GENERACIÓN AUTOMÁTICA DE RESÚMENES DE MÚLTIPLES DOCUMENTOS BASADA EN EL ALGORITMO GHS+LEM

WILLIAN ANDRES TAMAYO MONJE
MELISSA LYNNETTE VELA CORAL

Director: Dra. (c) MARTHA ELIANA MENDOZA BECERRA

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
DEPARTAMENTO DE SISTEMAS
GRUPO DE I+D EN TECNOLOGÍAS DE LA INFORMACIÓN
RECUPERACIÓN DE LA INFORMACIÓN
POPAYÁN, Abril 2012

Agradecimientos

A Dios, por ser nuestro creador, por permitirnos seguir nuestro camino y darnos la fuerza para continuar cada día.

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.

A la Dra (c). Martha Eliana Mendoza Becerra y al Ph.D. (c). Carlos Alberto Cobos Lozada por su dedicación, tiempo, apoyo y enorme conocimiento para guiarnos en este reto.

A nuestros amigos y educadores, fieles compañeros en este proceso con 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 TABLAS

LISTA DE FIGURAS

Presentación

Con el desarrollo de las tecnologías de comunicación, el incremento de capacidades en almacenamiento y el crecimiento de usuarios que comparten información en internet, actualmente se encuentra gran cantidad de información disponible en la red. Por esto, puede ocurrir que noticias, artículos, blogs o documentos relevantes para un usuario no sean leídos y que se dificulte la extracción de información útil al momento de investigar acerca de un tema específico, debido al tiempo y esfuerzo que es requerido. Un área de investigación que intenta dar solución a este problema es la generación automática de resúmenes.

La generación automática de resúmenes es un área del Procesamiento del Lenguaje Natural (PLN) cuyo objetivo es el análisis de texto basado en un conjunto de teorías y tecnologías, la cual ha tenido una larga trayectoria de investigación y desarrollo, sin embargo, la comprensión y procesamiento del lenguaje a nivel computacional es una tarea aún no resuelta debido a la complejidad que implica leer un texto, comprenderlo y extraer información deseada.

La generación automática de resúmenes de múltiples documentos puede ayudar a que se comprendan los principales tópicos de un tema sin tener que leer cada uno de los documentos originales. Desde hace muchos años, se han venido explorando diversos métodos para la creación automática de resúmenes para uno o más documentos y esto ha tomado mayor importancia a medida que se incrementa la información disponible en la Web.

Con el objetivo de explorar una nueva forma de generación automática de resúmenes de múltiples documentos, en el presente trabajo de grado se describe un nuevo algoritmo de generación automática de resúmenes de múltiples documentos basada en la mejor búsqueda armónica global y modelos evolutivos que aprenden.

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

En el capítulo 1 se presenta la problemática que motivó el planteamiento de este proyecto, la justificación del desarrollo del mismo, los objetivos que se definieron y los principales resultados obtenidos.

El capítulo 2 describe las bases teóricas que enmarcan el proyecto, teniendo en cuenta los conceptos básicos en el área de generación de resúmenes, las principales investigaciones realizadas alrededor de ésta área, la forma como se evalúan los

algoritmos planteados, la forma de representar los documentos para que puedan ser procesados por los algoritmos, los esquemas de pesado de términos, las medidas de similitud y la descripción de los algoritmos de búsqueda armónica que se utilizaron como base para el desarrollo del algoritmo planteado en este proyecto.

En el capítulo 3 se presenta el algoritmo de generación de resúmenes basado en algoritmos de búsqueda armónica y modelos evolutivos que aprenden, también el proceso de afinación realizado para la optimización de los parámetros usados en los algoritmos HS y GHS+LEM.

En el capítulo 4 se presentan las etapas de pre-procesamiento realizadas sobre los documentos, la descripción de los conjuntos de documentos y de las métricas utilizadas para la prueba. Además se muestra la evaluación de la calidad de los resúmenes generados por el algoritmo propuesto y el análisis de los resultados obtenidos.

El capítulo 5 describe las conclusiones que se establecieron a partir de la experiencia adquirida en el desarrollo del proyecto y se proponen varias ideas de trabajo futuro para la continuidad del proyecto.

En el capítulo 6 se muestra la bibliografía y documentación empleada en la realización del proyecto.

Capítulo 1

1 INTRODUCCIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

La generación automática de resúmenes tiene aplicación en diversas áreas, por ejemplo, puede ser usada para preparar información que será mostrada en dispositivos móviles pequeños como los PDA[1], por medio de un resumen del texto original [1]. También ha sido ampliamente usada por las agencias de noticias que necesitan resumir la información que les llega sobre una misma noticia desde diferentes agencias de noticias. En general cuando un usuario realiza una exploración acerca de un tema en particular, una noticia o información de cualquier tipo, se encuentra con gran cantidad de documentos relacionados con este tema, teniendo que leer muchos de ellos para comprender los aspectos fundamentales del tema, lo que requiere tiempo y esfuerzo para el lector. Para dar solución a este problema, la generación automática de resúmenes de múltiples documentos busca obtener un resumen estos que permita identificar los temas principales que se encuentran en los documentos originales, sin tener que leer cada uno de ellos.

Se han utilizado diversos enfoques para la generación automática de resúmenes[2] de múltiples documentos, entre ellos se puede encontrar uno de los primeros en generar resúmenes multi-documento [2], éste usa técnicas de extracción de información para facilitar el proceso de identificación de similitudes y diferencias entre documentos; los métodos basados en *centroides* como MEAD [3] y LexRank [4] generan resúmenes usando centroides producidos por un sistema de detección y seguimiento; otros sistemas son basados en *Grafos* [5], en los cuales se representa el texto en forma de grafo y se interconectan las palabras u otras entidades de texto con relaciones significativas a través de las cuales se determina la relevancia de las oraciones, dándole mayor puntuación a aquellas que tienen mayor cantidad de enlaces hacia otras entidades de texto; existen métodos que se enfocan en el manejo de *redundancia* como MMR [6] el cual hace énfasis en lo que llaman “novedad relevante” dándole un peso a las oraciones más relacionadas con la consulta del usuario y restando puntuación a aquellas muy similares a las oraciones ya seleccionadas; también se han propuesto otros métodos basados en *reducción algebraica* como LSA [1] y NMF [7]. Existen métodos basados en *algoritmos evolutivos* como el sistema CBSEAS [8], y MCMR [9] que hace uso del un algoritmo de optimización de enjambre de partículas binario. Actualmente la mayoría de enfoques son orientados a la generación de resúmenes multi-documento como se puede observar en

[1] Personal Digital Assistant (asistente digital personal): es un dispositivo de pequeño tamaño que combina un ordenador, teléfono/fax, Internet y conexiones de red.

[2] Resumen se define como una breve representación de uno o varios documentos que contiene la información más destacada de los documentos originales [12].

las tareas definidas para cada año por DUC[3] y TAC[4], lo que demuestra la gran importancia que éstos han adquirido [8, 10-12].

Otros trabajos que han sido basados en modelos evolutivos para resúmenes mono-documento son propuestos en [13] y en [14] en donde usan algoritmos de enjambre de partículas para determinar la efectividad de características para encontrar la relevancia de una oración. En [15] utilizan un algoritmo genético para encontrar el mejor resumen y en [16] hacen uso del algoritmo de Búsqueda Armónica (HS por sus siglas en inglés, Harmony Search) para encontrar el mejor resumen, éste trabajo presenta mejores resultados que los obtenidos con algoritmos genéticos y resalta la importancia de tener en cuenta factores de un buen resumen en la función objetivo, debido a que juega un papel importante a la hora de evaluar los resúmenes extraídos.

Teniendo en cuenta que el algoritmo meta-heurístico HS [17], se desarrolló para encontrar una solución óptima a problemas complejos, se han propuesto varias mejoras entre las cuales se destaca la mejor búsqueda armónica global (GHS por sus siglas en inglés, Global-best Harmony Search) [18]. HS ha sido satisfactoriamente usado en una variedad de problemas de optimización, presentando ventajas con respecto a técnicas tradicionales de optimización[5] [19, 20] y además tiene pocos requerimientos matemáticos que pueden ser fácilmente aplicados a áreas específicas como la generación de resúmenes.

Por otra parte, los Modelos Evolutivos que Aprenden (LEM) se han utilizado para aumentar la precisión y disminuir el tiempo de convergencia de la solución óptima en problemas complejos por medio del algoritmo GHS [21], demostrando que cuando hay alta dimensionalidad, se obtienen mejoras con respecto al algoritmo GHS. Esto hace prometedor el uso del algoritmo GHS+LEM aplicado al problema de generación automática de resúmenes puesto que se presenta alta dimensionalidad (dada por las frases de los documentos).

Dado lo anterior, en este trabajo de investigación basados en los resultados del algoritmo HS para la generación automática de resúmenes y en que el algoritmo GHS+LEM obtiene mejores resultados para alta dimensionalidad (como ocurre en este trabajo), se propone un algoritmo GHS+LEM para la selección de oraciones que permite crear resúmenes extractivos de múltiples documentos. El tipo de resumen que se genera tiene las siguientes características de acuerdo a la taxonomía propuesta en [22]: *Extractivo* porque únicamente se seleccionan oraciones para conformar el resumen, el *nivel de procesamiento* es superficial debido a que se utilizan características poco profundas, *genérico* porque no depende de la audiencia a la que va dirigido el resumen, *mono-lenguaje* ya que solo se resumirán documentos en inglés, *múltiples documentos* y no depende del tipo de documento a resumir (*científico, noticias, blogs*) sin embargo las pruebas se realizan con fuentes de DUC, organización que proporciona documentos de noticias con sus respectivos resúmenes realizados por expertos. Para evaluar la calidad de los resúmenes generados automáticamente por GHS+LEM se utiliza la herramienta ROUGE [23].

[3] DUC *Document Understanding Conference* http://duc.nist.gov/
[4] *Text Analysis Conference* http://www.nist.gov/tac/
[5] Proceso matemático y estadístico cuya finalidad es obtener la mejor de todas las soluciones posibles a un problema

Aunque la investigación en generación automática de resúmenes ha tenido una larga trayectoria y abarca trabajos con diversos enfoques y/o métodos, los trabajos en este tema aún continúan, pues hasta el momento no se ha llegado a un algoritmo óptimo debido a la complejidad de esta tarea, por lo tanto, con el presente trabajo se aporta al conocimiento científico global la exploración de una nueva alternativa para dar solución al problema planteado, para lo cual se intenta resolver la siguiente pregunta: ¿Cómo hacer uso de la heurística GHS+LEM para la generación automática de resúmenes extractivos de múltiples documentos?

1.2 JUSTIFICACIÓN

La generación automática de resúmenes es una tarea que se investiga desde finales de los años 50, sin embargo, actualmente ha tomado mayor importancia debido a que la cantidad de documentos electrónicos accesibles desde diversos lugares y dispositivos crece de manera exponencial permitiendo así procesar toda esa información, presentarla de manera resumida permitiría reducir el tiempo y esfuerzo que el usuario debe invertir en ello.

Debido a la complejidad que tiene la tarea de generación automática de resúmenes, el conseguir resúmenes igual que los haría un humano es todavía un reto por lo que aún no se llega a resultados óptimos en la generación de resúmenes de múltiples documentos, por tanto existe la necesidad de continuar investigando alternativas diferentes para identificar la información más relevante de los documentos y presentarla a manera de resumen.

Para lograr una mayor calidad en el resumen generado por el sistema planteado en este proyecto, se hace uso de los diferentes recursos actuales que se tienen para el PLN (Procesamiento del lenguaje natural) como bases de datos léxicas (WordNet), herramientas de pre-procesamiento (Lucene), y herramientas para la evaluación como ROUGE.

El aporte principal de este proyecto es proponer un nuevo algoritmo de generación automática de resúmenes de múltiples documentos, mediante la aplicación del algoritmo de la Mejor Búsqueda Armónica Global con los modelos de aprendizaje evolutivo (LEM) [24] y una variación del algoritmo PRISM [25]; para su planteamiento fue necesario tener en cuenta varios aspectos importantes como son: investigación y aplicación de diferentes alternativas en la etapa de pre-procesamiento, definición de una función objetivo apropiada para obtener un resumen adecuado, adaptación del algoritmo GHS y LEM para el problema de generación automática de resúmenes de múltiples documentos y el afinamiento de los parámetros del algoritmo para un mejor rendimiento del mismo.

Las herramientas tecnológicas (Microsoft: Microsoft Visual Studio 2010 y Microsoft Project 2007) necesarias para la realización de este proyecto, fueron seleccionadas con base en la experiencia que el Grupo de Tecnologías de la Información (GTI) ha obtenido en los últimos ocho (8) años de trabajo, específicamente en experiencias exitosas en el desarrollo de proyectos relacionados con Recuperación de la información y optimización, como por ejemplo: Buscador Inteligente Basado en Minería de Datos, Hibridación de la Mejor Búsqueda Armónica Global y el Algoritmo K-Means para el Clustering de documentos Web, Hibridación del algoritmo GHS con Modelos LEM. Además dada la

disponibilidad del software, documentación y materiales de aprendizaje que se tiene en la Universidad del Cauca, gracias al programa MSDN Academic Alliance[6].

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

Proponer un algoritmo de generación automática de resúmenes de múltiples documentos basado en la Mejor Búsqueda Armónica Global con Modelos Evolutivos que Aprenden, afinar sus parámetros y evaluar la calidad de los resúmenes generados.

1.3.2 OBJETIVOS ESPECÍFICOS

- Adaptar el algoritmo de generación automática de resúmenes extractivos de un documento basado en HS [16], para generar resúmenes extractivos fuera de línea sobre múltiples documentos.

- Proponer un algoritmo de generación automática de resúmenes de tipo extractivo, informativo, superficial, genérico y monolenguaje para múltiples documentos que trabaje fuera de línea, basado en GHS+LEM (GHS: Global-best Harmony Search, LEM: Learning Evolutionary Models).

- Afinar los parámetros de memoria armónica, número de improvisaciones, porcentaje de generación de las reglas del algoritmo propuesto, utilizando la mejor búsqueda armónica global (GHS) [18].

- Evaluar la calidad[7] de los resúmenes generados con el algoritmo propuesto y comparar los resultados con el algoritmo HS adaptado, utilizando documentos de noticias de DUC (Document Understanding Conference).

1.4 RESULTADOS OBTENIDOS

- Código fuente de los algoritmos de generación automática de resúmenes de múltiples documentos mediante HS, GHS y GHS+LEM y evaluación de la calidad de los mismos.

- Código fuente utilizado en las pruebas de afinación de parámetros de HS y GHS+LEM.

- Artículo: Generación automática de resúmenes de múltiples documentos basada en el algoritmo GHS+LEM.

[6] Acuerdo que vincula a Microsoft con entidades educativas universitarias, en la cual se permite tener acceso a software de desarrollo con propósitos académicos.

[7] Mediante ROUGE-2 y ROUGE-SU4

- Monografía del trabajo de grado. Corresponde al presente documento, donde se describe lo que dio origen a la realización del proyecto, las bases teóricas que lo enmarcan, el proceso seguido en el desarrollo del proyecto, el algoritmo propuesto, sus características, requerimientos y limitaciones, los resultados obtenidos, los aportes más sobresalientes, las conclusiones y las recomendaciones para el desarrollo de futuras investigaciones en el área.

Capítulo 2

2 CONTEXTO TEÓRICO

2.1 GENERACIÓN AUTOMÁTICA DE RESÚMENES

2.1.1 Definición

Es importante primero definir qué es un resumen, según Radev [26] “es un texto producido a partir de uno o más textos, que transmite información importante, que no es más largo que la mitad del texto(s) original(es) y usualmente es menos significativo”. Según Spärck Jones [27], un resumen es “una transformación reductiva del texto fuente al resumen a través de la condensación mediante la selección y/o generalización de lo que es importante en el texto original”.

De estas definiciones se puede decir que un resumen es una breve representación de uno o varios documentos que contiene la información importante de los documentos originales, por lo tanto, la generación automática de resúmenes consiste en la generación de un resumen a través de un programa de computador.

2.1.2 Taxonomía

Existen diversas taxonomías de resúmenes [22]; entre ellas se encuentran: i) de acuerdo a la forma del resumen pueden ser, de extracción o abstracción, los resúmenes de extracción son formados a partir de la reutilización de porciones del texto original, resúmenes basados en abstractos son un poco más complejos, pues requieren de herramientas de análisis lingüístico para construir nuevas frases a partir de las ya extraídas; ii) en cuanto al nivel de procesamiento: superficial o profundo, los superficiales son aquellos que representan el documento utilizando características poco profundas como posicionamiento de frases, frases claves, etc. Profundos requieren de técnicas como el procesamiento de lenguaje natural para su construcción; iii) propósito del resumen: indicativos o informativos, los resúmenes indicativos son aquellos que dan una idea al lector sobre el documento para que éste decida si es conveniente leerlo, resúmenes informativos buscan reemplazar el documento original para que el lector no tenga la necesidad leer todo el documento; iv) audiencia a la que va dirigido el resumen: genéricos, basados en consultas, enfocados en el usuario o en tópicos, los resúmenes genéricos son aquellos que no dependen de la audiencia a la que va dirigido el resumen, resúmenes basados en consultas buscan dar respuesta a una consulta que el usuario realiza, resúmenes enfocados en el usuario buscan dar importancia a las necesidades especificas del usuario; v) la cantidad de documentos que se procesan: Un documento o múltiples documentos; vi) el lenguaje del documento: Monolenguaje o Multilenguaje; vii) tipo de documento: científico, noticias, blogs, entre otros.
Dentro de la taxonomía general de los tipos de resumen, el de la presente investigación clasifica de la siguiente manera: *Extractivo* por que este tipo de resúmenes en general tiene mejores resultados para generación automática de resúmenes de múltiples

documentos, además los resúmenes abstractivos poseen una alta complejidad ya que consisten en la generación de nuevas oraciones lo que conlleva a obtener un resumen incoherente si las oraciones no son tratadas y enlazadas adecuadamente, el *nivel de procesamiento* es superficial debido a que se utilizan características poco profundas (frecuencia de palabras), *genérico* porque no depende de la audiencia a la que va dirigido el resumen, *mono-lenguaje* ya que solo se resumirán documentos en inglés dado que el pre-procesamiento está condicionado a este idioma, sin embargo el algoritmo puede ser fácilmente adaptado a otros idiomas, *múltiples documentos* y no depende del tipo de documento a resumir (*científico, noticias, blogs*)

2.1.3 Métodos de Generación Automática de Resúmenes Extractivos

La generación automática de resúmenes extractivos es un área que surgió a finales de los años 50 [28], durante estos años se han propuesto diferentes enfoques que buscan generar resúmenes de calidad, entre los principales métodos están:

2.1.3.1 Primeros métodos

Uno de los primeros trabajos en generación automática de resúmenes de documentos surge en 1958 y fue propuesto por Luhn [29], quien utiliza la ley de distribución de las palabras de Zipf[8] para establecer que si la cantidad de palabras de una oración sobrepasa cierto umbral, entonces esta oración contiene muy probablemente información relevante. Inicialmente elimina pronombres, preposiciones y artículos. Posteriormente realiza una normalización de términos en donde se fusionan los que son similares, para esto se cuenta el número de caracteres diferentes entre pares de palabras, y si la cuenta es inferior a seis, los dos términos se consideran iguales. Finalmente se ordenan las oraciones de acuerdo a su ponderado dado por la frecuencia de sus palabras significativas y por la distancia de estas en la oración.

Años más tarde, Edmundson en 1969 [30], parte de que una sola característica no era suficiente para determinar la importancia de una oración como lo propuso Luhn (*frecuencia de palabras),* por lo que propone el uso de cuatro métodos básicos para la asignación de pesos numéricos a las oraciones: Método indicador (Cue method) en el cual se tiene en cuenta la presencia de ciertas palabras dentro del texto como “significativo”, “importante”, “en conclusión”. Para esto usa un diccionario con una lista predeterminada de dichas palabras denominado “cue dictionary”. El método clave (Key method) se basa en el método propuesto por Luhn (frecuencia de palabras), para esto usa un glosario de palabras clave obtenidas a partir del documento a resumir. El método del título (Title method) se basa en las características del esqueleto del documento (título, subtítulos). En el método de la Ubicación (location method) plantea que las oraciones principales tienden a ocurrir en diversas partes del documento. Se probaron las características por separado y también la combinación de todas obteniendo los mejores resultados con la combinación “cue-title-location” y de manera individual el mejor resultado se obtuvo con el método de la posición.

[8] La ley de Zipf establece que en uno o varios documentos se cumple que: un conjunto pequeño de palabras aparecerán de manera muy frecuente dentro del documento, un conjunto de palabras de mayor tamaño aparecerán con una frecuencia menor y por último un conjunto grande de palabras aparecerán con muy poca frecuencia

En los primeros métodos hay muchas características que por limitaciones de rendimiento no se abordaban, ya que carecían de herramientas de apoyo para el procesamiento de los datos para obtener una buena precisión en la extracción de oraciones. Además solamente trabajaban resúmenes de un documento, para un idioma y generalmente eran indicativos.

2.1.3.2 Métodos basados en Aprendizaje de Máquina

En [30], [31], [32], [28] se muestra la utilización del clasificador Naive Bayes para generación automática de resúmenes de documentos, esta es una técnica que consiste en capturar una serie de características tomadas desde el texto y hacen que el algoritmo aprenda de estas, una de las desventajas que tiene esta técnica es que para la realización de un resumen se necesitan datos de entrenamiento y además depende del lenguaje en el que se ha hecho el entrenamiento o de las características que se han tomado para la extracción de las oraciones. Otra técnica es un clasificador de redes Neuronales [33], [32], [34], este algoritmo se utiliza para la extracción de las oraciones más importantes desde los documentos originales, al igual que Naive Bayes, hace uso de características tomadas desde los documentos las cuales nos permiten después inferir cuales oraciones son las más representativas del texto original, su desventaja es la necesidad de tener datos de entrenamiento y la realización de resúmenes que estén hechos en el mismo lenguaje en que se realizó el aprendizaje. Otro método de aprendizaje de máquina es el modelo planteado por Conroy y O'leary (2001) [35] el cual se basa en el Modelo Oculto de Markov (HMM por sus siglas en inglés, Hidden Markov model), una de las características importantes de este trabajo es la utilización de pocas características para la identificación de oraciones y el uso de un modelo secuencial para identificar las independencias locales, en esta técnica se calcula la probabilidad a posteriori de que cada oración sea incluida en el resumen. Una de las desventajas de este método es que cuando no hay muchas oraciones relacionadas entre sí se hace difícil establecer la totalidad de probabilidades de transición.

2.1.3.3 Métodos basados en Grafos

En éste tipo de métodos básicamente se representan los documentos en forma de grafo para capturar los conceptos centrales [4]. La manera de decidir sobre la importancia del vértice dentro del grafo toma en cuenta la información global recursivamente calculada desde el grafo completo más que enfocándose en la información de un vértice local específico. Cada nodo puede ser una frase, una oración, un párrafo o incluso un documento dependiendo del objetivo del resumen o del algoritmo. Los bordes entre pares de nodos son las conexiones que pueden existir entre las entidades de texto. Cuando se trata de múltiples documentos [36], para cada documento se genera un resumen usando el algoritmo de clasificación basado en grafos y posteriormente se produce un resumen de resúmenes usando el mismo u otro algoritmo de clasificación. Podría presentarse que todos dirijan al mismo tema o a temas relacionados, por lo tanto, de las relaciones más fuertes se podrían obtener oraciones muy similares o idénticas. Para evitar esos pares de oraciones, los cuales podrían decrementar la legibilidad y la extracción de información relevante, se introduce un máximo umbral sobre la medida de similitud de las oraciones. Consecuentemente en la etapa de construcción del grafo, no enlazan oraciones (vértices) cuya similitud exceda cierto umbral.

Los métodos basados en grafos tienen la ventaja de ser altamente independientes del lenguaje y cuando se pretende obtener una mayor precisión no resulta tan compleja la adaptación a un idioma en particular, sin embargo el método no tiene en cuenta las relaciones semánticas entre las palabras por ser independiente del lenguaje a menos que se lleve a un idioma o a un tema específico. Además, cuando la dimensionalidad es alta (muchas frases o términos en el documento) el costo en términos de ejecución se incrementa considerablemente.

2.1.3.4 Métodos basados en conectividad del texto

Los métodos basados en conectividad del texto intentan tener en cuenta propiedades cohesivas para establecer relaciones entre las expresiones del texto, por ejemplo, las expresiones anáforas que se refieren a partes del texto mencionadas previamente requieren de esos antecedentes para ser comprensibles, si la expresión se extrae sin el contexto previo, podría resultar difícil la comprensión del resumen. Uno de los diferentes enfoques que explora éstas propiedades es el modelo basado en *cadenas léxicas* [37]. Éste modelo consiste en realizar un procesamiento profundo del texto mezclando diferentes herramientas como: *WordNet*[9] para determinar los significados de las palabras y las relaciones entre ellas, *etiquetado gramatical* (POST por sus siglas en ingles, Part of Speech Tagging*)* para determinar los nombres simples**,** análisis sintáctico superficial (en ingles *shallow parser*) para la identificación de palabras compuestas y un algoritmo de segmentación para el pre-procesamiento del texto. Se plantean cuatro pasos para la generación automática de resúmenes, el primero es la segmentación del texto original, el segundo la construcción de las cadenas léxicas, el tercero la identificación de las cadenas más fuertes y finalmente las oraciones significativas son extraídas. Para la construcción de las cadenas, primero se selecciona el conjunto de palabras candidatas, después, para cada palabra candidata, hay que encontrar una cadena apropiada basándose en un criterio de relación entre los miembros de las cadenas y finalmente si se encuentra la cadena apropiada, se inserta la palabra. Para llevar esto a cabo, se seleccionan los nombres simples o compuestos, se identifican secuencias de términos agrupados mediante relaciones como sinónimos, homónimos, hiperónimos, entre otras, obtenidas a través de WordNet conformando interpretaciones y componentes a partir de éstas. Al final, de cada componente se selecciona la interpretación más fuerte. Para el tratamiento de múltiples documentos se propone un modelo para generación de resúmenes de documentos escritos en Chino, indicativos y con una fluidez moderada [38]. El algoritmo construye las cadenas léxicas para cada documento por medio de una base de datos de conocimiento, identifica las cadenas más fuertes, las oraciones significativas son extraídas de cada documento, se mezclan, se ordenan y para el resumen se seleccionan las cadenas con mayor puntuación y con la menor similitud hacia las oraciones ya seleccionadas.

[9] Es una gran base de datos léxica para el idioma Inglés, desarrollado bajo la dirección de George a. Miller, que agrupa las palabras en conjuntos de sinónimos llamados “synsets”, proporcionando definiciones cortas y generales, y almacenando las relaciones semánticas entre estos conjuntos de sinónimos. http://wordnet.princeton.edu/

El problema de este enfoque radica en que requiere de técnicas complejas de procesamiento del texto, que son dependientes del lenguaje, y necesita de varias herramientas externas para el pre-procesamiento del texto.

Otra técnica es la utilización de la teoría de la estructura retórica [39], en donde las relaciones retóricas se refieren al tipo de relación que existe entre dos segmentos de texto los cuales se conectan entre sí como unidades de texto formando un árbol, donde cada hoja del árbol (nodo) son las oraciones del documento original y cada nodo puede ser un satélite o un núcleo, siendo los núcleos los nodos más representativos. Una desventaja de este método es la construcción compleja de la estructura retórica, la cual genera dependencia del idioma y de la estructura del documento.

Estos métodos le dan un mayor trabajo al pre-procesamiento de los datos haciendo que el método aplicado dependa de las herramientas usadas para encontrar las relaciones existentes entre las diferentes entidades del texto.

2.1.3.5 Métodos basados en reducción Algebraica

Con esta técnica se han propuesto varios métodos que utilizan la reducción de la dimensionalidad para la obtención de las oraciones más importantes, una de las técnicas utilizadas es el Análisis Semántico Latente (LSA), esta se define como una técnica matemático-estadística que permite la creación de vectores multidimensionales para el estudio de las relaciones existentes entre palabras y párrafos.

En [40] realizan una extensión del método base propuesto en [41] en el cual realizan compresión de oraciones a través de la eliminación de clausulas que se consideran menos relevantes, para ello hacen uso de LSA mediante la aplicación de dos pasos, (i) creación de la matriz A para los pesos de los términos, (ii). Aplicar descomposición de valor singular (SVD por sus siglas en inglés, Singular Value Decomposition). Para el enfoque que aplica a múltiples documentos, el primer paso es crear una matriz para los pesos de las oraciones del grupo de documentos, el lugar de los términos. Se ejecuta la clasificación de oraciones, cada oración adquiere una puntuación la cual se calcula por medio de SVD. Finalmente se seleccionan las oraciones con mejor puntuación. Para el manejo de redundancia, antes de agregar una oración al resumen, determinan si la oración es similar a alguna de las oraciones ya seleccionadas. Miden la similitud a través de la medida cosenoidal y establecen un umbral de similitud. La desventaja de éste método es que la eliminación de clausulas podría generar oraciones gramaticalmente incorrectas ya que no se tiene control de características propias de un resumen como sinonimia [10]y polisemia[11].

Otra técnica basada en reducción algebraica es el uso de Factorización no-negativa de matrices (NMF por sus siglas en inglés, Non-negative Matrix Factorization) presentado inicialmente en [7], éste es un método no supervisado que obtiene la relevancia de una oración, como resultado el método selecciona oraciones más significativas que las seleccionadas a través de LSA. Posteriormente, se propone un método para generación de resúmenes multi-documento que hace uso de NMF [42] para el agrupamiento de los

[10] Sinonimia es una relación de semejanza de significados entre determinadas palabras.

[11] Polisemia se presenta cuando una misma palabra o signo lingüístico tiene varias acepciones.

documentos, y para la selección de las oraciones más representativas descomponen una oración en una combinación lineal de características semánticas no-negativas dispersas de manera que se pueda representar la oración como la suma de esas características.

2.1.3.6 Generación Automática de Resúmenes con Modelos Evolutivos

Los algoritmos evolutivos [43] surgieron como simulación de procesos de evolución natural, tuvieron su origen en el año 1960 y fueron introducidos por John Holland quien incorporó métodos de selección natural y supervivencia a la resolución de problemas de Inteligencia Artificial (IA). Un algoritmo evolutivo consta de una función objetivo que permite evaluar las soluciones candidatas dando mecanismos de selección que permitan crear nuevas soluciones al problema que se desea resolver.

Los algoritmos evolutivos han sido aplicados a la generación automática de resúmenes, dado que hasta el momento no se han presentado muchos trabajos basados en algoritmos evolutivos para generación de resúmenes de múltiples documentos, se dará primero una breve descripción de los que aplican para un solo documento.

Uno de los métodos basados en modelos evolutivos para generación automática de resúmenes mono-documento hace uso de un algoritmo de optimización de enjambre de partículas cuyo propósito principal es darle una puntuación a las oraciones enfatizándose en las características del texto, realizan un entrenamiento para obtener unos pesos que son usados para ajustar las puntuaciones de las características del texto, lo cual ayuda a que el algoritmo seleccione las oraciones importantes a ser incluidas en el resumen final [13]. En un trabajo similar se propone de igual forma hacer uso de la optimización de enjambre de partículas para determinar la efectividad de cinco características (centralidad, título, puntuación de palabras en la oración, palabras clave, similitud con la primera oración) para encontrar la relevancia de una oración [14]. En cada iteración, el algoritmo selecciona algunas características y sus correspondientes pesos son usados para puntuar las oraciones.

Otro método hace uso de un algoritmo genético para realizar la selección de las oraciones más relevantes donde la población es generada a través de cruce y mutación simple y es evaluada teniendo en cuenta tres factores que constituyen la función objetivo: relación con el tema, cohesión y legibilidad [15]. El *factor de relación con el tema* mide la similitud de las oraciones con el título del documento; en el *factor de cohesión*, cada oración es interpretada en relación con las demás, para esto se mide la similitud entre las oraciones, donde las oraciones se representan como una matriz triangular superior para realizar la comparación; el *factor de legibilidad*, reconoce la facilidad de lectura del resumen mediante la relación entre una oración y la siguiente. Estos tres factores también fueron tomados en cuenta, para la formulación de la función objetivo, en un trabajo posterior que hace uso del algoritmo de búsqueda armónica HS para la generación del resumen [16], en este trabajo dan especial importancia a la función objetivo, la cual permite organizar y clasificar los resúmenes candidatos. Como resultado obtiene mejores resultados que el algoritmo genético propuesto en [15] que a su vez fue comparado con otros resultados de DUC 2002 siendo superiores los de GA. La ventaja de este algoritmo es que no requiere de cálculos matemáticos complejos y puede ser adaptado para diferentes problemas de optimización.

En cuanto a la generación de resúmenes multi-documento basada en algoritmos evolutivos se puede encontrar el trabajo presentado por Bossard, A. [8] el cual realiza una combinación de un sistema de generación de resúmenes multi-documento con un algoritmo genético en donde hacen agrupamiento y el método de selección se basa en la centralidad local para extraer una oración por grupo, adicionalmente hacen uso de un algoritmo genético para realizar la optimización de los catorce parámetros utilizados para la creación del resumen. Otro trabajo que cabe resaltar es presentado por Alguliev, R. , en éste se intenta seleccionar las oraciones con la mayor cobertura y tener la menor redundancia en el resumen, para esto se propone una función objetivo que es probada con un algoritmo evolutivo de optimización de enjambre de partículas binario [9].

2.1.4 Criterios para la selección del algoritmo de generación de resúmenes de múltiples documentos.

Dado que en la investigación realizada por Cobos C. [44] se pudo establecer que GHS+LEM presentaba mejores resultados que otros algoritmos para optimización en general cuando se presenta alta dimensionalidad y teniendo conocimiento que ya existía un algoritmo de búsqueda armónica para generación de resúmenes planteado por Sharegui, se decidió aplicar el GHS+LEM partiendo de los criterios planteados en la Tabla **1**

Criterio	GHS+LEM
Criterio 1	Según lo planteado por Cobos C. este algoritmo es mejor que HS IHS(*Improved Harmony Search* o IHS por sus siglas en inglés) y GHS dado que utiliza aprendizaje de máquina para generar reglas las cuales delimitan el espacio de búsqueda de nuevos individuos.
Criterio 2	El algoritmo presenta mejores resultados cuando se tiene alta dimensionalidad, como es el caso de la generación de resumen de múltiples documentos, donde la dimensionalidad está dada por la cantidad de oraciones.

Tabla 1. Criterios selección del algoritmo GHS+LEM

2.1.5 Evaluación de la Calidad de los Resúmenes

La evaluación de los resúmenes generados automáticamente era una tarea difícil debido a las irregularidades que se presentan en los distintos idiomas y que no existía un resumen ideal para realizar la comparación, por lo tanto se requería del juicio de una persona experta capaz de detectar la coherencia, la consistencia gramatical, la legibilidad y el contenido del documento. En los primeros métodos la evaluación se realizó de forma manual [31], [45], [32] lo que era una tarea compleja y requería de un gran esfuerzo humano, además se presentaba mucha subjetividad por su dependencia en la opinión de los jueces. Actualmente, se han desarrollado otras formas de evaluar la calidad de los resúmenes así como también se han creado herramientas para realizar ésta tarea de manera más eficiente.

2.1.5.1 Tipos de Evaluación

Los métodos para evaluar la calidad de los resúmenes se pueden dividir en dos grandes grupos: evaluación Intrínseca y evaluación extrínseca [46].

2.1.5.1.1 Evaluación intrínseca

La evaluación intrínseca mide la calidad del sistema de generación de resúmenes sin tener en cuenta la audiencia, es decir, a quién va dirigido el resumen, dando mayor peso a aspectos como la coherencia o lo informativo del resumen generado. La mayoría de sistemas de evaluación son intrínsecos. En estos métodos de evaluación generalmente se tiene un conjunto de resúmenes ideales denominados “gold standard corpus” uno por cada documento o por cada conjunto de documentos para el caso de resúmenes de múltiples documentos. Después se compara el resumen generado con éste, midiendo el solapamiento del contenido. Este tipo de evaluación se puede hacer por medio de medidas estándar de Precisión y Recuerdo [1] y medidas de ROUGE [47]. Dado que no existe un resumen “perfecto", algunas investigaciones para la evaluación del resumen generado utilizan más de un resumen realizado por humanos para cada documento de prueba o por cada conjunto de documentos de prueba, y promedian el puntaje obtenido por el sistema a través del conjunto de resúmenes ideales.

2.1.5.1.2 Evaluación Extrínseca

La evaluación extrínseca se basa en ciertas métricas como coherencia, consistencia gramatical, legibilidad, a diferencia de la intrínseca, ésta evaluación se enfoca en el usuario final midiendo la eficiencia y aceptabilidad de los resúmenes generados, por ejemplo, la relevancia así como también el esfuerzo y el tiempo requerido para la comprensión de la lectura del resumen, también se tiene más en cuenta la utilidad que este puede tener sobre un usuario que su calidad como resumen. Éste tipo de evaluación requiere del esfuerzo de muchas personas y más aún cuando se trata de resumir muchos documentos extensos. Se han propuesto diversos escenarios para la evaluación extrínseca como el Juego de Shannon, el Juego de la Pregunta, el juego de la Categorización o Clasificación y Asociación de claves [1].

2.1.5.2 Evaluación con ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

La evaluación de los resúmenes generados por un sistema requiere de cualquiera de los dos tipos de evaluación, intrínseca o extrínseca, para la presente investigación se va a realizar la evaluación por medio de ROUGE [23], que es una técnica de evaluación intrínseca basada en BLEU [48] que califica la proximidad de un resumen candidato (generado automáticamente) contra un conjunto de resúmenes de referencia o “ideales” por medio de la co-ocurrencia de n-gramas. ROUGE se ha convertido en una herramienta muy usada para la evaluación de los resúmenes generados automáticamente y obtiene

los valores de precisión, recuerdo y medida-F[12]. El objetivo es que los textos con un significado similar, deben contener palabras o frases comunes.

Existen variaciones de ROUGE como ROUGE-N, ROUGE-L, ROUGE-W, ROUGE-S y ROUGE-SU; la diferencia entre ellas está en la cantidad o en la forma como se tomen los n-gramas. Para generación automática de resúmenes de múltiples documentos, las técnicas más usadas son ROUGE-1, ROUGE-2 y ROUGE-SU4.

ROUGE en su versión 1.5.5 realiza la puntuación de la siguiente manera:

- ROUGE-N: Mide la superposición de n-gramas de palabras entre el resumen del sistema y los resúmenes de referencia. Se calcula como se muestra en la Ecuación 1:

$$\text{ROUGE} - \text{N} = \frac{\sum_{S \in \{ResumenesModelo\}} \sum_{grama_n \in S} Conteo_{n-gramas\ que\ coinciden}}{\sum_{S \in \{ResumenesModelo\}} \sum_{grama_n \in S} Conteo_{n-gramas}} \quad (1)$$

  Donde n es la longitud del n-grama y $Conteo_{n-gramas\ que\ coinciden}$ es el número máximo de n-gramas co-ocurrentes en el resumen candidato y el conjunto de resúmenes modelo. El denominador de la ecuación es el número total de n-gramas ocurrentes en el resumen modelo.
  Es indiferente el orden en que se encuentren los n-gramas, por ejemplo un n-grama que esté al principio en el resumen generado por el sistema puede estar al final del resumen modelo y tendrá la misma puntuación.

- ROUGE-L (Sub-secuencia común más larga, LCS): Emplea la longitud de las secuencias más largas que son comunes entre el resumen generado por el sistema y el de referencia.

  Tomando una oración del resumen como una secuencia de palabras, se propone LCS para estimar la similitud entre dos resúmenes, X de longitud m y Y de longitud n asumiendo X como la oración del resumen de referencia y Y como la oración candidata del resumen. El recuerdo se calcula como se muestra en la Ecuación 2:

$$R_{lcs} = \frac{\sum LCS\ (X,Y)}{m} \quad (2)$$

  Donde m es la longitud del resumen modelo y LCS es la longitud de la sub-secuencia común más larga de X y Y.
  Por ejemplo dado:

  $\text{X} = \{\text{A}, \text{B}, \text{C}, \text{D}, \text{E}, \text{F}, \text{G}\}$
  $\text{Y}_1 = \{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \text{H}, \text{I}, \text{K}\}$
  $\text{Y}_2 = \{\mathbf{A}, \text{H}, \mathbf{B}, \text{K}, \mathbf{C}, \text{I}\ \mathbf{D}\}$

  La puntuación para Y_1 es 4/7 = 0,571 y para Y_2 es 1/7+ 1/7+ 1/7+1/7 = 0,571. Lo que muestra que Y_1 y Y_2 en este caso tienen la misma puntuación.

[12] Es una combinación de Precisión y Recuerdo.

El problema de ROUGE-L, como se puede ver en el ejemplo es que no diferencia las relaciones espaciales dentro de sus secuencias embebidas, en el ejemplo anterior se puede ver claramente que Y_1 debería tener mayor peso que Y_2.

- ROUGE-W: es una versión ponderada de ROUGE-L que además de la longitud de la secuencia valora la ausencia de "huecos" en la misma.

- ROUGE-S (Skip-Bigram Co-Occurrence Statistics)

Un Skip-Bigram es cualquier para de palabras en el orden de la oración, permitiendo desfases arbitrarios. *Skip-Bigram Co-Occurrence Statistics* mide la superposición de los Skip-Bigrams entre el resumen candidato y el resumen modelo.

Dados dos resúmenes X de longitud m y Y de longitud n, asumiendo que X es el resumen modelo y Y el resumen candidato se puede medir el Recuerdo mediante ROUGE-S como se muestra en la Ecuación 3:

$$R_{skip2} = \frac{SKIP2(X,Y)}{C(m,2)} \quad (3)$$

Donde $SKIP2(X,Y)$ es el número de Skip-Bigram que coinciden entre X y Y y $C(m,2)$ es la cantidad de Skip-Bigram obtenidos de X.

Por ejemplo, dados:

X: police killed the gunman
Y: *police kill the gunman*

Se pueden obtener los siguientes Skip-Bigram:

- Skip-Bigram(X) = { police killed, police the, police gunman, killed the, killed gunman, the gunman}
- Skip-Bigram(Y) = { police kill, police the, police gunman, kill the, kill gunman, the gunman}

Los Bigramas en común entre X y Y son: {pólice the, pólice gunman, the gunman}, entonces el valor de $R_{skip2} = \frac{3}{6} = 0{,}5$

El resultado obtenido con R_{skip2} es más intuitivo que ROUGE-L, la ventaja es que no requiere de coincidencias consecutivas aunque aún se conserva cierta sensibilidad al orden de las palabras. R_{skip2} cuenta todas las concordancias de pares de palabras en orden mientras que ROUGE-L solo cuenta la sub-secuencia común más larga.

- ROUGE-SU (Extensión de ROUGE-S)

Un problema potencial de ROUGE-S es que no da ningún crédito a las oraciones candidatas si la oración no tiene ninguna pareja de palabras co-ocurrentes entre el

resumen modelo y el resumen candidato. Por ejemplo dado el resumen modelo *X = {police killed the gunman}* y el resumen candidato *Y = {gunman the killed police}* en ROUGE-S tendrían una similitud de cero, por lo que podría ser conveniente calcular la co-ocurrencia de palabras no a través de bigramas sino de unigramas.

2.1.5.3 Corpus de Evaluación

DUC[13] (*Document Understanding Conference*) es un foro encargado de evaluar los algoritmos de generación automática de resúmenes. Uno de los principales objetivos de éste foro es lograr la estandarización sobre el modo de evaluación de este tipo de sistemas, los participantes pueden entrenar y evaluar sus sistemas sobre la gran cantidad de corpus debido a que desde sus inicios (desde el año 2001), los encargados del foro se han ocupado de la recolección y evaluación de diferentes sistemas de generación de resúmenes, lo que ha permitido la creación de múltiples corpus.

A partir del surgimiento de ROUGE, DUC decidió tomarlo como medida estándar debido a los buenos resultados en el manejo de la correlación de los resúmenes generados automáticamente y los resúmenes generados por humanos [28].

2.2 REPRESENTACIÓN DE LOS DOCUMENTOS

2.2.1 Modelo Vectorial

La representación comúnmente utilizada para tareas de Recuperación de la Información es el modelo de espacio vectorial [49] mediante el cual se representa cada unidad de texto (oraciones[14]) como un vector de términos ponderados. El modelo vectorial está basado en que cada oración de la colección está representada por un vector n-dimensional (n es la cardinalidad del conjunto de términos de indexación elegido para toda la colección de oraciones), en el que cada componente representa el peso del término asociado a esa dimensión. Este peso representa un estimado (usualmente estadístico, aunque no necesariamente) de la utilidad del término como descriptor del documento, es decir, de la utilidad para distinguir esa oración del resto de unidades de la colección. Un término recibe un peso de 0 en las oraciones en las cuales éste no ocurre. Normalmente los términos muy comunes y los poco frecuentes son eliminados y las formas diferentes de una palabra son reducidas a su forma canónica. Para tomar en consideración unidades textuales de diferentes longitudes, es usual, que los vectores sean normalizados causando que la mayoría de los vectores sean dispersos.

Dada un documento D formado por *N* unidades textuales, entonces una oración es representada según el modelo vectorial, como un vector $\overrightarrow{s_j} = (w_{1j}, w_{2j}, \ldots, w_{nj})$ donde *n* es el número total de términos de D y w_{nj} representa la importancia de término dentro de la oración.

[13] http://duc.nist.gov/

[14] Oración: Para nuestro caso es una oración simple, o compuesta de una o varias frases.

2.2.2 Esquemas de Pesado de Términos

Existen diferentes maneras de asignarle el peso a los términos [50], entre ellas está la técnica de pesado *booleano* donde los pesos $w_i \in \{0,1\}$ indican la presencia o ausencia del término t_i en el documento, otra es la técnica de *frecuencia de un término [51]* que indica el número de veces que el término t_k aparece en el documento, denotado *TF(t,d).*

La técnica usada en este trabajo (*ponderación por frecuencia relativa TF-ISF*) normalmente es usada en trabajos de generación automática de resúmenes, esta es una mezcla de *TF* con la *frecuencia inversa de la oración (ISF*[15]*)* que tiene que ver con la poca frecuencia de un término en la colección de oraciones, el cálculo de los pesos se hace de la siguiente manera:

$$W_{t,s} = TF_{t,s} \times ISF_t \quad (4)$$

Donde:

$$TF_{t,s} = \frac{Freq_{t,s}}{Max\ Freq_s} \quad (5)$$

$$ISF_t = \log\left(\frac{N}{n_t}\right) \quad (6)$$

Donde $TF_{t,s}$ es la frecuencia del término *t* en la oración *s,* ISF_t es la frecuencia invertida del término *t*, $Freq_{t,s}$ es la frecuencia del término *t* en la oración *s*, $Max\ Freq_s$ es la máxima frecuencia de términos de la oración *s*, *N* es la cantidad de oraciones en la colección y n_t es la cantidad de veces que aparece el término *t* en todas las oraciones.

2.2.3 Procesamiento de Múltiples Documentos

La representación de múltiples documentos se hace basado en el modelo de espacio vectorial y de acuerdo a [9], donde $D = \{d_1, d_2, \dots, d_n\}$ es el conjunto de documentos y *n* es el número total de documentos; por facilidad se representa la colección de documentos como un conjunto de todas las oraciones de todos los documentos en la colección, es decir, $D = \{s_1, s_2, \dots, s_m\}$ donde *m* es el número total de oraciones de la colección de documentos y *s* es la colección de términos $s_{i,} = \{t_{1,i}, t_{2,i}, \dots, t_{k,i}\}$. El objetivo es obtener un subconjunto de *D* con las oraciones que satisfagan los tres factores para la generación de un buen resumen. Los factores serán definidos más adelante. Cada oración s_i es representada como un vector con los pesos de los términos, $\vec{s_i} = \{w_{i,1}, w_{i,2}, \dots, w_{i,m}\}$ donde *m* es el número de términos en la colección de documentos, $w_{i,k}$ es el peso de término t_k en la oración s_i. El componente w_{ik} se define usando la matriz de términos por oración *tf-isf*, la cual asigna un valor mediante la combinación de la frecuencia de términos y la frecuencia inversa de términos, como se presentó en la sección 2.2.2.

[15] inverse sentence frequency"

2.2.4 Medidas de Similitud

Una manera de determinar qué tanto se parecen dos documentos u oraciones es estableciendo una relación entre ellos a través de la comparación del vocabulario que los componen. Para realizar la comparación se puede hacer uso de distintas medidas de similitud, las cuales juegan un rol muy importante en el área de generación de resúmenes. Se debe tener en cuenta que para aplicar cualquiera de las medidas descritas a continuación, es necesario definir una manera de representación de los documentos donde éstas puedan ser aplicadas. Para el caso de este proyecto, el modelo de representación vectorial. A continuación se presenta el estudio realizado para la selección de la medida que más se adaptará a éste problema.

2.2.4.1 Medida de Cosenos

Esta medida es ampliamente usada en generación de resúmenes debido a su sensibilidad, a la importancia relativa de cada palabra, y ha sido usada en [28], [16], [52], [53], [54], [55], [56], [57], entre otros. La idea básica es medir el ángulo entre el vector de $\overrightarrow{s_i}$ y de $\overrightarrow{s_j}$, para hacerlo, calculamos:

$$SC(s_i, s_j) = \frac{\sum_{k=1}^{t} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{t} w_{jk}^2 \ \sum_{k=1}^{t} w_{jk}^2}} \qquad (7)$$

Donde *k* va de 1 al número total de términos del vocabulario *t*, W_{ik} indica el peso del término *k* en la oracion s_i y W_{jk} el peso del término *k* en la oración s_j.

La medida cosenoidal es una función trigonométrica que mide el Coeficiente de Similitud entre dos documentos o dos oraciones, representados en un espacio vectorial, Mide el angulo *(0º<beta<90º)*, que indica que tan cercano esta el uno del otro, en terminos de la dimensionalidad [58]. Entre mas pequeño sea el angulo mayor será la similitud, (inversamente Proporcional), al aplicar la formula, el valor estará entre 0 y 1 siendo 1 el mayor grado de similitud entre dos oraciones.

2.2.4.2 Medida DICE

Esta medida determina el número de terminos comunes entre dos vectores. Donde *k* va de 1 al número total de terminos del vocabulario *t*, W_{ik} indica el peso del término *k* en la oración s_i, y W_{jk} el peso del término *k* en la oración s_j. El coeficiente de DICE es obtenido por medio de:

$$SC(S_i, S_j) = \frac{\sum_{k=1}^{t} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{t} w_{jk}^2 + \sum_{k=1}^{t} w_{jk}^2}} \qquad (8)$$

2.2.4.3 Medida de Jaccard

Jaccard es una función estadística que permite comparar términos en común entre dos documentos [59]. Si dos documentos (o unidades de texto) que son representados mediante el modelo de espacio vectorial presentan cierta cantidad de términos en común, entonces los documentos serán similares. Al comparar con Jaccard dos documentos de palabras se obtiene un número entre cero y uno. Siendo 1 el valor que se obtiene cuando los documentos son completamente semejantes y 0 cuando son completamente diferentes. Esta es una medida esencialmente combinatoria [60], pues se fija más en el

tamaño que en lo que contienen los conjuntos. El coeficiente de Jaccard es calculado de la siguiente manera:

$$SC(S_i, S_j) = \frac{\sum_{k=1}^{t} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{t} w_{jk}^2 + \sum_{k=1}^{t} w_{jk}^2 - \sum_{k=1}^{t} w_{ik} w_{jk}}} \qquad (9)$$

Donde *k* va de 1 al número total de terminos del vocabulario *t*, W_{ik} indica el peso del término *k* en la oración S_i y W_{jk} el peso del término *k* en el oración S_j.

2.2.4.4 Distancia de Google Normalizada (NGD)

NGD toma el número de hits retornados por motor de búsqueda de Google [9], para calcular la distancia semántica entre conceptos, los cuales son representados como etiquetas que alimentan la búsqueda de términos en el motor de google, para el cálculo de la similitud entre dos oraciones s_i y s_j donde $s_{i,} = \{t_{1,i}, t_{2,i}, \dots, t_{k,i}\}$, $s_j = \{t_{1,j}, t_{2,j}, \dots, t_{k,j}\}$ y t_k es un término, es calculada de la siguiente manera:

$$Sim_{NGS}(S_i, S_j) = \frac{\sum_{t_k \epsilon s_i} \sum_{t_i \epsilon s_j} Sim_{NGD}(t_k, t_l)}{|s_i| . |s_j|} \qquad (10)$$

$$Sim_{NGD}(t_k, t_l) = \exp(-NGD(t_k, t_l)) \qquad (11)$$

$$\mathrm{NGD}(t_k, t_l) = \frac{\max\{\log(f_k), \log(f_l) - \log(f_{kl})\}}{\log n - \min\{\log(f_k), \log(f_l)\}} \qquad (12)$$

Donde, f_k es el número de oraciones que contiene el término t_k, $f_{k,l}$ es el número de oraciones que contienes los dos términos t_k, t_l y *n* es el número de oraciones del documento.

2.2.4.5 Distancia Euclidiana

La Distancia Euclidiana mide la distancia en línea recta entre dos puntos, entre más lejanos de cero estén menos semejantes son los vectores. Esta medida es calculada de la siguiente manera:

$$DE = \sqrt{\sum_{ui=1}^{srcUIs} (sry(ui) - src(ui)^2)} \qquad (13)$$

Donde *sry y src* son los vectores a comparar, *ui* es un ítem o unidad, *srcUIs* y *sryUIs* son todas las unidades del texto o el resumen.

2.3 ALGORITMOS DE BÚSQUEDA ARMÓNICA

La programación evolutiva involucra algoritmos de optimización de meta heurísticas, los cuales son: Algoritmos evolutivos, que comprenden los algoritmos genéticos, programación evolutiva, la estrategia de evolución y la programación genética; y la inteligencia de enjambre. En el estudio de la programación evolutiva se encuentran los algoritmos evolutivos, y entre estos, un algoritmo de interés particular para el presente

trabajo, es la Búsqueda Armónica (Harmony Search) desarrollado por Lee K. y Geem Z. [61], el cual ha mostrado buenos resultados en generación automática de resúmenes de documentos [16]. En el presente trabajo se ha usado el Algoritmo de Búsqueda Armónica y algunas variaciones del mismo.

2.3.1 Búsqueda Armónica

El algoritmo básico de búsqueda armónica HS planteado por Zong W. y Kang S. [61], es un algoritmo meta heurístico que requiere pocos cálculos matemáticos y puede ser fácilmente adaptado para problemas de optimización, el algoritmo busca encontrar una solución global determinada por una función objetivo mediante procedimientos iterativos que guían una heurística combinando de forma inteligente distintos conceptos para explorar y explotar adecuadamente el espacio de búsqueda. HS simula el proceso de improvisación musical, en el cual los músicos buscan producir una armonía agradable determinada por el estándar estético auditivo [62]. Cuando un músico esta improvisando, realiza las siguientes acciones:

- Toca alguna melodía conocida que ha aprendido anteriormente.
- Toca algo parecido a la melodía anteriormente mencionada, ajustándola poco a poco al tono deseado.
- Posteriormente compone una nueva melodía basándose en sus conocimientos musicales a partir de nuevas notas seleccionadas aleatoriamente.

Estas tres acciones corresponden a los componentes de la búsqueda armónica: Uso de la memoria armónica, ajuste de tono y aleatoriedad, respectivamente.

Del mismo modo en el proceso de optimización en ingeniería, cada variable de decisión inicialmente toma valores aleatorios dentro del rango posible, formando un vector solución. Si dicho vector, es decir, dicho conjunto de valores que lo conforman son una buena solución según la función objetivo del problema, entonces estos valores son almacenados (en la memoria armónica) y la posibilidad de formar una buena solución se incrementa para la siguiente iteración. Por ejemplo si tenemos tres armonías conformadas por tres variables de decisión.

- Variable 1: {100,300,500}
- Variable 2: {700,500,200}
- Variable 3: {600,400,100}

De la primera se toma {100} de la segunda {500} y de la tercera {100} el nuevo vector armónico será {100, 500,100} si esta nueva armonía es mejor que la peor armonía en memoria, esta es remplazada.

En este proceso existen dos parámetros importantes que se deben tener en cuenta al momento de ejecutar el algoritmo HS, uno es el porcentaje de consideración de la memoria armónica (HMCR por sus siglas en inglés, harmony memory considering rate) que sirve para determinar si el elemento a seleccionar se toma de la memoria armónica o de todo el conjunto de oraciones, y la tasa de ajuste del tono (PAR por sus siglas en inglés Pitch Adjusting Rate) que permite ajustar un determinado ancho de banda mediante una pequeña cantidad aleatoria relativa al tono existente o una solución existente de la memoria armónica. El algoritmo HS funciona de la siguiente manera:

Paso 1. Inicializar los parámetros: El problema de optimización se define como minimizar (o maximizar) $f(x)$ tal que $LB_i < x_i < UB_i$ donde, $f(x)$ es la función objetivo, x es una solución candidata que consiste de N variables de decisión (x_i), y LB_i y UB_i son el límite de decisión más bajo y el más alto de cada variable, respectivamente. Los parámetros de HS se especifican en este paso. Estos parámetros son el tamaño de la memoria armónica (HMS), la tasa de consideración de la memoria armónica (HMCR), la tasa de ajuste del tono (PAR), el ancho de banda de ajuste del tono (BW) y el número de improvisaciones (NI).

Paso 2. Inicializar la memoria armónica: La memoria armónica inicial es generada desde una distribución uniforme en los rangos [LB_i, UB_i], donde $1 \leq i \leq N$. Esto se realiza de la siguiente manera: $x_i' = LB_i + r \times (UB_i - LB_i)$, donde j = 1,2,…, HMS y $r \sim U(0,1)$. La variable r hace referencia a un número aleatorio y $U(0,1)$ a la función que genera el número aleatorio uniforme.

Paso 3. Improvisar una nueva armonía: El proceso de generación de una nueva armonía es llamado improvisación. El nuevo vector armónico, $x' = (x'_1, x'_2, \ldots, x'_N)$, se genera utilizando las siguientes reglas: consideración de la memoria, ajuste del tono y selección aleatoria. Este procedimiento se muestra en la Figura 1. En la línea 006, r es un número aleatorio uniforme entre 0 y 1, y el valor BW es un ancho de banda arbitrario de la distancia para variables de diseño continuas [63].

```
001 para cada i ∈ [1, N] hacer
002     si U(0,1) < HMCR entonces /*consideración de la memoria*/
003         inicio
004             x_i' = x_i^j, donde j ~ U(1, …, HMS)
005             si U(0,1) ≤ PAR entonces/*ajuste del tono*/
006                 x_i' = x_i' + r × bw
007             fin_si
008         fin
009     sino /*selección aleatoria*/
010         x_i' = LB_i + r × (UB_i − LB_i)
011     fin_si
012 continuar_para
```

Figura 1. Improvisación de una nueva armonía.

Paso 4. Actualizar la memoria armónica: El vector armónico generado, $x' = (x'_1, x'_2, \ldots, x'_N)$, reemplaza la peor armonía almacenada en la memoria armónica si el fitness (o valor de aptitud del vector armónico actual, medido en términos de la función objetivo) es mejor que el de la peor armonía.

Paso 5. Verificar el criterio de parada: Los pasos 3 y 4 son repetidos hasta terminar el criterio de satisfacción. La condición de terminación puede ser definida de dos formas:

1. Dejar que el resultado se vaya a un estado estable.
2. Configurar un número de iteraciones para su cálculo.

2.3.2 Mejor Búsqueda Armónica Global

En el año 2008, Omran M. y Mahdavi M. propusieron el algoritmo de mejor búsqueda armónica global (GHS por sus siglas en inglés: Global-best Harmony Search) [18], el cual hibrida la búsqueda armónica original con el concepto de inteligencia de enjambre, este algoritmo presentó mejores resultados que la Búsqueda Armónica (HS), modificando el paso de ajuste del tono en HS de modo que la nueva armonía puede imitar a la mejor armonía en la memoria armónica. Esto permite a GHS trabajar eficientemente en problemas continuos y discretos. En general GHS es mejor que HS cuando se aplica a problemas de gran dimensionalidad y cuando hay presencia de ruido [18] como es el caso de la generación automática de resúmenes de múltiples documentos.

GHS tiene exactamente los mismos pasos que HS con la salvedad de la modificación del paso 3 que corresponde a la improvisación de un nuevo armónico, donde el ancho de banda BW no es tomado en cuenta y la generación de la nueva armonía imita la mejor improvisación de la memoria armónica, para mayor claridad consultar la Figura 2.

001 **para cada** $i \in [1, \mathrm{N}]$ **hacer**
002 **si** $U(0,1) < HMCR$ **entonces** */*consideración de la memoria*/*
003 **inicio**
004 $x_i^{'} = x_i^{j}$, donde $j \sim U(1, \ldots, HMS)$
005 **si**$U(0,1) \leq PAR(t)$ **entonces** */*ajuste del tono para la generación t*/*
006 $x_i^{'} = x_k^{best}$, *donde* $best$ *es el índice de la mejor armonía en* HM *y* $k \sim U(1, N)$
007 **fin_si**
008 **fin**
009 **sino** */*selección aleatoria*/*
010 $x_i^{'} = LB_i + r \times (UB_i - LB_i)$
011 **fin_si**
012 **continuar_para**

Figura 2. Improvisación en el algoritmo de la mejor búsqueda armónica global (GHS).

2.3.3 Modelos Evolutivos que Aprenden

LEM (Learnable Evolution Model), o Modelo Evolutivo que Aprende es un modelo no darwiniano (no se basan únicamente en la teoría de evolución darwiniana) propuesto por Ryszard S. Michalski[64] en el año 2000, que hace parte de la computación evolutiva y que emplea aprendizaje de máquina con el objetivo de determinar la generación de nuevos individuos (soluciones candidatas al problema). Al usar el modo de aprendizaje de máquina, LEM puede determinar cuáles individuos de una población (o un conjunto de individuos de poblaciones antiguas) son superiores a otros en la realización de ciertas tareas designadas (dichas tareas suelen verse como las características primordiales que deben tener los individuos de la población). Estas razones, expresadas como hipótesis inductivas (es decir, características identificadas y consideradas como esenciales), se usan para la generación de nuevas poblaciones [65]. En el modo de evolución darwiniana, el algoritmo usa operaciones aleatorias o semi-aleatorias para la generación de nuevos individuos (usando técnicas de mutación y/o recombinación de la teoría de evolución darwiniana). Una característica notable de LEM es su capacidad para dar saltos cuánticos (saltos perspicaces que pueden realizar un ajuste) en la función objetivo [65].

2.3.4 Búsqueda Armónica Global con Modelos Evolutivos que Aprenden

Este es un algoritmo propuesto en [44] en el cual se emplean técnicas de aprendizaje de máquina para generar nuevas poblaciones, este método puede determinar cuáles individuos de una población (o un conjunto de individuos de poblaciones anteriores) son mejores a otros en la realización de ciertas tareas. Estas razones, expresadas como hipótesis inductivas, se usan para la generación de nuevas poblaciones. Luego, cuando el algoritmo se ejecuta en modo de evolución darwiniana, usa operaciones aleatorias o semi-aleatorias para la generación de nuevos individuos (usando técnicas de mutación y/o recombinación tradicionales).

El modo de aprendizaje de maquina propuesto en GHS+LEM hace uso de una variación del algoritmo de inferencia de reglas PRISM, el cual toma como entrada un conjunto de entrenamiento ordenado por los valores de cada atributo, los resultados de PRISM se emiten como reglas individuales para cada una de las clasificaciones que figuran en términos de los atributos descritos, una de las ventajas de este algoritmo es que puede trabajar tanto con variables continuas como discretas y está destinado a imitar las decisiones simples que manejan los algoritmos de la familia armónica, el algoritmo de inferencia de reglas se ejecuta por primera vez inmediatamente después de creada la memoria armónica inicial, este algoritmo además de los parámetros ya utilizados en GHS contiene los siguientes:

- Tasa de consideración de reglas (RCR rule consideration rate): Este parámetro decide en que porcentaje de las veces se utilizarán las reglas, de lo contrario se ejecutará el método tradicional.
- Tasa de actualización de reglas (RRU rule update): Especifica en que porcentaje de las veces se deben actualizar las reglas, Si un número aleatorio entre 0 y 1 es menor que el valor de RRU, se ejecuta el proceso de inferencia de reglas nuevamente
- Tamaño de los grupos de alto y bajo rendimiento (HLGS high and low group size): indica el tamaño de los grupos de alto desempeño y los grupos de bajo desempeño, este valor debe ser $\leq \lfloor \mathrm{HMS}/2 \rfloor$.

Capítulo 3

3 SISTEMA DE GENERACIÓN DE RESUMENES MULTI-DOCUMENTO CON ALGORITMOS DE BÚSQUEDA ARMÓNICA

3.1 FACTORES PARA UN BUEN RESUMEN

Recientes investigaciones [9, 10, 66, 67] muestran que para la generación automática de resúmenes de múltiples documentos, encontrar la similitud entre las oraciones es un proceso muy importante, dado que permite determinar qué tan relevantes son las oraciones extraídas para el resumen, este proyecto parte de los factores para un buen resumen planteados por Shareghi E. [16] los cuales son:

- Relación con el tema: las palabras que conforman el título pueden ser elementos clave al momento de seleccionar las oraciones, pues generalmente el título contiene características específicas del contenido del documento, este factor trata de encontrar las oraciones que mejor representen al documento. Para ello se mide la similitud entre las oraciones del resumen y el titulo del documento.
- Cohesión: es un factor para reconocer si las oraciones en el resumen están discutiendo sobre el mismo tema o no. Para ello se calcula la similitud entre todas las oraciones que están en el resumen candidato.
- Legibilidad: un resumen legible es aquel, cuyas oraciones son altamente relacionadas con sus oraciones predecesoras.

Tomando como referencia los pasos para la generación de un buen resumen de múltiples documentos que se encuentran en las últimas investigaciones [9, 10], en este proyecto se hace la siguiente reconfiguración de los factores anteriormente mencionados:

- Cobertura: Un resumen con cobertura es aquel que contiene los aspectos principales del documento con la menor pérdida de información, por tanto, las oraciones seleccionadas deben abarcar la mayor cantidad de información contenida dentro del conjunto total de oraciones. Por otra parte, el factor de relación con el tema planteado por Sharegui [16] selecciona las oraciones más relevantes hacia el título, lo cual no aplica para múltiples documentos, debido a que no se tiene un único título que represente a todos los documentos. En este trabajo, la cobertura intenta seleccionar las oraciones más relevantes hacia todas las oraciones del documento, para lo cual es importante medir la similitud de cada oración del resumen con respecto a toda la colección de documentos. Alguliev R. [9] plantea una fórmula con el factor de cobertura la cual se tomo para el presente trabajo, pero se adicionó la parte del denominador para poder normalizar los valores obtenidos, como se presenta en la ecuación (14).

$$\mathrm{FCb} = \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}[\mathrm{sim}(D,Si)+\mathrm{sim}(D,Sj)]}{m*2} \qquad (14)$$

Donde *Si* y *Sj* son oraciones del resumen, *D* es la colección de oraciones de todos los documentos, *sim(D,Si)* es la similitud de la oración *i* del resumen con todas las

oraciones del conjunto de documentos, *sim(D,Sj)* es la similitud de la oración *j* con todas las oraciones del conjunto de documentos, *n* es la cantidad de oraciones que hay en el resumen, *m* es la cantidad de combinaciones entre pares de oraciones del resumen y el *2* es una constante que se utiliza para sumar todas las similitudes calculadas entre las oraciones, debido a cada combinación de las sumatorias (*m*) involucra dos oraciones.

- Eliminación de redundancia: Un resumen sin redundancia es aquel que contiene oraciones que no expresan la misma información, por el contrario, un resumen es cohesivo si las oraciones que están en el resumen tratan sobre el mismo tema. El factor de cohesión para un solo documento planteado por Shareghi [16], busca que las oraciones que estén en el resumen estén relacionadas entre ellas. Dado que el algoritmo planteado en este proyecto es para generar resúmenes de múltiples documentos, y en este tipo de algoritmos la redundancia debe ser eliminada, el factor de cohesión planteado por Shareghi debe restar y no sumar en la función objetivo.

  En el trabajo planteado por Alguliev R. [9] se propone una manera de eliminar redundancia que es similar al factor de cohesión planteado por Shareghi, la diferencia se encuentra en que el factor de cohesión se encuentra normalizado y además se le aplica una función de logaritmo. En la presente investigación se realizaron pruebas para determinar cuál de los dos factores podría influir mejor en la generación del resumen (remitirse al anexo D), y el resultado fue mejor cuando se resta el factor de cohesión que cuando se resta el factor de eliminación de redundancia planteado por Alguliev R. De acuerdo a lo anterior se selecciona la Ecuación (15) para realizar la eliminación de redundancia.

$$\mathrm{FER} = \frac{\log(\mathrm{C}*9+1)}{\log(\mathrm{M}*9+1)} \qquad (15)$$

  Donde C y M son el promedio y la máxima similitud de las oraciones que están en el resumen respectivamente.

El factor de legibilidad planteado para la generación de resúmenes para un solo documento propuesto por Sharegui, busca que las oraciones del resumen estén relacionadas con sus predecesoras, en el caso de la generación de resúmenes de múltiples documentos este factor haría que se seleccionen oraciones similares y además que sea más probable que se tomen oraciones del mismo documento y no de todos los documentos, generando redundancia. Por esta razón, el factor de legibilidad no se toma en cuenta en la función objetivo planteada en la presente investigación.

Como se mencionó en el Capítulo 2, existen diversas medidas de similitud que son usadas en generación de resúmenes, pero según la investigación realizada la más ampliamente usada es la similitud de cosenos [1, 16, 28], la cual es considerada como la medida estándar que hasta el momento ha producido buenos resultados, por lo tanto se usará esta medida para realizar todos los cálculos de similitud.

3.2 FUNCIÓN OBJETIVO

La formulación o selección de la función objetivo es muy importante en los algoritmos evolutivos, dado que es la función que busca optimizar, lo mismo ocurre para la generación de resúmenes con búsqueda armónica. Para la definición de esta función se

tienen en cuenta los dos factores mencionados anteriormente: cobertura y eliminación de redundancia, los cuales están ponderados por el coeficiente beta, que hace flexible esta función permitiendo que se le dé mayor o menor importancia a cada uno de los factores. Esta función objetivo se define a continuación:

$$f(x) = \beta * FCb - (1 - \beta) * FER \qquad (16)$$

En esta función el coeficiente β varía entre cero y uno.

3.3 ADAPTACIÓN DEL ALGORITMO HS PARA GENERACIÓN DE RESÚMENES DE MÚLTIPLES DOCUMENTOS

Shareghi E. [16] propone la búsqueda armónica (HS por sus siglas Harmony Search) para generación de resúmenes de un documento que se adaptó para el problema de múltiples documentos. Los pasos del algoritmo no se cambiaron para mantener la lógica del algoritmo de HS, la diferencia fundamental está centrada en los factores para la generación de un buen resumen, los cuales se plantearon en la sección 3.1. El proceso del algoritmo HS para encontrar el resumen ideal (ver Figura 5) consta de 5 pasos, los cuales son descritos a continuación:

- **Inicialización de parámetros** (*Paso 1*): En el capítulo 2 se describió el método habitual para encontrar el óptimo global para HS. Para el caso de generación de resúmenes la representación vectorial del resumen, se considera un vector de longitud *n* donde *n* es el número de oraciones candidatas y se hace una representación uno a uno entre el vector que contiene todas las oraciones y el vector resumen, cada posición del vector resumen está representada de forma binaria (0,1) (ver Figura 3. Representación vectorial del resumen) donde uno indica que la oración se toma en cuenta para el resumen y cero que la oración es descartada. Los parámetros de HS se especifican en este paso. Estos parámetros son el tamaño de la memoria armónica (HMS), la tasa de consideración de la memoria armónica (HMCR), la tasa de ajuste del tono (PAR), el ancho de banda de ajuste del tono (BW) y el número de improvisaciones (NI). Sin embargo, el PAR y el BW no se tienen en cuenta porque los valores de la memoria armónica son discretos.

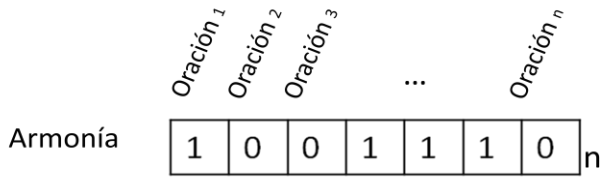

Figura 3. Representación vectorial del resumen

- **Inicialización de la memoria armónica** (*Paso 2*): Cada armonía de la memoria armónica (HM) es inicializada de forma aleatoria hasta completar la longitud del resumen deseado, generando de esta manera una cantidad de resúmenes igual al tamaño de la memoria armónica (HMS).

- **Improvisar una nueva armonía** (*Paso 3*): El proceso de generación de una nueva armonía es llamado *improvisación*. El nuevo vector armónico, $S' = (s'_1, s'_2, \dots, s'_n)$, donde *S'* es el vector resumen, s_i' es una oración candidata y *n* es la cantidad de oraciones candidatas, se genera utilizando las siguientes reglas: consideración de la memoria armónica (HMCR) y selección aleatoria.

  Primero se genera un número aleatorio entre 0 y 1, si este valor es menor a HMCR, se genera un nuevo aleatorio entre 0 y HMS para seleccionar el vector resumen, después se genera otro número aleatorio entre 0 y *n,* si la posición del vector seleccionado para la nueva improvisación es uno se toma de lo contrario es descartada. En caso contrario si el valor HMCR es menor al número aleatorio generado, la oración es tomada del conjunto de oraciones candidatas, siempre y cuando no haya sido tomada. Este proceso se repite iterativamente hasta que se llegue aproximadamente a la longitud del resumen deseada (por ejemplo, 200 o 400 palabras) y finalmente se calcula el *fitness* de la nueva armonía por medio de la ecuación (16).

- **Actualizar la memoria armónica** (*Paso 4*): Se compara el fitness del nuevo vector solución con el fitness de los vectores que se encuentran en la memoria armónica, si el valor para el nuevo vector es mejor que el peor de la memoria armónica, entonces se elimina el peor y se inserta el nuevo vector.

- **Repetir hasta que el criterio de parada es alcanzado** (*Paso 5*): Repetir los pasos 3 y 4 hasta que un número de iteraciones (NI) sea alcanzado.

Después de que el algoritmo ha encontrado el criterio de parada, solo basta con buscar la mejor armonía en la memoria armónica, que es aquella que tiene mejor fitness.

3.4 GENERACIÓN DE RESÚMENES DE MÚLTIPLES DOCUMENTOS APLICANDO MEJOR BÚSQUEDA ARMÓNICA GLOBAL

La única variación que tiene el algoritmo HS para aplicar la Mejor Búsqueda Armónica Global (GHS por sus siglas en ingles) para generación de resúmenes está en el *paso 3* (*improvisación de la nueva armónica*), adicional a esto se hace uso del parámetro PAR que no se tomo en cuenta para HS, por consiguiente solo describiremos este paso:

- **Improvisar una nueva armonía** (*Paso 3 GHS*): El proceso de generación de una nueva armonía es llamado *improvisación*. El nuevo vector armónico, $S' = (s'_1, s'_2, \dots, s'_n)$, donde *S'* es el vector resumen, s_i' es una oración candidata y *n* es la cantidad de oraciones candidatas, se genera utilizando las siguientes reglas: consideración de la memoria armónica (HMCR), tasa de ajuste del tono (PAR) y selección aleatoria, el proceso de nuevo improviso se muestra en la Figura 5.

Figura 4. Proceso de optimización para el algoritmo de Búsqueda Armónica para generación de resúmenes

1. Generar un número aleatorio entre 0 y 1 para HMCR
*1.1. **Si** el número aleatorio es menor o igual que HMCR **entonces***
1.1.1. Generar un número aleatorio entre 0 y 1 para el PAR.
*1.1.2. **Si** el valor del número aleatorio es menor que PAR **entonces***
Tomar el vector de la memoria armónica con mejor fitness. De lo contrario se genera un número aleatorio entre 0 y HMS para seleccionar el vector resumen.
fin_si
1.1.3. Después de seleccionar el vector, se genera un número aleatorio entre 0 y n para seleccionar la oración candidata para el resumen, si la posición del vector seleccionado para la nueva improvisación es uno se toma de lo contrario es descartada.
fin_si
*1.2. **Si** el número aleatorio generado para HMCR es mayor **entonces***
Generar un número aleatorio entre 0 y n tomando la oración del conjunto de oraciones candidatas, siempre y cuando no haya sido tomada.
fin_si

Este proceso se repite iterativamente hasta que se llegue aproximadamente a la longitud del resumen deseada (por ejemplo, 250 palabras) y finalmente se calcula el fitness de la nueva armonía por medio de la ecuación (16)

Figura 5. Generación de resúmenes con mejor búsqueda armónica global

3.5 GENERACIÓN DE RESÚMENES DE MÚLTIPLES DOCUMENTOS APLICANDO MEJOR BÚSQUEDA ARMÓNICA GLOBAL Y MODELOS EVOLUTIVOS QUE APRENDEN

Inspirados en el concepto de Modelo Evolutivo que Aprende (LEM, Learnable Evolution Model) propuesto por Michalsky [68], Cobos C. propone una nueva variación del algoritmo GHS, denominada GHS+LEM [44], en esta sección se describe el proceso realizado para aplicar el algoritmo a la tarea de generación de resúmenes de múltiples documentos. (Para ver en detalle la metodología de desarrollo del sistema GHS+LEM remitirse al Anexo B).

3.5.1 Modelo Evolutivo que Aprende para Generación de Resúmenes

El proceso de aprendizaje de máquina que se propone en [44] utiliza una variación del algoritmo PRISM propuesto por Cendrowska [25]. Debido a que la representación de las oraciones en la generación de resúmenes es binaria, no se puede aplicar el algoritmo con las mismas características, por consiguiente fue necesario hacer una adaptación teniendo en cuenta las necesidades particulares de este problema.

La aproximación que se utiliza del algoritmo PRISM en esta propuesta, tiene como objetivo imitar las decisiones simples que manejan los algoritmos de la familia armónica. Para tal fin se cuenta con reglas conjuntivas (P←R_1, R_2), que delimitan las regiones alrededor de las cuales hay una mayor posibilidad de encontrar un mejor valor para cada s_i. Por consiguiente para cada dimensión se limita el espacio de búsqueda a las regiones con más posibilidades de generar un óptimo global. El algoritmo de inferencia de reglas se ejecuta por primera vez inmediatamente después de creada la memoria armónica inicial, en el Anexo C se muestra un ejemplo del proceso de LEM para generación de resúmenes. Los pasos del algoritmo de inferencia de reglas, se resumen a continuación:

A. Se hace una copia de la memoria armónica actual la cual denominaremos matriz E, esta es ordena con respecto al fitness y se escoge la población de alto rendimiento y la población de bajo rendimiento de acuerdo a la siguiente fórmula.

$$Hgroup = P_{actual}(1, \dots, i), \quad (17)$$

$$Lgroup = P_{actual}(HMS - i, \dots, HMS), \quad (18)$$

Donde *i* = *HLGS*, HMS es el tamaño de la memoria armónica (HM) y HLGS es el tamaño de los grupos de alto y bajo rendimiento.

B. Teniendo los grupos de bajo y alto rendimiento, se calcula la frecuencia de ceros y unos, de la siguiente manera:

- Grupo de altos: Se calcula frecuencia de unos en altos (*F1A*) y frecuencia de ceros en altos (*F0A*).
- Grupo de bajos: Se calcula frecuencia de unos en bajos (*F1B*) y frecuencia de ceros en bajos (*F0B*).

Este cálculo se realiza para cada una de las dimensiones (oraciones) de la matriz E.

C. Se crea una lista de reglas R donde se almacena la probabilidad de ocurrencia de ceros y unos para cada dimensión de acuerdo a la siguiente fórmula:

$$\mathbf{P_{s_i}}(\mathbf{1}) = \begin{cases} 50\% + (\mathrm{F1A_{s_i}} - \mathrm{F1B_{s_i}})/\mathrm{TTG_{s_i}} & \mathrm{F1A_{s_i}} - \mathrm{F0A_{s_i}} \geq 0 \\ 50\% - (\mathrm{F0A_{s_i}} - \mathrm{F0B_{s_i}})/\ \mathrm{TTG_{s_i}} & \text{d. o. m.} \end{cases} \quad (19)$$

$$\mathbf{P_{s_i}}(\mathbf{0}) = 1 - P_{s_i}(1)$$

Donde s_i es una dimensión y TTG es tamaño total de grupos (Hg+Lg). El resultado final es una lista de reglas R del tipo $\mathrm{P} \rightarrow \mathrm{Q}$, donde P es un par de reglas (P0,P1) y Q corresponde a la clase seleccionada(0,1) grupo de alto rendimiento.

3.5.2 Algoritmo GHS+LEM Para Generación de Resúmenes

Los pasos (ver Figura 7) para la generación de resúmenes de múltiples documentos utilizando la mejor búsqueda armónica global y los modelos evolutivos que aprenden (GHS+LEM) se describen a continuación.

Inicializar los parámetros del problema y los parámetros de GHS *(Paso 1)**:*** Este paso es parte del algoritmo HS, pero se agregan tres parámetros que se explicarán más adelante, a saber: la tasa de actualización de las reglas (RRU), el tamaño de los grupos de alto y bajo rendimiento (HLGS) y la tasa de consideración de reglas (RCR).

Inicializar la memoria armónica *(Paso 2)**:*** Se ejecuta el proceso de inicialización propuesto en HS (*sesión 5.1*) sin ningún cambio.

Ejecutar el proceso de inferencia de reglas por primera vez *(Paso 3)**:*** Este paso pertenece a LEM, en el cual se ejecuta por primera vez el proceso de inferencia de reglas con base en la memoria armónica inicial y el tamaño de los grupos de alto y bajo rendimiento (HLGS). Este parámetro indica el tamaño de los grupos de alto desempeño y los grupos de bajo desempeño, este valor debe ser menor o igual que $\lfloor HMS/2 \rfloor$.

Improvisar la nueva armonía *(Paso 4)**:*** En este paso se introduce el uso de las reglas generadas en el paso anterior para la definición de los valores de la dimensión del nuevo improviso. Lo anterior se ejecuta basado en el parámetro denominado tasa de consideración de reglas (*RCR*). Este parámetro define el porcentaje de veces que se utilizan las reglas, de lo contrario se ejecuta el método tradicional de generación aleatoria basado en el espacio general de búsqueda (original en HS), el proceso de generación del nuevo improviso se describe en la Figura 6.

Actualizar la memoria armónica *(Paso 5):* Al igual que en HS la nueva armonía generada, $S' = (s'_1, s'_2, \dots, s'_N)$ reemplaza la peor armonía almacenada en la memoria armónica sólo si el fitness (o valor de aptitud de la nueva armonía, medido en términos de la función objetivo) es mejor que la peor armonía.

Verificar el criterio de actualización de reglas *(Paso 6):* Este paso hace parte de LEM y se realiza a través del parámetro RRU, el cual especifica en qué porcentaje de las veces se deben actualizar las reglas. Si un número aleatorio generado uniformemente entre 0 y 1 es menor que el valor de RRU, se ejecuta el proceso de inferencia de reglas nuevamente.

Verificar el criterio de parada *(Paso 7):* Al igual que en HS la ejecución del algoritmo termina cuando el número máximo de improvisaciones (NI) se alcanza, de lo contrario repetir los pasos 4,5 y 6.

1. ***Mientras*** *no se haya completado la longitud del resumen deseado,* ***hacer***
   1.1. ***si*** *U(0,1) es menor que HMCR* ***entonces***
      1.1.1. ***si*** *U(0,1) es menor o igual que PAR* ***entonces***
         Seleccionar el vector resumen que tiene el mejor fitness de la memoria armónica.
      fin_si
      1.1.2. ***si*** *U(0,1) es mayor que PAR* ***entonces***
         Seleccionar el vector resumen de forma aleatoria en el rango (0,…,HMS)
      fin_si
      1.1.3. *Cuando ya se ha seleccionado el vector resumen, se genera un número aleatorio entre 1 y n, para seleccionar la oración candidata para el resumen, si el valor de la posición del vector seleccionado para la nueva improvisación es uno se toma de lo contrario se descarta.*
   fin_si
   1.2. ***si*** *U(0,1) para HMCR es mayor ,* ***entonces***
      1.2.1. ***si*** *U(0,1) es menor que RCR* ***entonces***
         Se genera un número aleatorio entre 1 y HMS para encontrar un vector resumen y posteriormente de forma aleatoria se selecciona una oración del vector resumen.
         1.2.1.1. ***si*** *U (0,1) es menor que P(1)* ***entonces***
            Seleccionar la oración siempre y cuando no haya sido seleccionada, de lo contrario descartarla.
            fin_si
         fin_si
      1.2.2. ***si*** *el número aleatorio generado para RCR es mayor,* ***entonces***
         Se genera un número aleatorio entre 0 y n para seleccionar la oración del conjunto de oraciones candidatas, siempre y cuando no haya sido tomada
      fin_si
2. *Repetir el paso 1 hasta completar la longitud del resumen.*
   Fin_mientras

Figura 6. Pasos generación nuevo improviso con GHS+LEM

Figura 7. Proceso de optimización para el algoritmo GHS+LEM para generación de resúmenes

En la Figura 8 se muestra el diagrama general de los procesos del sistema de generación de resúmenes.

Figura 8. Diagrama general de procesos del sistema de generación automática de resúmenes

3.6 AFINACIÓN DE PARÁMETROS

Para encontrar la mejor combinación de parámetros para GHS+LEM, se realizó la afinación por medio del algoritmo GHS el cual denominamos GHSO (mejor búsqueda armónica global para optimización), los valores de los parámetros usados para este algoritmo fueron los más comúnmente usados [18].

La afinación consistió en tomar cada parámetro de HS y GHS+LEM como variables del vector armónico es decir $X_i = ((NI)_i, (HMS)_i, (HMCR)_i, \dots)$. Donde X_i es un vector armónico.

En las pruebas se tomaron de forma aleatoria 30 conjuntos de DUC2007, el cálculo del fitness de una armonía consistió en ejecutar cinco veces el algoritmo GHS+LEM con lo que se obtuvieron cinco medidas de ROUGE-2 por cada conjunto de documentos y se promediaron los resultados. A cada conjunto se le realizó el mismo proceso y al final se promediaron los resultados de los 30 conjuntos, el cual representa el fitness de la armonía actual, en la Figura 9 se muestra un ejemplo para dos conjuntos de documentos. (Para ver en detalle el proceso de afinación de parámetros remitirse al Anexo B)

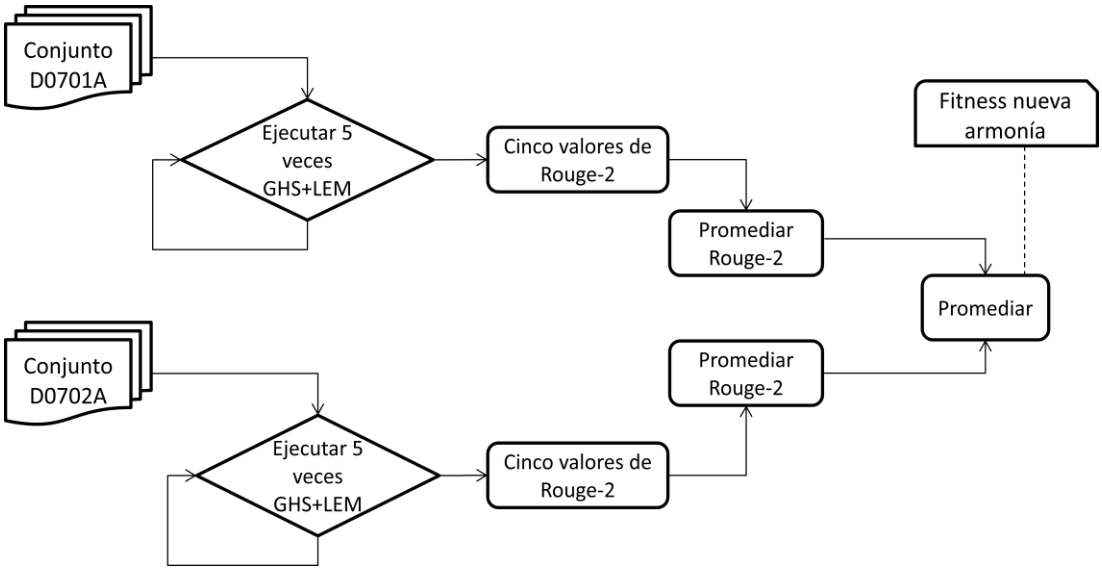

Figura 9: Ejemplo de Afinación de Parámetro

3.6.1 Afinación Parámetros HS adaptado a múltiple documentos

Los parámetros que se afinaron para este algoritmo fueron:

- Número de iteraciones (NI): Basados en [16] se partió de 1500 iteraciones hasta 10000, con incrementos de 500 iteraciones, es decir, {1500, 2000, …, 10000}.
- Tamaño de la memoria armónica (HM): Para este valor se partió de un tamaño de 100 con incrementos de 100 hasta llegar a 400.
- Tasa de consideración de la memoria armónica (HMCR): Partiendo de los resultados obtenidos en los trabajos más relevantes para ésta investigación [44, 69, 70], este parámetro puede tomar los valores de {0.9, 0.93, 0,95, 0.98}
- Coeficiente beta: Se varía de cero a uno. Esta variación se realiza de manera similar a la realizada en el trabajo de Sharegui. E. [16].

3.6.1.1 Resultados de la afinación

En la **Tabla 2** se muestra la mejor combinación de parámetros que se obtuvo como resultado del proceso de afinación. Esta combinación es utilizada para la generación de resúmenes mediante el algoritmo HS.

NI	HMS	HMCR	Beta
6000	300	0.98	0.6

Tabla 2. Mejores parámetros obtenidos para HS

Como se puede observar en la **Tabla 3**, las combinaciones iniciales para la afinación de parámetros de HS son diferentes para cada improvisación de la memoria armónica, pero después de 400 iteraciones del algoritmo GHSO (como se observa en la **Tabla 4**), el proceso de optimización converge a un número de iteraciones de 6000, tamaño de memoria armónica de 300, HMCR de 0.98, y Beta de 0.6. Por consiguiente se puede

concluir que se han encontrado los posibles mejores parámetros para la ejecución del algoritmo HS.

NI	HMS	HMCR	Beta	Rouge-2
8000	200	0,96	0,2	0,1006095
6000	400	0,93	0,6	0,10268725
4000	300	0,96	0,3	0,0921564
6000	200	0,96	0,6	0,10180275
6000	300	0,96	0,2	0,099348
8000	100	0,98	0,6	0,0989435
6000	300	0,98	0,6	0,10068725
4000	200	0,96	0,6	0,09986075
6000	200	0,96	0,6	0,10180275
5000	200	0,98	0,3	0,10012375

Tabla 3. Memoria armónica Inicial para HS

NI	HMS	HMCR	Beta	Rouge-2
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687
6000	300	0,98	0,6	0,104687

Tabla 4. Memoria armónica Final para HS

3.6.2 Afinación Parámetros GHS+LEM

Los parámetros afinados para GHS+LEM fueron:

- Número de iteraciones (NI): basados en [69] se partió de 1500 iteraciones hasta 10000 con incrementos de 500 iteraciones es decir {1500, 2000, …, 10000}.
- Tamaño de la memoria armónica (HM): Para este valor se partió de un tamaño de 100 con incrementos de 100 hasta llegar a 400.
- Tasa de consideración de la memoria armónica (HMCR): Partiendo de [44, 69, 70], este parámetro podía tomar los valores de {0.9, 0.93, 0,95, 0.98}
- Tasa de ajuste del tono (PAR): Partiendo de [18, 44] los valores que podía tomar esta parámetro fueron {0.1, 0.3, 0.5, 0.7, 0.9}
- Tasa de consideración de reglas (RCR): basados en [44] este parámetro podía variar de la siguiente manera { 0.5, 0.6, 0.7, 0.8, 0.9}
- Tasa de actualización de reglas (RRU): partiendo de [44] este parámetro podía tomar los siguientes valores {0.1, 0.2, 0.3, 0.4, 0.5}.
- Tamaño de los grupos de alto y bajo rendimiento: este es un nuevo parámetro el cual se decidió afinar considerando que es importante saber cuál es el tamaño más adecuado de los grupos de alto y bajo rendimiento, para afinar este valor se decidió variar la distancia que puede existir entre los dos grupos de 0% y 33%, el criterio que se escogió para decidir que el porcentaje máximo puede ser 33% es porque es la tercera parte de la memoria armónica y un valor más grande dejaría pocos datos para realizar un buen análisis para la creación de las reglas.
- El coeficiente Beta se varió de cero a uno igual que en [69].

3.6.2.1 Resultados de la afinación

En la **Tabla 5** se muestra la mejor combinación de parámetros que se obtuvo como resultado del proceso de afinación. Esta combinación es utilizada para la generación de resúmenes mediante el algoritmo GHS+LEM.

NI	HMS	HMCR	PAR	RCR	RRU	%HLGS	Beta
6000	300	0,98	0,7	0,7	0,3	47	0,3

Tabla 5. Mejores parámetros obtenidos en la afinación de GHS+LEM

Como se puede observar en la **Tabla 6** las combinaciones iniciales para la afinación de parámetros de GHS+LEM, son diferentes para cada improvisación de la memoria armónica, después de 600 iteraciones del algoritmo GHSO se llega a una memoria armónica como la que se muestra en la **Tabla 7**, en la cual no todas las improvisaciones son iguales, por lo tanto para encontrar una posible combinación de parámetros para GHS+LEM se toma el mejor valor de Rouge-2.

NI	HMS	HMCR	PAR	RCR	RRU	%HLGS	Beta	Rouge-2
4000	100	0,98	0,7	0,8	0,2	49,5	0,5	0,1148250
8000	100	0,98	0,3	0,7	0,2	36,5	0,7	0,1143578
6000	400	0,98	0,7	0,7	0,3	47	0,2	0,1159273
3000	100	0,9	0,7	0,8	0,3	34	0,1	0,1058535
5000	100	0,98	0,5	0,7	0,2	42,5	0,1	0,1093860
5000	200	0,93	0,7	0,8	0,3	50	0,4	0,1079890
4000	100	0,93	0,7	0,7	0,4	47	0,7	0,1063280
3000	300	0,9	0,7	0,8	0,2	49,5	0,7	0,1017650
8000	100	0,98	0,5	0,7	0,4	46,5	0,9	0,1034583
4000	200	0,93	0,5	0,7	0,4	48	0,9	0,1002400

Tabla 6 Memoria armónica inicial para GHS+LEM

NI	HMS	HMCR	PAR	RCR	RRU	%HLGS	Beta	Rouge-2
6000	300	0,98	0,7	0,7	0,3	47	0,3	0,121968
5000	300	0,98	0,6	0,8	0,3	43	0,3	0,120232
5000	300	0,98	0,5	0,7	0,3	47	0,3	0,120232
5000	300	0,98	0,5	0,7	0,3	47	0,3	0,120232
5000	300	0,98	0,7	0,8	0,3	47	0,3	0,118968
6000	300	0,98	0,5	0,8	0,2	47	0,3	0,117802
5000	300	0,98	0,7	0,7	0,3	49,5	0,3	0,117793
5000	400	0,98	0,5	0,8	0,2	47	0,3	0,117564
6000	100	0,98	0,5	0,7	0,3	47	0,3	0,116789
5000	300	0,98	0,5	0,7	0,2	45	0,3	0,116384

Tabla 7 Memoria armónica final para GHS+LEM

Capítulo 4

4 EVALUACIÓN

4.1 PRE-PROCESAMIENTO DE DOCUMENTOS

Antes de la ejecución del algoritmo propuesto para la generación del resumen, se realizo la etapa de pre-procesamiento, en la cual, se procesa el texto para obtener unidades más pequeñas (frases, oraciones), clasificarlo, etiquetarlo y para filtrar las palabras u oraciones que podrían constituir ruido en la etapa de selección de las oraciones más representativas del resumen, ésta etapa de pre-procesamiento normalmente genera cierto tipo de dependencia del lenguaje, por ejemplo las palabras vacías deben ser en el idioma específico a resumir. Las etapas de pre-procesamiento usadas en este trabajo son:

4.1.1 Segmentación

El texto original debe ser dividido en unidades más pequeñas con el objetivo de realizar la extracción de las oraciones que se van a presentar en el resumen. Generalmente es necesario dividir el texto en palabras o tokens para poder identificar correctamente los límites entre las unidades de texto, frases u oraciones. Esta no es una tarea trivial debido a las irregularidades encontradas en los diferentes lenguajes naturales.

Una de las técnicas para subdividir textos en unidades más pequeñas o sub-tópicos se denomina Texttiling [71]. Este hace uso de patrones de co-ocurrencia léxica y distribución. El algoritmo tiene tres partes principales: pre-procesamiento, cálculo de puntuaciones léxicas e identificación de los límites. En la primera parte se eliminan las palabras vacías, se realiza un análisis morfológico del texto y los documentos se dividen en secuencias de palabras significativas (oraciones), sin considerar signos de puntuación. Después se determina una puntuación léxica para los espacios entre grupos de oraciones y finalmente se realiza una identificación de límites.

Para la segmentación de oraciones en el presente trabajo se hace uso del segmentador diseñado originalmente para AnswerBus Question Answering System y que ahora es utilizado por Seven Tones Search Engine y por muchas otras aplicaciones de PLN. Fue seleccionado debido a que ha sido utilizado previamente por el grupo de investigación GTI (Grupo de Tecnologías de la Información) y se encuentra disponible en http://www.answerbus.com/sentence/.

4.1.2 Filtro de palabras vacías

Las palabras vacías o stopwords son aquellas palabras que son muy frecuentes en un documento pero no contribuyen particularmente al contenido del mismo, por ejemplo, palabras como “the”, “is”, “and”, entre otras; de manera individual no son buenos discriminantes cuando se quiere determinar la relevancia de una unidad de texto en un documento y en la mayoría de los casos constituyen ruido. De hecho, una palabra que ocurre en el 80% de los documentos en la colección es poco útil para propósitos de recuperación de la información. Una lista de palabras vacías puede ser extendida para

incluir artículos, preposiciones y conjunciones y además algunos verbos, adverbios y adjetivos pueden ser tratados como palabras vacías.

La eliminación de palabras vacías reduce considerablemente el tamaño del texto de origen. Es muy común obtener una compresión en el tamaño del texto origen del 40% o más sólo con la eliminación de palabras vacías. Ésta es una técnica muy común y necesaria en el procesamiento del lenguaje natural. Para remover éstas palabras, en este proyecto se tomó una lista de palabras vacías[16] que ha sido usada en el grupo de investigación GTI y también en [9].

4.1.3 Stemming

Es una técnica de reducción que permite detectar variantes morfológicas de un mismo término y reemplazarlas por el término raíz o lema. En un texto, la misma palabra usualmente ocurre en muchas variantes morfológicas. Esas formas variantes son gobernadas por el contexto, es decir, si esta se presenta en forma plural o singular, tiempo presente o pasado, etc. En muchos casos, esas diferencias de formas léxicas tienen interpretaciones semánticas diferentes y pueden a menudo ser consideradas como equivalentes para el propósito de procesar mucha información. Para que un sistema de gestión de la información sea capaz de tratar esas formas variantes como un *stem* o *lema*, es común usar un algoritmo de stemming o stemmer, que es un procedimiento computacional que reduce todas las palabras con una misma raíz a una forma común, es decir, si por ejemplo en el texto original se encuentran palabras como *computational* y *computing,* ambas palabras se pueden representar como *comput.*

El efecto no es solo que diferentes variantes de un término puedan ser llevadas a una forma simple de representación, sino que también reduce el tamaño del vocabulario necesario para la representación del documento o del conjunto de documentos. En muchos casos, la considerable reducción del tamaño del diccionario es útil porque reduce el espacio de almacenamiento, el tiempo de procesamiento, reduce el tamaño de la estructura de indización, así como también hace que la representación del documento sea menos ruidosa y más versátil. 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. Este problema puede ser cubierto con la aplicación del stemming.

Las reglas que forman los algoritmos clásicos de stemming dependen del idioma de las colecciones de los documentos a procesar. El algoritmo clásico es Porter Stemmer[17] cuya versión original está para el idioma inglés [72].

4.1.4 Eliminación de oraciones que tienen similitud menor a un umbral

Cuando se calcula la similitud entre una oración y la colección de documentos, aplicando la ley de cosenos, este valor se encuentra ente 0 y 1, donde cero nos indica que no tienen términos en común y uno que tienen todos los términos en común. Dado este concepto y lo planteado por Radev D. [73] donde establecen un umbral para definir si un documento

[16] ftp://ftp.cs.cornell.edu/pub/smart/english.stop
[17] http://tartarus.org/martin/PorterStemmer/

es incluido en un cluster, en la presente investigación se decidió hacer una prueba (anexo D) donde se eliminaba las oraciones que tenían una similitud con el documento menor a un umbral, según las pruebas realizadas se pudo concluir que se presentaban mejores resultados cuando se eliminaba las oraciones que tienen una similitud menor a 0.13.

4.1.5 Lucene

Las anteriores etapas mencionadas para el pre-procesamiento son un proceso fundamental en tareas de procesamiento del lenguaje natural, pues permiten que se haga un tratamiento más preciso del texto. Existen herramientas o bibliotecas de código que realizan algunas de las etapas del pre-procesamiento, muchas de ellas son de Código Abierto, por lo tanto su código fuente está a disposición de la comunidad y puede ser reutilizado en cualquier aplicación. Es importante aclarar que las bibliotecas de funciones no son aplicaciones que se descargan, ejecutan e instalan sino que son APIs (Aplication Programming Interface) a través de las cuales se añaden, con esfuerzos de programación, capacidades de indización y búsqueda a cualquier sistema que se esté desarrollando.

Lucene es una tecnología para la Recuperación de Información que realiza procesos de indización y búsqueda, es creada bajo una metodología orientada a objetos y cuenta con una API implementada en Java, también está disponible en otros lenguajes de programación, soporta la indización de documentos con formatos: txt, pdf, doc, ppt, rtf, xml y html. Existen otras herramientas que permiten realizar la indización y búsqueda de documentos pero se utilizan para usos concretos, lo que implica que el intentar adaptarlas a un proyecto específico es una tarea realmente difícil. La idea que engloba Lucene es completamente diferente, ya que su principal ventaja es su flexibilidad, permite su utilización en cualquier sistema que lleve a cabo procesos de indización o búsqueda, tiene versiones para otros lenguajes como Perl, C#, Ruby y C++. Lucene está disponible bajo la licencia Apache Software Licence en [74]. En general, Lucene posee mejores características en comparación con otras librerías y es la única que utiliza como modelo de representación de los documentos el modelo de espacio vectorial, el cual se considera el más apropiado para el desarrollo del algoritmo GHS+LEM. Por éstas razones y basados en previos estudios realizados por el grupo de investigación GTI (Grupo de Tecnologías de la Información) acerca de las diferentes herramientas para pre-procesamiento se decidió usar Lucene.

El resumen final está conformado por las oraciones originales, es decir, las obtenidas a partir del proceso de segmentación. Antes de aplicar las etapas de filtro de palabras vacías y stemming se hace una copia de las oraciones originales para que éstas queden en el resumen final.

4.2 CORPUS DE EVALUACIÓN

Para realizar la evaluación del algoritmo propuesto, se utilizaron los conjuntos de datos de DUC 2005 y DUC 2007, los cuales pertenecen al dominio periodístico obtenido de TREC[18] (The Text Retrieval Conference) y de AQUAINT[19]; y proponen tareas orientadas a resúmenes de múltiples documentos. Además estos conjuntos de datos también fueron

[18] http://trec.nist.gov/overview.html

[19] http://www-nlpir.nist.gov/projects/aquaint/

utilizados por MCMR [9], que es un método de referencia para esta investigación dado que hace uso de un algoritmo evolutivo y su función objetivo tiene en cuenta los mismos factores (cobertura y redundancia) para la generación del resumen.

DUC proporciona los conjuntos agrupados por temas además de varios resúmenes ideales para un mismo conjunto de documentos, esto es, porque no existe un "único" resumen para un documento o un conjunto de documentos, por lo que proporcionar varios resúmenes ideales permite que el resumen generado automáticamente tenga mayor o menor similitud con alguno de ellos. Todos los documentos fueron segmentados en oraciones, se eliminaron aquellas que tenían similitud con el documento menor a 0.13 y a cada una de ellas se le aplica stemming y eliminación de palabras vacías (stopwords) [75] por medio de la librería lucene.net [74].

DUC 2005 cuenta con 50 conjuntos de datos y cada conjunto contiene de 25 a 50 documentos, DUC 2007 cuenta con 45 conjuntos y cada conjunto con 25 documentos. La tarea es crear un resumen de no más de 250 palabras para DUC 2005 y 2007. En la **Tabla 8** se muestra una breve descripción de los datos.

	DUC 2005	**DUC 2007**
Número de grupos	50	45
Número de documentos	1593	1125
Fuente de datos	TREC y TIPSTER	AQUAINT
Longitud del resumen	250 palabras	250 palabras

Tabla 8. Conjuntos de datos

4.3 MÉTRICAS DE EVALUACIÓN

Para la evaluación de la calidad de los resúmenes generados por el algoritmo propuesto se utilizó la herramienta ROUGE (ver Capítulo 2) en su versión 1.5.5 [47], que ha sido adoptada desde sus orígenes (Lin, 2004) como métrica oficial para evaluar la calidad de los resúmenes generados por los métodos propuestos en las conferencias de DUC. Para este trabajo se utilizan las medidas de ROUGE-1, ROUGE-2 y ROUGE-SU4 por ser las más usadas en las investigaciones de generación automática de resúmenes de múltiples documentos [9, 76-78], especialmente por MCMR.

4.4 RESULTADOS Y ANÁLISIS

En la evaluación se ejecutó cada algoritmo treinta veces por cada grupo[20] de documentos, para esas treinta ejecuciones se encontró un promedio por grupo y posteriormente se promediaron todos esos resultados. Para el caso de los documentos de DUC 2005 el promedio se realizó teniendo en cuenta los 50 grupos de documentos y para 2007 los 45 grupos. Los parámetros usados para cada algoritmo fueron los mencionados en la sección 0.

[20] Grupos de documentos que están contenidos dentro del conjunto de DUC 2005 o DUC 2007.

4.4.1 Evaluación con DUC 2005

En la **Tabla 9** se presentan los resultados de las pruebas de evaluación de los 50 conjuntos de documentos de DUC 2005 realizadas con el algoritmo GHS+LEM con el objeto de medir la calidad de los resúmenes con respecto a HS, las pruebas indican que GHS+LEM supera a HS en la media de Rouge-1 en un 5.45%, en Rouge-2 en un 7.49% y en la medida de Rouge-Su4 en un 6.01%. Adicional a esto también se realiza la evaluación con GHS y como se puede observar, los resultados son superiores a los de HS pero no mayores a GHS+LEM. Esto muestra que la propuesta se comporta mejor que GHS y HS en generación automática de resúmenes de múltiples documentos.

Algoritmo	**ROUGE-1**	**ROUGE-2**	**ROUGE-SU4**
GHS+LEM	**0,41626504**	**0,0785762**	**0,1426726**
GHS	0,40689604	0,07587084	0,1386914
HS	0,39356992	0,07269092	0,13409324

Tabla 9. Resultados ROUGE DUC 2005

En la **Tabla 10, Tabla 11, y Tabla 12** se muestra el comportamiento de cada uno de los conjuntos de DUC 2005 con respecto a las medidas de ROUGE-1, ROUGE-2 y ROUGE-SU4 respectivamente. Como se puede observar los tres algoritmos presentan buenos resultados pero en promedio GHS+LEM presenta mejores resultados que los otros dos algoritmos.

Conjunto	**HS**	**GHS**	**GHS+LEM**		**Conjunto**	**HS**	**GHS**	**GHS+LEM**
d301i	0,328388	0,34916	**0,358656**		d398e	0,42338	0,432024	**0,460708**
d307b	0,368008	**0,39291**	0,37797		d400b	0,369934	**0,401506**	0,394236
d311i	0,454934	**0,479184**	0,476404		d401e	0,385228	0,39042	0,411376
d313e	0,441676	**0,464208**	**0,464208**		d404g	0,402594	0,427718	**0,428514**
d321f	**0,377974**	0,115834	0,291838		d407b	0,413492	0,440838	**0,447006**
d324e	0,400088	**0,44485**	0,435612		d408c	0,259092	**0,293478**	0,280042
d331f	0,439444	**0,46275**	0,458166		d413a	0,461096	0,452528	**0,464354**
d332h	**0,34606**	0,336562	0,33838		d422c	0,409202	**0,435282**	0,434746
d343c	0,356	**0,4184**	0,416		d426a	0,31476	0,37111	**0,3721**
d345j	0,382036	0,405842	**0,419558**		d428e	0,342454	0,356942	**0,362576**
d346h	0,345956	0,384416	**0,40138**		d431h	0,371586	0,377086	**0,377088**
d347b	0,402204	0,42963	**0,430832**		d434b	0,395474	**0,421436**	0,413372
d350a	0,4156	0,4678	**0,4714**		d435f	0,412614	0,437148	**0,447516**
d354c	0,368518	0,391202	**0,395136**		d436j	0,284076	**0,342434**	0,313948

d357i	0,48597	0,489352	**0,502686**		d438g	0,44361	0,486838	**0,490298**
d360f	0,44758	0,457894	**0,466526**		d442g	0,332742	0,34359	**0,348518**
d366i	0,393924	**0,395192**	0,380044		d446j	0,361062	0,369914	**0,37404**
d370i	0,425	**0,4608**	0,4552		d632i	0,46797	0,452606	**0,478994**
d374a	0,384194	0,38075	**0,39919**		d633g	0,498404	0,511456	**0,527698**
d376e	0,455784	0,457296	**0,482206**		d654f	0,27455	0,202816	**0,33943**
d383j	0,39362	0,405966	**0,408848**		d671g	0,434284	0,46737	**0,48511**
d385g	0,509948	**0,532358**	0,5267		d683j	0,437262	**0,461538**	0,45567
d389h	0,39211	0,385076	**0,39339**		d694j	0,338268	0,360272	**0,37741**
d391h	0,366906	0,384972	**0,393738**		d695c	0,435748	**0,460826**	0,459938
d393f	0,370156	**0,38438**	0,376354		d699a	0,357536	**0,370842**	0,348142

Tabla 10. Resultados DUC 2005 para Rouge-1

Conjunto	**HS**	**GHS**	**GHS+LEM**		**Conjunto**	**HS**	**GHS**	**GHS+LEM**
d301i	0,060178	0,05561	**0,059582**		d398e	0,095068	0,092112	**0,100394**
d307b	0,065384	**0,067694**	0,062308		d400b	0,058592	**0,060194**	**0,060194**
d311i	0,082722	0,086302	**0,086388**		d401e	0,062326	0,064728	**0,070942**
d313e	0,098098	**0,106906**	0,101102		d404g	0,093294	0,096096	**0,103304**
d321f	**0,066928**	0,01974	0,051924		d407b	0,062986	0,070076	**0,075818**
d324e	0,088096	0,110476	**0,11333**		d408c	0,042858	0,053374	**0,055556**
d331f	0,1062	**0,1166**	0,1144		d413a	0,09832	0,101936	**0,10727**
d332h	**0,053346**	0,03996	0,046044		d422c	0,078386	**0,08278**	0,077486
d343c	0,084738	0,09277	**0,105624**		d426a	0,04591	**0,061004**	0,057842
d345j	0,068504	**0,080854**	0,079608		d428e	0,04424	0,046668	**0,051516**
d346h	0,03861	0,044948	**0,048712**		d431h	0,042482	0,039584	**0,0423**
d347b	0,066734	0,067938	**0,068542**		d434b	0,074628	**0,086082**	0,082526
d350a	0,054414	0,057028	**0,063454**		d435f	0,089852	**0,092714**	0,092282
d354c	0,033256	0,03535	**0,037904**		d436j	0,017674	**0,03416**	0,023834
d357i	0,127274	0,125674	**0,131668**		d438g	0,11321	**0,135582**	0,133076
d360f	0,074628	**0,076954**	0,074842		d442g	0,047128	**0,056238**	0,054854
d366i	**0,070402**	0,065666	0,066938		d446j	0,05923	**0,060414**	0,058834
d370i	0,094178	**0,10984**	0,099798		d632i	0,11695	0,107104	**0,118694**
d374a	0,101932	0,096032	**0,109866**		d633g	0,130882	0,137294	**0,148514**
d376e	0,132738	0,127734	**0,139616**		d654f	0,032656	0,019152	**0,056752**
d383j	0,052482	**0,062396**	**0,062396**		d671g	0,082696	0,097552	**0,10503**
d385g	**0,092536**	0,081596	0,085806		d683j	0,097858	**0,10866**	0,10214
d389h	**0,077732**	0,073018	0,069806		d694j	0,034018	0,036754	**0,036952**

d391h	0,037898	**0,053526**	0,051008		d695c	0,074822	**0,083214**	0,078214
d393f	0,05957	**0,060446**	0,058434		d699a	0,049902	**0,055012**	0,045386

Tabla 11. Resultados DUC 2005 para Rouge-2

Conjunto	HS	GHS	GHS+LEM		Conjunto	HS	GHS	GHS+LEM
d301i	0,11173	0,115196	**0,118492**		d398e	0,146128	0,15063	**0,162278**
d307b	0,134066	**0,13774**	0,133806		d400b	0,12174	**0,131584**	0,129106
d311i	0,149524	0,156098	**0,156274**		d401e	0,127772	0,129758	**0,13786**
d313e	0,15576	**0,167016**	0,160298		d404g	0,143066	0,148572	**0,156904**
d321f	**0,12663**	0,03597	0,098044		d407b	0,131394	0,14127	**0,148238**
d324e	0,136594	**0,162042**	0,160682		d408c	0,083886	**0,096606**	0,091014
d331f	0,164294	**0,176342**	0,16933		d413a	0,1657	0,165728	**0,17166**
d332h	**0,10868**	0,097106	0,100238		d422c	0,14757	**0,158404**	0,15332
d343c	0,133152	0,157852	**0,158422**		d426a	0,098878	0,120854	**0,121398**
d345j	0,12427	0,132514	**0,139446**		d428e	0,105084	0,10885	**0,113796**
d346h	0,106248	0,1208	**0,12402**		d431h	0,119534	0,11824	**0,121006**
d347b	0,130556	0,13811	**0,138986**		d434b	0,133226	**0,147036**	0,13935
d350a	0,127526	0,146158	**0,150236**		d435f	0,14535	0,154722	**0,158082**
d354c	0,110624	0,118672	**0,120354**		d436j	0,074776	**0,0971**	0,083804
d357i	**0,206908**	0,201208	0,20667		d438g	0,165772	**0,190942**	0,188904
d360f	0,149502	0,153408	**0,155608**		d442g	0,1	0,10548	**0,10638**
d366i	**0,137674**	0,131912	0,126886		d446j	0,112654	0,116496	**0,118516**
d370i	0,16166	**0,18029**	0,17207		d632i	0,181944	0,174418	**0,185804**
d374a	0,14476	0,140388	**0,1535**		d633g	0,19242	0,198622	**0,207676**
d376e	0,185252	0,177938	**0,194828**		d654f	0,079046	0,055798	**0,112884**
d383j	0,114804	0,121464	**0,123304**		d671g	0,155034	0,167828	**0,17878**
d385g	0,172254	**0,172644**	0,17148		d683j	0,157244	**0,17299**	0,167492
d389h	**0,130156**	0,121854	0,128432		d694j	0,099574	0,10482	**0,108364**
d391h	0,119332	**0,128888**	0,128328		d695c	0,144374	0,151176	**0,15332**
d393f	0,11815	0,118884	**0,119356**		d699a	0,11239	**0,116152**	0,108604

Tabla 12. Resultados DUC 2005 para Rouge-Su4

Adicional a la evaluación realizada con los parámetros obtenidos en la afinación, se realizó una prueba para GHS+LEM con 15000 iteraciones con el propósito de realizar la comparación con el sistema MCMR dado que en éste también hace uso de un algoritmo evolutivo cuyo número de iteraciones es similar. Como se aprecia en la **Tabla 13** GHS+LEM supera a MCMR con PSO, en la medida de ROUGE-2 en un 1.95% y en la de Rouge-SU4 en un 4.76%. Esto indica que el método propuesto tiene un buen desempeño en generación de resúmenes de múltiples documentos, sin embargo, a pesar de que la cantidad de iteraciones se incrementa en más del doble con respecto a los parámetros de

la afinación (de 6000 a 15000) la diferencia entre los resultados obtenidos para Rouge-1 y para Rouge-SU4 es muy pequeña pues GHS+LEM(15000) supera a GHS+LEM(6000) en Rouge-SU4 en un 0.07% lo que demuestra que los parámetros de la afinación son suficientes para tener buenos resultados.

Algoritmo	ROUGE-2	ROUGE-SU4
GHS+LEM	**0,0769**	**0,1428**
MCMR(PSO)	0,0754	0,1360

Tabla 13. Resultados ROUGE DUC 2005, 15000 iteraciones

4.4.2 Evaluación con DUC 2007

En la **Tabla 14** se presenta los resultados de las pruebas de evaluación de los 45 conjuntos de documentos de DUC 2007 aplicadas al algoritmo GHS+LEM con el objeto de medir la calidad de los resúmenes con respecto a HS, las pruebas aplicadas indican que GHS+LEM supera a HS en la media de Rouge-1 en un 4.34%, en Rouge-2 en un 5.11% y en la medida de Rouge-Su4 en un 4.60%. Adicional a esto también se realiza la evaluación con GHS y como se puede observar los resultados son superiores a los de HS pero no mayores a GHS+LEM. Esto muestra que la propuesta se comporta mejor que GHS y HS en generación automática de resúmenes de múltiples documentos

Algoritmo	ROUGE-1	ROUGE-2	ROUGE-SU4
GHS+LEM	**0,46073147**	**0,11698356**	**0,17485933**
GHS	0,45782533	0,11661609	0,17313231
HS	0,44074787	0,11099471	0,16680596

Tabla 14. Resultados ROUGE DUC 2007

En la **Tabla 15**, **Tabla 16** y **Tabla 17** se muestra el comportamiento de cada uno de los conjuntos de DUC 2007 con respecto a la medida de ROUGE-1. Como se puede observar los tres algoritmos presentan buenos resultados pero en promedio GHS+LEM presenta mejores resultados que los otros dos algoritmos.

ROUGE-1								
Conjunto	**HS**	**GHS**	**GHS+LEM**		**Conjunto**	**HS**	**GHS**	**GHS+LEM**
D0701A	0,429716	0,438954	**0,452814**		D0724F	0,411198	0,419058	**0,421808**
D0702A	0,443774	0,465664	**0,470684**		D0725F	0,457422	0,484	**0,494086**
D0703A	**0,468062**	0,454528	0,452938		D0726F	0,451302	0,45952	**0,466534**

D0704A	0,44252	**0,462598**	0,448622		D0727G	0,452352	**0,476278**	0,462986
D0705A	0,450546	**0,503484**	0,497314		D0728G	0,44363	**0,46328**	0,46004
D0706B	0,409514	0,433592	0,437086		D0729G	0,476892	0,47225	**0,492432**
D0707B	0,469892	**0,501284**	0,490622		D0730G	0,536058	0,543616	**0,55097**
D0708B	0,362298	0,39879	**0,402622**		D0731G	0,474418	**0,4908**	0,483314
D0709B	0,25352	0,25352	0,25352		D0732H	0,435948	**0,461768**	0,460776
D0710C	0,441898	0,474308	0,480436		D0733H	**0,459368**	0,450886	**0,459368**
D0711C	0,49421	**0,5**	0,496208		D0734H	0,418998	0,43382	**0,444052**
D0712C	0,461676	**0,523742**	0,513692		D0735H	0,475272	**0,47726**	0,456404
D0713C	0,496142	0,510582	**0,522654**		D0736H	0,352028	**0,359474**	0,352904
D0714D	0,504016	**0,507344**	0,496768		D0737I	0,360888	0,376212	**0,420764**
D0715D	0,388758	**0,436884**	0,4144		D0738I	0,439574	0,46615	**0,485744**
D0716D	**0,504706**	0,50255	0,499018		D0739I	0,441126	0,449662	**0,45393**
D0717D	0,444254	**0,466668**	0,45356		D0740I	0,422308	0,442466	**0,449314**
D0718D	0,42099	0,431888	**0,44541**		D0741I	0,441312	0,462162	**0,472008**
D0719E	0,416248	0,44896	0,47271		D0742J	**0,495602**	0,486806	0,48566
D0720E	0,483384	0,489154	0,494926		D0743J	0,396016	0,399052	**0,406446**
D0721E	0,473542	**0,51401**	0,513228		D0744J	0,476682	**0,507906**	0,483992
D0722E	0,4706	0,477754	**0,508318**		D0745J	0,407808	0,434666	**0,444952**
D0723F	0,377186	0,38879	**0,406882**					

Tabla 15. Resultados DUC 2007 para Rouge-1

ROUGE-2								
Conjunto	**HS**	**GHS**	**GHS+LEM**		**Conjunto**	**HS**	**GHS**	**GHS+LEM**
D0701A	0,114112	0,116532	**0,124396**		D0724F	0,0572	0,061144	**0,062722**
D0702A	0,079436	**0,08871**	0,085886		D0725F	0,110612	0,118796	**0,12191**
D0703A	**0,142458**	0,12767	0,13047		D0726F	**0,118512**	0,100602	0,105432
D0704A	**0,099802**	0,096838	0,097234		D0727G	0,117864	0,123818	**0,137372**
D0705A	0,125276	**0,150648**	0,144258		D0728G	0,08308	0,095662	**0,098266**
D0706B	0,093956	0,09961	0,099612		D0729G	0,121174	0,1309	**0,137384**
D0707B	0,120318	**0,144104**	0,131618		D0730G	**0,162872**	0,159384	0,161844
D0708B	0,073886	**0,077528**	0,0751		D0731G	0,142742	**0,157562**	0,147612
D0709B	**0,02503**	**0,02503**	**0,02503**		D0732H	0,110668	0,118046	**0,119842**
D0710C	0,124604	0,13512	**0,1498**		D0733H	0,10792	0,113068	**0,116038**
D0711C	0,103806	**0,107614**	0,107412		D0734H	0,079034	0,089726	**0,10105**
D0712C	0,135904	**0,179622**	0,159448		D0735H	**0,127418**	0,11984	0,110666
D0713C	0,169016	**0,175772**	0,173982		D0736H	0,08229	**0,08471**	0,07987
D0714D	0,140216	**0,14415**	0,139822		D0737I	0,062148	0,068422	**0,088058**
D0715D	0,09307	**0,112078**	0,10792		D0738I	0,102046	**0,10964**	0,109248

D0716D	**0,154528**	0,147638	0,14193		D0739I	0,103798	0,115288	**0,124246**
D0717D	0,139754	**0,15062**	0,142804		D0740I	0,09705	0,098036	**0,104126**
D0718D	0,109826	0,10233	**0,113476**		D0741I	0,117832	0,115118	**0,121898**
D0719E	0,078452	0,083684	**0,084726**		D0742J	**0,17044**	0,16622	0,167566
D0720E	0,16164	**0,162838**	0,14925		D0743J	0,074214	**0,074406**	0,067934
D0721E	0,133398	**0,140234**	0,13926		D0744J	0,136706	**0,148412**	0,131746
D0722E	0,117088	0,125824	**0,129906**		D0745J	0,102294	0,111088	**0,113956**
D0723F	0,071272	0,073642	**0,082134**					

Tabla 16. Resultados DUC 2007 para Rouge-2

ROUGE-Su4								
Conjunto	**HS**	**GHS**	**GHS+LEM**		**Conjunto**	**HS**	**GHS**	**GHS+LEM**
D0701A	0,156664	0,160286	**0,168572**		D0724F	0,127298	0,132394	**0,133586**
D0702A	0,155174	**0,16235**	0,161536		D0725F	0,178242	0,189186	**0,193596**
D0703A	**0,18696**	0,180084	0,178344		D0726F	**0,17505**	0,164516	0,173836
D0704A	0,158554	**0,161638**	0,155736		D0727G	0,175394	0,184218	**0,188146**
D0705A	0,173584	**0,203888**	0,19973		D0728G	0,155608	0,160526	**0,164932**
D0706B	0,141236	0,154022	**0,1551**		D0729G	0,177796	0,180142	**0,190274**
D0707B	0,180512	**0,201396**	0,194344		D0730G	0,233324	0,23391	**0,234078**
D0708B	0,130536	0,140318	**0,141372**		D0731G	0,192062	**0,205996**	0,199182
D0709B	**0,06613**	**0,06613**	**0,06613**		D0732H	0,172196	0,18277	**0,183306**
D0710C	0,172538	0,188346	0,195474		D0733H	**0,175348**	0,1696	0,174618
D0711C	0,18067	**0,181272**	0,175116		D0734H	0,140428	0,14511	**0,156792**
D0712C	0,188116	**0,233986**	0,217128		D0735H	0,194044	**0,190534**	0,180866
D0713C	0,222392	0,227624	**0,232422**		D0736H	0,122422	**0,12416**	0,118468
D0714D	**0,204124**	0,203168	0,20165		D0737I	0,11583	0,116406	**0,141202**
D0715D	0,145318	**0,161562**	0,157376		D0738I	0,160666	0,171024	**0,178536**
D0716D	**0,216148**	0,208156	0,208916		D0739I	0,155244	0,162138	**0,168248**
D0717D	0,17512	**0,187114**	0,179374		D0740I	0,15168	0,157482	**0,157976**
D0718D	0,16168	0,160966	**0,173376**		D0741I	0,167164	0,169116	**0,17721**
D0719E	0,138976	0,150596	**0,159478**		D0742J	**0,205088**	0,200964	0,201706
D0720E	0,202076	**0,20265**	0,194166		D0743J	**0,13029**	0,127196	0,1286
D0721E	**0,183944**	0,197476	0,197084		D0744J	0,194042	**0,20406**	0,192974
D0722E	0,183616	0,189154	**0,207784**		D0745J	0,163374	0,171522	**0,177902**
D0723F	0,11961	0,125802	**0,132428**					

Tabla 17. Resultados DUC 2007 para Rouge-Su4

Para este conjunto de datos también se hizo una evaluación adicional para realizar la comparación con el sistema MCMR con PSO, como se puede observar en la **Tabla 18**, GHS+LEM supera a MCMR en la medida de Rouge-2 en un 2.75% y en la medida de Rouge-SU4 en un 4.07%, esto indica que la propuesta tiene un buen desempeño en generación de resúmenes de múltiples documentos.

Algoritmo	**ROUGE-2**	**ROUGE-SU4**
GHS+LEM	**0,1198**	**0,1769**
MCMR(PSO)	0,1165	0,1697

Tabla 18. Resultados ROUGE DUC 2007, 15000 iteraciones

4.4.3 Comportamiento de GHS+LEM con diferentes iteraciones.

Después de realizar las pruebas con 6000 iteraciones y con 15000 se pudo observar que al aumentar la cantidad de iteraciones en GHS+LEM, el algoritmo converge a una mejor solución para los documentos de DUC2007. Adicionalmente se realizó una prueba con este conjunto de datos con la medida de ROUGE-SU4 y diferentes números de iteraciones, entre 500 y 15.000, para observar el comportamiento de esta medida al incrementar la cantidad de iteraciones. En la Figura 10 se muestran los resultados que se obtuvieron al aplicar el algoritmo GHS+LEM, en la gráfica se observa que la medida Rouge-SU4 mejora cuando aumenta la cantidad de iteraciones. Cabe resaltar que después de 12.000 iteraciones la curva se hace más suave, esto indica que el algoritmo comienza a converger a una posible solución. Las otras medidas como Rouge-1 y Rouge-2 no se muestran debido a que tienen un comportamiento similar. Este proceso de evaluación no se realizó con los conjuntos de DUC 2005 por falta de tiempo, debido a que los documentos son más grandes que los de DUC 2007 lo que implica un mayor tiempo de ejecución.

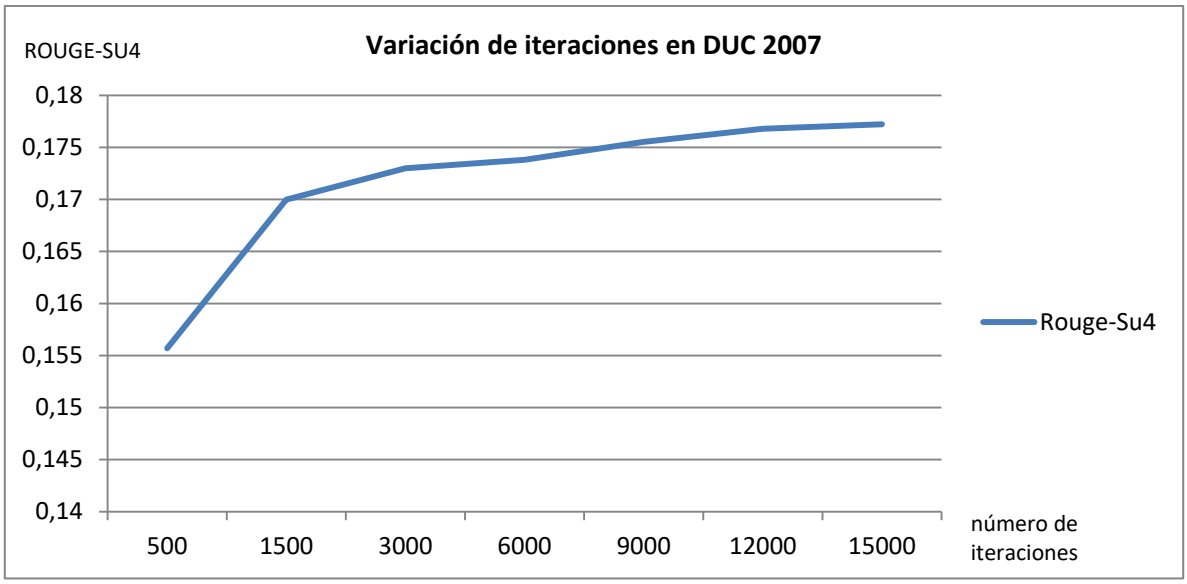

Figura 10. Duc 2007 con diferente numero de iteraciones.

4.4.4 GHS+LEM con respecto a otros sistemas

En la **Tabla 19** se muestran los resultados de la evaluación realizada al algoritmo GHS+LEM con los documentos de DUC 2007 y otros trabajos encontrados en el estado del arte con una breve descripción de cada uno. Como se puede observar, MCMR (B&B) tiene el mejor resultado en ROUGE-2, y GHS+LEM tiene el mejor resultado en ROUGE-SU4.

	DUC 2007				
Método	**Autor**	**Año**	**ROUGE-2**	**ROUGE-SU4**	**Descripción**
GHS+LEM (15.000)	Tamayo y Vela	2012	0,1198	**0,1769**	Método propuesto con 15000 iteraciones.
GHS+LEM (6.000)	Tamayo y Vela	2012	0,1170	0,1749	Método propuesto con 6000 iteraciones.
MCMR (B&B)	Alguliev, [9]	2011	**0,1221**	0,1753	El sistema se basa en el algoritmo de Ramificación y Poda para encontrar la solución óptima.
MCMR (PSO)	Alguliev[9]	2011	0,1165	0,1697	El sistema se basa en el algoritmo de Optimización de Partículas de Enjambre para encontrar la solución óptima.
HybHSum	Celikyilmaz [79]	2010	0,114	0,172	Utilizan dos etapas de aprendizaje: la construcción de un modelo generativo para descubrir patrones y un

					modelo de regresión para inferencia.
CBC	Celikyilmaz [80]	2011	0,1170	0,175	Se usa un modelo extractivo semi-supervisado basado en clasificación de conceptos latente. Utiliza un clasificador de entrenamiento y también el modelo Bayesiano: Modelo de tópicos enfocado en resúmenes (The summary-focused topic model).

Tabla 19. Valores de ROUGE obtenidos sobre los conjuntos de DUC 2007

En la **Tabla 20**, se puede ver que existe una mejora de GHS+LEM con 15.000 iteraciones con respecto a GHS+LEM con 6000 iteraciones. Por lo tanto el desempeño del algoritmo GHS+LEM depende de la cantidad de iteraciones que se ejecuten. También se puede observar que para los valores obtenidos en ROUGE-SU4, GHS+LEM (15.000) supera a todos los demás métodos, pero esto no ocurre en ROUGE-2, pues el valor negativo que se obtiene al realizar la comparación con MCMR (B&B) indica que éste supera los resultados de GHS+LEM. Sin embargo, es importante destacar que MCMR utiliza una función objetivo que involucra dos medidas de similitud (Cosenos y NGD), lo cual implica que el factor de cobertura y de eliminación de redundancia se calcula dos veces; en cambio GHS+LEM solo requiere de una medida de similitud (Cosenos) para obtener mejores resultados de Rouge. También se puede notar que aunque se requieren 15.000 iteraciones para superar los resultados de MCMR (B&B), basta con realizar 6000 iteraciones para superar a los demás métodos.

	Mejora del método GHS+LEM (%)	
Métodos	**ROUGE-2**	**ROUGE-SU4**
GHS+LEM (15.000)	0	0
GHS+LEM (6.000)	2,33	1,13
MCMR (B&B)	-1,91	0,90
MCMR (PSO)	2,75	4,07
HybHSum	4,84	2,77
CBC	2,33	1,07

Tabla 20. Comparación GHS+LEM (15.000) con otros métodos para conjuntos de DUC2007

En la **Tabla 21** se muestran los resultados de la evaluación realizada al algoritmo GHS+LEM con los documentos de DUC 2005 y otros trabajos encontrados en el estado del arte con una breve descripción de cada uno. Como se puede observar, al igual que en los resultados obtenidos con los documentos de DUC 2007, MCMR (B&B) tiene el mejor resultado en ROUGE-2, y GHS+LEM tiene el mejor resultado en ROUGE-SU4.

	DUC 2005				
Método	**Autor**	**Año**	**ROUGE-2**	**ROUGE-SU4**	**Descripción**
GHS+LEM (15000)			0,0769	**0,1428**	Método propuesto ejecutado con 15000 iteraciones.
GHS+LEM (6000)		2012	0,0786	0,1427	Método propuesto ejecutado con 6000 iteraciones.
MCMR(B&B)	Alguliev[9]	2011	**0,0790**	0,1392	Método no supervisado que para la generación del mejor resumen tiene en cuenta dos propiedades principales: cobertura y redundancia. El sistema se basa en el algoritmo de Ramificación y Poda para encontrar la solución óptima.
MCMR(PSO)	Alguliev[9]	2011	0,0754	0,1360	Método no supervisado que para la generación del mejor resumen tiene en cuenta dos propiedades principales: cobertura y redundancia. El sistema se basa en el algoritmo de Optimización de Partículas de Enjambre para encontrar la solución óptima.
TranSumm	Amini y Usunier [81]	2009	0,0755	0,1366	Método que utiliza un enfoque transductivo que identifica los temas de los tópicos dentro de la colección de documentos lo que ayuda a identificar conjuntos de oraciones relevantes e irrelevantes para una consulta. Para encontrar la relevancia de las oraciones, el algoritmo se basa en una función de probabilidad.
QEA	Zhao [58]	2009	0,0749	0,1333	Método que hace uso de un algoritmo basado en grafos para la clasificación de las oraciones. También utiliza un método de

					expansión de consultas y para la redundancia aplica una penalización sobre las oraciones.

Tabla 21. Valores de ROUGE obtenidos sobre los conjuntos de DUC 2005

Como se aprecia en la **Tabla 22** para los valores obtenidos en ROUGE-SU4, GHS+LEM con 15.000 iteraciones supera a todos los demás métodos, pero esto no ocurre en ROUGE-2, pues el valor negativo que se obtiene al realizar la comparación con GHS+LEM (6.000) y MCMR (B&B) indica que éstos superan los resultados de GHS+LEM (15.000). También se puede notar que basta con realizar 6000 iteraciones para superar a todos los demás métodos en Rouge-SU4.

	Mejora del método GHS+LEM (%)	
Métodos	**ROUGE-2**	**ROUGE-SU4**
GHS+LEM (15.000)	0	0
GHS+LEM (6.000)	-2,21	0,07
MCMR (B&B)	-2.73	2,52
MCMR (PSO)	1,95	4,76
TranSumm	1,82	4,34
QEA	2,6	6,65

Tabla 22. Comparación GHS+LEM con otros métodos para conjuntos de DUC 2005

Cabe resaltar que como se había mencionado anteriormente, MCMR utiliza una función objetivo que involucra dos medidas de similitud la cual es controlada por un valor $\propto$ que varía entre 0 y 1 con incrementos de 0,05. MCMR toma el mejor resultado obtenido a partir de la variación de ese valor, el cual es diferente para DUC2007, DUC2005, y para las medidas de ROUGE, mientras que en la función objetivo utilizada para GHS+LEM se tiene un coeficiente estándar para todas las evaluaciones realizadas.

Capítulo 5

5 CONCLUSIONES Y TRABAJO FUTURO

5.1 CONCLUSIONES

- En este trabajo se propuso un algoritmo GHS+LEM para generación automática de resúmenes extractivos de múltiples documentos, el cual utiliza técnicas de modelos evolutivos que aprenden para crear un conjunto de reglas que permiten inferir la selección de nuevas oraciones y de esta forma no surgen solamente de la exploración aleatoria. Además el algoritmo propuesto se evaluó con los conjuntos de datos de DUC 2005 y 2007 y se comparó con HS adaptado obteniendo mejores resultados debido a que GHS+LEM utiliza un conjunto de reglas a diferencia de HS, las cuales hacen que converja a una mejor solución con el mismo número de iteraciones.

- Los efectos de los parámetros HCMR, HMS, RCR, RRU, HLGS y NI en el desempeño del algoritmo propuesto, fueron los siguientes:

    - Tamaño de la memoria armónica (HMS): basados en lo establecido por Sharegui, quien propone un tamaño de memoria armónica de 100, y después de haber realizado pruebas preliminares de afinación se pudo establecer que el rango para HMS es entre 100 y 300, esto se debe a que para resúmenes de múltiples documentos la dimensionalidad es muy alta.

    - En cuanto al valor para HCMR se pudo establecer que debe ser superior a 0.95, esto se debe a que el tamaño de la memoria armónica es grande lo que permite tener en la memoria una exploración amplia del espacio de búsqueda, es decir, los mejores resúmenes, y por lo tanto sea mejor que el nuevo improviso se tome de la memoria y tenga en cuenta las probabilidades de unos y ceros almacenados en la memoria armónica en lugar de un aleatorio.

    - Con respecto a la variación del parámetro RCR se observó que se tiene un mejor desempeño general del algoritmo cuando el proceso de aplicación de las reglas es ejecutado con una probabilidad de 0.7, lo cual implica tomar con más frecuencia las reglas para la generación del resumen. Esto indica que el algoritmo de LEM propuesto en esta investigación es eficiente en la generación de resúmenes de múltiples documentos, además de esto requiere de pocos cálculos para la generación de las reglas.

    - Con relación al parámetro RRU se pudo establecer que debe ser un valor pequeño menor a 0.3, esto indica que no es necesario generar reglas muy seguido, además un valor pequeño de RRU ayuda a que el algoritmo tenga un mejor rendimiento debido a que realiza menos cálculos.

  - Numero de iteraciones (NI), según el proceso de afinación y pruebas realizado estableció que el algoritmo converge a mejores resultados cuando el número de iteraciones se encuentra entre 6000 y 15000.

  - Tamaño de los grupos de alto y bajo rendimiento, según las pruebas de afinación se determinó que el tamaño de los grupos debe estar entre 45% y 49.5% esto indica que la distancia que debe existir entre los dos grupos es de 10% a 1%.

- La eliminación de las oraciones que tienen una similitud con el documento menor a 0.13, que se realizó durante el pre-procesamiento de los documentos, permitió una convergencia más rápida a una mejor solución del algoritmo, esto se debe a que existen oraciones que no son significativas o pueden contener caracteres especiales que no son relevantes para la generación del resumen.

- La utilización de un coeficiente para variar el peso que se le da a cada uno de los factores ayuda a que la función objetivo encuentre mejores resultados, pues de esta manera se puede establecer a cuál de los factores se le da mayor importancia. Según las pruebas de afinación se concluyo que la eliminación de redundancia es un factor muy importante en la generación de resúmenes de múltiples documentos, dado que este factor presenta un peso del 70% del peso en la función objetivo y tan solo de un 30% para el factor de cobertura.

- En cuanto al rendimiento, la aplicación se tarda aproximadamente 2 días en generar los resúmenes para conjuntos de DUC 2005 y para DUC 2007 se tarda aproximadamente un día incluyendo la evaluación con las medidas de ROUGE-1, ROUGE-2 y ROUGE-SU4, adicionalmente se pueden obtener resúmenes para ambos conjuntos de documentos en paralelo. Si este proceso no es realizado automáticamente sino por una persona, requerirá de mucho tiempo y esfuerzo, teniendo en cuenta que DUC2005 tiene aproximadamente 1593 documentos y DUC 2007 tiene aproximadamente 1125 documentos.

- La generación automática de resúmenes es de gran utilidad cuando la información debe ser reducida para ser presentada por ejemplo en dispositivos pequeños (PDA's) en donde no sería adecuado mostrar gran cantidad de texto

- Con respecto a la evaluación del algoritmo GHS+LEM se puede concluir lo siguiente:

  - Las pruebas realizadas con conjuntos de datos de DUC 2005 y 2007 muestran que el algoritmo GHS+LEM es mejor a que HS y GHS en generación de resúmenes de múltiples documentos dado que converge a una mejor solución con el mismo número de iteraciones.

  - A pesar de que se requirieron 15.000 iteraciones para sobrepasar los resultados obtenidos por MCMR, basta con ejecutar el algoritmo GHS+LEM con las iteraciones obtenidas en la afinación para tener buenos resultados, lo que muestra que la afinación de parámetros fue lo suficientemente buena para que el algoritmo GHS+LEM en general se comporte mejor que los demás métodos.

- Una de las razones por la cual MCMR con ramificación y poda obtiene mejores resultados es porque en este trabajo se ejecuta el mismo algoritmo con dos medidas de similitud aumentando la posibilidad de encontrar mejores soluciones pero a cambio de un tiempo mucho mayor de procesamiento, mientras que con GHS+LEM no es necesario probar las dos medidas de similitud para obtener resultados satisfactorios.

- La diferencia en las medidas de Rouge-2 y Rouge-SU4 se debe al proceso de segmentación, ya que cuando esta difiere mucho entre el resumen automático y el resumen ideal se reduce la posibilidad de encontrar bi-gramas, mientras que con unigramas como lo hace Rouge-SU4 se le da menos importancia al orden de las oraciones

5.2 RECOMENDACIONES Y TRABAJO FUTURO

Como trabajo futuro se propone variar el proceso de inferencia de reglas teniendo en cuenta históricos de la memoria armónica y modificar el proceso de generación de reglas para que se actualicen cada vez que se ejecute un cambio en la memoria armónica con el fin de evaluar el desempeño del algoritmo en estas nuevas condiciones.

Modificar o adaptar la función objetivo para que contemple otras fórmulas para los factores de cobertura y eliminación de redundancia; y también que tenga en cuenta otros factores que puedan ser importantes para generar un buen resumen, que permitan mejorar la calidad de los resúmenes.

Evaluar el algoritmo con otros conjuntos de datos de las conferencias de DUC y TAC, que permita determinar si la calidad de los resúmenes con estos datos también se mejora con respecto a otros algoritmos de generación automática de múltiples documentos del estado del arte.

Capítulo 6

6 BIBLIOGRAFÍA

1. Hassel, M., *Resource Lean and Portable Automatic Text Summarization*, in *Computer Science and Communication*. 2007, KTH School of Computer Science and Communication: Stockholm, Sweden p. 144.
2. McKeown, K. and D.R. Radev. *Generating summaries of multiple news articles*. 1995: ACM.
3. Radev, D.R., et al., *Centroid-based summarization of multiple documents.* Information Processing & Management, 2000 **40**(6): p. 919-938.
4. Radev, D.R. and G. Erkan. *Lexpagerank: Prestige in multi-document text summarization*. 2004.
5. Mihalcea, R. *Graph-based ranking algorithms for sentence extraction, applied to text summarization*. 2004: Association for Computational Linguistics.
6. Xiao-Chen, M., Y. Gui-Bin, and M. Liang. *Multi-Document Summarization Using Clustering Algorithm*. in *Intelligent Systems and Applications, 2009. ISA 2009. International Workshop on*. 2009.
7. Lee, J.H., et al., *Automatic generic document summarization based on non-negative matrix factorization.* Information Processing & Management, 2009. **45**(1): p. 20-34.
8. Bossard, A. and C. Rodrigues, *Combining a Multi-document Summarization System with a Genetic Algorithm.* 2011.
9. Alguliev, R.M., et al., *MCMR: Maximum coverage and minimum redundant text summarization model.* Expert Systems with Applications, 2011. **In Press, Corrected Proof**.
10. Kowsalya, R., R. Priya, and P. Nithiya, *Multi Document Extractive Summarization Based On Word Sequences.* International Journal of Computer Science, 2011. **8**.
11. Lin, C.Y., *Multi-document Summarization via budgeted maximization of submodular Functions.* 2010.
12. L. Hennig, S.A., *Personalized Multi-Document Summarization using N-Gram Topic Model Fusion.* 2010.
13. Binwahlan, M.S., N. Salim, and L. Suanmali. *Swarm based text summarization*. 2009: IEEE.
14. Binwahlan, M.S., N. Salim, and L. Suanmali, *Swarm based features selection for text summarization.* International Journal of Computer Science and Network Security, 2009. **9**(1): p. 175-179.
15. Qazvinian, V., L. Sharif, and R. Halavati, *Summarization Text with a Genetic Algorithm-Based Sentence Extraction.* International Journal of Knowledge Management Studies (IJKMS, 2008. **4**(2): p. 426-444.
16. Shareghi, E. and L.S. Hassanabadi. *Text summarization with harmony search algorithm-based sentence extraction*. 2008: ACM.
17. Loganathan, G.V., *A New Heuristic Optimization Algorithm: Harmony Search.* SIMULATION, 2001. **76**(2): p. 60.
18. Omran, M.G.H. and M. Mahdavi, *Global-best harmony search.* Applied Mathematics and Computation, 2008. **198**(2): p. 643-656.

19. Coelho, L.S. and V.C. Mariani, *An improved harmony search algorithm for power economic load dispatch.* Energy Conversion and Management, 2009. **50**(10): p. 2522-2526.
20. Hoang, D.C., et al. *A Robust Harmony Search Algorithm based Clustering Protocol for Wireless Sensor Networks*: IEEE.
21. Carlos Cobos, Darío Estupiñan, and J. Perez, *GBHS-LEM: Hibridación del algoritmo GBHS con Modelos LEM.* Popayán, Colombia. Universidad del Cauca. Facultad de Ingenieria Electrónica y Telecomunicaciones, 2011.
22. Jezek, K. and J. Steinberger. *Automatic Text Summarization (The state of the art 2007 and new challenges)*. in *Znalosti 2008*. 2008 Bratislava, Slovakia
23. Lin, C.Y. *Rouge: A package for automatic evaluation of summaries*. 2004.
24. Michalsky, R.S., *Learnable Evolution Model.* Machine Learning, 2000(38): p. 9-40.
25. Cendrowska, J., *PRISM: An algorithm for inducing modular rules.* International Journal of Man-Machine Studies, 1987. **27**(4): p. 349-370.
26. Radev, D.R., E. Hovy, and K. McKeown, *Introduction to the special issue on summarization.* Computational Linguistics, 2002. **28**(4): p. 408.
27. Spärck Jones, K., *Automatic summarising: The state of the art.* Information Processing & Management, 2007. **43**(6): p. 1449-1481.
28. Ciencias, M.E.N., E.N. La Especialidad, and C. Computacionales, *Generacion Automatica de Resumenes de Multiples Documentos.* 2007.
29. Luhn, H., *The automatic creation of literature abstracts.* IBM Journal of Research and Development, 1958: p. 159-165.
30. Edmundson, H.P., *New Methods in Automatic Extracting.* J. ACM, 1969. **16**(2): p. 264-285.
31. Aone, C., Okurowski, M. E., Gorlinsky, J., and Larsen, B. s, *Trainable, Scalable Summarization Using Robust NLP and Machine Learning*.* Advances in Automatic Text Summarization, 1999 **Mani, I. and Maybury, M. T.**: p. 71-80.
32. Chuang, W.T. and J. Yang. *Extracting sentence segments for text summarization: a machine learning approach*. 2000: ACM.
33. Svore, K., L. Vanderwende, and C. Burges. *Enhancing single-document summarization by combining RankNet and third-party sources*. 2007.
34. Mohamed Abdel, F. and R. Fuji, *GA, MR, FFNN, PNN and GMM based models for automatic text summarization.* Comput. Speech Lang., 2009. **23**(1): p. 126-144.
35. Conroy, J.M. and D.P. O'Leary. *Text summarization via hidden markov models*. 2001: ACM.
36. Mihalcea, R. and P. Tarau. *A language independent algorithm for single and multiple document summarization*. 2005.
37. Barzilay, R., Elhadad, M, *Using Lexical Chains for Text Summarization. .* In Proceedings of the ACL/EACL'97 Workshop on Intelligent Scalable Text Summarization, Madrid, Spain., 1997: p. 10–17.
38. Yan-Min, C., W. Xiao-Long, and L. Bing-Quan. *Multi-document summarization based on lexical chains*. in *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*. 2005.
39. Marcu, D., *Improving summarization through rhetorical parsing tuning.* Proceedings of The Sixth Workshop on Very Large Corpora. Montreal, Canada, 1998: p. 206-215.
40. Steinberger, J. and M. K iš an. *Lsa-based multi-document summarization*. 2007.
41. Steinberger, J. and K. Jezek. *Sentence Compression for the LSA-based Summarizer*. 2006: Citeseer.
42. Sun, P. and C. ByungRae. *Query-Based Multi-Document Summarization Using Non-Negative Semantic Feature and NMF Clustering*. in *Fourth International*

Conference on Networked Computing and Advanced Information Management, 2008. NCM '08. . 2008.

43. Bäck, T., *Evolutionary algorithms in theory and practice*. 1996: Oxford University Press New York.
44. Cobos, C., D. Estupiñan, and J. Pérez, *GHS+LEM: Global-best Harmony Search using learnable evolution models.* Applied Mathematics and Computation. **218**(6): p. 2558-2578.
45. Julian, K., P. Jan, and C. Francine, *A trainable document summarizer*, in *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. 1995, ACM: Seattle, Washington, United States.
46. Jones, K.S. and J.R. Galliers, *Evaluating natural language processing systems: An analysis and review*. Vol. 1083. 1995: Springer Verlag.
47. Lin, C.Y. *Rouge: A package for automatic evaluation of summaries*. 2004.
48. Lin, C.Y. and E. Hovy. *Automatic evaluation of summaries using n-gram co-occurrence statistics*. 2003: Association for Computational Linguistics.
49. Baeza-Yates, R. and B. Ribeiro-Neto, *Modern information retrieval*. Vol. 82. 1999: Addison-Wesley New York.
50. Porrata, A.P., R.B. Llavori, and J.R. Shulcloper, *Desarrollo de Algoritmos para la Estructuración Dinámica de Información y su Aplicación a la Detección de Sucesos.* Castellón, España, 2004.
51. Salton, G., *Automatic Text Processing: The Transformation, Analysis, and Retrieval of*. 1989: Addison-Wesley.
52. Song, W., et al., *Fuzzy evolutionary optimization modeling and its applications to unsupervised categorization and extractive summarization.* Expert Systems with Applications.
53. Ali, M., M.K. Ghosh, and A. Al-Mamun. *Multi-document Text Summarization: SimWithFirst Based Features and Sentence Co-selection Based Evaluation*. 2009: IEEE.
54. Xiaojun, W., *An exploration of document impact on graph-based multi-document summarization*, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 2008, Association for Computational Linguistics: Honolulu, Hawaii.
55. Bollegala, D., N. Okazaki, and M. Ishizuka, *A machine learning approach to sentence ordering for multidocument summarization and its evaluation.* Natural Language Processingâ€“IJCNLP 2005, 2005: p. 624-635.
56. Hennig, L. and D.A.I. Labor. *Topic-based multi-document summarization with probabilistic latent semantic analysis*. 2009.
57. Sun, P. and C. ByungRae. *Query-Based Multi-Document Summarization Using Non-Negative Semantic Feature and NMF Clustering*. in *Networked Computing and Advanced Information Management, 2008. NCM '08. Fourth International Conference on*. 2008.
58. Zhao, L., L. Wu, and X. Huang, *Using query expansion in graph-based approach for query-focused multi-document summarization.* Information Processing & Management, 2009. **45**(1): p. 35-41.
59. Carrillo, P.A.A., I.F.V. López, and E.F. González, *Análisis Comparativo de las Medidas de Semejanza Aplicadas al Contenido de Documentos Web.*
60. Lee, L. *Measures of distributional similarity*. 1999: Association for Computational Linguistics.

61. Lee, K.S. and Z.W. Geem, *A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice.* Computer methods in applied mechanics and engineering, 2004. **194**(36-38): p. 3902-3933.
62. Geem, Z. and X.-S. Yang, *Harmony Search as a Metaheuristic Algorithm*, in *Music-Inspired Harmony Search Algorithm*. 2009, Springer Berlin / Heidelberg. p. 1-14.
63. Lee, K. and Z. Geem, *A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice.* Computer Methods in Applied Mechanics and Engineering, 2005. **194**(36-38): p. 3902-3933.
64. Ryszard, S.M., *LEARNABLE EVOLUTION MODEL: Evolutionary Processes Guided by Machine Learning.* Mach. Learn., 2000. **38**(1-2): p. 9-40.
65. Janusz, W. and S.M. Ryszard, *The LEM3 implementation of learnable evolution model and its testing on complex function optimization problems*, in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. 2006, ACM: Seattle, Washington, USA.
66. Lloret, E. and M. Palomar, *COMPENDIUM: Una herramienta de generaciÃ³n de resÃºmenes modular.* Procesamiento de Lenguaje Natural, 2011. **47**(0): p. 107-115.
67. Zhang, Y., D. Wang, and T. Li, *iDVS: an interactive multi-document visual summarization system.* Machine Learning and Knowledge Discovery in Databases, 2011: p. 569-584.
68. Michalski, R.S., *LEARNABLE EVOLUTION MODEL.* Machine Learning, 2000. **38**(1-2).
69. Ehsan, S. and H. Leila Sharif, *Text summarization with harmony search algorithm-based sentence extraction*, in *Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology*. 2008
ACM: Cergy-Pontoise, France.
70. Mahdavi, M., et al., *Novel meta-heuristic algorithms for clustering web documents.* Applied Mathematics and Computation, 2008. **201**(1-2): p. 441-451.
71. Hearst, M.A., *TextTiling: Segmenting text into multi-paragraph subtopic passages.* Computational Linguistics, 1997. **23**(1): p. 33-64.
72. Porter, M.F., *An algorithm for suffix stripping*. 1980, Program.
73. Radev, D.R., et al., *Centroid-based summarization of multiple documents.* Information Processing & Management, 2004. **40**(6): p. 919-938.
74. Lucene, *Sitio web de Lucene: Disponible en http://lucene.apache.org.*
75. stoplist:, E., *ftp://ftp.cs.cornell.edu/pub/smart/english.stop.*
76. Zhou, Z., *Combined Features to Maximal Marginal Relevance Algorithm for Multi-document Summarization.* Journal of Convergence Information Technology, 2011. **6**(5).
77. Litvak, M., M. Last, and M. Friedman. *A new approach to improving multilingual summarization using a genetic algorithm*. 2010 Association for Computational Linguistics.
78. Gong, S., Y. Qu, and S. Tian. *Subtopic-based Multi-documents Summarization*. in *Third International Joint Conference on Computational Science and Optimization (CSO), 2010.* 2010.
79. Celikyilmaz, A. and D. Hakkani-Tur. *A hybrid hierarchical model for multi-document summarization*. 2010: Association for Computational Linguistics.
80. Celikyilmaz, A. and D. Hakkani-Tur. *Concept-based classification for multi-document summarization*. 2011: IEEE.
81. Amini, M.R. and N. Usunier. *Incorporating prior knowledge into a transductive ranking algorithm for multi-document summarization*. 2009.