Universidad de las Ciencias Informáticas

Facultad # 3

Título: Implementación de algoritmos de agrupamiento

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Darel Camps Diaz

Tutor: Lic. Manuel Vázquez Acosta

Junio 2007

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de Junio del año 2007.

Darel Camps Diaz

_____

Lic. Manuel Vázquez Acosta

_____

“Todo tiempo futuro tiene que ser mejor”

(Julio A. Mella)

AGRADECIMIENTOS

A mis padres y a mi familia, por su apoyo incondicional durante todos estos años.

A la Revolución, por haberme dado la posibilidad de estudiar.

A nuestro Comandante en Jefe Fidel Castro por sus ideas.

A la UCI, quien me permitió forjarme como profesional y como parte de una generación de porvenir, de un proyecto de futuro.

A todos los profesores que tuvieron que ver con mi formación y especialmente a mi tutor.

A mis amigos todos, por confiar en mi amistad y brindarme su apoyo.

A mi hermano de la universidad el Bad Boy, por todo lo vivido.

Finalmente, un recuerdo a todas las personas que me quieren, a quienes seguro he abandonado demasiado tiempo y en algún caso decepcionado. Aunque a veces puedan dudarlo, siempre han estado, están y estarán conmigo.

Gracias por su amor y cariño

DEDICATORIA

El esfuerzo realizado durante mis años de estudio,
se los dedico al sueño de mis padres y hermano,
de verme realizado como profesional.

RESUMEN

La digitalización de la información y la automatización de los servicios, han provocado la generación de grandes volúmenes de información. Esta capacidad de producir información ha modificado el soporte del saber y el conocimiento. Con ello, se han visto afectadas nuestras habilidades para asimilar y convertir esa información en conocimiento.

Uno de los procesos que involucra el procesamiento de la información es la clasificación. Donde su objetivo es asignar a la fuente de información una o más categorías según el contenido tratado. El presente trabajo tiene como propósito automatizar este proceso a través de técnicas de agrupamiento de forma no supervisada. En él se realiza una revisión de las técnicas de agrupamiento más utilizadas, se diseña e implementa una biblioteca con algoritmos de agrupamiento, específicamente el algoritmo K-Means y Simultaneous Keyword Identification and Clustering of Text Documents. Además, se desarrolló un caso de estudio para evaluar la calidad de los resultados de los agrupamientos.

TABLA DE CONTENIDOS

INTRODUCCIÓN

En el mundo actual, la llamada Sociedad de la Información ha provocado un cambio en la humanidad, el cual está reflejado en la forma de utilizar la información a través de las nuevas tecnologías. Podemos afirmar entonces que la información ahora regula el comportamiento de la sociedad y se ha convertido en un recurso clave de la economía, de las organizaciones, del mundo cultural y de la política (JOYANES, 1997).

Algunos de los resultados que ha provocado la utilización de las tecnologías de la información y las comunicaciones, es la digitalización creciente de la información y la automatización de los servicios (TIC). Donde cualquier clase de información o servicio puede estar disponible en cualquier parte del mundo, sin que existan barreras para la comunicación (BARRETT, 1999). Esta capacidad de producir información en grandes volúmenes, ha modificado el soporte del saber y el conocimiento, pero también ha generado consecuencias. Estas están reflejadas en las capacidades que poseen los seres humanos para asimilar y convertir esa información en conocimiento (RODRÍGUEZ, 1999).

Refiriéndose a la gestión del conocimiento la doctora cubana Rosa Elena Simeón expresó: “Nuestro propósito es contribuir a la creación en Cuba de organizaciones inteligentes, que son aquellas capaces de recibir y procesar información, crear conocimiento a partir de la información procesada y usar el conocimiento para la toma de decisiones de manera eficaz, a fin de cooperar de modo creativo al desarrollo del país” (NEGRÍN, 2002).

Como se aprecia el conocimiento es considerado un recurso valioso para la raza humana. Gran parte de este conocimiento existe en forma de lenguaje natural: libros, informes, periódicos, noticias, etcétera. Para poder apropiarse de dicho conocimiento es necesario contar con ciertas habilidades, para poder realizar operaciones con la información como: buscar y filtrar la información necesaria, comparar fuentes de información diferentes y obtener conclusiones, clasificar por sus contenidos la información y realizar procesamiento con los textos; que pudiera ser editarlos ó resumirlos (GÓMEZ, 2003).

Cuando ocurre el procesamiento de la información surge el proceso de clasificación, donde el consumidor de información una vez que haya leído y analizado la fuente de información debe ser capaz de asociar la misma a una categoría que exprese una relación con su contenido.

Pero debido a la gran cantidad de información que existe, resulta imposible que los humanos puedan llevar a cabo esta tarea de clasificación, y es entonces donde surge la necesidad de contar con mecanismos automatizados que se encarguen de realizar este trabajo.

Existen en el mundo aplicaciones que hacen uso de estos métodos automatizados para lograr la clasificación mediante el agrupamiento de documentos afines por sus contenidos. Un ejemplo de estos es el sitio Google Noticias (GOOGLE, 2007a) de la compañía Google, en sus diferentes idiomas. El mismo es generado automáticamente, recopilando titulares de noticias de todo el mundo, dependiendo del idioma. Este agrupa aquellas noticias que sean similares usando un sistema software de clasificación. Además cuenta con un sistema empresarial llamado Google Search Appliance que tiene como objetivo agrupar los resultados de las búsquedas en subcategorías para ayudar a los trabajadores de cualquier oficina a refinar sus consultas(GOOGLE, 2007b).

Los buscadores hacen uso de estas tecnologías para brindar resultados de búsquedas más específicas a sus usuarios, permitiéndoles navegar por un árbol de categorías en sus resultados, además permiten optimizar el tiempo de respuesta y calidad de las búsquedas, dentro de los buscadores más conocidos con estas características se encuentra Clusty (VIVÍSIMO, 2000).

Los buscadores hacen uso de estas tecnologías para brindar resultados más específicos a sus usuarios, permitiéndoles navegar por ellos usando un árbol de categorías, además de posibilitar la optimización del tiempo de respuesta y la calidad de las búsquedas. Dentro de los buscadores más conocidos con estas características se encuentra Clusty (VIVÍSIMO, 2000).

En Cuba y especialmente en la Universidad de las Ciencias Informáticas (UCI) sería de gran importancia contar con estos métodos automatizados. Actualmente se desarrolla en la UCI un trabajo de tesis no publicado que tiene como objetivo sentar las bases tecnológicas para la creación de un observatorio de información (LÓPEZ, 2007), que pudiera utilizar estas técnicas

para realizar agrupamientos de las fuentes de información comunes para posteriormente se creen resúmenes de las mismas.

Uno de los requerimientos que deberá abordar el sistema en su desarrollo será el agrupamiento de la información observada. Para llevar a cabo su implementación, se ha decidido realizarlo sobre la plataforma Zope (ZOPE-CORPORATION, 2007) y específicamente sobre el sistema de gestión de contenidos (CMS) Plone (PLONE-FOUNDATION), aprovechando los excelentes mecanismo de extensión y las ventajas que el mismo brinda. Esta plataforma ha sido desarrollada completamente orientada a objetos e implementada en el lenguaje Python (PYTHON-SOFTWARE-FOUNDATION, 2007) lo cual facilita su entendimiento y desarrollo.

Por las razones anteriormente expuestas, se deriva el siguiente **problema** a resolver, la inexistencia de implementaciones en Python de algoritmos que permitan realizar de forma automática el agrupamiento de documentos por sus contenidos.

El presente trabajo está encaminado a dar solución al problema planteado. De modo que, tomando como **objeto de estudio** la Minería de texto, (*Text Mining)* la cual es una nueva área de investigación en el procesamiento de textos, que se encargada de descubrir conocimientos que no existían explícitamente en ningún texto de la colección, pero que surgen de relacionar el contenido de varios de ellos (GÓMEZ, 2002).

El **campo de acción** de este trabajo está enmarcado en el agrupamiento de documentos, el cual consiste esencialmente en asignar documentos a clases o categorías que no están previamente definidas, sino que son descubiertas sobre la base del análisis del contenido de los documentos.

En correspondencia con el problema planteado se formula como **objetivo general** del trabajo:

Realizar la implementación de una biblioteca de clases en el lenguaje Python que contenga algoritmos de agrupamiento de documentos.

A partir del objetivo general se derivaron los **objetivos específicos** siguientes**:**

1. Realizar una revisión de la literatura nacional e internacional actualizada y otras fuentes relacionadas con la temática en estudio, realizando un análisis del estado actual de los algoritmos de agrupamiento de documentos con el objetivo de construir el marco conceptual de este trabajo.

2. Seleccionar los algoritmos de agrupamiento de documentos e implementarlos en una biblioteca de clases.
3. Exponer los resultados obtenidos a partir de las implementaciones realizadas de los algoritmos.

Se formuló la siguiente **hipótesis**:

Con la implementación de una biblioteca de clases en Python que contenga algoritmos que permitan el agrupamiento de documentos; será posible aglomerar documentos de forma automática por sus contenidos, pudiendo ser integrada fácilmente con las aplicaciones que exijan estos requerimientos.

La **significación práctica** del presente trabajo esta dado por el hecho que, a partir de sus resultados:

- Se tendrán implementaciones de algoritmos de agrupamiento de documentos en el lenguaje de programación Python.
- Estarán sentadas las bases para continuar enriqueciendo la biblioteca con más implementaciones de algoritmos.
- Será posible realizar proyectos donde se apliquen estas técnicas.

El documento está estructurado en dos capítulos:

El Capítulo 1 describe el marco teórico referencial del trabajo, realizando un análisis del estado del arte del agrupamiento de documentos; se aborda al agrupamiento como un sub-proceso dentro de la minería de texto, se describen los principales métodos de clustering y se realiza una revisión de los algoritmos más usados.

El Capítulo 2 presenta la selección de los algoritmos a implementar y se detalla el diseño y la concepción de la biblioteca implementada en el lenguaje Python. Por último se evalúan los resultados obtenidos con la implementación de los algoritmos de agrupamiento de documentos. Esto es realizado usando unas métricas de calidad aplicadas a una colección de documentos previamente clasificada a través de un caso de estudio definido.

CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

Este capítulo describe los principales conceptos y teorías sobre el agrupamiento de documentos. Primeramente se presenta la minería de texto para enmarcar la temática en estudio, después se muestran los sistemas de aprendizaje como la rama de la Inteligencia Artificial que se ocupa de implementar estas técnicas, específicamente el clustering como una aplicación de los algoritmos de aprendizaje no supervisados. Luego, se explican las principales teorías del agrupamiento de datos, para después centrar la atención en el agrupamiento de documentos. Son presentadas las diferentes clasificaciones de usos del clustering, los métodos más comunes de clustering, las medidas de similitud que pueden ser usadas, se hace una revisión de los algoritmos más usados en la minería de texto para realizar el clustering de documentos y finalmente se presentan los requerimientos que exige realizar el clustering.

1.2 Minería de Texto

La minería de texto, también conocida como minería de datos textuales o descubrimiento de conocimiento desde bases de datos textuales, se define como el proceso de extraer patrones interesantes y nuevos conocimientos desde documentos de textos no estructurados. Tiene sus principales usos en la exploración de como evoluciona el mercado, en la observación de nuevas ideas o tópicos relacionados (GÓMEZ, 2002; , 2003). Esta constituye un campo multidisciplinario al tener involucrado otras disciplinas como: la recuperación de información, el análisis de los textos, la extracción de información, el clustering, la categorización, la visualización, el almacenaje de datos, y los sistemas de aprendizaje (VISA, 2001).

La minería de texto consta de dos etapas principales. La primera es llamada etapa de pre-procesamiento, donde los textos se transforman en algún tipo de representación estructurada o semi-estructurada que facilite su posterior análisis. La segunda es llamada etapa de descubrimiento, donde las representaciones intermedias se utilizan para descubrir los patrones interesantes y los nuevos conocimientos (TAN, 1999).

La forma más natural de almacenar información es en textos. Hay estudios realizados que demuestran que más del 80% de la información de una compañía está contenida en documentos de texto. Esto supone una alta complejidad, solo por el hecho de trabajar con datos textuales, que son esencialmente no estructurados y borrosos (FÉLIX, 2002).

1.3 Aprendizaje supervisado vs. Aprendizaje no supervisado

La inteligencia artificial es la ciencia que se ocupa de crear programas para las máquinas, con el objetivo de imitar el comportamiento y la comprensión humana (RUSSELL and NORVIG, 1995). Posee, como una de sus ramas, a los sistemas de aprendizaje, que tienen como objetivo el desarrollo de algoritmos y técnicas que le permitan a las computadoras aprender (NILSSON, 2005; WIKI, 2007b).

Dentro del conjunto de tipos de algoritmos que desarrollan, están los algoritmos de aprendizaje supervisado y no supervisado. Estos últimos son basados en una estrategia ambiciosa y difícil puesto que su planteamiento original no presupone ningún conocimiento previo sobre lo que se quiere aprender, a diferencia de los algoritmos de aprendizaje supervisado que si exigen tener conocimiento previo para llevar a cabo sus objetivos. Ellos inducen la descripción de un concepto a partir de la presentación de diferentes instancias de este. Una de las aplicaciones de los algoritmos de aprendizaje no supervisado, que será objeto de estudio, es el clustering (CLARKE, 2001).

1.4 Clustering de Datos

El clustering es la clasificación de objetos dentro de diferentes grupos de forma automática, donde los objetos de un mismo grupo o cluster son similares entre ellos y disimilares con los objetos pertenecientes a otros clusters. De los objetos se conocen sus descripciones pero se desconocen las relaciones que existen entre ellos. La cantidad de grupos a formar puede estar previamente definida o no (FRAKES and YATES, 1992; LUKE, 2007).

Al realizar el clustering es necesario tener en cuenta una serie de factores. En primer lugar el concepto de similitud que va a permitir comparar los objetos para poder decidir si están en el

mismo agrupamiento; en segundo lugar, sobre qué espacio de representación se va a trabajar dicho concepto; y en tercer lugar, cómo se va a trabajar dicho concepto en el espacio de representación (SHUCLOPER and TRINIDAD, 1995).

Según (SHUCLOPER and TRINIDAD, 1995) existen tres paradigmas para encontrar la estructura de los agrupamientos: el paradigma del coeficiente, el del solapamiento y el difuso. El paradigma del coeficiente plantea la formación de una partición del conjunto de objetos, donde las propiedades que caracterizan los objetos de un agrupamiento dado contradicen las propiedades de los demás agrupamientos. El paradigma del solapamiento permite que las agrupaciones tengan objetos comunes, o sea, que las propiedades que caracterizan un agrupamiento pudieran ser satisfechas por otros agrupamientos. Y el paradigma difuso plantea que, los objetos satisfacen en un cierto grado de pertenencia los agrupamientos formados, partiendo de la suposición conceptual de que un objeto no pertenece únicamente a un conjunto especifico.

Estos paradigmas son estudiados bajo dos enfoques diferentes: el clasificatorio y el conceptual. En el enfoque clasificatorio se tiene un universo de objetos y se necesita agruparlos de modos tal, que los objetos del mismo agrupamiento, se parezcan más entre sí, que con objetos de otros agrupamientos. En el enfoque conceptual se tiene un universo de objetos y es preciso agruparlos de modo tal que los objetos que estén en el mismo agrupamiento cumplan (en cierto grado) la propiedad que caracteriza al agrupamiento.

Para poder comparar los objetos, se necesita una medida que permita evaluar diferencias y similitudes para poder formar los clusters. La similitud es un concepto que expresa una medida de correspondencia o semejanza entre los objetos, permitiendo que ellos puedan compararse. La selección de la medida de similitud a usar depende del tipo de dato. Algunas medidas como las de correlación y las de distancia requieren datos métricos, mientras que las medidas de asociación requieren datos no métricos (NORES, 2000).

Si los objetos son representados en un plano geométrico Figura 1, cada punto representa un objeto, y entonces es posible identificar fácilmente cuatro clusters. Teniendo una función de distancia como la medida de similitud usada para determinar los cluster, dos o más objetos pertenecen al mismo cluster si ellos están cercanos de acuerdo a la distancia determinada.

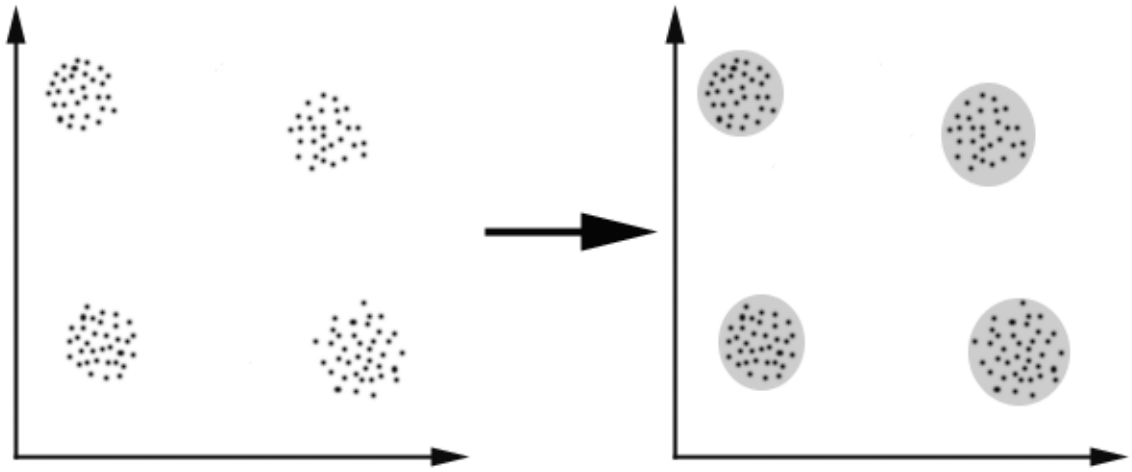

Figura 1 Objetos en el espacio geométrico

1.5 Clustering de Documentos

El clustering de documentos es ampliamente aplicado en diversas áreas como en la recuperación de información, en los motores de búsquedas y en la minería de texto. Estos métodos permiten mejorar el rendimiento de muchos sistemas.

Para aplicar los algoritmos de clustering se necesita que los documentos estén descritos o representados en un formato adecuado. En este trabajo se usará la representación del modelo de espacio vectorial (VSM) (YATES and NETO, 1998). En dicha representación cada documento es identificado como un vector de rasgos en un espacio en el cual cada dimensión corresponde a términos indexados distintos.

Un vector documento dado tiene, en cada componente, un valor numérico para indicar su importancia (STEINBACH *et al.*). Este valor es comúnmente determinado como una función de cuán frecuente aparece el término correspondiente en el documento particular y cuánto este aparece en total en la colección de documentos (ROBERTSON, 2004). Usando diferentes funciones de peso se obtendrán diferentes representaciones (RUBIO *et al.*, 2005).

Aunque muchos algoritmos de clustering han sido propuestos en la literatura, no todos satisfacen los requerimientos especiales que el clustering de documentos exige (EDNA, 2006; MATTEUCCI, 2003; RIJSBERGEN, 1999):

Escalabilidad: Deben permitir trabajar con grandes volúmenes de datos.

Habilidad para tratar con datos ruidosos: Ser eliminada la influencia de los datos con ruido (son datos que no poseen poder discriminante, ejemplo de ellos son las palabras de parada) en los resultados de los clusters.

Insensibilidad al orden de las observaciones de entrada: Que no afecte el orden de entrada de los datos al algoritmo en los resultados de los clusters que se obtengan.

Requerimientos mínimos en el conocimiento del dominio para determinar los parámetros de entrada: Que no sea necesario un experto para predefinir los agrupamientos.

Alta dimensionalidad: Permitir trabajar con espacios de gran dimensionalidad donde los objetos estén descritos por una gran cantidad de rasgos.

Precisión: La similitud intra-cluster debe ser alta y la similitud inter-cluster debe ser baja, por ejemplo los documentos que pertenecen al mismo cluster deben ser tan similares como se pueda y los documentos que pertenecen a clusters diferentes deben ser tan diferentes como se pueda.

1.6 Clasificación de los tipos de Clustering

Es importante conocer los tipos de clustering de documentos que son necesarios y factibles para una determinada aplicación. En esta sección se describen las principales clasificaciones de clustering, que permitirán después explicar mejor los métodos de agrupamiento y justificar la elección de cuales serán implementados (BERRY, 2004; VESTER and MARTINY, 2005).

1.6.1 Clustering duro vs. Clustering flexible

En muchas ocasiones la información puede ser relevante para varias categorías al mismo tiempo. A los métodos de clustering que pueden agrupar documentos en varias categorías se les llama métodos flexibles de clustering. En la Figura 2 se representa este concepto donde los documentos pueden aparecer en más de un cluster.

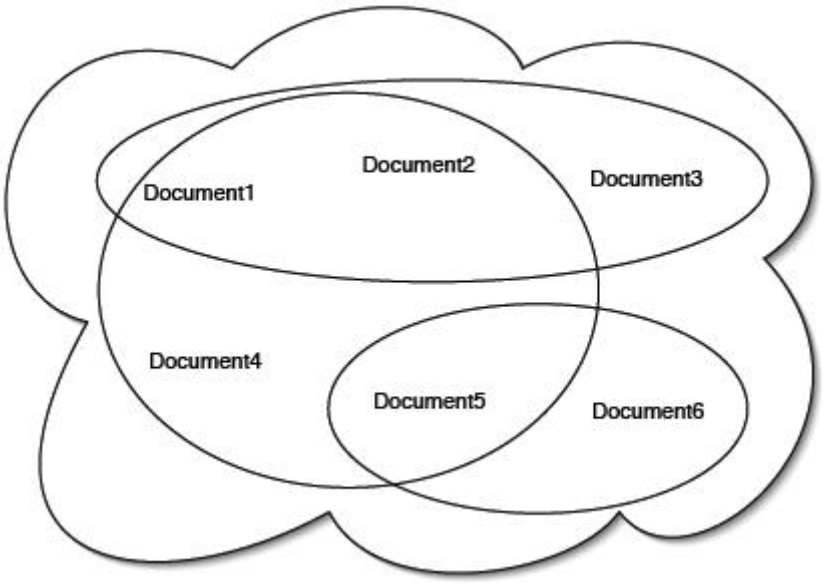

Figura 2 Documentos agrupados usando agrupamiento flexible

Pero no toda la información es clasificada de la misma forma. Ejemplo de ello es lo que ocurre en una librería donde los libros son ubicados en la categoría donde son más relevantes. Estos son llamados métodos duros de clustering. Este concepto es ilustrado en la Figura 3 donde los documentos cercanos son agrupados en un único cluster.

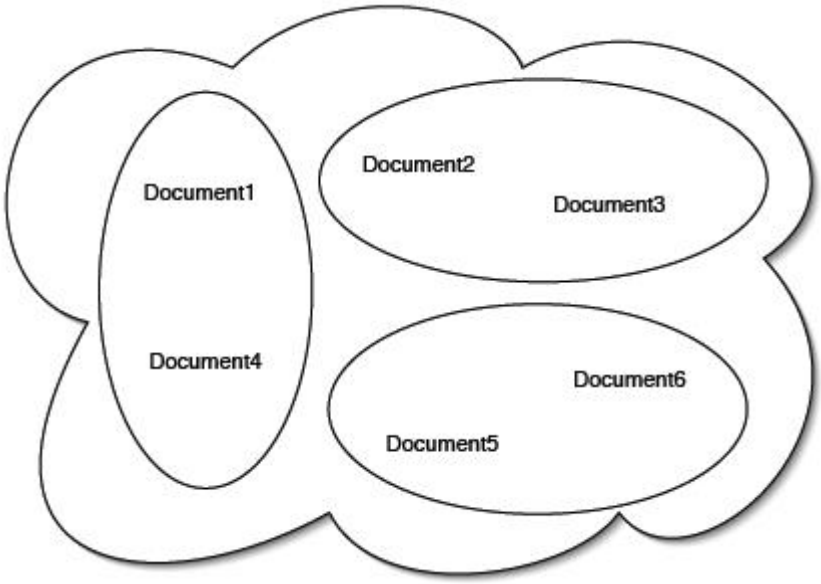

Figura 3 Documentos agrupados usando agrupamiento duro

1.6.2 Clustering jerárquico vs. Clustering no jerárquico

Al organizar información es importante saber el tipo de organización que se necesita. Cuando todos los documentos son puestos dentro de diferentes grupos que poseen tamaños aproximados y al solicitar un documento, el grupo donde está contenido se devuelve completamente, a estos métodos se les llaman, métodos de clustering no jerárquicos. Sin embargo cuando la información es organizada de forma que pueda recuperarse de diferentes formas, donde los grupos tienen diferentes tamaños y estos pueden dividirse en otros grupos, a estos métodos se les llaman, métodos de clustering jerárquicos. Estos pueden ser un caso especial de clustering flexible.

1.6.3 Clustering en línea vs. Clustering fuera de línea

Esta clasificación es para diferenciar cuándo debe ser ejecutado el clustering, el cual depende de varios factores como el tamaño de la dimensionalidad, la representación y la complejidad del algoritmo usado. En el caso donde sea necesario un alto procesamiento y las actualizaciones de los documentos sean frecuentes, el clustering fuera de línea es el más apropiado. Cuando no

existan actualizaciones frecuentes, un conjunto de datos reducidos y se cuente con rápidos algoritmos de clustering, entonces el más apropiado será el clustering en línea.

1.7 Métodos Comunes de Clustering

En la revisión de la literatura aparecen diferentes acercamientos al agrupamiento de documentos. Se puede mencionar cuatro categorías principales basadas en (VESTER and MARTINY, 2005).

1.7.1 Clustering basado en particiones

Los algoritmos basados en particiones dividen la colección de documentos en un número de clusters duros usando una medida de distancia dada. Estos son divididos en métodos de simple pasada y métodos basados en la reasignación (FRAKES and YATES, 1992; MENDONCA *et al.*, 1999).

1.7.1.1 Métodos de simple pasada

Los algoritmos de simple pasada usan un enfoque ambicioso (WIKI, 2007a) de asignar cada documento a un cluster de una sola vez. Ellos de esta forma son muy rápidos pero el costo de la calidad es afectado, pues no dejan margen para corregir errores.

La más simple implementación consiste en asignar el primer documento a un nuevo cluster y los siguientes documentos a los cluster existentes más cercanos, dependiendo de la distancia entre estos, y si esta excede un umbral predeterminado entonces es asignado a un nuevo cluster.

1.7.1.2 Métodos de reasignación

Los algoritmos basados en reasignación corren en múltiples pasadas. Primero inicializan las particiones a formar y luego corren de forma iterativa, aplicando técnicas de reasignación para tratar de mejorar las particiones moviendo documentos entre particiones.

1.7.2 Clustering jerárquico

El clustering jerárquico se enfoca en intentar crear una jerarquía de descomposición de la colección de documentos. Dependiendo de cómo se vaya a construir la estructura de la jerarquía

los métodos se clasifican en aglomerativos y divisivos (MENDONCA *et al.*, 1999; STEINBACH *et al.*, 2002).

1.7.2.1 Métodos aglomerativos

Los métodos aglomerativos comienzan considerando cada documento de la colección como un cluster separado. Los clusters cercanos son agrupados continuamente usando una medida de similitud inter-clusters hasta obtener un solo cluster con todos los elementos de la colección u obtener un número de clusters predefinido.

Ellos son clasificados de acuerdo a la medida de similitud inter-cluster que usen. Las más populares son: Single-Link, Complete-Link, Group Average y Ward.

1.7.2.1.1 Single-Link

Son algoritmos basados en esta medida de similitud que juntan dos cluster que contienen dos documentos cercanos que todavía no están en el mismo cluster (Figura 4). La similitud entre dos grupos es la máxima de las similitudes entre todos los pares de documentos, tales que uno está en un grupo y el otro en el segundo grupo. Cada documento de un cluster será más similar, como mínimo, a otro miembro de su mismo cluster, que a cualquier miembro de otro.

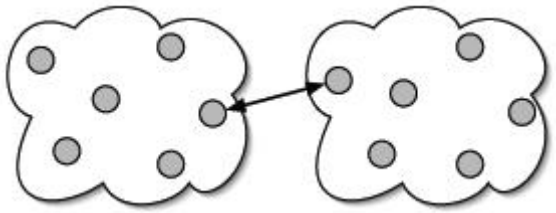

Figura 4 La medida de similitud de Single-Link

1.7.2.1.2 Complete-Link

Son algoritmos de clustering que usan esta medida de similitud para unir los clusters basándose en la mínima distancia obtenida entre los documentos más alejados, que todavía no están en el mismo cluster (Figura 5). Cada documento de un grupo será más similar al documento más diferente de su grupo que al más diferente de cualquier otro grupo.

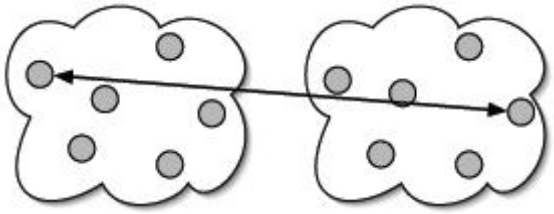

Figura 5 La medida de similitud de Complete-Link

1.7.2.1.3 Group Average

Son algoritmos que usan esta medida de similitud para unir cluster con la mínima distancia promedio entre los centros de los clusters Figura 6.

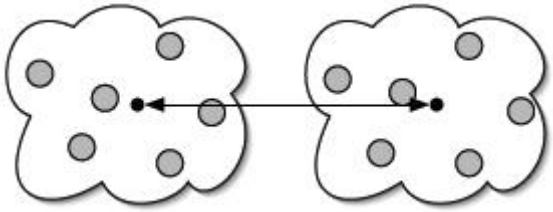

Figura 6 La medida de similitud de Group Average

1.7.2.1.4 Ward

Es también conocido como el método de la mínima varianza porque este, junta en cada etapa el par de clusters que minimicen el incremento del total de la suma del error cuadrado dentro del grupo, basado en la distancia Euclidiana entre centroides. Este tiende a producir clusters homogéneos y organización simétrica.

1.7.2.2 Métodos divisivos

Los algoritmos divisivos comienzan con un simple cluster donde están contenidos todos los documentos. Este es continuamente dividido hasta obtener cada documento en un cluster u obtener un número predefinido de clusters. Estos son significativamente más rápidos que los aglomerativos.

1.7.3 Clustering basado en palabras claves

Estos no son realmente una clase de métodos de algoritmos, sin embargo estos algoritmos son factibles cuando existen limitaciones en el conjunto de características por documentos. Estos trabajan asignando un conjunto de palabras claves por cada documento y usan estos términos como características para los siguientes agrupamientos. Estos siguen la idea de la hipótesis Topic-binder; donde un conjunto de documentos que compartan términos comunes y estos aparezcan únicamente en el mismo tema, entonces estos documentos deberán ser semejantes entre ellos y por tanto pueden estar en el mismo cluster.

1.7.4 Clustering basados en modelos

En estos métodos se sigue la hipótesis de los modelos, donde los documentos ocupan el cluster donde el modelo se ajuste mejor. Estos pueden ser duros y flexibles así como jerárquicos o no jerárquicos. Ellos cubren una variedad de acercamientos, pero los principales son los métodos estadísticos y los métodos basados en redes neuronales. Los métodos estadísticos usan análisis estadístico sobre la colección de documentos. Los métodos basados en redes neuronales emulan internamente el funcionamiento del cerebro humano.

1.7.5 Otros métodos de clustering

Están los métodos borrosos, los cuales son un tipo especial de clustering flexible donde se asigna un grado de pertenencia a los documentos por clusters que indican la medida en que el documento pertenece al cluster. La suma de las pertenencias de un documento a todos los clusters es igual a uno (KLAWONN and HÖPPNER, 2003).

1.8 Medidas de Similitud

Cuando se agrupa objetos es necesario conocer el grado de asociación que poseen para poder determinar a cuál cluster serán asociados. Esta puede estar dada por una medida de distancia ó una medida de similitud ó disimilaridad. Algunos algoritmos de clustering exigen una medida específica pero otros dejan a elección, cual usar (FRAKES and YATES, 1992).

Existen diferentes medidas y estas pueden influir en el resultado de los clusters, lo cual ha sido demostrado en estudios realizados (STREHL, 2002). En la comparación de vectores documentos, los coeficientes Dice, Jaccard y el Coseno han mostrado los mejores resultados (FRAKES and YATES, 1992).

El Coeficiente Dice plantea:

$$S(D_i, D_j) = \frac{2\sum_{k=1}^{L}\left(peso_{ik} \cdot peso_{jk}\right)}{\sum_{k=1}^{L} peso^2_{ik} + \sum_{k=1}^{L} peso^2_{jk}}$$

Si el peso de los términos es binario la ecuación se reduce a:

$$S(D_j, D_j) = \frac{2C}{A+B}$$

Donde *C* es el número de términos que D_i y D_j tienen en común, y *A* y *B* son el número de términos de D_i y D_j respectivamente.

El coeficiente Jaccard se define como:

$$S(D_i, D_j) = \frac{2\sum_{k=1}^{L}\left(peso_{ik} \cdot peso_{jk}\right)}{\sum_{k=1}^{L} peso^2_{ik} + \sum_{k=1}^{L} peso^2_{jk} - \sum_{k=1}^{L} peso_{ik} - peso_{jk}}$$

El Coeficiente Coseno se define como sigue:

$$S(D_i, D_j) = \frac{\sum_{k=1}^{L}\left(peso_{ik} \cdot peso_{jk}\right)}{\sqrt{\sum_{k=1}^{L} peso^2_{ik} \cdot \sum_{k=1}^{L} peso^2_{jk}}}$$

Esta medida es ampliamente usada y basa su funcionamiento en el coseno del ángulo entre dos documentos. Si el ángulo es cercano a uno, entonces estos documentos son iguales, y diferentes si son cercanos a cero.

Otras medidas son referenciadas en la literatura (ABDI, 2007) como la distancia Euclidiana, la cual es altamente usada. Y se determina mediante la ecuación:

$$D(O_i, O_j) = \sqrt{\sum_{k=1}^{L} (O_{ik} - O_{jk})^2}$$

Donde los objetos son iguales si la distancia entre ellos es cercana a cero y diferentes si es cercana a uno. La métrica de Minkowski (BERGO, 2001) se calcula mediante la fórmula:

$$D(O_i, O_j) = \left(\sum_{k=1}^{L} \left| O_{ik} - O_{jk} \right|^{\gamma} \right)^{\frac{1}{\gamma}}$$

Cuando $\gamma \geq 1$ es llamada métrica de Manhattan, si $\gamma = 2$ esta se refiere a la distancia Euclidiana y cuando $\gamma > 2$ estamos en presencia de la métrica de Supermum.

La métrica de Canberra (BERGO, 2001) también permite medir la distancia entre dos objetos y se calcula mediante la ecuación:

$$D(O_i, O_j) = \sum_{k=1}^{L} \frac{\left| \vec{O}_{ik} - \vec{O}_{jk} \right|}{\left(\left| \vec{O}_{ik} \right| + \left| \vec{O}_{jk} \right| \right)}$$

1.9 Algoritmos de clustering en la Minería de Texto

El algoritmo K-Means (BERRY, 2004) ha sido considerado un estándar dentro del clustering y es probablemente el más popular. Este se ha convertido en la base de muchos otros algoritmos, dentro de ellos están los algoritmos Batch K-Means, Incremental K-Means y el Bisecting K-Means.

El Batch K-Means (BERRY, 2004) tiene dos desventajas, una de ellas es que la calidad de la partición final depende de una buena selección de la partición inicial y además puede quedar atrapado en mínimos locales. El Incremental K-Means resuelve las dos desventajas del Batch K-

Means, pero es más lento. Como resultado de mejoras realizadas a estos dos últimos algoritmos se obtuvo el algoritmo Means (BERRY, 2004).

El algoritmo Bisecting K-Means (SAVARESI *et al.*, 2002) consiste en dividir recursivamente un cluster en dos clusters partiendo del conjunto de datos. Obteniendo una estructura jerárquica y basándose en el centroide al igual que el K-Means.

El K-Means presenta la desventaja de tener que estimar los parámetros de entrada. Esto puede traer una pobre precisión al agrupar, además de no ser adecuado para descubrir clusters con grades variaciones en el tamaño, Este algoritmo es sensitivo al ruido que pueda tener una influencia significativa en el centro del cluster, la cual a su vez puede bajar la precisión del clustering (FUNG *et al.*). El algoritmo K-Medoids (WIKI, 2006) surgió para resolver el problema del ruido, pero es computacionalmente mucho más costoso y no escala bien para grandes conjuntos de documentos.

El algoritmo Principal Direction Divisive Partitioning (PDDP) (BERRY, 2004) es uno de los más recientes. El mismo hace un uso eficiente de la memoria, no es basado en una medida de distancia o medida de similitud. Este algoritmo toma ventajas de lo dispersa que es la matriz de términos por documentos. Su funcionamiento consiste en dividir la colección en dos clusters, y cada cluster en dos clusters usando un proceso recursivo que se detiene cuando el cluster alcanza un cierto grado de calidad predefinido. Otro que sigue este mismo funcionamiento es el algoritmo Spherical Principal Directions Divisive Partitioning (sPDDP), ambos se conjugan con el algoritmo Means para obtener mejores resultados, pero este último ofrece los mejores (BERRY, 2004).

El algoritmo Simultaneous Clustering and Attribute Discrimination (SCAD) (FRIGUI and NASRAOUI, 2002) está basado en el algoritmo K-Means y tiene como ventajas que realiza los agrupamientos y el pesado de los rasgos simultáneamente, permitiendo una representación más rica de relevancia de los rasgos que la selección binaria. Además, aprende de las diferentes relevancias de los rasgos en cada cluster, y usa como medida de similitud la distancia Euclidiana. Está comprobado en el agrupamiento de documentos que sus resultados no son los más apropiados, debido a que no lo es para altas dimensionalidades. El algoritmo ha sido extendido y

el nuevo acercamiento es llamado Simultaneous Keyword Identification and Clustering of Text Documents (SKWIC). Este algoritmo es conceptualmente y computacionalmente simple y funciona mejor que el K-Means cuando no todos los rasgos son igualmente relevantes. Usa una medida de disimilaridad basada en el coeficiente coseno, y puede ser adaptado a otras medidas de similitud. (FRIGUI and NASRAOUI, 2002).

El algoritmo Simultaneous Soft Clustering and Term Weighting of Text Documents (Fuzzy SKWIC), (BERRY, 2004) sigue el mismo funcionamiento que el algoritmo SKWIC pero incorpora el grado de asociación que tiene un documento con cada cluster. Esto parte del concepto que un documento no pertenece únicamente a una categoría, sino que ellos pueden tender a estar en dos o más categorías. Con esta propuesta se puede conocer además del pesado de los rasgos en cada cluster, en cuanto un documento pertenece a cada cluster.

De forma tradicional el clustering genera particiones, donde cada objeto pertenece a un único cluster. Extendiendo esta noción surge el clustering borroso, el cual asocia cada objeto a cada cluster usando una función de pertenencia para asignar los objetos (KLAWONN and HÖPPNER, 2003).

Un clásico dentro la extensión anterior para obtener clustering borroso lo constituye el algoritmo Fuzzy C-Means (MATTEUCCI, 2003; MUCHA and SOFYAN, 2003). Este constituye un acercamiento común y muy popular dentro de dicha clasificación, exigiendo como requisito que la cantidad de clusters a obtener sea fijado a priori. La suma de las pertenencias de cada objeto a cada clusters es igual a 1.

Otro algoritmo llamado Suffix Tree Clustering (ZAMIR and ETZIONI, 1998) no trata los documentos como un conjunto de palabras sino más bien como cadenas y hace uso de la proximidad entre palabras. Este identifica eficientemente conjuntos de documentos que compartan frases comunes y usa esta información para crear los clusters y resumir estos contenidos para los usuarios.

1.10 Cuestiones del Clustering

Para obtener los mejores resultados en los agrupamientos se debe enfrentar ciertos retos, los cuales son descritos a continuación (FRAKES and YATES, 1992):

1. Seleccionar apropiadamente las características de los documentos que deben ser usadas para el clustering.
2. Seleccionar apropiadamente la medida de similitud entre los documentos.
3. Seleccionar el método de clustering apropiado para la medida de similitud seleccionada.
4. Implementar el algoritmo de clustering de forma eficiente que sea factible en requerimiento de memoria y recursos de CPU.
5. Buscar formas de evaluar la calidad de los clusters obtenidos.
6. Buscar formas factibles para actualizar los clusters en caso de aparecer nuevos documentos para ser adicionados a la colección.

Por tanto en las técnicas de clustering actuales la efectividad de los métodos depende de la medida de similitud seleccionada. Al aumentar las dimensiones y elevarse los volúmenes de datos se genera una complejidad que puede convertirse en un problema. En muchos casos el resultado de las técnicas de clustering puede ser arbitrario y, por tanto, ser interpretados de formas diferentes (MATTEUCCI, 2003).

1.11 Conclusiones parciales

Después de haber realizado la revisión de la literatura se puede concluir que la representación textual es una etapa fundamental para el correcto funcionamiento del proceso de clustering. Ha quedado evidenciado que es una generalización usar en los procesos de agrupamiento de documentos el coeficiente coseno como medida de similitud. Se ha seleccionado para su implementación los algoritmos K-Means y SKWIC teniendo en cuenta que el primero constituye la base de muchos algoritmos y el último porque logra calcular la relevancia de las palabras por grupos y agrupar simultáneamente. Este resultado es de gran importancia debido a que dichos agrupamientos serán usados para realizar resúmenes de texto.

CAPÍTULO 2 IMPLEMENTACIÓN DE LA BIBLIOTECA DE ALGORITMOS DE AGRUPAMIENTO

2.1 Introducción

Después de realizar la revisión de la literatura sobre los distintos métodos y algoritmos de clustering de documentos, este capítulo se propone describir la implementación de una biblioteca de clases en el lenguaje Python con los algoritmos de clustering seleccionados. El mismo está estructurado por tres epígrafes principales. El primero está dedicado a la elección de los algoritmos que posteriormente serán objeto de implementación. El segundo está orientado a la explicación de cómo se ha concebido el diseñó e implementación de la biblioteca y en el tercero serán mostrados los resultados de la evaluación de las implementaciones.

2.2 Selección de los algoritmos a implementar

A partir de la bibliografía consultada sobre los algoritmos de clustering de documentos, se ha decidido llevar a cabo la implementación de los algoritmos K-Means y Simultaneous Keyword Identification and Clustering of Text Documents (SKWIC).

Para su elección partimos inicialmente de analizar el contexto del problema a resolver, donde la principal dificultad es que no existen implementaciones en Python de estos algoritmos que permitan poder usarlos en productos que sean desarrollados bajo estos requerimientos.

Además se conoce que estos serán usados para agrupar documentos de texto, lo cual supone tener que tratar con una alta dimensionalidad en su representación. También se tiene presente cuáles serán los usos que tendrán los agrupamientos una vez realizado el proceso de clustering.

Los algoritmos seleccionados pertenecen a la clasificación de algoritmos particionales y dentro de ellos a los algoritmos de reasignación. Si es verdad que los algoritmos jerárquicos obtienen una organización en forma de árbol llamada dendograma, la cual facilita saber cuán cerca se encuentra un grupo respecto a otros, estos presentan un inconveniente significativo y es que cuando asignan un documento a un cluster este no vuelve a ser movido. Esto hace que errores cometidos no puedan ser corregidos.

En cambio los algoritmos de reasignación pueden corregir en sucesivas iteraciones errores introducidos en las asignaciones anteriores. Ellos se caracterizan por ser más eficientes en cuanto a coste computacional y rendimiento. Ambos algoritmos parten de la representación del VSM y devuelven una colección de clusters que su estructura está en dependencia del algoritmo de agrupamiento.

2.2.1 Algoritmo K-Means

2.2.1.1 Descripción

Por largo tiempo K-Means ha sido considerado el estándar dentro del clustering y permanece con gran fuerza en este campo (VESTER and MARTINY, 2005). Lo primero que necesita es que sea especificado con anticipación el número de cluster deseado: este es el parámetro k. Entonces k documento son elegidos aleatoriamente y asignados como centros de clusters. Seguidamente todos los documentos son asignados al centros de cluster más cercano de acuerdo a una medida de distancia. Después los centroides o medios de los centro de clusters son calculados. Estos centroides son tomados para ser los nuevos centros para sus respectivos clusters. Finalmente se repite el proceso completo con los nuevos centros.

Esto ocurrirá iterativamente hasta que los centros se estabilicen o estos no cambien mucho. Este método es simple, efectivo y fácil de probar, busca en su funcionamiento minimizar una función objetivo dada por el cuadrado total de la distancia desde cada documento de cada cluster a cada centro de cluster.

La función objetivo es

$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$, donde $\left\| x_i^{(j)} - c_j \right\|^2$ es la medida de distancia elegida entre un objeto x_i y el cluster c_j. Este es un indicador de la distancia de cada objeto a partir de los respectivos centros.

2.2.1.2 Algoritmo

El algoritmo fue propuesto por (MACQUEEN, 1967), y está compuesto por los siguientes pasos:

1. Elegir el número de clusters, k
2. Generar aleatoriamente k clusters y determinar el centro de los clusters, o directamente tomar aleatoriamente k objetos como los centros del clusters.
3. Asignar cada objeto al centro de cluster más cercano.
4. Recalcular los nuevos centros de clusters.
5. Repetir el paso 3 y 4 hasta que el criterio de convergencia sea alcanzado (usualmente que las asignaciones no cambien entre una iteración y otra o los centros no cambien mucho)

2.2.1.3 Tiempo y complejidad

Según las funciones implementadas para los pasos del algoritmo, se tiene la siguiente complejidad:

1. La inicialización de los centros de cluster es O(k), donde k es el número de clusters deseados.
2. El principal ciclo se repite a lo sumo t veces, donde t es el paso límite fijado (donde un documento es movido continuamente de un centro a otro hasta cumplir con el paso límite).
3. Para cada documento, se calculan todas las distancias a los centros de cluster, esta es O(kn), donde n es el número de documentos en el espacio de documentos.
4. Para cada cluster, el centroide debe ser calculado, pero como este es clustering particional solo se tienen documentos y la complejidad es O(n).
5. Finalmente, la función de error es calculada, y otra vez esto ocurre para todos los documentos y se arriba a una complejidad O(n).

La complejidad del tiempo de corrida de K-Means es $O(tkn)$

2.2.1.4 Ventajas

La más importante que presenta este algoritmo es su simplicidad y rapidez; la cual permite correr sobre largos conjuntos de datos. Esto es tenido en cuenta para minimizar el problema de tomar aleatoriamente los centros de cluster, permitiendo realizar varias corridas del algoritmo y tomar aquellos centros de cluster que favorezcan más a los agrupamientos.

2.2.1.5 Desventajas

La principal dificultad es que los resultados obtenidos en cada corrida son diferentes. Debido a que el resultado encontrado depende de la asignación aleatoria inicial de los centros de clusters. Este puede quedar atrapado en mínimos locales y afectar la calidad final de la solución. Además el hecho de tener que fijar a priori el número de clusters a obtener puede conllevar a resultados desconocidos.

2.2.1.6 Medida de similitud

Este algoritmo permite que el investigador elija cual medida de similitud usar, lo cual posibilita ser aplicado a varios dominios de aplicación. En este caso, cuando se hace frente al agrupamiento de documentos donde existe una alta dimensionalidad, ha sido utilizado para su implementación la medida del coeficiente coseno del ángulo entre los vectores que representan a los documentos. Siendo esta la más usada en dominios textuales por lograr mejores resultados.

2.2.2 Algoritmo SKWIC

2.2.2.1 Descripción

El SKWIC es un algoritmo basado en K-Means. Tiene como finalidad buscar una alternativa para el pesado de las palabras claves de los documentos, puesto que en ocasiones se ve afectado el resultado del clustering por considerar a todas las palabras por igual en cada cluster. El mismo parte de considerar que grupos diferentes tendrán un conjunto de palabras claves diferentes. De ahí que se necesite descubrir simultáneamente las palabras claves que representan a los grupos a medida que se obtienen estos.

2.2.2.2 Algoritmo

El algoritmo fue propuesto por (FRIGUI and NASRAOUI, 2002) y consta de los siguientes pasos:

1. Fijar el número inicial de clusters.
2. Inicializar aleatoriamente los centros de clusters.
3. Inicializar las particiones y la relevancia de los términos.

REPETIR

4. Calcular distancia entre características de cada documento y el centro de cluster (Ecuación 1).
5. Actualizar la relevancia de los términos por clusters v_{ik} (Ecuación 3).
6. Actualizar cada partición cluster Xi (Ecuación 1 Ecuación 5).
7. Actualizar los pesos δ_i por clusters (Ecuación 4).
8. Actualizar los centros de cluster c_{ik} (Ecuación 6).

HASTA (centros estabilizados)

En dicho algoritmo se fija el número de clusters a obtener y se asigna aleatoriamente un documento como centro de cada cluster. Seguidamente se inicializan las particiones (X_i) pertenecientes a cada cluster basándose en la (Ecuación 5) y considerando que los pesos de las palabras son iguales a $\frac{1}{n}$. Para calcular la medida de disimilitud que está basada en el coeficiente coseno se necesita la (Ecuación 1) y la ecuación (Ecuación 2).

$$\tilde{D}_{wcij} = \sum_{k=1}^{n} v_{ik} D^{k}_{wcij} \qquad \text{Ecuación 1}$$

$$D^{k}_{wcij} = \frac{1}{n} - \left(x_{jk}.c_{ik}\right) \qquad \text{Ecuación 2}$$

Donde n el es el número total de términos en la colección de *N* documentos, c_{ik} es el *k*-ésimo componente del vector centro del *i*-ésimo cluster, y *V*= [v_{ik}] es la relevancia del término *k* en el

cluster *i*. El cálculo de la relevancia de los términos por clusters se realiza a partir de la (Ecuación 3), donde el peso δ_i se calcula mediante la (Ecuación 4):

$$v_{ik} = \frac{1}{n} + \frac{1}{2\delta_i} \sum_{x_j \in X_i} \left[\frac{1}{n} \sum_{x_j \in X_i} \tilde{D}^{k}_{wc_{ij}} - D^{k}_{wc_{ij}} \right] \qquad \text{Ecuación 3}$$

$$\delta_i^{(t)} = K_\delta \frac{\sum_{x_j \in X_i} \sum_{k=1}^{n} v_{ik}^{(t-1)} \left(D^{k^{(t-1)}}_{wc_{ij}} \right)}{\sum_{k=1}^{n} v_{ik}^{(t-1)^2}} \qquad \text{Ecuación 4}$$

Donde K_δ es una constante y el subíndice (*t*-1) es usado sobre v_{ik} y c_{ik} para denotar sus valores en la iteración (*t*-1). Para asignar los documentos a los clusters se toma la menor distancia de cada documento a cada centro de cluster basado en la (Ecuación 5). Al final se deben recalcular los centros de cluster mediante la (Ecuación 6)

$$X_i = \left\{ x_j \middle| \tilde{D}_{wc_{ij}} \leq \tilde{D}_{wc_{kj}} \forall\, k \neq i \right\} \qquad \text{Ecuación 5}$$

$$c_{ik} = \begin{cases} 0, & if\ v_{ik} = 0 \\ \dfrac{\sum_{x_j \in x_i} x_{jk}}{\lfloor x_i \rfloor}, & if\ v_{ik} > 0 \end{cases} \qquad \text{Ecuación 6}$$

2.2.2.3 Tiempo y complejidad

Según las funciones implementadas para los pasos del algoritmo se tiene la siguiente complejidad:

1. La inicialización de los centros de cluster es O(k), donde k es el número de clusters deseados.
2. La inicialización de la relevancia de cada cluster es O(k).
3. La inicialización de las particiones es O(k n^3), donde n es el número de documentos en el espacio de documentos.
4. El principal ciclo se repite a lo sumo t veces, donde t es el paso límite fijado (donde un documento es movido continuamente de un centro a otro hasta cumplir con el paso limite).
5. Para cada cluster, el coeficiente δ_i es calculado, esta es O(n).
6. Para cada documento y cada centro de cluster, se calculan todas las distancias entre términos esta es O(k n^3).
7. Para cada clusters, la relevancia de cada término es calculada; esta es O(n).
8. Para cada documento, se calculan las distancias a los centros de cluster, esta es O(kn).
9. Para cada cluster el centroide debe ser calculado, pero como este es clustering particional solo se tienen documentos y la complejidad es O(n).
10. Finalmente, la función de error es calculada, y otra vez, esto ocurre para todos los documentos y se arriba a una complejidad O(n).

La complejidad del tiempo de corrida de SKWIC es $O(tkn^3)$

2.2.2.4 Ventajas

Su principal ventaja es que realiza el pesado continuo de los términos, mejorando así la relevancia de la representación. Además permite que no todos los términos sean igualmente relevantes para todos los grupos. El pesado continuamente ayuda a particionar la colección de

documentos dentro de las categorías más significativas para los términos que más la representen. También puede ser generada una descripción compacta de cada cluster en términos no solo de los valores de las palabras claves sino también de su relevancia.

2.2.2.5 Desventajas

Posee al igual que K-Means la restricción de tener definido con anticipación los clusters que se desean obtener. De ahí que la calidad de la agrupación estará afectada, así como los diferentes resultados que serán obtenidos en dependencia de la inicialización de los centros.

2.2.2.6 Medida de similitud

El mismo ha sido implementado con una variación de la medida de distancia coeficiente coseno, teniendo en cuenta que se tratará con altas dimensionalidades. El mismo permite que pueda ser adaptado a la medida de similitud que se desee implementar.

2.3 Diseño e implementación de la biblioteca de clases

Una vez elegido cuáles algoritmos implementar y conociendo sobre los diferentes requerimientos que se necesitan para abordar su implementación, se hace necesario realizar un diseño para la biblioteca de clases que permita la implementación de los algoritmos de clustering de documentos seleccionados. Además, el diseño debe brindar el soporte sobre una infraestructura capaz de permitir extender dicha biblioteca de clases con más algoritmos en el futuro.

2.3.1 Herramientas y lenguaje utilizado

Al tener como uno de los requisitos en el desarrollo de la biblioteca de clases, la utilización del lenguaje de programación Python, se ha utilizado la flexibilidad que este aporta para aumentar la productividad del código. Este lenguaje es conocido por soportar el paradigma orientado a objetos, sin embargo es posible hacer uso del paradigma estructurado que a veces resuelve algunos problemas. Además ha sido aprovechada su característica de ser interpretado para realizar pruebas durante el desarrollo del código.

Para llevar adelante el desarrollo de la biblioteca de clases han sido usadas un conjunto de herramientas, las cuales han facilitado el proceso de diseño e implementación. Para la codificación de la biblioteca de clases en dicho lenguaje; si bien existen una variedad de ambientes de desarrollo, se ha elegido la plataforma extensible de desarrollo eclipse (ECLIPSE-FOUNDATION, 2007). Es una plataforma desarrollada en el lenguaje Java que funciona sobre varios sistemas operativos. Su principal característica es la de poder ser extendido mediante un mecanismo de plugins que implementa. Estos plugins son funcionalidades extra que se le pueden adicionar. Para las de este trabajo se ha adicionado el plugin Pydev (COMMUNITY-PYDEV, 2007), el cual aporta el soporte para la codificación en el lenguaje Python.

Durante la implementación se ha decido llevar las versiones del código para una mejor gestión de las versiones y trabajo con el código fuente, aprovechando las ventajas que nos brindan los sistemas de control de versiones. Para ello se ha usado un servidor de control de versiones muy usado actualmente por los desarrolladores: subversion (COMMUNITY-SUBVERSION, 2007). Para lograr una integración entre subversion y el entorno de desarrollo fueron adicionados un plugin llamado Subeclipse (COMMUNITY-SUBECLIPSE, 2007) que funciona como cliente del subversion y permite la interacción con el servidor de control de versiones desde el Eclipse.

Durante el proceso de diseño para llevar a cabo los diagramas de clases se hizo uso de la herramienta de ingeniería de software asistida por computadoras (CASE), Enterprise Architect (EA) (SPARX-SYSTEMS, 2007) en su versión 6.1. EA es una herramienta flexible, completa y potente de modelado para el lenguaje unificado de modelo (UML), que permite realizar ingeniería de código inversa y reversa para varios lenguajes, entre ellos Python.

2.3.2 Modelo de diseño de la biblioteca de clases

La biblioteca sigue un diseño basado en interfaces para lograr una arquitectura fácil de extender y reutilizar. En su estructura se ha decidido tener una organización a nivel de módulos donde estos están especializados en implementar una determinada responsabilidad. En la Figura 7 se puede observar las dependencias entre los módulos que conforman la biblioteca. En esta sección se describe detalladamente el funcionamiento de los módulos.

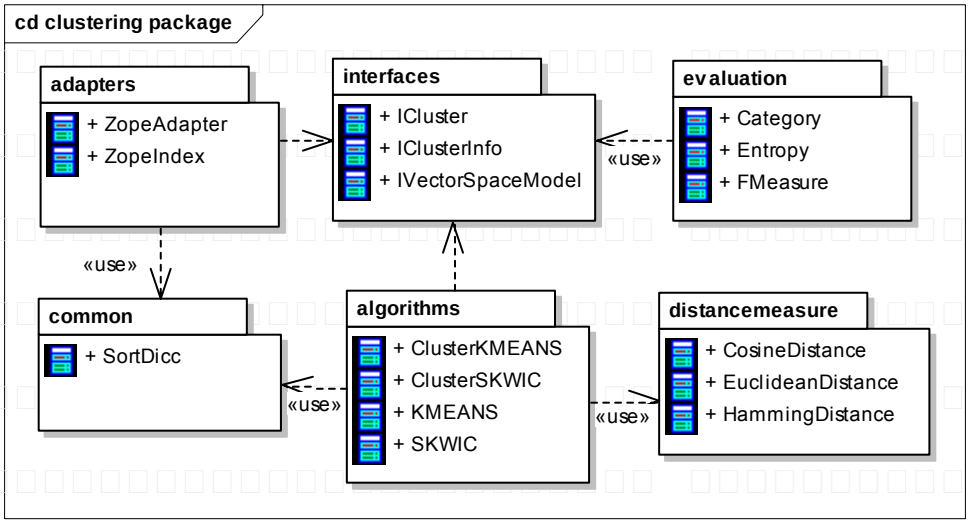

Figura 7 Dependencias entre módulos

2.3.2.1 Módulo de interfases

El módulo de interfases tiene la responsabilidad de alojar cualquier interfaz que se vaya a definir para la biblioteca. En el mismo se han definido tres interfaces principales ver Figura 8:

La interfaz ClusterInfo ha sido diseñada para estandarizar los resultados básicos que cualquier algoritmo de cluster debe devolver sobre la información básica que un cluster debe aportar. Entre los resultados está el documento centro del cluster y la lista de documentos que pertenecen al cluster. Con la información que aporta dicha interfaz es posible estandarizar el cálculo de las medidas de calidad para evaluar los agrupamientos realizados por los algoritmos.

La interfaz ICluster tiene como responsabilidad manejar el comportamiento que deben exponer los algoritmos de clustering implementados. Se han defino dos responsabilidades principales; dado la representación del VSM devolver un listado con los clusters obtenidos después de aplicar el proceso de agrupamiento cumpliendo con la interfaz ClusterInfo; y segundo devolver el nombre del algoritmo de clustering implementado.

La interfaz IVectorSpaceModel se ha diseño para estandarizar la comunicación con los índices que almacenan el espacio vectorial y su responsabilidad es abstraer de la implementación a los

algoritmos de clustering en la utilización del VSM. Esto ha sido logrado con la utilización de un patrón de diseño llamado Adapter; en la sección 2.3.2.7 será abordado en detalle.

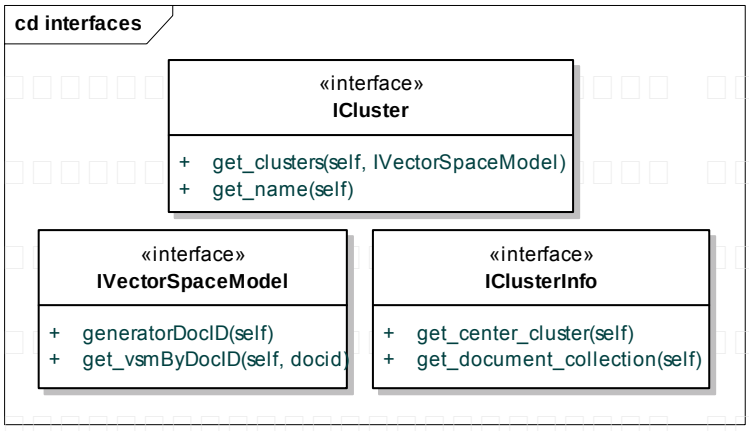

Figura 8 Diagrama de clases del módulo de interfaces

2.3.2.2 Módulo de medidas de distancia

Este módulo ha sido concebido para implementar las diferentes medidas de distancias que serán utilizadas por los algoritmos en el proceso de agrupamiento. Se encuentran implementadas tres medidas: la distancia de Hamming, la euclidiana y la distancia coseno. Esta última es las más usada en agrupamiento de documentos.

2.3.2.3 Módulo común

Como su nombre lo indica este módulo ha sido concebido para concentrar las responsabilidades compartidas por otros módulos para de este modo reutilizar porciones de código. Actualmente hay una funcionalidad implementada que permite ordenar una estructura de un diccionario de Python por su valor.

2.3.2.4 Módulo de evaluación

Este es el módulo responsable de implementar las medidas que se vayan a usar para evaluar la calidad de los agrupamientos obtenidos por los algoritmos de clustering implementados. Actualmente están implementadas dos medidas de calidad: la entropía y F-Measure ver Figura 9.

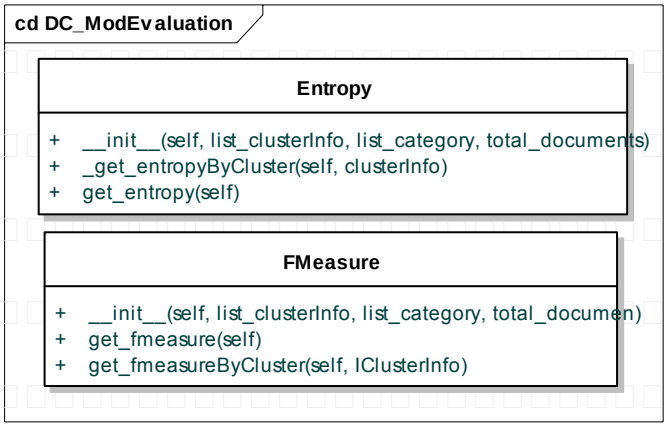

Figura 9 Diagrama de clases del módulo de evaluación

2.3.2.5 Módulo de adaptadores

Es el módulo responsable de contar con las implementaciones de la interfaz IVectorSpaceModel para los diferentes índices de los cuales se desee extraer la representación VSM. Actualmente existe una implementación para acceder al índice generado por Zope ver (Figura 10).

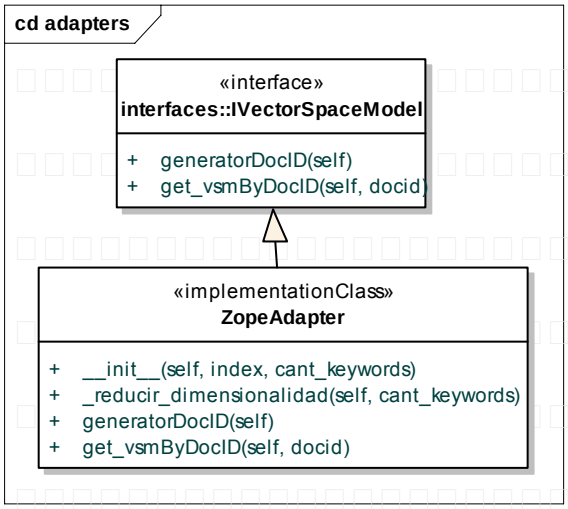

Figura 10 Diagrama de clases del módulo de adaptadores

2.3.2.6 Módulo de algoritmos

Este módulo fue pensado para ser responsable de contener las implementaciones de los algoritmos de clustering y aquellas dependencias que estos tengan ver (Figura 11). Actualmente están implementados los algoritmos seleccionados en el epígrafe 2.2.

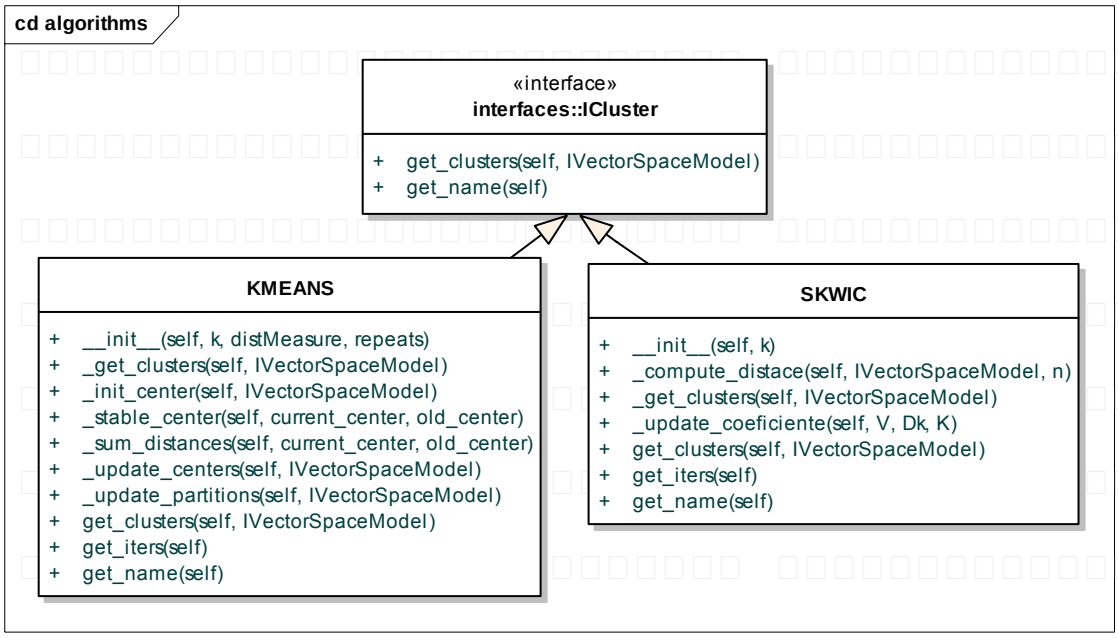

Figura 11 Diagrama de clases del módulo de algoritmos

2.3.2.7 Aplicación del patrón Adapter

En los diseños de software es importante tratar de rehusar los patrones de diseño pues estos han sido creados con el fin de resolver un problema que se repite una y otras vez. En nuestra biblioteca se ha identificado el contexto donde se aplica un patrón llamado Adapter (GAMMA *et al.*, 1994). El cual pretende adaptar un interfaz de una clase a la cual no es posible modificar para ser utilizada por un cliente (para un mejor entendimiento ver Figura 12). Con este patrón se ha logrado adaptar cualquier índice para obtener la interfaz necesaria para acceder al VSM. Se ha implementado el adaptador para zope.

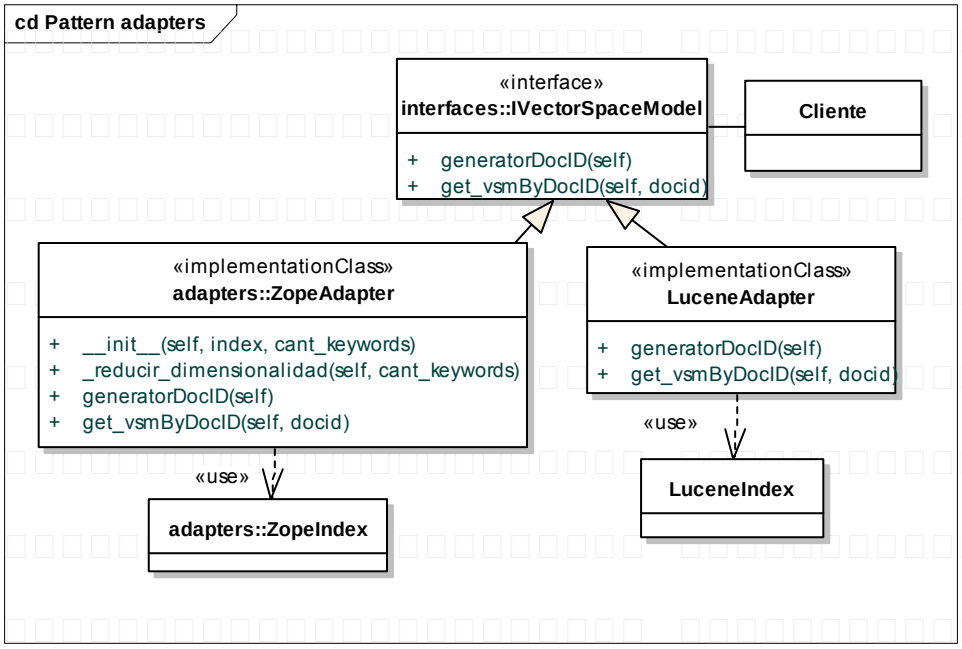

Figura 12 Diagrama de clases del patrón de diseño Adapter

2.3.2.8 Detalles de implementación de los algoritmos

En la implementación realizada se decidió implementar la estrategia de tomar aleatoriamente los centros de cluster desde la colección de documentos. En la Figura 13 se muestra dicho código. Primeramente se elige del total de documentos un listado de k identificadores de documentos y posteriormente se asigna cada vector documento elegido a un centro de cluster diferente.

```
85      def _init_center(self, ivsm):
86          """Inicializar los centros de clusters"""
87
88          docids = random.sample(list(ivsm.generatorDocID()),self.__k)
89          print 'Centros inicializados %s' % (docids)
90          for docid, c in zip(docids,range(self.__k)):
91              term2weight = ivsm.get_vsmByDocID(docid)
92              self.__center[c] = term2weight
```

Figura 13 Cálculo de los centroides

Otro aspecto importante en la implementación son los criterios de convergencia de los algoritmos. El más básico es fijar un número de iteraciones tras las cuales debe concluir el proceso. Otra

sería comprobar que entre una iteración y otra los centros de cluster no cambien. En la Figura 14 se muestra el código donde se hace un bucle a la vez para ambos conjuntos de vectores centros y se comprueba que todas sus componentes son iguales.

```
127     def _stable_center(self, current_center, old_center):
128         """Comprobar que los centros no cambian
129         """
130 
131         for current_t2k, old_t2k in zip(current_center, old_center):
132             v = numpy.array(current_t2k.values())
133             u = numpy.array(old_t2k.values())
134             if not numpy.alltrue(u == v):
135                 return False
136         return True
```

Figura 14 Centros estabilizados

Y la otra estrategia es implementar que los nuevos centroides disten de los centroides obtenidos en la iteración previa menos que una determinada distancia o se minimice esta distancia. En la Figura 15 se muestra el código donde se recorren ambos conjuntos de centros calculando la distancia entre cada par y se suma, y este valor final se trata de minimizar.

```
138     def _sum_distances(self, current_center, old_center):
139         """Devuelve la suma total de las distancias
140            entre los centros entre iteraciones
141         """
142         difference = 0.0
143         for current_t2w, old_t2w in zip(current_center,old_center):
144             u, v = current_t2w.values(), old_t2w.values()
145             difference += self._distance(u, v)
146         return difference
```

Figura 15 Minimizando la distancia entre los centros

Una parte importante del código es recalcular los centroides. En la Figura 16 se muestra la implementación. Para cada partición creada se realizó un promedio entre todas las características de los vectores documentos entre el total de características.

```
117    def update_centers(self, ivsm):
118        """Actualizar los centros"""
119
120        for c in range(self.__k): #Para cada cluster
121            vector_term = 0
122            keys = 0
123                                  __
124            for docid in self.__Part[c]:
125                keys = ivsm.get_vsmByDocID(docid).keys()
126                vector_term += numpy.array(ivsm.get_vsmByDocID(docid).values())
127                vector_term = vector_term / len(keys)
128            self.__center[c] = dict([(k,v) for k,v in zip(keys,vector_term)])
```

Figura 16 Actualizar los centros de cluster

2.4 Evaluación de los resultados obtenidos con la implementación de los algoritmos

En este epígrafe se pretende evaluar la calidad de los agrupamientos obtenidos por los algoritmos implementados, K-Means y SKWIC, con la aplicación de las métricas de evaluación de los agrupamientos. Se ha de llevar a cabo una prueba de hipótesis para demostrar que los resultados obtenidos por las implementaciones realizadas cumplen en cierta medida lo abordado en la literatura. Para esto se definió un caso de estudio con una colección de documentos previamente etiquetados.

2.4.1 Métricas para evaluar la calidad de los agrupamientos

Con el objetivo de poder comprobar la validez de la implementación realizada respecto a lo abordado en la literatura, se ha propuesto tratar unas métricas de evaluación que permitan validar la calidad de los agrupamientos, y así demostrar que los resultados pueden ser aplicados.

Existen dos tipos de métricas que son ampliamente usadas (STEINBACH *et al.*, 2002). Un tipo de métrica que permite comparar diferentes conjuntos de grupos sin referenciar a conocimiento externo, llamada métrica de calidad interna. Un ejemplo de métrica interna es la métrica overall similarity basada en la similitud de pares de documentos en un cluster.

El otro tipo de métricas que permiten evaluar cuán bueno es el agrupamiento, se basan en la comparación de los grupos producidos por las técnicas de agrupamiento, donde se conocen las clases de la colección. Este tipo de métrica es llamada métrica de calidad externa. Ejemplos de métricas externas son la entropía y F-Measure.

Algunos investigadores plantean que al aplicar las métricas de calidad el resultado puede variar sustancialmente dependiendo de cual métrica fue usada. Pero afirman que si un algoritmo de agrupamiento funciona mejor que otros para la mayoría de las métricas, entonces se puede decir que ciertamente ese algoritmo es el mejor para la situación que fue evaluada.

2.4.1.1 Overall Similarity

En la ausencia de información externa, tales como las etiquetas de clases, la cohesión de los clusters puede ser usada como una métrica de similitud de clusters (STEINBACH *et al.*, 2002). Un método para calcular la cohesión de un cluster es usar la similitud pesada de la similitud interna del cluster,

$$OverallSimilarity = \frac{1}{|S|^2} \sum_{\substack{d \in S \\ d' \in S}} dis\tan ce(d', d)$$

donde $|S|$ es el número de documentos que pertenecen al cluster a evaluar. En (STEINBACH *et al.*, 2002) utilizan el cociente Coseno para calcular la distancia entre los vectores.

2.4.1.2 Entropía

En (STEINBACH *et al.*, 2002) usan la entropía como métrica de calidad de los clusters (con la advertencia de que la mejor entropía es obtenida cuando cada cluster contiene exactamente un documento). Sea *CS* un resultado de agrupamiento, para cada cluster es calculada primero la distribución de las clases (por ejemplo para un cluster *j* se calcula p_{ij}, la probabilidad que un miembro del clusters *j* pertenezca a la clase *i*)

$$p_{ij} = \frac{n_j^i}{n_j}$$

donde n_j^i es el número de documentos de la clase *i* que están asignados al cluster *j*. Entonces, usando esa distribución de la clase, la entropía de cada cluster *j* es calculada usando la fórmula estándar

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

donde la sumatoria es aplicada sobre todas las clases. La entropía total para el conjunto de clusters *i* es calculada como la suma de las entropías de cada cluster y han sido pesadas 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 números total de documentos.

2.4.1.3 F-Measure

La segunda métrica de calidad externa es F-Measure (STEINBACH *et al.*, 2002). Es una métrica que combina las ideas de precision y recall de la recuperación de información (FRAKES and YATES, 1992; RIJSBERGEN, 1999) . En (STEINBACH *et al.*, 2002) se trata cada cluster como si este fuera el resultado de una consulta y cada clase como si esta fuera el conjunto de documentos deseados para una consulta. Así, se calcula precision y recall de los clusters para cada clase dada, más específicamente para el cluster j y la clase i

$$precision(i, j) = n_{ij} / n_j$$

$$recall(i, j) = n_{ij} / n_j$$

donde n_{ij} es el número de miembros de la clase *i* en el cluster *j*, n_j es el número de miembros del cluster *j* y n_j es el número de miembros de la clase *i*.

Entonces, según (STEINBACH *et al.*, 2002) F-Measure, del cluster j y la clase i es entonces dada por

$$F(i,\ j)\ =\ \frac{2\cdot recall\ (_{i,j})\cdot precision\ (_{i,j})}{recall\ (_{i,j})+ precision\ (_{i,j})}$$

Un valor de *F-Measure* es calculado para toda la colección calculando un promedio pesado de todos los valores de las métricas *F-Measure* según la siguiente expresión

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

Es bien conocido de la práctica diaria de la recuperación de información que los niveles más altos de *precision* generalmente se obtienen con valores bajos de *recall*. La siguiente expresión permite calcular el valor *F-Measure*, proporcionando un parámetro $\alpha(0\le\alpha\le1)$ que permite ponderar *precision* y *recall*.

$$F\alpha(i,\ j)\ =\ \frac{1}{\alpha\frac{1}{precision\ (i,j)}+(1-\alpha)\frac{1}{recall\ (i,j)}}$$

En esta fórmula α puede verse como el grado relativo de importancia atribuida para *precision* y *recall*. Si $\alpha=1$ entonces $F\alpha(i,j)$ coincide con *precision*, si $\alpha=0$ entonces $F\alpha(i,j)$ coincide con *recall*. El valor $\alpha=0.5$ es usado usualmente, así se considera igual importancia para *precision* y *recall*.

Para la evaluación serán empleadas las medidas que usan documentos previamente clasificados Entropía y F-Measure.

2.4.2 Definición del caso de estudio para realizar la evaluación

Para realizar la evaluación de los agrupamientos se definió un caso de estudio, el cual debía estar formado por una colección de documentos, donde estos estuvieran previamente clasificados en categorías. Para poder aplicar las medidas elegidas en la sección anterior, con el fin de seleccionar esta colección, fue necesario revisar las colecciones disponibles que existen en

Internet y están reconocidas por los investigadores que pueden ser usadas para realizar este tipo de tareas. En la siguiente sección se aborda la selección.

2.4.2.1 Selección de la colección para realizar los agrupamientos

Existen publicados en Internet varias colecciones de documentos de texto que son recomendadas en la literatura para realizar pruebas a los algoritmos en las diversas áreas de la minería de texto. Entre las más recomendadas se encuentran:

- The 20 Newsgroups data set (LANG, 2005)
- TDT2 Multilanguage Text Version 4.0 (WAYNE *et al.*, 2001)
- Reuters-21578 (LEWIS, 1999)

En el caso de 20 Newsgroups data set es una colección de alrededor de 20 000 textos etiquetados en 20 categorías. Posee un formato sencillo pero provienen de listas de discusión, lo cual la hace contar con bastante información no útil y texto de poca longitud. Algunas de las categorías están estrechamente relacionadas unas de otras, mientras otras están muy separadas.

La colección TDT2 solo está clasificada en dos tipos de clases: news story y miscellaneous text. En su formato usan etiquetas del Lenguaje de Marcado Estándar Generalizado (SGML) por lo que sería fácil el procesamiento.

Por último, la colección de Reuters-21578 está compuesta por 22 ficheros de datos que contienen 21 578 documentos multi-clasificados en 135 tópicos y 6 ficheros que describen las categorías y estos ficheros siguen el formato SGML.

Después de analizar cada corpus y viendo los inconvenientes presentes en cada colección se ha decidido la colección de Reuters-21578 porque cumple la mayoría de los requerimientos necesarios para evaluar la calidad de los agrupamientos.

2.4.2.2 Caso de estudio: Grupos de documentos de la Agencia de noticias Reuters

Con el objetivo de conformar la muestra para realizar la evaluación. Se tomaron de la colección de Reuters-21578 un total de 800 documentos pertenecientes a 4 categorías, con una distribución de 200 documentos por categoría.

Los documentos fueron obtenidos realizando un procesamiento de la colección y tomando solo aquellas noticias que estaban clasificadas en una sola categoría y que estuvieran recomendadas para realizar entrenamientos. Cada uno de los documentos fue almacenado en un fichero texto.

Posteriormente se llevó a cabo el proceso de representación de los documentos al modelo vectorial apoyándonos en el motor de indexado de Zope. Dentro de las técnicas de pre-procesamiento, se usaron la separación en palabras, la conversión de las palabras a minúsculas, la eliminación de las palabras de parada, el proceso de reducir una palabra a su raíz y el cálculo del peso de las palabras.

Para reducir la dimensionalidad del vector documento se realizó el cálculo de la frecuencia inversa del documento (IDF) para cada término y se tomaron en orden descendente los 5000 términos con mayor valor de IDF.

2.4.3 Experimento de prueba

Con el siguiente experimento se pretende llegar a la conclusión de que la implementación realizada de los algoritmos se comporta según lo planteado en la literatura (BERRY, 2004).

Para el mismo se ha realizado una prueba de hipótesis estadísticas analizando la entropía obtenida por lo implementado y fijando como indicador la obtenida en (BERRY, 2004) ver Tabla 1. Para realizar el experimento se ha tomado una muestra aleatoria simple de 200 documentos de la población de los 800 documentos del caso de estudio descrito.

Algoritmo	**Entropía**
K-Means	0.771
SKWIC	0.750

Tabla 1 Datos de la literatura

La prueba fue realizada para un nivel de confianza del 95%, con un nivel de significación del 0.05. Este experimento fue repetido para 5 corridas de los algoritmos K-Means y SKWIC (Tabla 2 Tabla 3). A continuación se define la hipótesis:

$H_0 : E_L \leq E_I$

$H_1 : E_L > E_I$

Donde E_L es la entropía local calculada para los algoritmos y E_I es la entropía de Internet calculada en (BERRY, 2004). Es aplicado el estadígrafo de prueba (Ecuación 8) para la región crítica (Ecuación 7):

$$Z = \frac{\hat{P} - P_0}{\sqrt{\frac{P_0(1 - P_0)}{n}}}$$ Ecuación 8

$$Z > Z_{1-\alpha}$$ Ecuación 9

$Z > Z_{0.95}$ = 1.65

Corridas	K-Means Entropía	K-Means F-Measure	Ho Aprobada	Ho Rechazada
#1	0.798692	0.378541	X	
#2	0.875552	0.353766		X
#3	0.825024	0.376320		X
#4	0.764955	0.341311	X	
#5	0.847506	0.314745		X

Tabla 2 Métricas calculadas para el algoritmo K-Means

Corridas	SKWIC Entropía	SKWIC F-Measure	Ho Aprobada	Ho Rechazada
#1	0.831333	0.419383		X
#2	0.811155	0.476989		X
#3	0.793423	0.389121	X	
#4	0.806535	0.388263		X
#5	0.846589	0.435104		X

Tabla 3 Métricas calculadas para el algoritmo SKWIC

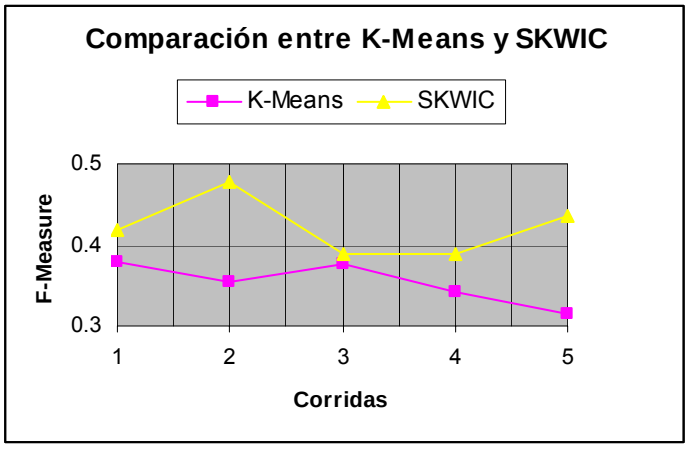

Figura 17 Comparación entre los algoritmos K-Means y SKWIC para la métrica F-Measure

Después de realizar las pruebas de hipótesis para cada corrida de cada algoritmo se llega a la conclusión de que para el nivel de significación fijado de 0.05 y bajo un nivel de confianza del 95%, la calidad de los agrupamientos teniendo en cuenta el valor de entropía se comporta mejor que el referenciado en la literatura. Y analizando la métrica F-Measure llegamos al resultado de que el SKWIC es mejor que el K-Means para las condiciones sobre las que fue evaluada Figura 17. Por tanto se puede se puede decir que las implementaciones realizadas están en correspondencia con lo planteado en la literatura.

2.5 Conclusiones parciales

En este capítulo se cumplió el segundo objetivo relacionado con la elección de los algoritmos ha implementar y el diseño de la biblioteca de clases formada por seis módulos. Fue posible describir los módulos en detalle y describir las partes fundamentales del código. Se ha notado una diferencia significativa en los resultados obtenidos por los algoritmos al tener que fijar el número de grupos a obtener. Se logró demostrar que la implementación de los algoritmos implementados está en correspondencia con lo planteando en la literatura.

CONCLUSIONES

Al finalizar el trabajo se cumplió el objetivo general propuesto arribando a las siguientes conclusiones:

1. La etapa de pre-procesamiento constituye una etapa fundamental en el proceso de clustering.
2. Se logró implementar la biblioteca de algoritmos de agrupamiento de documentos.
3. Existe una deficiencia en la calidad de los agrupamientos obtenidos debido a la naturaleza de los algoritmos de definir a priori los grupos a formar.
4. Se demostró que el K-Means y SKWIC fueron implementados de acuerdo a lo planteado en la literatura.

RECOMENDACIONES

1. Implementar algoritmos que no necesiten definir de antemano lo grupos a formar.
2. Implementar algoritmos que combinen lo mejor de las técnicas jerárquicas y particionales.
3. Profundizar en las técnicas de reducción de la dimensionalidad.

REFERENCIAS BIBLIOGRÁFICAS

ABDI, H. *Distance*, 2007. [2007]. Disponible en: http://www.utdallas.edu/~herve/Abdi-Distance2007-pretty.pdf

BARRETT, N. *El Estado del la Cibernación, Consecuencias Culturales, Políticas y Económicas de Internet*. 1999. p.

BERGO, A. *Text Categorization and Prototypes*, 2001. [Disponible en: http://www.illc.uva.nl/Publications/ResearchReports/MoL-2001-08.text.pdf

BERRY, M. W. *Survey of Text Mining: Clustering, Classification, and Retrieval*. 2004. p. Cap3-Simultaneous Clustering and Dynamic Keyword Weigting for Text Document

Cap4-Means Algorithm, Principal Directions Divisive Partitioning, Principal Direction Divisive Partitioning.

Cap5 modelo vectorial y su relacion con los cluster mining. 0-387-95563-1

CLARKE, A. C. *Aprendizaje no supervisado: Estado del arte*, 2001. [Disponible en: http://www.lsi.upc.edu/%7Ebejar/aaac/Apren_no_sup.ps

COMMUNITY-PYDEV. *PyDev for Python*, 2007. [2007]. Disponible en: http://pydev.sourceforge.net/

COMMUNITY-SUBECLIPSE. *Plugin subeclipse*, 2007. [Disponible en: http://subclipse.tigris.org/

COMMUNITY-SUBVERSION. *Subversion*, 2007. [Disponible en: http://subversion.tigris.org/

ECLIPSE-FOUNDATION Eclipse, 2007

EDNA, H. V. *Algoritmo de clustering basado en entropía para descubrir grupos en atributos de tipo mixto*. CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITECNICO NACIONAL DEPARTAMENTO DE INGENIERIA ELECTRICA SECCION DE COMPUTACION, 2006. p.

FÉLIX, L. C. M. *Data mining: torturando a los datos hasta que confiesen*, 2002. [Disponible en: http://www.uoc.edu/web/esp/art/uoc/molina1102/molina1102.html

FRAKES, W. B. and R. B. YATES. *Information Retrieval : Data-structures and Algorithms*, Prentice-Hall, 1992.

FRIGUI, H. and O. NASRAOUI. *Simultaneous Categorization of Text Documents And Identification of Cluster-dependent Keywords*, 2002. [Disponible en: http://citeseer.ist.psu.edu/frigui02simultaneous.html

FUNG, B. C. M.; K. WANG*, et al. Hierarchical Document Clustering*, 2003. [Disponible en: http://www.lsi.upc.edu/%7Ebejar/aaac/articulos/document%20clustering%20Encyclopedia.pdf

GAMMA, E.; R. HELM*, et al. Design Patterns Elements of Reusable Object-Oriented Software*, 1994. [Disponible en: http://hillside.net/patterns/DPBook/DPBook.html

GÓMEZ, M. M. *Minería de texto empleando la Semejanza entre Estructuras Semánticas*, 2002. p.

---. *Minería de texto: Un nuevo reto computacional*, 2003. [Disponible en: http://ccc.inaoep.mx/~mmontesg/publicaciones/2001/MineriaTexto-md01.pdf

GOOGLE, C. *Google News*, 2007a. [2007]. Disponible en: http://news.google.es/

---. *Google Search Appliance* 2007b. [2007]. Disponible en: http://www.google.es/enterprise/gsa/features.html

JOYANES, L. *Cibersociedad. Los retos sociales ante un nuevo mundo digital*, 1997.

KLAWONN, F. and F. HÖPPNER. *What is Fuzzy About Fuzzy Clustering: Understanding and Improving the Concept of the Fuzzifier*, 2003. [Disponible en: http://public.fh-wolfenbuettel.de/~hoeppnef/paper/Klawonn-IDA-2003.pdf

LANG, K. *The 20 Newsgroups data set*, 2005. [2007]. Disponible en: http://people.csail.mit.edu/jrennie/20Newsgroups/

LEWIS, D. D. *Reuters-21578*, 1999. [2007]. Disponible en: http://www.daviddlewis.com/resources/testcollections/reuters21578

LÓPEZ, A. H. *Observatorio Información*, UCI, 2007.

LUKE, B. T. *A Tutorial on Clustering Algorithms*, 2007. [2007]. Disponible en: http://fconyx.ncifcrf.gov/%7Elukeb/cluster.html

MACQUEEN, J. B. *Some Methods for classification and Analysis of Multivariate Observations*. Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1967. 281-297 p.

MATTEUCCI, M. *A Tutorial on Clustering Algorithms*, 2003. [2007]. Disponible en: http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/

MENDONCA, M.; A. V. W. BUILDING*, et al. Mining Software Engineering Data: A Survey*, 1999. 193.

MUCHA, H.-J. and H. SOFYAN. *Nonhierarchical Clustering*, 2003. [2007]. Disponible en: http://www.quantlet.com/mdstat/scripts/xag/html/xaghtmlframe149.html

NEGRÍN, R. E. S. *Gestión del Conocimiento: Conceptos, Aplicaciones y experiancias*. Cuba, 2002. 222.

NILSSON, N. J. *Introduction to Machine Learning*, 2005. [Disponible en: http://robotics.stanford.edu/people/nilsson/mlbook.html

NORES, J. E. G. *Análisis Cluster*, 2000. [Disponible en: http://www.estadistico.com/arts.html?20001023-1

PLONE-FOUNDATION. *Plone*, 2007. [2007]. Disponible en: http://plone.org/

PYTHON-SOFTWARE-FOUNDATION. *Python Language*, 2007. [Disponible en: http://www.python.org

RIJSBERGEN, C. J. V. *Information Retrieval*, 1999. [Second Edition]. Disponible en: http://www.dcs.gla.ac.uk/~iain/keith/

ROBERTSON, S. *Understanding Inverse Document Frequency: On theoretical arguments for IDF*, 2004. [Disponible en: http://www.soi.city.ac.uk/~ser/idfpapers/Robertson_idf_JDoc.pdf

RODRÍGUEZ, L. G. Tecnologías de información y comunicación: la gestión de un impacto social positivo 1999.

RUBIO, A. C.; V. F. FERNÁNDEZ*, et al. Evaluación del clustering de páginas web mediante funciones de peso y combinación heurística de criterios*, 2005.

RUSSELL, S. J. and P. NORVIG. *Artificial Intelligence A Modern Approach*, 1995.

SAVARESI, S. M.; D. L. BOLEY*, et al. Cluster selection in divisive clustering algorithms*, 2002. [Disponible en: http://www.siam.org/meetings/sdm02/proceedings/sdm02-18.pdf

SHUCLOPER, J. R. and J. F. M. TRINIDAD. *Clasificación sin Aprendizaje y con Aprendizaje Parcial*, Centro de Investigaciones y de Estudios Avanzados: Instituto Politécnico Nacional de México, 1995.

SPARX-SYSTEMS. *Enterprise Architect*, 2007. [2007]. Disponible en: http://www.sparxsystems.com

STEINBACH, M.; G. KARYPIS*, et al. A Comparison of Document Clustering Techniques*, 2002. [Disponible en: http://citeseer.ist.psu.edu/steinbach00comparison.html

STREHL, A. *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*, 2002. [2007]. Disponible en: http://www.lans.ece.utexas.edu/~strehl/diss/node53.html

TAN, A.-H. Text Mining: The state of the art and the challenges, 1999.

VESTER, K. L. and M. C. MARTINY. *Information Retrieval in Document Spaces Using Clustering* 2005. p.

VISA, A. Technology of Text Mining, 2001.

VIVÍSIMO, C. *Clusty Web Search* 2000. [2007]. Disponible en: http://clusty.com/

WAYNE, C.; G. DODDINGTON*, et al. DT2 Multilanguage Text Version 4.0*, 2001. [Disponible en: http://projects.ldc.upenn.edu/TDT2/

WIKI. *Greedy algorithm*, 2007a. [Disponible en: http://en.wikipedia.org/wiki/Greedy_algorithm

---. *K-Medoids*, 2006. [Disponible en: http://en.wikipedia.org/wiki/K-medoids

---. *Machine learning*, 2007b. [Disponible en: http://en.wikipedia.org/wiki/Machine_learning

YATES, R. B. and B. R. NETO. *Modern Information Retrieval*, 1998.

ZAMIR, O. and O. ETZIONI. *Web Document Clustering: A Feasibility Demonstration*, 1998. [Disponible en: http://www.lsi.upc.edu/%7Ebejar/aaac/articulos/document%20clustering%20sigir98.pdf

ZOPE-CORPORATION. *Zope*, 2007. [2007]. Disponible en: http://www.zope.org/

BIBLIOGRAFÍA

BERKHIN, P. *Survey of Clustering Data Mining Techniques*, 2002. [Disponible en: http://citeseer.ist.psu.edu/berkhin02survey.html

BRONCANO, R. G. *Modelo de Recuperación*, 2006. [Disponible en: http://modelosrecuperacion.tripod.com/

CUTTING, D.; O. GOSPODNETIC*, et al. Lucene in Action*. 2004. 347 p. Implementacion de un api, para indexacion y busqueda de informacion 1-932394-28-1

CUTTING, D. R.; D. R. KARGER*, et al. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections* 1999. [Disponible en: http://citeseer.ist.psu.edu/cutting92scattergather.html

ESCRIVÁ, D. M. L. *Extracción y Recuperación de Información Temporal*, 2002. p.

GASPERIN, C. *Using Syntactic Contexts for Measuring Word Similarity*, 2001. [Disponible en: http://www.cl.cam.ac.uk/~cvg20/esslli01.pdf

HALGAMUGE, S. K. and L. WANG. *Classification and Clustering for Knowledge Discovery*, Springer-Verlag, 2005.

HEARST, M. *What Is Text Mining?*, 2003. [Disponible en: http://www.ischool.berkeley.edu/~hearst/text-mining.html

HEARST, M. A. *Untangling Text Data Mining*, 1999 [Disponible en: http://www.ischool.berkeley.edu/~hearst/papers/acl99/acl99-tdm.html

JAIN and DUBES. *Clustering Method and Algortihms*, 1999a. [Disponible en: http://www.lsi.upc.edu/%7Ebejar/aaac/Jain_Dubes_cap3.pdf

---. *Data Types and Scales*, 1999b. [Disponible en: http://www.lsi.upc.edu/%7Ebejar/aaac/Jain_Dubes_cap2.pdf

JAIN, A. K.; M. N. MURTY*, et al. Data Clustering: A Review* 1999. [Disponible en: http://www.lsi.upc.edu/%7Ebejar/aaac/jain99data.pdf

JAME, A. *Natural Language Understanding*, 1995.

KOGAN, J.; C. NICHOLAS*, et al. Grouping Multidimensional Data, Recent Advances in Clustering*, 2006.

LATTEIER, A.; M. PELLETIER*, et al. The Zope Book*, 2004.

MANNING, C. D.; P. RAGHAVAN*, et al. An Introduction to Information Retrieval*, 2006.

MARCUCCI, A. and A. GAUTHIER Implementación y Evaluación de Métedos de Clasificación: Estadísticos, de Agrupamiento, Árboles de Decisión y MetaHeurísticos. , 2005.

MIRANDA, M. I. *Clustering methods and algorithms*, 1999. [2007]. Disponible en: http://www.cse.iitb.ac.in/dbms/Data/Courses/CS632/1999/clustering/dbms.html

PONS, A.; R. BERLANGA*, et al.* Técnicas de agrupamiento semántico-temporal para la identificación de sucesos en bases de noticias digitales, 2001.

STATSOFT. *Cluster Analysis*, 2004. [2007]. Disponible en: http://www.statsoft.com/textbook/stcluan.html

STEIN, B. and S. M. Z. EISSEN. *Automatic Document Categorization Interpreting the Perfomance of Clustering Algorithms*, 2003. [Disponible en: http://www.uni-weimar.de/medien/webis/publications/downloads/stein_2003c.pdf

WANG, J. *Encyclopedia of Data Warehousing and Mining*. 2005. p. 1-59140-559-9

WITTEN, I. H. and E. FRANK. *Data Mining: Practical Machine Learning Tools and Techniques*, 2005. 2.

YOLIS, E.; P. BRITOS*, et al.* Algoritmos Genéticos Aplicados a la Categorización Automática de Documentos, 2003.

ZHANG, Y. Q.; A. KANDEL*, et al. Computational Web Intelligence, Intelligent Technology for Web Applications*. 2004. p. Web Document Analysis: Challenges and Opportunities vol55

Cap6 Fuzzy Clustering and intellingent search for a web-based fabric database 117p

981-238-827-3

ZHAO, Y. and G. KARYPIS. *Evaluation of Hierarchical Clustering Algorithms for Document Datasets*, 2002. [Disponible en: http://www.lsi.upc.edu/%7Ebejar/aaac/articulos/document%20clustering%20vaclusterCIKM02.pdf

---. *Hierarchical Clustering Algorithms for Document Datasets*, 2004. [Disponible en: http://www.lsi.upc.edu/%7Ebejar/aaac/articulos/document%20clustering%20vaclusterDMKD05.pdf