ejecutar el algoritmo K-Means y calcular el valor de la función fitness toma *O(n*D*k*L)* ciclos (donde n es el número de documentos, D es el número de términos y L es el número de iteraciones que tiene el algoritmo K-Means para converger). Por otra parte el etiquetado de los grupos toma *O(n*D)* veces. Realizar el solapamiento de los grupos toma *O(k*n*D)* veces. Por tanto la complejidad total del algoritmo es de *O((n*k*D)*(L)*S)*.

13 METODOLOGIA DE DESARROLLO DEL META BUSCADOR

Para la realización de este proyecto se ha tomado como proceso base a RUP con la adición de dos actividades, la primera relacionada con la elaboración del survey y la segunda con la evaluación del algoritmo. Teniendo en cuenta que RUP plantea cuatro fases iterativas, Iniciación, Elaboración, Construcción y Transición. Se incorporó a la fase de iniciación unas actividades de recopilación y análisis bibliográfico en el tema del clustering de documentos web para poder construir el survey. Además a la fase de transición se le adicionó las actividades de recolección de información, realización de pruebas experimentales y análisis de resultados de acuerdo con las medidas planteadas en los objetivos para la evaluación del algoritmo.

Algunas de las actividades se desarrollarán en forma secuencial, otras en paralelo y la fase de construcción se realizará basada en 6 ciclos de 21 días cada uno. A continuación se muestra un resumen de cada etapa:

13.1 Inicio

En esta etapa se realizará además de la recopilación y análisis bibliográfico sobre clustering de documentos web, un diseño preliminar del algoritmo y de la arquitectura general del sistema teniendo en cuenta los requisitos relacionados con la evaluación. Como resultado de esta fase se obtendrá: Un documento con el survey de clustering de documentos web, un documento con una lista de riesgos, un prototipo y un plan de la arquitectura.

13.2 Elaboración

Esta etapa consiste en entender en mayor detalle los requerimientos para el modelado y diseño del algoritmo de clustering de documentos web y el afianzamiento de la arquitectura general del sistema. Como resultado de esta fase se obtendrán: Casos de Uso de alto nivel, Diagrama de Clases, Diagrama Conceptual de la Base de Datos y Arquitectura base.

13.3 Construcción

El objetivo de esta fase es obtener un prototipo funcional del algoritmo que se quiere mostrar. Para el desarrollo de esta etapa se tendrán en cuenta las siguientes disciplinas:

1. **Análisis:** Se hace una profundización sobre los artefactos generados durante la fase de elaboración para la creación del sistema.
2. **Diseño:** Se realizan los casos de uso reales que servirán como guía para la construcción de las diferentes funcionalidades.
3. **Implementación:** Se implementará el sistema (en los primeros ciclos el algoritmo de clustering) con los artefactos obtenidos en las anteriores actividades (Análisis y Diseño), posteriormente se harán pruebas alfa para garantizar su funcionalidad.
4. **Pruebas:** Al finalizar la implementación del prototipo se definen un conjunto de pruebas. Los seis ciclos están pensados para cumplir con las siguientes funcionalidades: 1. Algoritmo de clustering inicial, teniendo en cuenta que los data sets se obtienen de un conjunto definido por la comunidad de IR, así mismo obtener los itemsets frecuentes encontrados en la matriz de Conceptos Frecuentes – Oraciones Documento, para obtener esta matriz se hará uso de la herramienta

lucene.net, realizando nuevos métodos que permitan construir la matriz TF-OD 2. Algoritmo de clustering final con manejo de stopwords (palabras no relevantes en las búsquedas), 3. Funcionalidad relacionada con stemming (raíces gramaticales de las palabras) en inglés, 4, Funcionalidad relacionada con criterios para definición del número grupos y etiquetas de grupos, 5. Funcionalidad relacionada con las ontologías, y 6. Funcionalidad oculta relacionada con la evaluación.

13.4 Transición

En esta fase se instalará el sistema de clustering en el servidor http://spar.unicauca.edu.co/gruweb. Se verificará la funcionalidad del sistema, se realizará la evaluación del algoritmo con base en las medidas planteadas en el tercer objetivo específico. En nuestro caso, no se diseñarán data sets de prueba, sino que se usarán conjuntos ya definidos por la comunidad de IR, específicamente Reuters-21578. Además se realizarán unas pruebas preliminares y básicas con usuarios reales, estudiantes del programa de Ingeniería de Sistemas de la Universidad del Cauca, y el algoritmo tomará como fuente de datos, los resultados de un motor de búsqueda como Yahoo, Google y/o MSN Live.

13.5 Documentación y Divulgación

En forma paralela al desarrollo de las anteriores fases, se realizará una actividad permanente de documentación y ciertos hitos de divulgación de resultados; entre ellos la presentación de un artículo en un evento o revista nacional o internacional, la monografía de grado y sus anexos, y la sustentación de la tesis ante los jurados definidos por el Consejo de Facultad de la FIET.

14 Analisis y Diseño

14.1 Casos de uso de alto nivel

En la Figura 10 se muestran las operaciones que el usuario del sistema (el actor Cliente) pueden realizar, a saber: Iniciar sesión, realizar búsqueda y calificar grupos.

14.2 Casos de uso reales.

A continuación se muestra el caso de uso real más importante del sistema (ver Tabla 6), los demás casos de uso se encuentran en el Anexo.

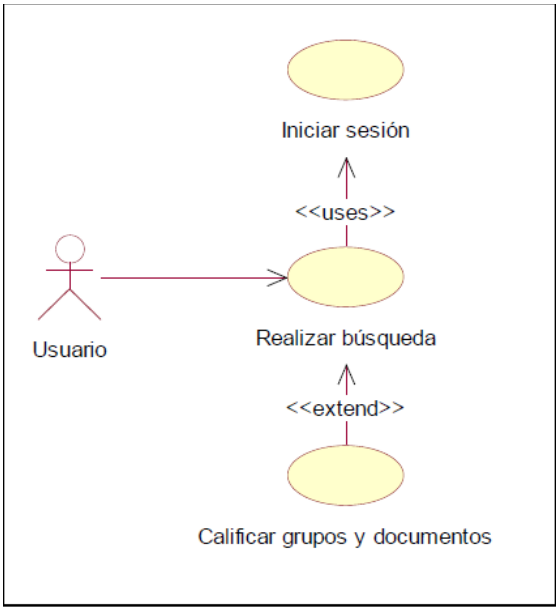


Figura 10. Casos de uso para los usuarios

CASO DE USO REAL: REALIZAR BÚSQUEDA	
Actores: Cliente.	
Propósito: Realizar el agrupamiento por temas según la consulta realizada.	
Resumen: El cliente ingresa la consulta deseada, seleccionado el grupo deseado y revisa los documentos dentro de este.	
Tipo: Primario.	
CURSO NORMAL DE LOS EVENTOS	
Acción del actor	**Respuesta del sistema**
1. El usuario (Cliente), digita la consulta deseada en el cuadro de texto	
2. El usuario da click en el botón Buscar.	3. El sistema obtiene los documentos de la Web por medio de Google, Yahoo! y MSN Live.
	4. El sistema realiza el pre-procesamiento de los documentos.
	5. El sistema crea los grupos.
	6. El sistema muestra los grupos creados.
7. El usuario selecciona el grupo deseado.	
8. El usuario selecciona y visualiza el documento deseado.	

CURSO ALTERNO 1	
Acción del actor	**Respuesta del sistema**
2. El usuario da click en el botón Buscar sin digitar la consulta en el cuadro de texto.	3. El sistema muestra un mensaje informándole que debe digitar el texto de la consulta.
CURSO ALTERNO 2	
Acción del actor	**Respuesta del sistema**
1. El usuario, digita la consulta en el cuadro de texto en un lenguaje diferente al inglés.	
2. El usuario da click en el botón Buscar.	3. El sistema muestra un mensaje informándole que debe digitar el texto de la consulta en inglés.

Tabla 6. Caso de Uso Real Realizar Búsqueda

14.3 Diagrama de clases

En la Figura 11 se muestra de manera general el diagrama de clases del sistema, en el Anexo se ilustran con más detalle cada una de las Clases. Más adelante, en la Tabla 8, se describe la funcionalidad de cada Clase.

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

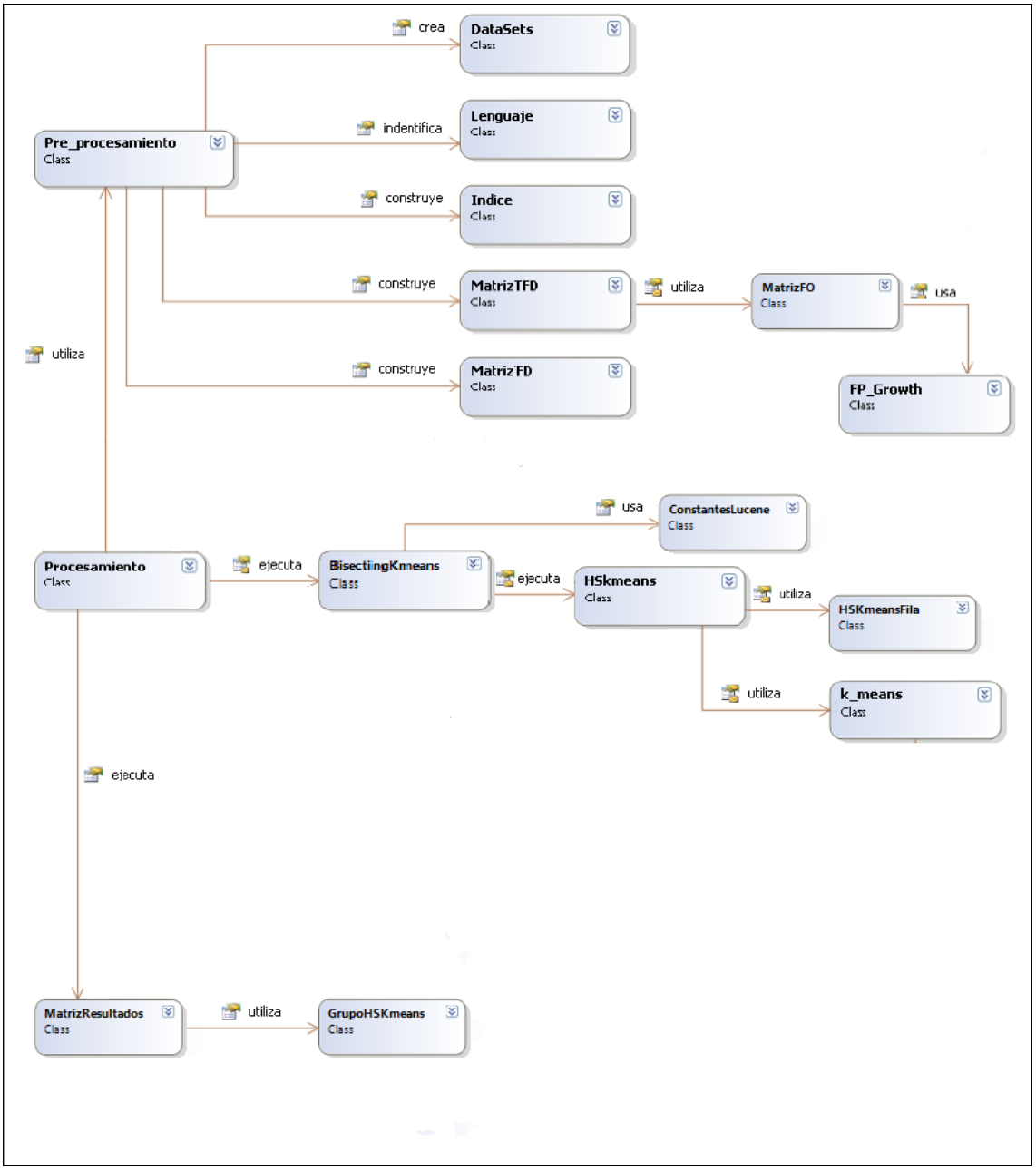


Figura 11. Diagrama General de Clases

Clase	Función
DataSets	Provee la funcionalidad para crear el Data Set con los documentos.
Índice	Provee la funcionalidad correspondiente a la indexación de los documentos tales como: análisis léxico,

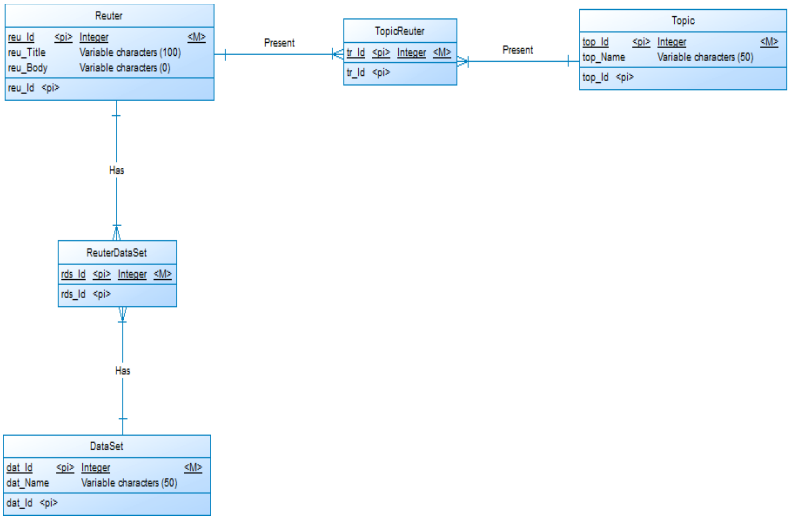
	eliminación de palabras vacías y stemming. Igualmente provee la funcionalidad correspondiente a la obtención de las raíces de los documentos para el proceso de etiquetado.
Fp-Growth	Obtiene la lista de Itemsets frecuentes de los documentos.
Matriz TFD	Provee la funcionalidad para crear la matriz de términos frecuentes por documentos.
Matriz FD	Provee la funcionalidad para crear la matriz de Frases por documentos.
Pre-procesamiento	Provee toda la funcionalidad correspondiente al pre-procesamiento de los documentos.
ConstantesLucene	Mantiene los parámetros para el algoritmo.
K-means	Provee la funcionalidad del algoritmo k-means
MatrizResultados	Se encarga de mantener las mejores soluciones del algoritmo WDC-PTFBK.
Evaluacion	Provee la funcionalidad para el cálculo de las medidas de relevancia (Precisión, Exhaustividad y Medida F). esta clase se utiliza solamente para la evaluación con el dataset de Reuters.
BisectingKmeans	Provee la funcionalidad principal para la ejecución del algoritmo WDC-PTFBK.
MatrizResultado	Provee la funcionalidad correspondiente a la asignación de etiquetas, calificación de grupos y documentos, ordenamiento de grupos y documentos.
Procesamiento	Es la clase principal, que procesa la consulta del usuario para obtener los documentos, realizar el pre-procesamiento de los mismos, ejecutar el algoritmo, crear los resultados y procesar la calificación de grupos y documentos.
Frase	Identifica la frase de cada uno de los documentos.
GrupoHSKmeans	Tiene todos los datos para generar los clusters.
HSKmeans	Provee la funcionalidad correspondiente a la creación de una solución del algoritmo WDC-PTFBK.
HSKmeansFila	Provee la funcionalidad correspondiente a la Creación de

	Centroides Aleatorios y a la Creación de un Improviso.
MatrizFFD	Provee la funcionalidad para crear la matriz de frases frecuentes por documentos.
MatrizFO	Provee la funcionalidad para crear la matriz de frecuencias observadas.
MatrizKmeans	Almacena el resultado después del proceso de cluster.
Noticia	Almacena información importante de la noticia.
Termino	Almacena la frecuencia del termino.

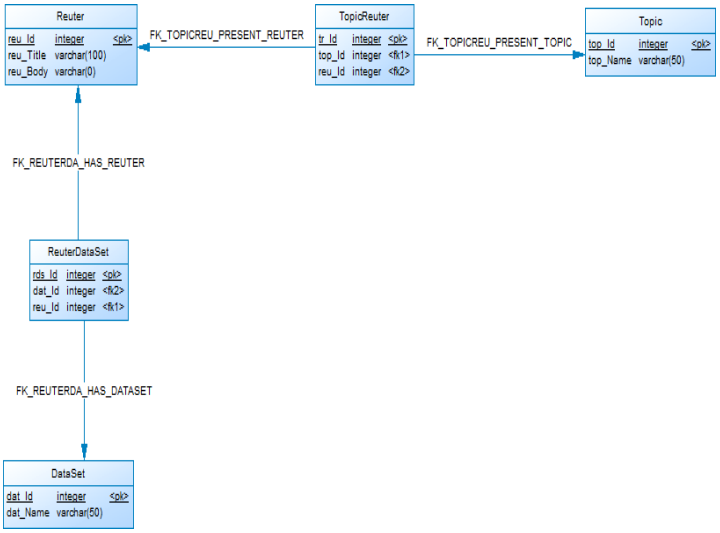
Tabla 7. Descripción de las clases.

14.4 Modelo de la base de datos

A continuación en la Figura 12 se muestra el modelo de la base de datos BDReuter y en la Tabla 9 se explica la finalidad de cada tabla de esta base de datos.



Modelo Conceptual BD-Reuters



Modelo Físico BD-Reuters

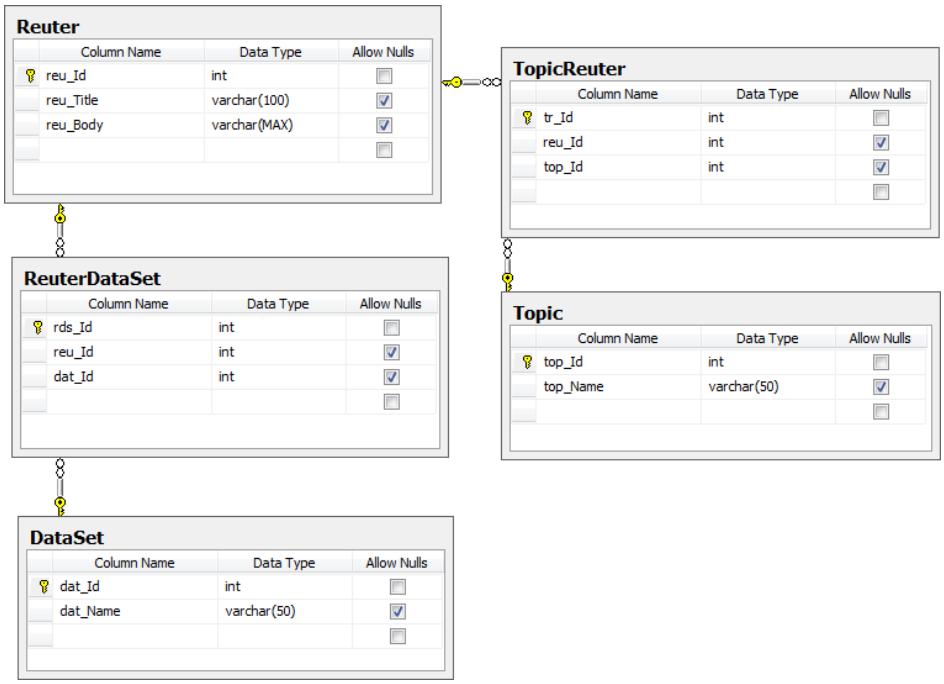


Figura 12. Modelos de la Base de Datos de BDReuter

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

TABLA	FUNCION
Topic	Mantiene la información correspondiente a los temas a los que pertenecen los documentos de la colección Reuters (colección de noticias).
TopicReuter	Mantiene la información correspondiente al tema al que pertenece cada uno de los documentos.
DataSet	Mantiene la información de los cuatro (4) Data Sets creados para realizar las evaluaciones.
ReuterDataSet	Mantiene la información correspondiente a los Data Sets a los que pertenece cada uno de los documentos.
Reuter	Mantiene la información de cada uno de los documentos de la colección Reuters.

Tabla 8. Descripción de las tablas de la base de datos BDReuter

Capitulo 4: Evaluación

15 DATA SETS

La fase de evaluación del algoritmo se utilizó una colección que ya es muy conocida por la comunidad académica y científica de RI, esta colección es Reuters 21578.

Para la esta colección de datos, Reuters 21578, se hizo instancia de cuatro conjuntos de datos (data sets) los cuales son descritos en la siguiente tabla:

	DS1	DS2	DS3	DS4
Tema	Money-Supply 55 Coffe 60 Sugar 45 Interest 40	Money-Supply 80 Coffe 90 Sugar 70 Interest 60 Ship 100	Money-Supply 50 Coffe 50 Sugar 50 Interest 50	Money-Supply 80 Coffe 80 Sugar 80 Interest 80 Ship 100
Numero documentos	40 - 60	60 - 100	50	80
Términos	2613	3991	2697	3977
Términos frecuentes	981	189	419	203

Tabla 9. Descripción de la colección Reuters 21578

El valor de K-Max es $\sqrt{\frac{N}{2}} + 1$, tomado de (Weiguo and Xiaohui, et al. (2008)) donde N es el número de documentos. Para el valor de soporte mínimo del algoritmo FP-Growth es de 10% para los data sets de Reuters 21578. Estos valores se escogieron teniendo en cuenta varios aspectos: 1) a medida que aumenta el soporte mínimo, la cantidad de itemsets en los documentos que cumplen con el porcentaje de soporte será menor, 2) el tamaño de los documentos a los que se les aplicará el algoritmo FP-Growth es importante, debido a que si los documentos son grandes y el soporte es pequeño el tiempo

de procesamiento aumenta, y 3) si los documentos son pequeños y el soporte es grande se obtienen pocos itemsets.

A continuación se presentan las preguntas realizadas a los usuarios con respecto a cada grupo generado y con respecto a cada documento presentado por grupo. Estas preguntas fueron efectuadas para realizar una experimentación controlada con usuarios finales y así, en lo posible, generalizar los resultados con facilidad.

Las preguntas para cada Grupo, Cluster o Agrupación fueron:

1. ¿El nombre del grupo representa los documentos que contiene?
   R1. Nada; R2.Poco; R3.Mucho.
2. ¿El grupo es útil para su consulta?
   R1.no útil; R2.moderadamente útil; R3.útil

Para cada documento dentro de cada grupo generado se preguntó:

3. ¿El documento se ajusta a la descripción del grupo?
   R1.nada; R2.moderadamente, R3.muy bien.
4. ¿La ubicación del documento en el grupo, según su importancia es?
   R1.incorrecta, R2.moderadamente correcta, R3.correcta.

Se buscó además, evaluar los resultados de la funcionalidad del algoritmo, basado en diferentes opciones de ejecución con el fin de obtener resultados que permitieran definir cual alternativa de procesamiento de documentos generaba mejores soluciones.

16 Resultados

El Algoritmo se ejecutó 10 veces con la matriz TDM con el criterio BIC y DB, los resultados promedios son mostrados en la Tabla 10. Luego se ejecutó el algoritmo sobre la matriz TFOD con el criterio BIC, DB 10 veces. Los resultados

promedios se muestran en la Tabla 11. Esto para ver si el algoritmo es eficiente, y se calculó la Precisión (P), la Recuerdo (R) y la Medida F (F), se busca obtener valores altos de P, R y F

	K			P		R		F	
	Ideal	**BIC**	**DB**	**BIC**	**DB**	**BIC**	**DB**	**BIC**	**DB**
DS 1	4	8,7	9	84,5	80,4	51,6	55,3	62,7	65,2
DS 2	5	10,3	10,8	81,8	78,3	46,4	55,6	61,2	65,6
DS 3	4	7	7,8	83,5	78,6	49,8	53,4	62,8	63,4
DS 4	5	13,5	13,5	80,8	77,7	48	53,4	60,6	62,1
AVG	4,5	9,875	10,275	82,65	78,75	48,95	54,425	61,825	64,075

Tabla 10. Precisión (P), Recuerdo(R) y Medida F (F) para dos criterios (BIC y DB) en la Matriz de Términos por Documentos (TDM).

Después, el algoritmo se corrió 10 veces sobre la Matriz de Términos Frecuentes por Oraciones Documentos (TFOD) con el criterio BIC, de la misma forma se calcularon los promedios de las ejecuciones. Luego, el mismo algoritmo se ejecutó 10 veces usando el índice DB sobre la misma matriz (TFOD). Estos resultados promisorios se muestran en la Tabla 12.

	K			P		R		F	
	Ideal	**BIC**	**DB**	**BIC**	**DB**	**BIC**	**DB**	**BIC**	**DB**
DS 1	4	9.6	3	86,1	35,8	48,6	46,5	60,3	43,2
DS 2	5	14	12,4	87,2	84,3	42,4	46,2	55,3	61
DS 3	4	10.4	7,8	84,9	77,8	47,6	48,3	59,9	59,7
DS 4	5	13.8	7,6	88,7	74,3	42,6	49,7	59,7	52,7
AVG	4,5	14	7,7	86,725	68,05	45,3	47,675	58,8	54,15

Tabla 11. Precisión (P), Recuerdo (R) y Medida F (F) para dos criterios (BIC y DB) en la Matriz de Términos Frecuentes por Documentos (TFOD)

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

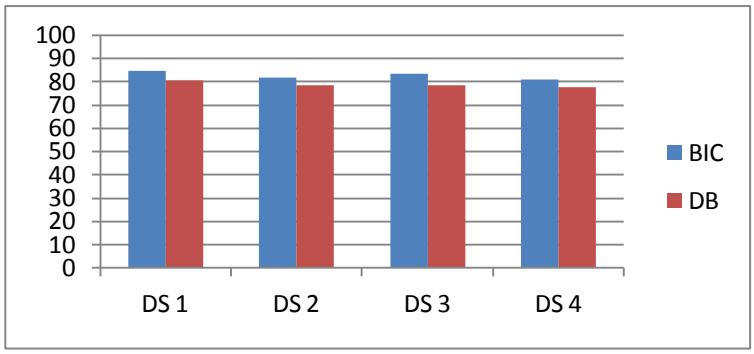


Figura 13. Precisión para los criterios BIC y DB de la matriz TDM.

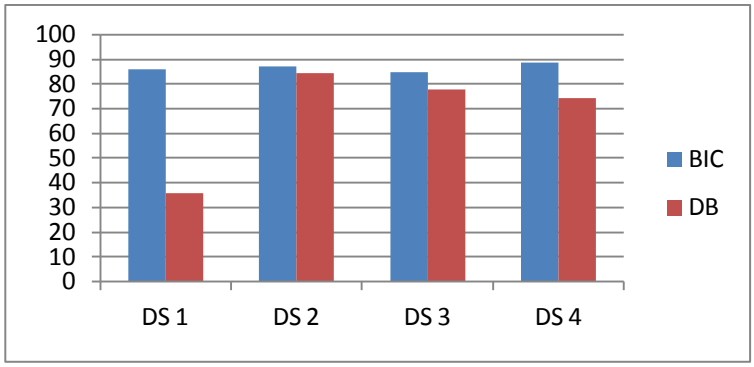


Figura 14. Precision para los criterios BIC y DB de la matriz TFOD

En general, el algoritmo obtuvo mejores resultados de precisión usando el criterio BIC sobre la Matriz de Términos Frecuentes por Oraciones Documentos, aunque el índice DB reportó valores más apropiados de k (número de grupos) en TFOD. Con relación a la Medida F, los resultados son mejores para la Matriz de Términos por Documentos (5% más), aunque en este camino de ejecución de nuestro algoritmo, no hay reducción de la dimensionalidad.

Esta experimentación sirvió para verificar que el criterio BIC es la mejor función de aptitud en comparación con DB, además sirvió para definir que el proceso de etiquetado que más se ajusta a la representación de los clusters.

Adicionalmente, se muestra la Tabla 13 que relaciona el tiempo promedio de ejecución del algoritmo respecto a las matrices TDM, TFOD y CDM. Es preciso

comentar que este tiempo de ejecución es alto para el entorno de clustering de documentos web, pero no lo es para la tarea de clustering de textos, como es el caso de los experimentos en laboratorio. Ya que el clustering de estos documentos se hace fuera de línea y con el texto completo de muchos documentos. Por el otro lado en clustering de documentos web se trabaja solo con los resúmenes y con un máximo aproximado de 180 documentos.

Modelo de Representación de Documentos	Tiempo Promedio de Ejecución del Algoritmo
TDM	4,7
TFOD	4,5
CDM	5,3
PROMEDIO	4,83

Tabla 12. Tiempos de ejecución.

17 Comparación con Lingo

Carrot2 es un buscador web que realiza agrupamiento de documentos web para mostrar los resultados de las consultas. Carrot2 implementa el modelo de Lingo. Este modelo se basaen la Descomposición de Valores Singulares y Frases Frecuentes. Para comparar nuestro algoritmo con Carrot2, se utilizó la última versión de Carrot2, llamada Workbench (http://project.carrot2.org). Esta versión de Carrot2 es una aplicación de escritorio y puede ser ejecutada con diferentes fuentes (no solo con los resultados de búsquedas web). Se utilizaron los data sets basados en Reuters 21578 (data set 1 a 2 descritos en la sección resultados) para comparar nuestro algoritmo con Carrot2. En el algoritmo propuesto en este proyecto se utilizaron los resultados de una sola ejecución de éste para compararlos con los resultados de Carrot2.

Nuestro algoritmo genera 8 grupos y sus respectivas etiquetas para el Data set 4. En estos grupos generados, 5 de ellos coinciden con los temas originales: Algorithms, Number Theory, Volleyball, Adventure, y Dogs. En la Figura 12 se muestran 11 grupos generados por Carrot2, en los cuales ninguna de las

etiquetas de grupo coincide con los temas originales del Data set 4, además el grupo “Otros temas” tiene 88 documentos, lo cual no es apropiado para las necesidades de los usuarios.

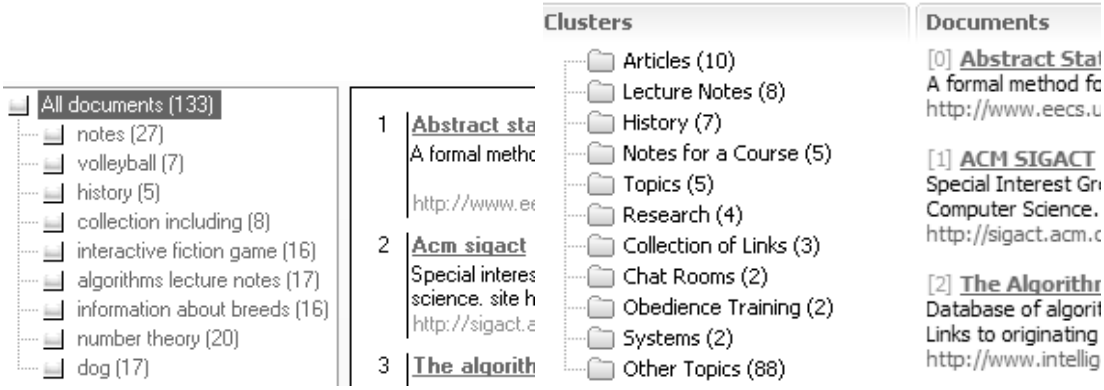


Figura 15. Resultados del Data set generados por Carrot2.

La Tabla 13 muestra los resultados de Precisión, Recuerdo, Medida F, el Número de Etiquetas que fueron realmente Representativas de cada grupo (NRL) y el Número de Documentos en el grupo con etiqueta “Otros temas” (OT). En general, nuestro algoritmo tiene mejores resultados (precisión, recuerdo y medida F) que Carrot. Nuestro algoritmo definió un valor mejor de k (número de grupos) que Carrot. Nuestro algoritmo tiene una proporción más alta de etiquetas representativas que Carrot, finalmente, Carrot2 reportó una gran cantidad de documentos en el grupo “Otros temas” lo cual no es adecuado para los usuarios finales.

.

		DS1	DS2	DS3	DS4
K	Real Carrot CDW	3 9 8	4 9 8	4 11 8	5 11 9
P	Carrot CDW	78,2% 90,1%	68,1% 81,2%	59,7% 85,5%	58,5% 76,5%
R	Carrot CDW	33,5% 42,2%	42,8% 48,4%	26,0% 45,8%	29,4% 50,2%
F	Carrot CDW	46,9% 59,4%	52,6% 61,4%	36,2% 59,8%	39,2% 65,9%
NRL	Carrot CDW	6(67%) 7(88%)	6(67%) 6(75%)	6(55%) 5(63%)	9(82%) 7(78%)
OT	Carrot CDW	72 4	74 0	84 1	88 3

Tabla 13. Precisión (P), Recuerdo (R) y Medida F (F), Número de Etiquetas Representativas (NRL) y el Número de Documentos en el grupo “Otros temas” (OT) para CDW y Carrot2.

18 Evaluación de Usuario

Esta sección se realizó con la colaboración de algunos estudiantes del programa Ingeniería de Sistemas, de diferentes semestres, en total fueron 15 estudiantes los que aplicaron la prueba.

Para realizar esta prueba se utilizó métricas estándar de IR, tales como: precisión y recuerdo, para cada cluster, y así evaluar las respuestas del usuario.

Para la pregunta numero 1 (Etiquetas representativas de cluster), se obtuvo mejores resultados cuando se ejecuta el algoritmo usando la matriz TDM y el índice BIC.

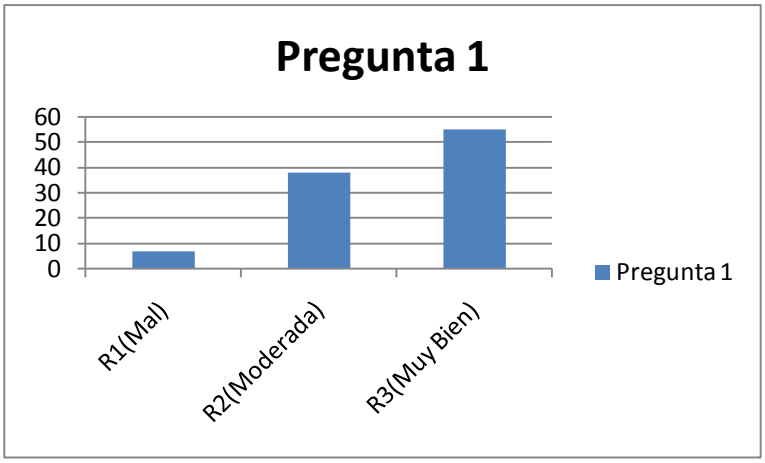


Figura 16. Resultados específicos para Pregunta 1 según los encuestados.

Para la pregunta 2 (¿El grupo es útil para su consulta?), los mejores resultados son obtenidos cuando el algoritmo utiliza la matriz de términos por documentos, y el índice BIC.

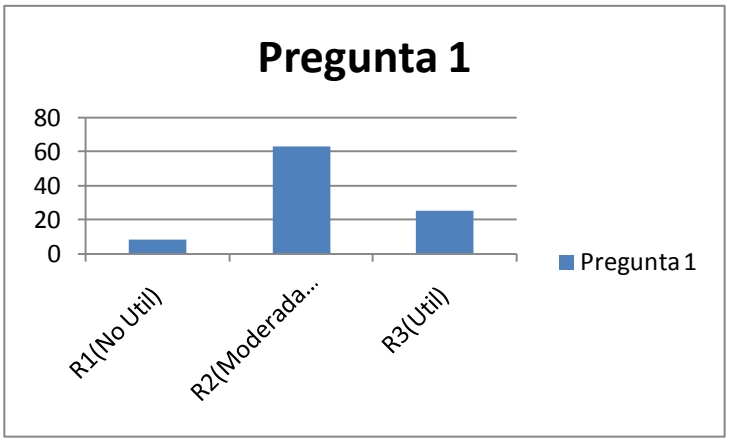


Figura 17. Resultados específicos para Pregunta 2 según los encuestados

Para la pregunta 3 (¿El documento se ajusta a la descripción del grupo?), los mejores resultados son obtenidos cuando el algoritmo utiliza la matriz de términos por documentos, el Criterio de Información Bayesiano.

Daniel Andrés Pino Gómez
Pablo Alejandro Zúñiga Muñoz
Ph. D. Carlos Alberto Cobos Lozada

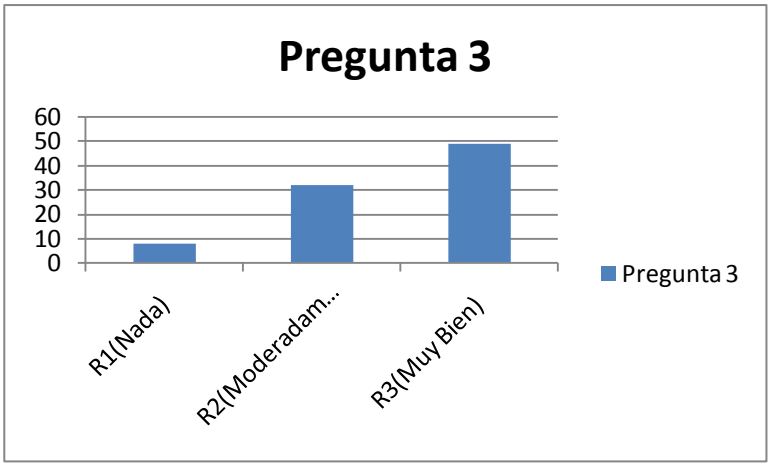


Figura 18. Resultados específicos para Pregunta 3 según los encuestados

se enfoca en indagar si el documento está bien asociado al grupo al cual pertenece. En la figuran anterior, se observa que los mejores resultados son la opción “Muy bien” (R3), además de un porcentaje aceptable de documentos “Moderadamente bien”. Así que, estas dos opciones son competitivas a la hora de realizar el agrupamiento de documentos

Para la pregunta 4 (¿La ubicación del documento en el grupo, según su importancia es:?), los mejores resultados se obtuvieron cuando el algoritmo usa la matriz de términos por documentos, y el índice BIC.

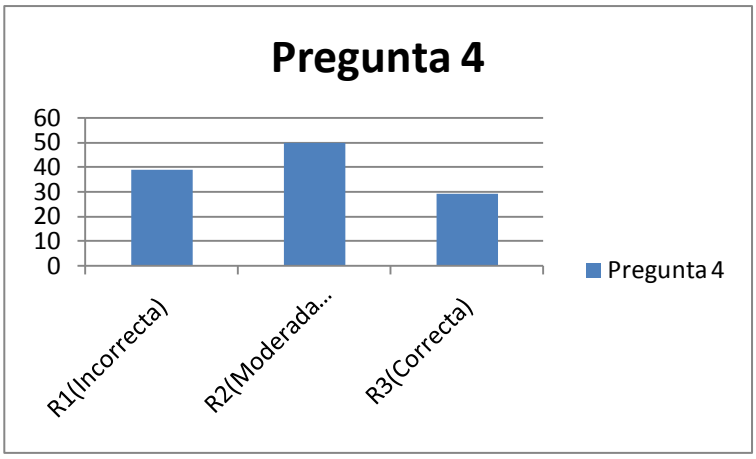


Figura 19. Resultados específicos para Pregunta 4 según los encuestados

Si se tiene que los documentos en su gran mayoría son presentados al usuario, por su relevancia en el tema consultado, en esta prueba nuestro algoritmo propuesto tiene un porcentaje relativamente alto en que lo realiza moderadamene bien, pero existe tambien la posibilidad de hacerlo de la manera incorrecta, esto se debe a la dimensionalidad de los documentos, y a que la información no siempre queda organizada de la misma manera en todas las ejecuciones, producto de la dependencia del orden de análisis de los documentos, el proceso de asignación de elementos a los grupos o simplemente del criterio de agrupamiento utilizado.

Con los anteriores resultados podemos afirmar que las matrices TFOD Y CDM, junto con el criterio BIC y el etiquetado de frases, provee resultados satisfactorios para el clustering de documentos web.

Capitulo 5: Conclusiones y Trabajo Futuro

- La arquitectura del meta-buscador se puede modificar fácilmente para agregar nuevos componentes relacionados con el pre-procesamiento, el algoritmo de clustering o el tipo de etiquetado, lo que la convierte en una arquitectura adaptable que puede servir como base para futuras investigaciones sobre el clustering de documentos Web.

- El modelo de representación de documentos "Términos Frecuentes por Oraciones de documentos" es más adecuado que el modelo estándar de "Términos por documentos", debido a que el primero, por una parte reduce la dimensionalidad de los documentos y por otra parte arrojó mejores resultados (mayores promedios de precisión) en las pruebas realizadas en la aplicación.

- Se diseñó, implementó y evaluó el algoritmo WDC-PTFBK que combina una matriz de Términos Frecuentes por Oraciones de Documentos y el algoritmo Bisecting Kmeans. Este algoritmo encuentra automáticamente el número de grupos, asigna nombres adecuados a los grupos de documentos y como se mostró en las pruebas, además de ser viable en el clustering de documentos Web mostró mejores resultados que LINGO, uno de los algoritmos más recientes y citados en la literatura de clustering de documentos Web.

- Los resultados obtenidos en el laboratorio con el data set de Reuters, en relación con la medida de precisión superaron los resultados obtenidos por la mayoría de los algoritmos propuestos en investigaciones previas para el clustering de documentos, los cuales se encuentran entre el 60% y 80%, WDC-PTFBK alcanzó en promedio 86.725%.

- Consideramos que lo más útil para un usuario que plantea búsquedas muy generales, es que los términos que caracterizan a cada grupo puedan acercarlo a precisar lo que realmente busca y con estos nuevos términos el usuario los pueda utilizar en subsiguientes búsquedas y de esta manera pueda cambiar de su estrategia de búsqueda muy general a una estrategia de búsqueda más específica que evite el problema de la sobresaturación de resultados.

19 Trabajo Futuro

- Hacer uso de la herramienta Segmenter para realizar una síntesis de los documentos, teniendo en cuenta elementos como, tema e idea de la frase, esto permite que se mejore aún más los resultados puesto que se reduce la cantidad de términos candidatos a indizar.

- Implementar otros algoritmos de stemming que permitan la búsqueda en idiomas diferentes al inglés (Recuperacion de Informacion Multi Lenguaje). Y analizar el impacto de manejar múltiples idiomas en el algoritmo de clustering.

- Utilizar otros valores de soporte y confianza para las reglas de Asociación y asi obtener cluster mucho mas compactos.

Capitulo 6: Glosario y Bibliografía

20 Glosario

Lucene: es un API de código abierto para recuperación de información, originalmente implementada en Java por Doug Cutting. Está apoyado por el Apache Software Foundation y se distribuye bajo la Apache Software License.
Esta API tiene versiones para otros lenguajes incluyendo C#, es útil para cualquier aplicación que requiera indexado y búsqueda de texto completo. Lucene ha sido ámpliamente usado por su utilidad en la implementación de motores de búsqueda.

Metabuscador: Sistema que hace uso de los motores de búsqueda más utilizados como Yahoo!, Google, Big, entre otros, para encontrar información. Estos buscadores carecen de base de datos propia y, en su lugar, aprovecha las de otros buscadores y muestra una combinación de las mejores páginas que ha devuelto cada buscador.

Meta heurística: Estrategia de alto nivel que guía otros métodos o heurísticas para buscar soluciones factibles y sencillas a un problema complejo, por lo general perteneciente al área computacional.

Survey: En tecnología, se refiere a un documento de investigación que recopila de manera general diferentes investigaciones presentadas por la comunidad científica, relacionadas con un tema o problema de investigación.

Sistemas IR: Proceso que accede a información previamente almacenada, mediante herramientas informáticas que permiten establecer ecuaciones de búsqueda específicas.

Snippet: En el contexto del tema de investigación de éste proyecto, hace referencia al fragmento de texto que describe cada documento resultado de una búsqueda.

Cluster: Este término se aplica a una colección de objetos que son "similares" entre si y son "distintos" a los objetos pertenecientes a otros grupos.

Clustering: Método utilizado para la creación de agrupaciones o clusters.

Etiquetas: En el proceso de clustering de documentos Web, este término hace referencia a los elementos encargados de identificar o describir un cluster.

Centroide: En el tema de esta investigación, este término hacer referencia al punto que define el centro del cluster.

SVD: Abreviatura que referencia “Singular Value Descomposition”, la cual es una técnica que sirve para realizar factorización de matrices.

Ontología: Éste término hace referencia a la formulación de un exhaustivo y riguroso esquema conceptual dentro de uno o varios dominios dados; con la objeto de facilitar la comunicación y el intercambio de información entre diferentes sistemas y entidades.

Taxonomía Léxica: Término que hace referencia al proceso de ordenar aquellas palabras con contenido referencial y semántico en un sistema de clasificación compuesto por una jerarquía de grupos anidados.

21 BIBLIOGRAFIA

Akbar, M. (2008). "FP-GROWTH approach for document clustering."

Andrade R., J. K. and W. A. Constain D. (2010). "Hibridación de la mejor búsqueda armónica global y el algoritmo k-means para el clustering de documentos web." 100.

Arunasalam, B., S. Chawla, et al. (2009). "Association Rules Network Definition and Aplications." Wiley Periodicals, Inc.

Baeza-Yates, R., A. and B. Ribeiro-Neto (1999). Modern Information Retrieval, Addison-Wesley Longman Publishing Co., Inc.

Baroni, M., S. Evert, et al. (2008). "ESSLLI 2008 Workshop on Distributional Semantics: Bridging the gap between semantic theory and computational simulations."

Beckwith, R., C. Fellbaum, et al. (1993). "Introduction to WordNet: An On-line Lexical Database."

Beil, F., M. Ester, et al. (2007). Frequent term-based text clustering. KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, Edmonton, Alberta, Canada, ACM.

Berkhin, P. (2002). Survey Of Clustering Data Mining Techniques.

Berry, M. W. and M. Castellanos (2007). "Survey of Text Mining: Clustering, Classification, and Retrieval " Springer.

BrightPlanet (2000). "The Deep Web: Surfacing Hidden Value."

Capparelli, A., Urtasun, M. (2004). "Clasificación automática de documentos basada en la métrica de similitud universal Vitányi." 164.

Carpineto, C., Osinski, S., Romano, G., Weiss, D. (2009). "A Survey of Web Clustering Engines." ACM Comput. Surv.

Carpineto, C. O. S., Romano Giovanni, Weiss Dawid. (2009). "A survey of web clustering engines." ACM Comput. Surv.

Casillas, A., Fresno, V., Martínez, R., Montalvo, S. (2005). "Evaluación del Clustering de Páginas Web mediante funciones de peso y combinación heurística de criterios." 8.

D. Pascual, F. Pla, et al. "Algoritmos de Agrupamiento." 13.

Dunn, J. C. (1974). "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters." Cybernetic and Systems **36**: 27.

Eckard, E. and J. C. Chappelier (2007). "Free Software for research in Information Retrieval and Textual Clustering." 27.

Fersini, E., Messina, E., Archetti, F. (2009). "A probabilistic relational approach for web document clustering." ELSEVIER: 14.

Fung, B., K. Wang, et al. (2003). Hierarchical document clustering using frequent itemsets. Proceedings of the SIAM International Conference on Data Mining.

Hammouda, K. (2001). Web Mining: Clustering Web Documents A Preliminary Review.

Hatcher, E. and O. Gospodnetic (2005). "Lucene in Action."

Hearst, M. A. (1997). "TextTiling Segmenting Text into Multi-Paragraph subtopic passages." Association for Conputational Linguistics: 32.

Hipp, J., U. Güntzer, et al. (2000). "Algorithms for Association Rule Mining – A General Survey and Comparison." ACM Comput. Surv.: 7.

J. Stefanowski. and D. Weiss. (2007). "Comprehensible and Accurate Cluster Labels in Text Clustering."

Jiang-Chun, S., S. Jun-Yi, et al. (2004). A new document clustering algorithm based on association rule. Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on.

Leouski, A. V. and W. B. Croft (1996). "An Evaluation of Techniques for Clustering Search Results." National Science Foundation.

Li, H., Wang, Y., Zhang, D. (2000). "Parallel FP-Growth for Query Recommendation." ACM Comput. Surv.: 8.

Li, H., Wang, Y., Zhang, D. (2008). "Parallel FP-Growth for Query Recommendation." ACM Comput. Surv.: 8.

Li, Y., S. M. Chung, et al. (2008). "Text document clustering based on frequent word meaning sequences." Data & Knowledge Engineering **64**(1): 381-404.

Liu, X. and P. He (2005). A Study on Text Clustering Algorithms Based on Frequent Term Sets. Advanced Data Mining and Applications**:** 347-354.

Mahdavi, M. and H. Abolhassani (2009). "Harmony K-means algorithm for document clustering." Data Mining and Knowledge Discovery **18**(3): 370-391.

Mahdavi, M., M. H. Chehreghani, et al. (2008). "Novel meta-heuristic algorithms for clustering web documents." Applied Mathematics and Computation **201**(1-2): 441-451.

MARTINEZ A., F. (2007). "An´alisis de las series temporales de los precios del mercado el´ectrico mediante tecnicas de clustering."

Mejía P., M. F. and C. A. Montealegre P. (2010). "Clustering de Documentos Web basado en Algoritmos Meméticos con Técnicas de Niching." 101.

Middleton, C. and R. Baeza-Yates (2008). "A Comparison of Open Source Search Engines." 46.

O. Zamir., O. Etzioni., et al. (1997). "Fast and intuitive clustering of web documents." American Association for Artificial Intelligence.

Oren, Z. and E. Oren (1998). Web document clustering: a feasibility demonstration. Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. Melbourne, Australia, ACM.

Osiński, S. (2003). An Algorithm for clustering of web search results. Poland, Poznań University of Technology,. **Master:** 91.

Osinski, S. and D. Weiss (2005). "A concept-driven algorithm for clustering search results." Intelligent Systems, IEEE **20**(3): 48-54.

Perez, J., M. F. Henriques, et al. (2007). "Mejora al algoritmo K-Means mediante un nuevo criterio de convergencia y su aplicación a bases de datos poblacionales de cáncer." 7.

Porter, M. F. (1980). "An Algorithm for suffix stripping." ACM Trans. Knowl. Discov. Data.

Ralambondrainy, H. (1995). "A conceptual version of the K-means algorithm." Pattern Recognition Letters **16**(11): 1147-1157.

Ramos H., J. P. and G. A. Hernandez (2008). "Indización y Búsqueda a través de Lucene."

Sasaki, M. and H. Shinnou (2004). "Web Document Clustering Using Threshold Selection Partitioning." National Institute of Informatics: 7.

Savoy, J. (2000). "Information retrieval on the Web." 4.

Shen, L., Shen, H., Cheng, L. (1999). "New algorithms for efficient mining of association rules." Information Sciences.

Steinbach, M., G. Karypis, et al. (2000). A comparison of document clustering techniques.

Tasoulis, S. K. and D. K. Tasoulis (2008). "Improving Principal Direction Divisive Clustering."

Weiguo, S., L. Xiaohui, et al. (2008). "A Niching Memetic Algorithm for Simultaneous Clustering and Feature Selection." Knowledge and Data Engineering, IEEE Transactions on 20(7): 868-879.

Wong, S. K. M. and C. J. Butz (2000). "A Bayesian Approach to User Profiling in Information Retrieval." Technology Letters **4**(1): 50-56.

Xiang-Wei, L., H. Pi-Lian, et al. (2005). The research of text clustering algorithms based on frequent term sets. Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on.

Zaki, M. (1999). "Generating Non-Redundant Association Rules." Rensselaer Polytechnic Institute.

Zaki, M. (1999). "generating non-redundant association rules." 18.

Zamir, O. and O. Etzioni (1999). "Grouper: a dynamic clustering interface to Web search results." Computer Networks **31**(11-16): 1361-1374.

Zhang, D. and Y. Dong (2004). Semantic, Hierarchical, Online Clustering of Web Search Results. Advanced Web Technologies and Applications**:** 69-78.