UNIVERSIDAD NACIONAL DE LA PLATA

TESINA DE LICENCIATURA EN INFORMÁTICA

Resumen extractivo de documentos. Un análisis comparativo de técnicas de puntuación

---

Autora:
Julieta Pilar CORVI

Directora:
Dra. Laura LANZARINI

Asesor profesional:
Dr. Augusto VILLA MONTE

Tesina presentada para obtener el grado de Licenciada en Informática

Facultad de Informática

5 de agosto de 2019

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

Prefacio

En la actualidad, es numerosa la cantidad de información digital disponible en diversos formatos, siendo el texto el más habitual. Desde libros y artículos científicos, pasando por diarios y revistas hasta llegar a las redes sociales e internet, en general, se encuentran repletas de documentos. Sin resúmenes sería prácticamente imposible para las personas tener acceso a la creciente masa de información disponible en internet (Saggion y Poibeau, 2013).

El resumen automático de un documento de texto es el proceso que, a través de una aplicación informática, reduce al mismo obteniendo sus partes más importantes.

Esta tesina tiene por objetivo general el desarrollo de una técnica capaz de resumir documentos en forma automática haciendo foco en el resumen extractivo. Este tipo de resumen se basa en identificar las partes más relevantes del documento de la misma forma que podría hacerlo un lector utilizando un resaltador. La respuesta obtenida puede ser o bien solo las oraciones seleccionadas o bien el documento completo con dichas oraciones marcadas con color. En ambos casos, el resumen a generar quedará formado por partes del texto original.

Una dificultad que surge al momento de resolver esta tarea es que, ante un mismo documento, no todas las personas consideran importantes las mismas oraciones. Por tal motivo, si bien existen en la literatura distintas formas para medir la importancia de las partes que componen un documento, es necesario considerar la opinión de quien será el receptor del resumen generado automáticamente.

El objetivo específico de esta tesina es definir una estrategia que sea capaz de ponderar la importancia de las sentencias que componen el documento de forma similar a lo que haría el usuario. Para ello será necesario contar con un resumen extractivo generado por dicho usuario a partir del cual se generará el modelo a utilizar en el resto de los documentos a resumir. Si bien la precondición de contar con un resumen puede parecer poco práctica, debe tenerse en cuenta que solo se le pide al usuario un resumen corto en comparación a la cantidad de texto sobre el que puede ser aplicada dicha estrategia posteriormente.

En base a todo lo antes expuesto puede decirse que **el objetivo de esta tesina es proponer una técnica de resumen automático extractivo basada en la manera de resumir del usuario**.

Este documento se encuentra organizado de la siguiente manera:

- En en el capítulo 1 se desarrollan los aspectos generales de la Minería de Datos y de Texto, temáticas principales de la tesina. Se describe como se relacionan entre sí y luego se profundiza en los conceptos de documentos de texto y al preprocesamiento correspondiente al tipo de dato textual. La finalidad del capítulo es introducir al lector un marco teórico mínimo antes de arribar al método propuesto. Por lo que, si el lector está familiarizado con este tipo de conceptos, este capítulo lo puede omitir.

- En el capítulo 2, el énfasis se encuentra puesto en explicar los tipos de resúmenes que se pueden generar, la comparación entre ellos y se abarca de una manera más detallada la técnica extractiva que fue la utilizada para el desarrollo.

- En el capítulo 3, se introduce el tema de redes neuronales, profundizando en el combinador lineal adaptativo de la red neuronal ADALINE, modelo utilizado para desarrollar el método propuesto.

- En el capítulo 4 se expone el método propuesto. En base a ejemplos de resúmenes realizados por un usuario humano, se desarrolló un método capaz de crear un modelo para resumir automáticamente de la manera más aproximada a cómo lo haría esa persona. A su vez, se definen las herramientas que se utilizaron y la manera en la que se las utilizó para alcanzar el objetivo. Se cierra el capítulo con la presentación de los resultados obtenidos y las conclusiones en base a ellos.

- En el capitulo 5 se plantean las conclusiones generales de la tesina.

Capítulo 1

Resúmenes de documentos

1.1. Introducción

Han pasado más de 60 años de la famosa publicación de Luhn (Luhn, 1958) sobre los resúmenes automáticos de texto. En todos esos años la necesidad de resumir textos se ha incrementado exponencialmente, sobretodo por el uso de las nuevas tecnologías.

El resumen de un documento se puede definir como el punto intermedio entre el título y el texto completo del documento. Así es que, la finalidad de resumir es generar una descripción concisa del texto, que sea más reveladora que el título pero no tan extensa como leer todo el documento (Kupiec, Pedersen y Chen, 1995).

Si bien en la web puede hallarse información representada de manera muy diversa, el formato texto sigue siendo el más popular tanto en lo que se refiere a representación y almacenamiento como a mecanismo de consulta. Es por esto que al momento de efectuar una búsqueda se pueden encontrar millones de recursos en formato texto sobre cualquier tema; el problema es saber qué documentos son relevantes para esa temática en particular y si realmente poseen la información que se está buscando.

Actualmente, el resumen automático de textos se ha constituido en una herramienta sumamente útil a la hora de almacenar y recuperar información. Su uso aumentó notablemente, por lo que se hizo imprescindible ahondar en el estudio y desarrollo del proceso de resumir automáticamente documentos.

Contar con estrategias capaces de resumirlos automáticamente facilitará su procesamiento, no solo en lo que se refiere a los lectores humanos quienes agilizarán su lectura sino en la reducción del tiempo de ejecución necesario para su posterior procesamiento.

A modo de ejemplo de uso general, si se tiene un conjunto de documentos a partir del cual se quisiera seleccionar los que están relacionados con una temática en particular, en general hoy en día sería preciso acceder a cada uno de ellos y realizar una lectura lineal para determinar cuáles corresponden a la búsqueda que se está efectuando. Para reducir la cantidad de documentos a procesar, una posible alternativa es analizar la relación que posean los respectivos títulos con el tema buscado. Esto solo puede ayudar en algunas ocasiones, ya que es posible que el título no se relacione con la búsqueda pero el contenido del documento si. Sin embargo, si se tuviera un resumen de cada uno de esos documentos o las palabras claves de cada uno de ellos, sería posible hacer una búsqueda mucho más eficiente y rápida.

Además de la eficiencia a nivel de tiempo de búsqueda, el contar con un resumen de los documentos brinda eficiencia en tiempo de recuperación de los mismos. Esto significa que, por ejemplo, si solo se considera que hay dos tipos de memorias, una RAM y un disco rígido, al no tener resúmenes de los documentos para poder buscar las palabras o temas que se necesitan, el sistema operativo debe cargar en memoria RAM los documentos completos del disco, lo que hace que al cambiar de documento probablemente se deba hacer una lectura al mismo implicando un costo muy grande a nivel de eficiencia de respuesta por el tipo de memoria.

Si se tienen los resúmenes de los documentos, se obtienen a simple vista dos ventajas. La primera es que no va a ser necesario leer los documentos completos para poder saber cuáles tratan el tema buscado. Este es el objetivo principal de tener resúmenes, no tener que leer todo el texto pero saber de qué se trata el mismo. También, no se corre el riesgo de no leer algún documento o de no prestar atención a alguno por no leer la parte más significativa del mismo.

La segunda ventaja es que al ocupar menos espacio, se podrían tener en RAM resúmenes de más documentos, esto hace que el acceso a disco se reduzca considerablemente. Esto se conoce comúnmente como *"lazy loading"* y se implementa mucho hoy en día para agilizar la respuesta al usuario. Se basa en que el usuario vea lo justo y necesario y solo si él lo requiere, se carga la información completa. Podrían tenerse réplicas de los resúmenes en las memorias de acceso más rápido y solamente acceder al disco cuando el usuario quiera leer un documento completo. De esta manera el lector puede abarcar una mayor cantidad de contenido y también hace más eficiente el tiempo de acceso a memoria de la computadora.

Otra de las nuevas aplicaciones en las que se está utilizando el resumen automático de textos es en la industria mediante técnicas de resumen automático parametrizado por el usuario con actualización dinámica. Por cada conjunto de consultas ingresadas por el usuario, se arma un resumen que responde a esa solicitud; el usuario interacciona con ese resumen y se le permite ingresar nuevas consultas, si es que lo desea. La técnica empleada toma las nuevas consultas y rearma el resumen teniendo en cuenta las consultas previas, considerando que esa información ya es conocida por el usuario. De esta manera se evita la redundancia entre consultas y el usuario puede profundizar su conocimiento sobre el conjunto de documentos (Baumel, Cohen y Elhadad, 2014), (Tauchmann y col., 2018).

Todas estas aplicaciones junto con el uso de nuevas tecnologías hacen que los resúmenes automáticos de texto tomen un papel protagónico en la investigación.

Es muy difícil para una persona hacer resúmenes de textos largos o de un conjunto grande de documentos. El objetivo del resumen automático de texto es el de condensar los documentos en una versión más corta preservando el contenido importante (A. Khan y Salim, 2014).

El objetivo de esta tesina es obtener un resumen de forma automática manteniendo la semejanza con el que hubiera generado el usuario humano. Para ello es preciso contar con un breve resumen de muestra a partir del cual se obtendrá un modelo que podrá ser aplicado a nuevos documentos dando lugar así a un resumen generado de forma similar a lo realizado por el usuario previamente.

El modelo se obtiene a través de técnicas que se enmarcan en el área conocida como Minería de Datos y de Textos, la cual se describirá a continuación.

1.2. El proceso de extracción de conocimiento

Obtener conocimiento a partir de información almacenada es un tema de interés desde hace décadas. En el caso de esta tesina se busca construir un modelo capaz de identificar las preferencias del usuario al momento de elaborar un resumen. Para lograrlo es preciso comprender las distintas etapas del proceso de extracción de conocimiento a partir de información almacenada o KDD *Knowledge Discovery in Database*.

El KDD es el proceso de descubrir conocimiento a partir de información almacenada. Este proceso comienza con la recolección de la información, ya que no siempre

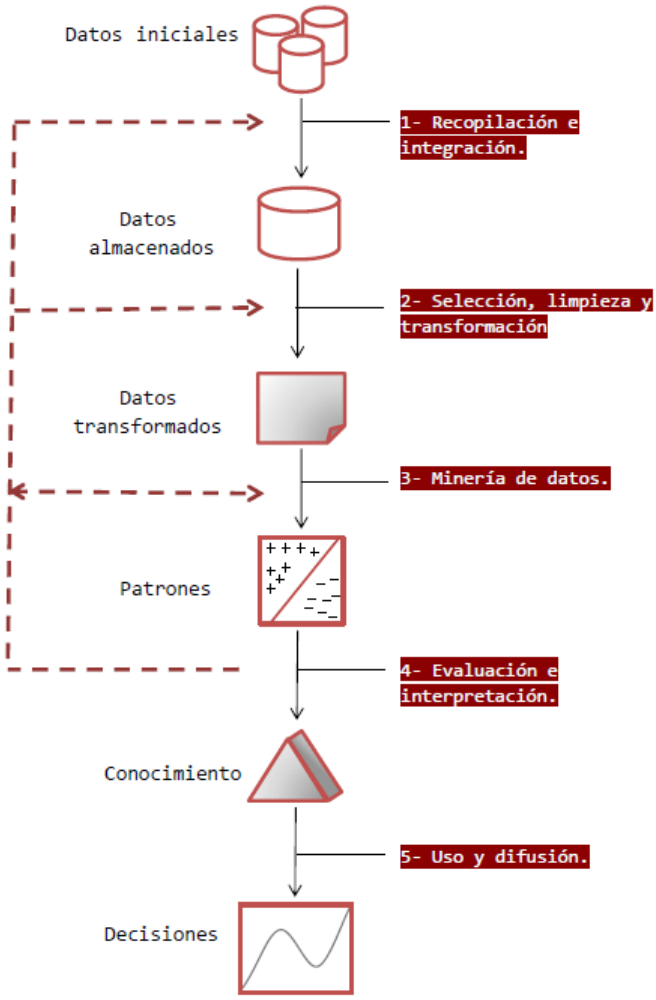


FIGURA 1.1: Fases del proceso de KDD (Fayyad, Piatetsky-Shapiro y Smyth, 1996).

se encuentra ubicada en un mismo punto y finaliza con el conocimiento extraído a partir de ella. Se trata de un proceso en fases que requiere de revisiones continuas, especialmente relacionada con los resultados intermedios obtenidos y la mirada puesta en el tipo de respuesta esperada.

Según (Fayyad, Piatetsky-Shapiro y Smyth, 1996), *"KDD es el proceso no trivial de identificar patrones válidos, novedosos, potencialmente útiles y en última instancia comprensibles a partir de los datos"*. El resultado de este proceso es lo que le da significado al almacenamiento de la información y ayudará a poder tomar decisiones en base a esa información.

El proceso de extracción de conocimiento consta de cinco fases, como se puede observar en la figura 1.1:

- La primera fase es la de *recopilación e integración de datos*. En ella se determinan las fuentes de información útiles y cómo obtenerlas. Es importante en este punto considerar que, por lo general, al momento de iniciar con el proceso de KDD ya se dispone de información histórica almacenada. Este aspecto tiene sus ventajas y sus desventajas. La parte positiva es que se reduce el tiempo y el esfuerzo necesarios para disponer de los datos. La parte negativa es que solo se dispone de "esos" datos almacenados no teniendo la oportunidad de diseñar ninguna estrategia alternativa. Independientemente del camino a seguir, el resultado estará condicionado por la información disponible.

- Luego, la segunda etapa es la *fase de selección, limpieza y transformación*. Aquí se identifican las características más relevantes para resolver la tarea planteada y se trabaja en la corrección o eliminación de errores y datos incompletos. El resultado de esta fase es la "vista minable" o información sobre a partir de la cual se continuará trabajando en las siguientes etapas.

- La tercera fase del proceso de KDD, es la *fase de Minería de Datos*. Aquí es donde se eligen las técnicas a utilizar en base al tipo de problema a resolver. Como resultado se obtendrá un modelo que, si bien es capaz de hallar relaciones o patrones entre los datos o ejemplos que conforman la vista minable, no siempre puede ser comprendido directamente por el usuario final encargado de tomar las decisiones. Por ello, la siguiente es una fase de análisis.

- En la *fase de evaluación e interpretación* se evalúan los patrones obtenidos en el punto anterior y se los analiza.

- Finalmente, la fase número cinco, *fase de difusión*, es donde se hace uso del nuevo conocimiento adquirido.

Tal como se observa en la figura 1.1 luego de cada fase puede ocurrir que los resultados no sean los esperados y sea preciso volver atrás. Puede hallarse una descripción más detallada de este interesante proceso en (Fayyad, Piatetsky-Shapiro y Smyth, 1996) o en capítulo 2 del libro (Hernández Orallo, Ramírez Quintana y Ferri Ramírez, 2004).

A continuación se desarrolla con mayor profundidad la fase de la Minería de Datos.

1.3. Minería de datos

La Minería de Datos es el conjunto de técnicas que pueden aplicarse sobre la *vista minable* (información ya preprocesada) para hallar relaciones o patrones novedosos y útiles brindando información sobre cómo están organizados los ejemplos. Esto permite extraer patrones, descubrir tendencias e incluso predecir comportamiento.

La búsqueda de patrones en datos es algo que se viene viendo hace mucho tiempo. Los cazadores buscan patrones en el comportamiento de los animales, los granjeros buscan patrones en la explotación agrícola, los políticos en la opinión de los votantes, y así en todas las áreas en las que se disponga de algún tipo de información. Análogamente, en la Minería de Datos, los datos se encuentran almacenados electrónicamente y la búsqueda de patrones es automática (o al menos una parte de la misma) (Witten y Frank, 2005).

El resultado de aplicar una técnica de Minería de Datos es la obtención de un modelo. La elección de la técnica a utilizar depende del tipo de problema a resolver.

Básicamente, puede decirse que hay dos tipos de problemas:

- Problema predictivo: Es aquél que ante un nuevo caso requiere predecir una respuesta. En este caso se debe disponer de información histórica etiquetada. Es decir que, se debe contar con ejemplos del problema a resolver y para cada caso se debe tener información de la respuesta esperada. Ejemplo de este tipo de problemas es la predicción de la probabilidad de que un paciente padezca cierta enfermedad conociendo su historia clínica o la predicción de la calidad de un determinado producto dadas sus características más relevantes.

- Problema descriptivo: Es aquél que ante un conjunto de ejemplos busca establecer relaciones o similitudes entre ellos para identificar patrones o perfiles útiles. Por ejemplo, una empresa podría estar interesada en conocer los distintos perfiles de clientes que compran sus productos con el objetivo de conducir la próxima campaña publicitaria. Aquí no interesa dar una respuesta ante un caso (cliente) puntual sino explicar cuáles son las características centrales que reúnen a quienes compran los productos.

Según el tipo de problema a resolver será la técnica que deberá aplicarse. Por ejemplo, las técnicas de agrupamiento son las más utilizadas a la hora de resolver un problema descriptivo. En cambio, si lo que se busca construir es un modelo predictivo, las soluciones no lineales resultan muy convenientes. Este es el caso de las

redes neuronales, modelo utilizado en este trabajo. Tanto las técnicas de agrupamiento como las redes neuronales, se encuentran explicadas con mayor detalle en el capitulo 3.

Cada técnica de Minería de Datos a utilizar tiene sus propias características. No todas pueden operar con datos faltantes o con información en cualquier tipo de formato. Por esto es preciso saber primero qué tipo de problema se quiere resolver y en base a eso elegir la técnica a aplicar. Una vez decidido esto, se conocerá la manera en que deben ser preprocesados los datos para obtener una vista minable adecuada. Esto es lo que sucede con la Minería de Texto, donde el tratamiento de los datos es bastante particular.

Las técnicas capaces de procesar información almacenada en formato de texto se engloban en lo que se conoce como Minería de Textos. Es aquí donde quedan comprendidas las tareas realizadas en el marco de esta tesina y por tal motivo, en la siguiente sección se dará mayor detalle a este tema.

1.4. Minería de Texto

Como se mencionó en una sección anterior, la Minería de Datos es la etapa del KDD a través de la cual se descubre de manera automática información útil a partir de datos o ejemplos históricos almacenados.

Se define a la Minería de Texto como un caso particular de la Minería de Datos, a través del cual se busca extraer conocimiento y realizar un análisis pero únicamente a partir de texto. Si bien el proceso de KDD no se modifica según el tipo de información del cual se trate, las operaciones de preparación de datos son específicas del procesamiento de textos. La diferencia fundamental radica en que el texto carece de estructura y por lo tanto, identificar y extraer sus características o partes importantes requiere de un tratamiento especial.

Uno de los enfoques más utilizados para procesar textos es transformarlos en datos estructurados. De esta forma, el énfasis para poder ser analizados y utilizados está puesto en la construcción de la vista minable.

A partir del texto preprocesado, se lo transforma y representa adecuadamente (generando características) para poder continuar con el proceso aplicando las técnicas específicas de Minería de Texto y/o de Minería de Datos (adaptadas de ser necesario) que permiten descubrir patrones (Liang y Tan, 2007).

Las herramientas de la Minería de Texto tienen el foco en grandes conjuntos de documentos de texto, por lo que a continuación se explican los conceptos de colección de documentos, el documento de texto y las técnicas de preprocesamiento necesarias para llevar a cabo el método propuesto de esta tesina.

1.4.1. La colección de documentos y el documento

El elemento más importante de la Minería de Texto es sin duda la batería de documentos a analizar, también llamada *CORPUS*. La mayoría de las soluciones de la Minería de Texto se consiguen a través del reconocimiento de patrones en grandes colecciones de documentos. Estas colecciones pueden contener entre miles y millones de documentos, por lo cual, intentar cruzar datos entre los distintos documentos o crear mapas de relación de los documentos de manera manual es una tarea casi imposible de conseguir. Los métodos automáticos para estos fines mejoran drásticamente la velocidad y eficiencia en la investigación.

Sin embargo para poder realizar el análisis, los sistemas de Minería de Texto necesitan colecciones de documentos preparados para tal fin. Para esto se realiza el preprocesamiento de los documentos, el cual incluye una variedad de técnicas de recuperación de información, extracción de información e investigación en la lingüística computacional que transforman el texto original en datos cuidadosamente estructurados y almacenados para su posterior análisis.

El documento

Un documento puede ser definido como una unidad de datos de texto que se encuentra dentro de una o más colecciones de documentos. Más allá de no estar estructurado en un principio para realizar el análisis, un documento de texto se puede ver como un objeto que posee un cierto grado de estructura. Poseen una estructura semántica y sintáctica, aunque estas son implícitas y de cierta manera están ocultas en el texto.

A su vez, existen elementos tipográficos como los signos de puntuación, el uso de mayúsculas, los caracteres numéricos y los caracteres especiales, los espacios en blanco, los textos subrayados, entre otros, que se pueden utilizar como marcas del lenguaje dentro del documento. Estas marcas sirven para identificar dentro del mismo componentes tales como párrafos, títulos, fechas, nombres de autores, tablas, encabezados, etc.

A los documentos que no poseen una estructura fuertemente tipográfica se los define como *documentos de formato libre o estructura débil*. Por otro lado, a los documentos que poseen elementos con formatos definidos en donde los metadatos se pueden identificar fácilmente (como un e-mail, HTML, XML, páginas web) son definidos como *documentos semiestructurados* (Feldman y Sanger, 2006).

Características de un documento

Para poder realizar un análisis de un documento se necesita identificar y almacenar la mayor cantidad de elementos que caracterizan al mismo. El preprocesamiento sirve para transformar el texto fuente, con su estructura irregular e implícita, en una representación explícitamente estructurada. Se necesitan identificar todas las partes de un documento tales como el título, los párrafos, los títulos de las secciones, si las tuviera, las oraciones, las palabras, los conceptos, los caracteres, etc.

Las partes de un documento varían según el tipo; por ejemplo un artículo científico tiene un resumen, una introducción, secciones y títulos de las secciones, pero un cuento infantil no posee estos mismos elementos. Por este motivo son muy importantes las definiciones que se toman en esta etapa. Se deben extraer las características que se consideren más representativas de los documentos de la colección y persistirlas en un modelo relacional para luego poder aplicar los procesos de la Minería de Texto.

1.4.2. Preprocesamiento de los documentos

Las siguientes operaciones fueron aplicadas en esta tesina para poder alcanzar el objetivo de la misma. La manera en la cual se las utilizó se explica en detalle dentro del capítulo 4.

Segmentación del texto original

La tarea más importante dentro del preprocesamiento es la de segmentar el documento para luego poder representarlo. Esta tarea consiste en dividir el texto en porciones más pequeñas, con algún criterio, a partir de uno o varios delimitadores.

Para facilitar la segmentación es deseable que los documentos estén de alguna manera estructurados o "marcados". En caso contrario, no será posible identificar las partes que lo componen dejando que todas las oraciones del texto posean la misma

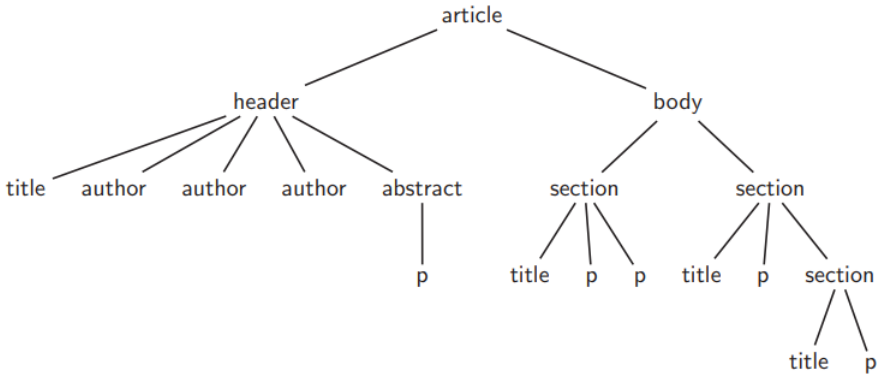


FIGURA 1.2: Árbol con la estructura de un documento XML - (Büttcher, Clarke y Cormack, 2010)

importancia. Es claro que no se obtendrá el mismo resultado si se pueden distinguir partes del documento (tales como el título, los subtítulos que dividen secciones, o las conclusiones) a disponer de un documento plano sin ningún tipo de identificación de su estructura interna. Por tal motivo, hoy en día se está empezando a tomar el recaudo de guardar los datos de manera adecuada para su posterior manipulación. Los documentos PDF, por ejemplo, no poseen ningún formato particular dentro del texto, lo cuál hace imposible la tarea de identificar partes del documento. En cambio los documentos con formato XML proveen una estructura que permite identificarlos y así también, da la posibilidad de obtenerlos (Véase anexo A para un mayor detalle sobre este tema).

En la figura 1.2 se puede observar la estructura de etiquetas de un documento XML simple.

Si los documentos que se desean resumir se encuentran marcados de alguna forma, se puede proceder a la división del texto. Para ello hay varias maneras; puede ser por palabras, oraciones, párrafos, secciones, etc. Depende mucho del tipo de documento que se quiere resumir y del tipo de resumen que se desee hacer.

Generalmente, a cada división se la llama unidad textual o "sentencia" y se la obtiene reconociendo los signos ortográficos del texto, tales como el punto, la coma, el punto y coma. En el caso de dividir el texto entre puntos, las unidades textuales serían las oraciones. Una oración es una unidad del lenguaje que tiene sentido por si misma y que contiene un verbo.

También pueden utilizarse otros signos como la coma, el punto y coma, los dos

puntos y los paréntesis. El uso de cada uno de ellos tendrá consideraciones especiales tal como también las tiene el punto en relación a sus otros usos: en abreviaturas, siglas y números.

Todas las características que se deseen obtener posteriormente para rearmar el documento o para analizarlo, deben almacenarse de alguna manera. Pero antes de esto, se debe adecuar el texto para poder extraer las características sin el ruido que existe en el lenguaje natural.

Antes de poder comenzar con cualquier tipo de procesamiento sobre el texto, se debe segmentar el mismo en unidades lingüísticas tales como párrafos, oraciones o palabras.

Una vez extraídas las oraciones del documento, se deben obtener cada uno de sus términos y guardarlos de tal manera que luego se pueda trabajar con ellos. Para esto se realiza el proceso de tokenización del texto.

Tokenización

Se llama tokenización a la tarea de separar el texto en estas unidades a las cuales se les da el nombre de "tokens". En sí, tokenizar un documento, es el método que sirve para simplificar el contenido textual para su posterior procesamiento.

Toda palabra del documento es considerada un posible término o token. Se considerará un término a cualquier secuencia de símbolos que comience o finalice con un delimitador o esté delimitado por ambos. El caracter blanco y los signos de puntuación son los delimitadores habituales pero pueden indicarse conjuntos especiales según el tipo de texto con el que se trabaje.

En este proceso, se suelen transformar todos los caracteres de las palabras a minúscula o mayúscula, ya que no es una característica significativa para el análisis pero simplifica algunas cuestiones del tratamiento del texto. Hay algunas excepciones como los nombres propios o los nombres de países, por ejemplo.

Otra de las decisiones que se toman en este punto es la forma en que se manejarán las abreviaciones en el texto. En la mayoría de los lenguajes cuando al final de un conjunto de caracteres hay un punto implica el fin de la sentencia; sin embargo, no siempre es así. Se deben considerar, por ejemplo, las abreviaciones seguidas de un punto, donde el punto no implica fin de oración por lo cual no hay que hacer el corte. Por ejemplo, en la oración "El Dr. García atiende lunes, martes y miércoles.",

si no se tienen en cuenta las abreviaciones, al cortar las sentencias quedaría dividida en dos oraciones "El Dr." y "García atiende lunes, martes y miércoles.". Otro tipo de abreviaciones son por ejemplo "adj." (adjetivo), "adv." (adverbio), "EE.UU." (Estados Unidos), "lat." (latín), entre otros.

Naturalmente, las decisiones que se tomen en esta etapa van a afectar directamente al resumen. Una de las opciones sería tener un listado de abreviaciones para así poder identificarlas y si es una abreviación se la almacena con el punto incluido, o dependiendo del tipo de abreviación, con el punto y la palabra siguiente. Otra solución sería quitar los puntos que no son finales de sentencia. No es la solución óptima pero si la más fácil y rápida. Con el listado de abreviaciones se pueden tomar recaudos para mantener la coherencia en los tokens lo más posible, pero nada asegura que en el listado estén todas las abreviaciones de un lenguaje.

Palabras vacías de contenido o stopwords

Una vez obtenidas todas las palabras del texto, se realiza un proceso de eliminación de palabras que no aportan, y hasta empeoran, el análisis para realizar el resumen del documento. Las *stopwords o palabras vacías* son palabras del lenguaje natural que no poseen contenido semántico, como artículos, preposiciones, pronombres personales, conjunciones, adverbios y adjetivos.

Son palabras que modifican a otras palabras o sirven para indicar relaciones gramaticales. Ignorar este tipo de palabras puede mejorar considerablemente la eficiencia del resumidor. Por ejemplo, utilizando como forma de representar a un documento el modelo de espacio vectorial, donde cada oración es un vector en el espacio y la proximidad entre vectores representa la semejanza entre las oraciones, si no se excluyen las palabras vacías, va a haber proximidad entre oraciones simplemente por que contienen gran cantidad de artículos en común, lo cual no es representativo en absoluto.

Al excluir las stopwords se le puede dar mayor énfasis a las palabras más representativas de cada oración. Lógicamente la cantidad de stopwords en un documento es mucho mayor que la cantidad de palabras que poseen un significado semántico, por lo que mantener las palabras vacías hace que se pierdan las palabras que deberían tenerse en cuenta. Por este motivo, tradicionalmente los sistemas de extracción

de información definen un listado de stopwords y se excluyen las palabras del documento que estén presentes en ese listado. La longitud de este listado varía de sistema en sistema, puede contener desde una docena de palabras hasta cientos de ellas.

Debido a que las stopwords son muy frecuentes en los textos, al eliminarlas se reduce considerablemente el tiempo de ejecución del análisis, ya que son menos palabras por oración. Cuando la colección de documentos es muy grande y se posee poco espacio de almacenamiento, si se excluyen todas las palabras vacías de los documentos, no sería necesario guardarlas junto a las demás palabras y esto reduce la cantidad de espacio a utilizar (Büttcher, Clarke y Cormack, 2010).

También es habitual descartar algunos verbos como "ser" y "estar" o palabras que tengan una longitud menor a tres letras. El problema de sacar palabras del texto es que hay ciertos textos que utilizan esas palabras como contenido de las oraciones. Por ejemplo, el soliloquio de Hamlet "Ser o no ser", extrayendo las palabras vacías o la palabra "ser", se pierde el significado de la oración, la cual es sumamente representativa en el texto. Este ejemplo es excepcional, pero no deja de ser posible que haya textos de este estilo. Por lo tanto, es importante tener en cuenta el tipo de documento que se quiere representar a la hora de aplicar el proceso de excluir las stopwords.

Tanto el proceso de tokenización como el proceso de extracción de las palabras vacías, dependen del idioma del texto. Hay algunas consideraciones que se deben tener en cuenta para algunos lenguajes que para otros no. Por ejemplo en el idioma inglés utilizan muchas palabras compuestas como *"well-known"*, si la colección de documentos es en inglés deben considerarse estos casos, ya que no es lo mismo guardar las palabras compuestas como un único token o guardar las palabras por separado. Este tipo de elecciones hacen que cada resumidor automático sea distinto y que los resúmenes varíen mucho entre un resumidor y otro. Dividir las unidades textuales en una o más palabras, considerar o no los puntos intermedios en una oración, hacen que las medidas que se calculan en el siguiente capítulo 2 varíen su puntaje para cada sentencia.

Raíces de las palabras o stemming

Habiendo realizado los procesos de tokenización y de extracción de las palabras vacías, se puede realizar el proceso de stemming. Esta técnica sirve para extraer la

raíz de las palabras, considerando la morfología, o estructura interna de los términos, reduciendo cada uno de ellos a su raíz por comparación. Por ejemplo, comparando palabras tales como "camionero", "camiones", "camioneta", se puede obtener la raíz "camion" de esos tres términos.

El stemming está relacionado con el concepto de la lingüística *lematización*, donde se reduce un término a su *lexema*. Un lexema es la parte que se mantiene invariable en todas las palabras de una misma familia; Lo que no significa que esta parte tenga un significado en sí. Posee un significado referencial, es decir, aporta a la palabra una idea distinguible del significado.

Por cada lexema hay una forma de la palabra, llamada *lema*. Un lema es una unidad semántica que puede constituir una entrada en el diccionario, es decir, posee un significado en si misma. Por ejemplo, el lema "correr" es elegido para representar en el diccionario a las palabras "corriendo", "corredor", "corren", etc. El lexema de estas palabras sería "corr".

Aunque se suele pensar que la técnica de stemming y la lematización son los mismo, esto no es así. El stemming es una técnica puramente operacional del proceso de extracción de información, no tiene en cuenta que la transformación sea válida lingüísticamente. Simplemente reduce un conjunto de palabras a su stem o raíz común, y así aumenta la frecuencia de aparición de las palabras en el documento, debido a que no se considera la palabra completa para la búsqueda, sino su raíz. En cambio, la lematización es un proceso lingüístico que consiste en, dada una forma flexionada (es decir, en plural, en femenino, conjugada, etc) hallar el lema correspondiente. Por lo tanto, no son sinónimos.

El stemming presenta, además de la inevitable ambigüedad semántica, dos problemas básicos: el overstemming (reducir demasiado, representando con un mismo stem formas que deberían ser representadas con varios, por ejemplo *cas-* para casa y caso) y el understemming (reducir poco, al obtener stems distintos para formas que se corresponderían únicamente con uno, por ejemplo con los stems *computaci-* y *computa-* si el stem fuera únicamente *computa-* cambiaría el análisis).

Uno de los algoritmos más usados para realizar el stemming en el idioma inglés es el *algoritmo de Porter* y es el utilizado en esta tesina.

El algoritmo de Porter actúa como un autómata de estados finitos que incluye un grupo de reglas que se emplean para eliminar terminaciones morfológicas y flexivas de palabras. Su idea básica era la reducción del plural al singular para normalizar

los términos, por lo que básicamente eliminaba las "s" finales. Este algoritmo fue desarrollado para el idioma inglés pero ya se han realizado adaptaciones para otras lenguas como el español, el portugués o el francés, que ajustan lo más posible a las reglas del idioma en concreto (Porter, 1980).

Este algoritmo puede parecer una técnica demasiado agresiva, por una cuestión de overstemming. Su manera de aplicar stemming es aplicar listas de reglas en una secuencia de pasos. Por ejemplo, la primera lista de reglas en el primer paso del algoritmo para el idioma inglés transforma las palabras plurales de la siguiente manera:

- sses $\rightarrow$ ss
- ies $\rightarrow$ i
- ss $\rightarrow$ ss
- s $\rightarrow$

Para que se aplique esta regla el sufijo de la palabra debe coincidir con el lado izquierdo de la regla. Si lo hace, se reescribe el sufijo reemplazando los caracteres del lado izquierdo por los caracteres del lado derecho.

La regla se aplica en la primera coincidencia, por lo que si una palabra posee un sufijo que coincide con la primera regla, ese sufijo se sobreescribe pero no sigue recorriendo las otras reglas. En general el algoritmo cuenta con nueve listas de reglas, donde la más larga tiene veinte reglas.

Es un proceso de stemming, por lo cual las raíces de las palabras que da como resultado no son necesariamente una palabra con significado, sino caracteres del comienzo que comparten una familia de palabras. Esto no es un problema, ya que las raíces de las palabras nunca llegan a la vista del usuario, sino que sirven para mejorar los resultados de análisis posteriores (Büttcher, Clarke y Cormack, 2010).

1.5. Conclusión

En este capítulo se han descripto los pasos que deben realizarse para extraer conocimiento a partir de información almacenada. La Minería de Datos constituye, sin lugar a dudas, la parte más conocida de este proceso reuniendo un conjunto de técnicas capaces de identificar patrones y relaciones subyacentes en los datos.

Es en este proceso en el que se enmarcan las tareas desarrolladas a lo largo de esta tesina aunque por tratarse de texto, en lugar de Minería de Datos es preciso referirse a estrategias pertenecientes al área de la Minería de Textos. Nótese que el proceso de KDD es lo suficientemente genérico como para no verse modificado por el tipo de dato a tratar. Son las etapas de preprocesamiento que generan la vista minable y eventualmente la técnica a utilizar los aspectos que se modifican.

Antes de iniciar las tareas de preprocesamiento se definieron conceptos relacionados con la colección de documentos, los tipos de documentos y la estructura de un documento en particular. Luego, se habló del tratamiento del texto previo a cualquier tipo de análisis. Se mostró la necesidad de aplicar distintos procesos sobre el texto para obtener la mayor cantidad de información de los documentos, tanto explícita como implícitamente.

A continuación, en el capitulo 2, se dará mayor detalle sobre los dos grandes tipos de resúmenes que se pueden construir. En particular se trabajará sobre la construcción de un resumen extractivo, es decir, de un resumen formado por las oraciones más representativas del documento. Para generarlo es preciso ponderar la importancia de las oraciones y luego establecer un criterio de selección. En el capítulo 2 se revisan las técnicas más utilizadas de la literatura ya que luego serán un insumo importante para el método propuesto en esta tesina.

Capítulo 2

Resumen extractivo

2.1. Introducción

A la hora de resumir automáticamente un documento existen distintas alternativas que pueden aplicarse. En este capítulo se desarrollarán los diferentes tipos de resúmenes automáticos, haciendo hincapié en la clasificación más utilizada: extractivos y abstractivos. Dentro de esta clasificación se abarcará con mayor detalle el tipo de resumen extractivo dado que fue el utilizado en el método propuesto de esta tesina.

Luego, se continúa con la problemática de guardar los documentos de texto de manera estructurada. Esto permite recuperar el documento fuente y realizar los cálculos deseados sobre los datos sin necesidad de volver a manipular el texto original. Algunos de estos cálculos se presentan en la segunda mitad del capítulo, detallando los utilizados en el método propuesto.

2.2. Tipos de generación de resúmenes

Hay muchas maneras de clasificar las distintas formas de generación de resúmenes. Generalmente se categorizan basados según la cantidad de documentos de entrada (puede ser un único documento o múltiples), según el propósito (resúmenes genéricos, con un dominio específico o basado en consultas particulares) o el tipo de resumen a generar (resúmenes extractivos o abstractivos).

El tipo de *resumen de un solo documento* se utiliza en general para artículos científicos, historias, transcripciones de emisiones de radio o televisión, conferencias o clases. En cambio, el *resumen de múltiples documentos* surge de la necesidad de manejar la redundancia de información en la web. Este tipo de resumen es clave para

organizar los resultados de las búsquedas reduciendo considerablemente el tiempo de las mismas, especialmente cuando el usuario desea obtener la mayor cantidad de información posible. Por ejemplo, cuando se busca información sobre un evento, aparecen muchísimos resultados pero con esta metodología se puede realizar un resumen de un conjunto de artículos de noticias que hablen sobre el evento. La idea principal es obtener la información más importante del evento de las distintas páginas y agruparla sin necesidad de acceder a cada una de ellas. Aunque el usuario pueda luego acceder a las páginas fuente si así se lo desea (Nenkova y K. McKeown, 2012).

En la clasificación según el propósito, el objetivo de los *resúmenes genéricos* es resumir los documentos sin darle importancia a la temática o dominio de los mismos; ve a todos los documentos como textos homogéneos. La mayoría de los trabajos realizados con resúmenes automáticos utilizan este tipo. Por otro lado, se desarrollaron sistemas de resúmenes para un dominio de interés en particular, *resúmenes por propósito*, donde la importancia está puesta en resumir en base al tema de interés del lector. Por ejemplo, vandalismo online (Wang y K. R. McKeown, 2010), investigación biomédica (Naderi, 2012), eventos terroristas (K. McKeown y Radev, 1995), entre otros. Este tipo requiere de un conocimiento previo de los tópicos para el proceso de selección de sentencias que irán al resumen. Dentro de la clasificación por propósito, se encuentran los resúmenes basados en consultas externas, los cuales solo contienen información requerida por el usuario explícitamente. Estas consultas son en general preguntas en lenguaje natural o palabras claves relacionadas con un tema en particular, que el resumidor debe conocer (Kumar, Goh y col., 2016).

La diferenciación más común es si un resumen es extractivo o abstractivo. El primer caso consiste en una estrategia en donde se seleccionan y concatenan oraciones del texto original, en cambio la estrategia abstractiva involucra la creación de nuevas oraciones. La generación de resúmenes automáticos abstractivos requiere cambios en el texto, ya sea eliminando, modificando o sustituyendo partes. Generar un resumen con una narrativa coherente es una tarea sumamente compleja (Kupiec, Pedersen y Chen, 1995; Hahn y Mani, 2000). A continuación se pasan a desarrollar con mayor detalle cada uno de ellos.

2.2.1. Resúmenes extractivos

Un resumen extractivo se crea a partir de la selección de unidades textuales del texto original, las cuales se copian al texto destino dando lugar así al resumen. Las unidades textuales que se extraen pueden variar su longitud; pueden ser palabras, frases, oraciones o incluso párrafos completos. Generalmente, se utilizan oraciones, ya que tienen significado auto contenido y tienen una longitud intermedia como para poder combinar varias.

El proceso de selección implica una evaluación y una asignación de puntaje a las oraciones para poder identificar el conjunto más relevante. Esta etapa es fundamental debido a que los mecanismos utilizados para evaluar y puntuar a las oraciones es en lo que se diferencian la mayoría de los sistemas extractivos.

Este tipo de resumen no garantiza una narrativa coherente, pero es muy útil para identificar lo más relevante del documento (Kupiec, Pedersen y Chen, 1995). Una vez seleccionadas, las oraciones se copian al resumen en el orden en el que se encontraban en el documento original para mantener la coherencia de la narrativa lo más posible (Litvak y col., 2010).

Una de las razones por las que es beneficioso utilizar resumen extractivo automático es que el mismo no se ve influenciado por el conocimiento u opinión del autor. Debido a esto, el resumen extractivo es producto de un análisis estadístico de las propias palabras del autor del documento, lo que lo hace fiable y consistente en cuanto al contenido (Luhn, 1958).

2.2.2. Resúmenes abstractivos

Determinar qué oraciones de un texto son significativas para formar parte del resumen, es una tarea que se puede realizar utilizando simplemente métodos extractivos, pero para obtener un resumen óptimo en términos de calidad de contenido y lingüística, se debe utilizar un enfoque abstractivo.

Cuando una persona realiza un resumen, selecciona partes del documento y las corta según lo que considere importante. Reutiliza lo restante, combinándolo o reescribiendo a una versión concisa del texto original. Para realizar resúmenes abstractivos se utilizan métodos semánticos y métodos de generación de resúmenes.

Dentro de los métodos semánticos se encuentran, por ejemplo, *los basados en ontología*. Este se basa en el conocimiento subyacente para mejorar los resultados de la

generación de resúmenes mediante la ontología. La ontología definida por Gruber (Gruber, 1993) *"...es una especificación explícita de algunos tópicos. Es una representación formal y declarativa, la cual incluye el vocabulario (o los nombres) para hacer referencia a los términos que se encuentran dentro de un área específica y las declaraciones lógicas que describen qué términos son y cómo se relacionan entre ellos."*

Esencialmente, la ontología sirve para descomponer un tema en varios objetos que lo describen; permite obtener la información semántica de los tópicos. La forma de describir estos objetos y los formalismos para su representación dependen de las aplicaciones individuales (Wu y C. Liu, 2003). Se utiliza la representación para darle mayor peso a las oraciones que poseen términos que pertenecen a la ontología utilizada. Esta técnica puede ser muy útil cuando los documentos tienen un tema específico de dominio donde se pueden identificar fácilmente los conceptos clave dentro del mismo. El mayor problema es la disponibilidad de la ontología en sí misma. En la mayoría de los casos la ontología es construida manualmente por expertos en el tema de dominio; no hay ontologías generales definidas para cada temática (Kumar y Salim, 2012).

Otro método semántico son *las técnicas de reordenamiento de sentencias*. El orden en que la información es presentada al lector influye críticamente en la calidad de un resumen. En los resúmenes escritos manualmente, las sentencias no siempre mantienen el orden del documento original. Aunque en los resúmenes de un solo documento se puede preservar el orden de aparición de las oraciones, en los resúmenes de múltiples documentos reordenar es una tarea muy significativa ya que las sentencias provienen de distintas fuentes (Yao, Wan y Xiao, 2017).

Existe otra metodología que incluye la compresión y fusión de las sentencias. La *compresión de sentencias* tiene como propósito acortar las oraciones seleccionadas para formar parte del resumen, permitiendo que se pueda tener una mayor cantidad de oraciones, y por lo tanto, abarcar más contenido del documento (Nenkova y K. McKeown, 2012). El problema de este enfoque es que, al quitar palabras, se pueden generar oraciones gramaticalmente incorrectas, lo cuál afecta directamente al resumen (Vanderwende y col., 2007). Aunque la compresión da como resultado sentencias que no son idénticas al texto de entrada, las diferencias son mínimas, ya que a lo sumo, la nueva sentencia es un subconjunto de palabras de la sentencia original. Los resúmenes realizados por humanos poseen oraciones que son reformulaciones del texto de entrada o una combinación de varias sentencias reescritas en

una que contiene la información más relevante de ambas. De esto se trata la *fusión de sentencias*, unir dos o más oraciones parecidas en una única oración con el texto más representativo (Nenkova y K. McKeown, 2012).

2.2.3. Resumen extractivo vs. resumen abstractivo

Puede decirse que, decidir entre resumir de manera extractiva o abstractiva es equivalente a decidir entre seleccionar oraciones en un texto o reescribir lo que se entiende de la lectura. Resumir de manera extractiva es como resaltar el texto formando el resumen con lo marcado y de manera abstractiva sería transcribir lo que se entiende del texto marcado. Vale aclarar que para poder realizar un resumen abstractivo primero se deben usar técnicas extractivas y luego se realiza el análisis y transformación del texto.

Es posible que el resumen abstractivo resulte más atractivo por su similitud con la manera en que resume el lector. Sin embargo, su generación es más compleja ya que requiere del análisis del contenido del documento. El principal problema es la etapa de representación. Los sistemas están limitados por la gran cantidad de representaciones que se pueden realizar y la capacidad de estas para generar sistemas que no agreguen al resumen lo que su representación no tiene que agregar. En dominios muy restringidos, puede concretarse la elección de estructuras apropiadas, pero una solución de propósito general depende de un análisis semántico de dominio general (L. Liu y Özsu, 2009).

Por su parte, los resúmenes extractivos pueden incluir detalles innecesarios, ya que una vez que el método determina que una oración contiene información importante, la incluye completa en el resumen. Esto no sucede con las técnicas abstractivas, ya que en general reformulan el texto fuente.

Realizar resúmenes puramente extractivos, en general, da mejores resultados que hacerlo de manera abstractiva. Esto se debe a que los problemas del abstractivo, como representación semántica o la generación de lenguaje natural, son más difíciles de resolver que la extracción de oraciones (Erkan y Radev, 2004). Sin embargo, cuando se trata de realizar un resumen sobre varios documentos, se ha observado que los resúmenes abstractivos abarcan mejor la temática, así como también cuando se trata de textos con múltiples opiniones contrarias. En ambos casos, los resúmenes extractivos suelen tener un sesgo hacia un documento o a una opinión, según el caso (Carenini, Chi y Cheung, 2008).

La presente tesina se desarrolló en base a los resúmenes automáticos extractivos, por lo que en las siguientes secciones se explicarán de manera más detallada el proceso de resumir y los conceptos relacionados con esta categoría.

2.3. El proceso de resumir texto automáticamente

Spark Jones propone el siguiente modelo tripartito como modelo de resumir automáticamente (Sparck Jones, 1999): (I) crear una representación intermedia para el texto original (*interpretación*), (II) valorar cada una de las sentencias a través de un puntaje (*transformación*) y (III) seleccionar la información que formará parte del resumen (*generación*).

El modelo de la figura 2.1 ilustra los pasos que se realizan generalmente para hacer un resumen.

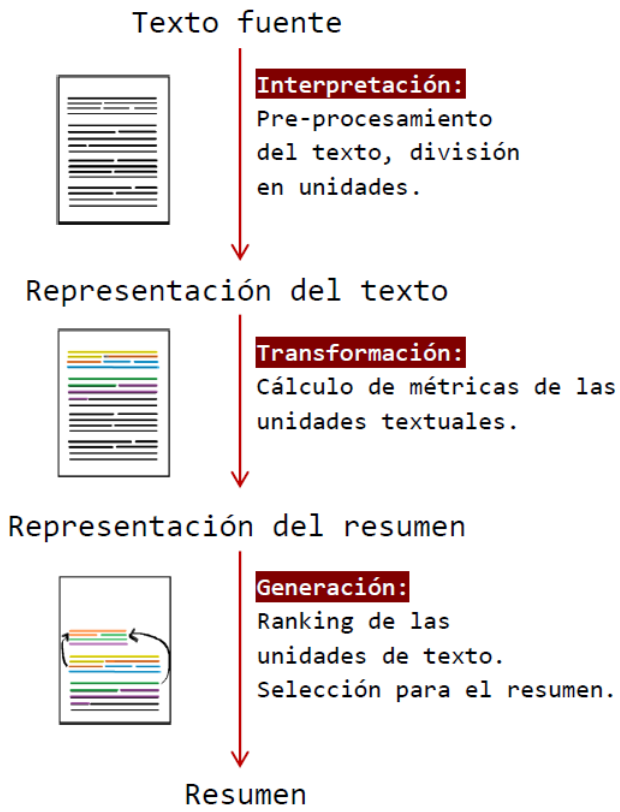


FIGURA 2.1: Modelo del proceso de resumen extractivo.

Dentro de la etapa de *interpretación* se realiza: la extracción del texto original completo, las modificaciones estructurales necesarias y se lo guarda de la manera más adecuada para poder procesarlo en la siguiente etapa (Jones, 2007).

Esta etapa comienza con las tareas de preprocesamiento detalladas en la sección 1.4.2. Como se mencionó anteriormente, muchas de las decisiones tomadas en la etapa de preprocesamiento de los documentos influyen en el cálculo de las medidas de las oraciones. No es correcto comparar resúmenes automáticos realizados con distintos métodos para calcular las métricas ya que la forma de extracción del texto del documento y su posterior guardado en una estructura dependen de muchas variables.

En el presente trabajo se utilizaron documentos estructurados de tipo XML: *Lenguaje de marcado extendible* o Extensible Markup Languaje. En el anexo A se describe de manera detallada la estructura de los mismos y la forma de recuperación de la información desde ese tipo de documentos.

En este punto en el que se dispone de los términos que componen el documento por separado, así como la identificación de las oraciones que lo forman, es conveniente definir una forma de almacenamiento que resulte adecuada para su posterior procesamiento. Dado que este aspecto no condiciona los resultados del método propuesto en esta tesina sino que sólo tiene por objetivo reducir el tiempo de cómputo, su explicación detallada ha sido realizada en el anexo B.

Una vez almacenada toda la información se debe determinar cuáles son las mejores porciones de un documento para que formen parte del resumen. Esto requiere una medida por la cual el contenido de todas las partes pueda ser comparado y ordenado. Un valor puede ser asignado a cada unidad textual de acuerdo con el criterio de importancia (Luhn, 1958).

Cuando el documento se encuentra guardado de manera apropiada, se comienza con la etapa de *representación* de los documentos. En la misma se realiza la extracción de información de los datos y se calculan las métricas por cada una de las unidades textuales del documento a resumir.

2.4. Métricas

Como se mencionó anteriormente para poder realizar un resumen automático, es necesario ponderar numéricamente la importancia de las unidades textuales seleccionadas respecto del documento. Es necesario aclarar en este punto que en el presente trabajo se seleccionaron las oraciones de los documentos de texto como unidades textuales para formar parte del resumen.

En esta sección se desarrollarán veintiún técnicas de medición que van a servir como medida para almacenar en la base de datos. A partir de allí se podrán utilizar herramientas de la minería de datos para buscar patrones dentro de estos resultados.

Se considera D variable representativa de un documento, S_i como sentencia del documento, i un número entre 1 y n asignado a cada sentencia secuencialmente respetando el orden de aparición en el texto de principio a fin y n como la cantidad de sentencias total del documento.

2.4.1. Métricas de posición

Las métricas por posición de Baxendale (Baxendale, 1958) se basan en que las sentencias más importantes tienen una tendencia a estar muy próximas al comienzo del documento o muy próximas al final del mismo. Este tipo de medida involucra la posición de la sentencia dentro del documento como parte del cálculo de la misma. Se puede variar entre tener en cuenta la posición de la oración en una sección, en un párrafo o en todo el documento, dependiendo de qué es lo que se quiera medir.

Métrica POS-F

Esta métrica puntúa a la oración dependiendo de cuán cercana esté la misma del comienzo del documento.

$$pos\_f(S_i) = \frac{1}{i} \tag{2.1}$$

Métrica POS-L

Esta métrica puntúa dependiendo de cuán cercana esté la oración del final del documento.

$$pos\_l(S_i) = \frac{1}{n-i+1} \tag{2.2}$$

Métrica POS-B

Esta métrica puntúa a la oración dependiendo de cuán cercana esté la misma a los extremos del documento. Cuanto más cercana esté al inicio o fin del documento, mayor peso tendrá.

$$pos\_b(S_i) = max\left(\frac{1}{i}, \frac{1}{n-i+1}\right) \tag{2.3}$$

2.4.2. Métricas de longitud

Este tipo de métricas pondera la importancia de una sentencia en base a la cantidad de palabras o caracteres que posee la misma (Nobatay y col., 2001). Su aplicación, por lo general, involucra el uso de un umbral definido por el usuario para decidir cuáles serán las sentencias que formarán parte del resumen. La importancia de una sentencia es proporcional a su longitud independientemente de si se utilizan palabras o caracteres. Dado un conjunto X donde $|X|$ denota su cardinalidad, se definen las siguientes métricas:

Métrica LEN-W

Según la cantidad de palabras de una oración retornada por la función $palabras(S_i)$.

$$len\_w(S_i) = |palabras(S_i)| \tag{2.4}$$

Métrica LEN-CH

Según la cantidad de caracteres de una oración retornada por la función $caracteres(S_i)$.

$$len\_ch(S_i) = |caracteres(S_i)| \tag{2.5}$$

2.4.3. Métricas de frecuencia

Este tipo de medición se basa en la frecuencia de aparición de las palabras, dentro de la oración o la proporción respecto del documento. Algunas de estas medidas utilizan otras métricas de este tipo para realizar cálculos más complejos y obtener así otro tipo de resultados respecto de la frecuencia.

Las fórmulas que representan a continuación fueron variando a lo largo del tiempo, a medida que se encontraron mejoras para hacerlas más precisas, aunque el concepto de cada una sigue siendo el mismo.

Métrica de LUHN

Toma como factores significativos de una sentencia a la frecuencia y la posición relativa de palabras significativas dentro de una oración. Luhn es considerado pionero en resúmenes de texto automatizado. A continuación se describe la manera en que mide la importancia esta métrica.

En la publicación de Luhn (Luhn, 1958) se propone que la frecuencia de aparición de una palabra en un artículo facilita una forma de medición muy útil para determinar la importancia de una sentencia en el artículo. También se considera que la posición de la palabra dentro de la sentencia tiene influencia en su importancia. En base a estas dos medidas es que se calcula la métrica.

Primero se deben obtener las palabras que se consideran más importantes dentro del documento, a las cuales se las denominará *palabras clave*. Utilizar la frecuencia de una palabra como medida de importancia se basa en el hecho de que un escritor normalmente repite ciertas palabras a medida que avanza o varía en sus argumentos para dar énfasis a lo que quiere decir.

Este método no tiene en cuenta las implicaciones lingüísticas como la gramática o la sintaxis. En general, no diferencia entre las formas de las palabras. Por lo tanto, utiliza las raíces de las palabras (mencionadas en la sección 1.4.2). Es decir que las variantes "diferencia", "diferenciación", "diferente" y "diferencial", podrían ser consideradas nociones iguales, bajo la raíz "difer" y contar como una única palabra. Tampoco se tienen en cuenta las relaciones lógicas ni semánticas que el autor estableció al escribir. En otras palabras, lo que se genera en primer instancia es una lista de palabras ordenadas en base a su frecuencia dentro del documento. Puede parecer un método simple, pero, visto desde un punto de vista más técnico, hay pocas probabilidades de que el autor use una palabra para reflejar más de una noción y hay menos probabilidades de que un autor use muchas palabras para una misma noción (Luhn, 1958).

La figura 2.2 muestra las frecuencias de todas las palabras del texto a analizar. Para identificar la zona de interés se calcula la frecuencia promedio y la desviación estándar, denotadas como μ y σ respectivamente. Luego, las palabras a considerar serán aquellas cuya frecuencia se encuentre comprendida en el intervalo $[\mu - \sigma, \mu + \sigma]$. Las rectas c y d determinan la región de interés. Aquellas palabras cuya frecuencia se encuentre por encima de $\mu + \sigma$ se consideran con una frecuencia de aparición excesiva (por ejemplo palabras como los artículos, las preposiciones, etc.) y no contienen información significativa para identificar la sentencia. Lo mismo ocurre con las que posean frecuencia por debajo de $\mu - \sigma$, ya que serían términos demasiado específicos de la sentencia y por consiguiente estarían poco relacionados con el tema general del documento. Otra forma de identificar las palabras claves o palabras más representativas del documento es utilizando intervalos de confianza con un cierto

nivel de significación. En esta tesina se optó por la media y la desviación estándar.

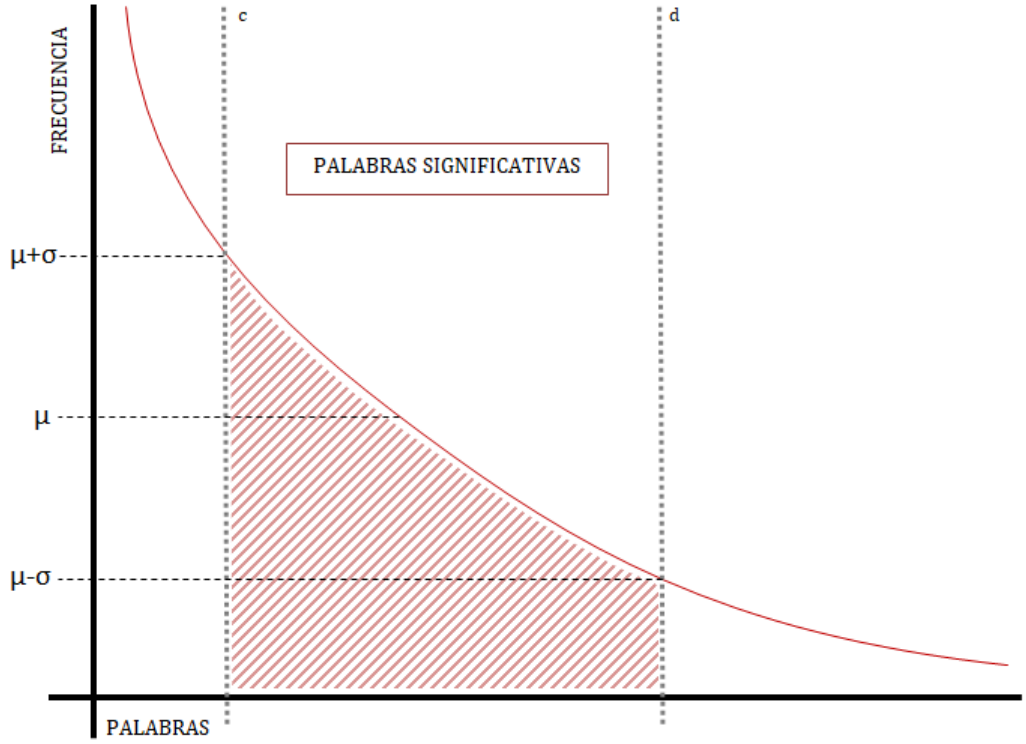


FIGURA 2.2: Diagrama de frecuencia de palabra. Rango de aceptación utilizado.

Una vez identificado el conjunto de palabras claves, Luhn propone puntuar cada sentencia según la ubicación y la cantidad de estas palabras que contenga. Cuanto mayor sea el número y más próximas se encuentren, más representativa será la sentencia para el documento en cuestión.

La importancia del grado de proximidad se basa en las características del lenguaje, en donde las ideas más relacionadas intelectualmente se encuentran desarrolladas por las palabras más próximas físicamente. Las divisiones del texto escrito en oraciones, párrafos, secciones, etc., tienen incidencia en los resultados a obtener (Luhn, 1957).

De esto se puede deducir un factor de importancia que se refleja no solo en el número de ocurrencias de las palabras clave en la sentencia, sino también en la distancia lineal entre ellas. Sea cual sea la temática, cuanto más cercanas estén las palabras, más específicamente se tratará ese aspecto. Por lo tanto, donde se encuentre el mayor número de palabras que ocurran frecuentemente con mayor proximidad física entre sí, mayor será la probabilidad de que esta información sea representativa del documento.

Debe considerarse que, el criterio es la relación de las palabras significativas entre sí en lugar de su distribución sobre toda la sentencia. Por lo tanto, parece apropiado considerar solo aquellas partes de las sentencias que están delimitadas entre la primera y última palabra clave, formando un conjunto y estableciendo así un límite para la distancia a la que cualquier par de palabras significativas se considerarán estrechamente relacionadas. A este conjunto se lo denomina rango de representatividad de las sentencias. Una palabra clave más allá de ese límite sería entonces ignorada de la consideración. Un análisis de muchos documentos ha indicado que un límite útil es de cuatro o cinco palabras no significativas entre palabras significativas. Si con esta separación resultan dos o más agrupaciones, se toma el más alto de los varios factores de significación como la medida para esa sentencia.

Desarrollo del método

- PASO 1: Obtener y almacenar en una estructura todas las raíces de las palabras del documento en cuestión.
- PASO 2: Calcular la frecuencia de cada una de las raíces y ordenar la estructura en base a lo calculado.
- PASO 3: Eliminar de la estructura las raíces con baja frecuencia. En este trabajo se consideró como baja frecuencia un valor de dos o menor a dos.
- PASO 4: Se establece una cota superior y una inferior, obteniendo así un rango de aceptación. Todas las raíces que se encuentren dentro del rango son consideradas palabras clave, por lo que permanecen en la estructura; las que no se encuentren dentro, se eliminan.
- PASO 5: Se analiza para cada oración del documento las raíces de las palabras, si estas se encuentran o no entre las palabras clave y el conjunto de palabras a considerar según la proximidad de las palabras claves dentro de la oración. Por cada raíz de la oración que se encuentre dentro del rango de representatividad y de la estructura de palabras clave, se aumentará el contador de palabras clave de esa sentencia. Paralelamente, se contabiliza también la cantidad de raíces de la oración. Una vez obtenidos ambos valores, se realiza el cálculo de la métrica; se divide la cantidad de raíces que son palabras clave elevada al cuadrado, por el total de raíces de la oración.

Al finalizar el proceso, el valor de importancia calculado para cada oración es el valor del método Luhn. En el algoritmo 1 se puede observar el proceso de asignación de pesos a las sentencias.

Algoritmo 1: Algoritmo de actualización de la métrica Luhn

Data: *keywords* = conjunto de palabras clave
raices = lista de raíces de un documento
oracion_metrica = matriz de cada oración del artículo por las métricas
Result: Actualización de la métrica Luhn para cada oración del artículo

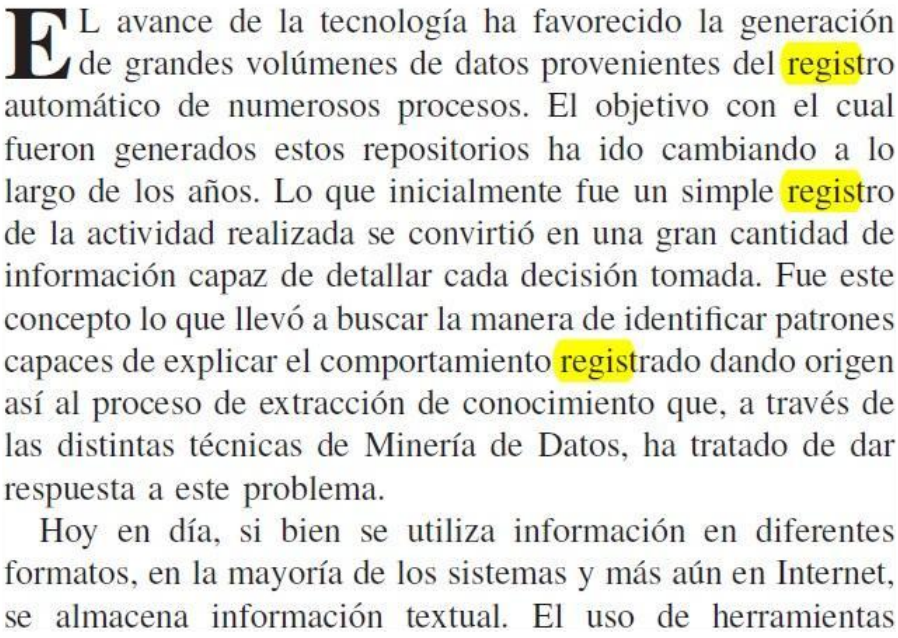
```
begin
    luhn[ ] = vector de contadores en 0
    while raices ≠ ∅ do
        id_oracion = raices.id_oracion
        total_palabras_oracion = 0

        while (raices ≠ ∅)and(id_oracion == raices.id_oracion) do
            raiz_texto = raices.texto
            if raiz_textoinkeywords then
                luhn[id_oracion]++
            total_palabras_oracion++
            raices.next()

        met_luhn = pow(luhn[id_oracion], 2)
        oracion_metrica[id_oracion][luhn] = met_luhn/cant_pal_oracion
```

Ejemplificación: Para ejemplificar se utilizará un fragmento del artículo "Document Summarization using a Scoring-Based Representation" (Villa-Monte, Lanzarini y col., 2016).

Cuando se habla de raíces, este es un buen ejemplo. Como se puede observar en figura 2.3, hay dos palabras diferentes resaltadas, `registro` y `registrado`; si se contaran las palabras, el valor para la palabra "registro" sería 2 y para la palabra "registrado" sería 1. Pero en LUHN lo que cuenta son las raíces de las palabras por lo que la palabra clave que se obtiene es "regis", esto hace que los valores se sumen y quede valor 3 para esta raíz. Cuando se realice el cálculo de cada oración, las oraciones 1, 3 y 4 van a tener 1 o más en el contador de palabras clave.

Métrica de palabras clave (KEY)

En esta métrica se calcula para cada sentencia la sumatoria de las frecuencias de las palabras clave que contiene la misma. Siendo tf_j la frecuencia de la palabra clave



EL avance de la tecnología ha favorecido la generación de grandes volúmenes de datos provenientes del registro automático de numerosos procesos. El objetivo con el cual fueron generados estos repositorios ha ido cambiando a lo largo de los años. Lo que inicialmente fue un simple registro de la actividad realizada se convirtió en una gran cantidad de información capaz de detallar cada decisión tomada. Fue este concepto lo que llevó a buscar la manera de identificar patrones capaces de explicar el comportamiento registrado dando origen así al proceso de extracción de conocimiento que, a través de las distintas técnicas de Minería de Datos, ha tratado de dar respuesta a este problema.

Hoy en día, si bien se utiliza información en diferentes formatos, en la mayoría de los sistemas y más aún en Internet, se almacena información textual. El uso de herramientas

FIGURA 2.3: Ejemplo con extracto de documento de texto de artículo científico.

j en el documento,

$$key(S_i) = \sum_{j \, \epsilon \, \text{keywords}(S_i)} tf_j \tag{2.6}$$

Desarrollo del método:

Para cada una de las oraciones se compara cada palabra con el conjunto de palabras clave del documento. Si la palabra está en el conjunto, a la oración S_i se le suma la frecuencia de la palabra clave respecto de todo el documento. Finalmente, se ordenan las oraciones de mayor a menor según su peso. Esta medida fue desarrollada por Edmundson (Edmundson, 1969).

En este trabajo se utilizó como conjunto de palabras clave el implementado en la métrica luhn 2.4.3.

Métrica COV

Esta métrica calcula la medida de importancia de cada sentencia dividiendo la cantidad de palabras clave de la sentencia actual con la cantidad de palabras clave total del documento. Se calcula el radio de la cantidad de palabras clave (Kallel F. J. y B., 2004).

Esta medida no se trabaja solo con las palabras de la sentencia en cuestión, sino con la sentencia respecto a todo el documento. Se mide la proporción de las palabras

clave contenidas en el documento y en cada sentencia. Por esto es la métrica de cobertura (*COV, coverage en inglés*) del documento.

COV se representa de la siguiente manera:

$$cov(S_i) = \frac{keywords(S_i)}{keywords(D)} \tag{2.7}$$

Métrica de frecuencia de término (TF)

Esta métrica calcula el promedio de las frecuencias de los términos de la oración S_i. Se basa en que si un término aparece muchas veces en un documento debe tener un mayor peso que en un documento donde aparece una cantidad de veces menor.

Para un documento D y un término t, se debe calcular la frecuencia del término tf_t en D. Una vez obtenida la misma, se calcula la medida TF para cada sentencia S_i de la siguiente manera:

$$tf(S_i) = \frac{\sum_{t\ \epsilon\ \text{keywords}(S_i)} tf_t}{\mid S \mid} \tag{2.8}$$

Es una bolsa de palabras, conocida comúnmente como *Bag of words* en inglés, si hay una palabra con frecuencia 4 va a tener 2 veces más peso esa palabra que una con frecuencia 2.

El problema con esta métrica es que se evalúan todos los términos como si tuvieran igual importancia. Esto no debería ser así, ya que, por ejemplo, en un documento de la industria automotriz, la palabra "auto" aparecerá en la mayoría de las sentencias. De esta manera el valor de TF para t = "auto" va a ser alto, pero probablemente muchas oraciones lo tengan y eso aumenta el valor de todas las oraciones por igual. Así, la palabra "auto" deja de ser un término relevante para medir importancia. Para solucionar el problema de los términos que se repiten demasiado en un documento, se utiliza la medida $TF - ISF$ (Manning, Raghavan y Schütze, 2008).

Métrica TF-ISF

Es una medida estadística que indica cuán importante es una palabra para una oración en un documento (Larocca Neto y col., 2000). Para poder definir $TF - ISF(t)$ (*term frequency - inverse sentence frequency*) es necesario introducir la medida $TF - IDF(t)$ (*term frequency - inverse document frequency*), ya que $TF - ISF(t)$ se basa en la teoría de la segunda.

La frecuencia de un término t en un documento D, denominada $TF(t, D)$, es el número de ocurrencias de esa palabra t en el documento D. Se basa en que mientras más alta sea la frecuencia, más representativo es el término t para el documento D.

La frecuencia del documento de un término t, se denota $DF(t)$ y es el número de documentos en los cuales aparece el término t.

Por ambas denominaciones se desprende que la frecuencia inversa de un documento de una palabra t, se define como $IDF(t)$ y se calcula con la siguiente fórmula:

$$IDF(t) = 1 + \frac{log|D|}{DF(t)} \tag{2.9}$$

El valor $IDF(t)$ de un término t es alto si la palabra se encuentra en algunos pocos documentos, indicando de esta manera que puede ser un término representativo del documento (Neto y col., 2000). Se busca obtener palabras con alto TF y alto IDF y esto se puede representar con la siguiente fórmula

$$TF - IDF(t, D) = TF(w, D)*IDF(t) \tag{2.10}$$

Obtener los valores de la medida $TF - ISF(t)$ se realiza de manera similar, reemplazando cantidad de documentos por cantidad de sentencias S. La comparación del término es con las sentencias de un documento y no con múltiples documentos. La hipótesis se basa en que cuantas más palabras específicas de la temática haya en una oración dada, la oración será más importante. Asigna un alto peso a un término si tiene varias ocurrencias en esa oración, pero rara vez en el documento completo.

El valor de TF-ISF es la frecuencia promedio de las palabras de la sentencia S_i. Siendo $SF(t)$ el número de oraciones en las cuales se encuentra la palabra t, el valor que se asigna a cada oración es la sumatoria del cálculo $TF - ISF(t)$ y se resuelve de la siguiente forma,

$$tf\_isf(S_i) = \sum_{\text{t } \epsilon \text{ words}(S_i)} tf_t \text{ x } isf_t \text{ con } isf_t = \frac{log|S|}{logSF(t)} \tag{2.11}$$

Ejemplificación:

Sean S_1 = "El auto circula por ruta." y S_2 = "El camión circula por autopista.", oraciones del documento D, se puede observar en la tabla 2.1 el TF-ISF de las palabras que poseen ambas oraciones tiene valor cero, lo que denota que no son palabras significativas. En cambio, las palabras como "auto", "camión", "ruta" y "autopista"

tienen un valor distinto de cero por lo que su medida de importancia será más alta.

Luego de los cálculos se realiza la sumatoria de cada columna TF-ISF, siendo este el valor resultado para la métrica de cada una de las oraciones.

Término	TF		ISF	TF-ISF	
	S_1	S_2		S_1	S_2
El	$1/5$	$1/5$	$log(2/2) = 1$	0	0
auto	$1/5$	0	$log(2/1) = 0{,}7$	0.14	0
camión	0	$1/5$	$log(2/1) = 0{,}7$	0	0.14
circula	$1/5$	$1/5$	$log(2/2) = 1$	0	0
por	$1/5$	$1/5$	$log(2/2) = 1$	0	0
ruta	$1/5$	0	$log(2/1) = 0{,}7$	0.14	0
autopista	0	$1/5$	$log(2/1) = 0{,}7$	0	0.14
				0.28	0.28

TABLA 2.1: Ejemplificación de la métrica TF-ISF.

2.4.4. Medidas de similitud

Cuando se trabaja con representaciones numéricas de documento, medir la similitud con distancia euclídea no suele arrojar buenos resultados. En su lugar, es común utilizar algunas de las siguientes métricas:

- **Similitud por superposición**: Este índice mide el nivel de intersección respecto al conjunto más pequeño. Sean A y B dos conjuntos, esta medida se define como $|A \cap B| / \text{mín}(|A|, |B|)$. La desventaja es que el valor obtenido se sobrevalora si uno de los conjuntos es muy pequeño respecto del otro.

- **Similitud jaccard:** Mide el grado de similitud entre dos conjuntos. Se define como el tamaño de la intersección de los conjuntos sobre el tamaño de su unión. El *coeficiente Jaccard* para medir similitud entre dos conjuntos A y B se define como $|A \cap B| / |A \cup B|$.

- **Similitud coseno:** Es una función trigonométrica utilizada para medir el ángulo entre dos vectores en el espacio vectorial. Este ángulo indica qué tan cercano está uno del otro. El resultado del cálculo varía entre cero y uno. Mide el coseno del ángulo entre dos vectores no nulos. Esta medida observa la orientación, no la magnitud de los vectores.

En base a estas medidas, se explicarán dos grupos de métricas. Primero las métricas relacionadas con el título que fueron las primeras en utilizarse, desarrolladas por (Edmundson, 1969). En base a éstas, se desarrollaron las métricas de cobertura, las cuales utilizan las medidas de similitud entre la sentencia y el resto del documento (Litvak y col., 2010).

Métricas de título (TITLE)

La idea principal de aplicar el método que mide en base a los títulos, es que el autor genera el título como el texto representativo del sujeto del documento. Esto vale también para título de secciones y párrafos. El método de Edmundson (Edmundson, 1969) asigna un valor positivo a una sentencia basándose en la cantidad de palabras en común que tiene con el título del documento, sección o párrafo, según se prefiera.

Las siguientes métricas están basadas en las medidas de similitud mencionadas en la sección anterior 2.4.4. Se considera a la similitud entre palabras como palabras iguales.

Sea T el título del documento y S_i una sentencia del mismo, se define al cálculo de las métricas de la siguiente manera:

$$title\_o(S_i) = \frac{|palabras(S_i) \cap palabras(T)|}{\text{mín}(|S_i|, |(T)|)} \tag{2.12}$$

$$title\_j(S_i) = \frac{|palabras(S_i) \cap palabras(T)|}{|palabras(S_i) \cup palabras(T)|} \tag{2.13}$$

$$title\_c(S_i) = \frac{\vec{S_i} \times (\vec{T})}{|\vec{S_i}| \times |\vec{T}|} \tag{2.14}$$

Métricas de cobertura (D-COV)

Las siguientes métricas miden a la sentencia de acuerdo a la similitud de la misma con el resto de las sentencias del documento $(D - S)$. Esto se basa en la idea de que mientras más contenido del documento esté en una oración, más importante va a ser esta oración para el resumen (Litvak y col., 2010). Se considera la similitud entre palabras como palabras iguales.

D-COV-O: Esta métrica mide la similitud de una sentencia S con el resto del documento $(D - S)$ utilizando la medida de superposición. La similitud se define a través de las palabras en común que posea la oración con el resto del documento

(Edmundson, 1969) (Villa-Monte, Lanzarini y col., 2016).

$$d\_cov\_o(S_i) = \frac{|palabras(S_i) \cap palabras(D - S_i)|}{\text{mín}(|S_i|, |(D - S_i)|)} \tag{2.15}$$

D-COV-J: Esta medida utiliza el coeficiente Jaccard para comparar la similitud entre dos sentencias. Cada sentencia es un conjunto de palabras únicas, por lo que la medida no varía si una palabra se repite muchas veces en el mismo documento.

Sean S_1 y S_2 las sentencias del ejemplo utilizado en TF-ISF (2.4.3), S_1 = "El auto circula por ruta." y S_2 = "El camión circula por autopista.", oraciones del documento D, se obtiene $3/(3+2+2) = 0{,}43$

$$d\_cov\_j(S_i) = \frac{\mid palabras(S_i) \cap palabras(D - S_i) \mid}{\mid palabras(S_i) \cup palabras(D - S_i) \mid} \tag{2.16}$$

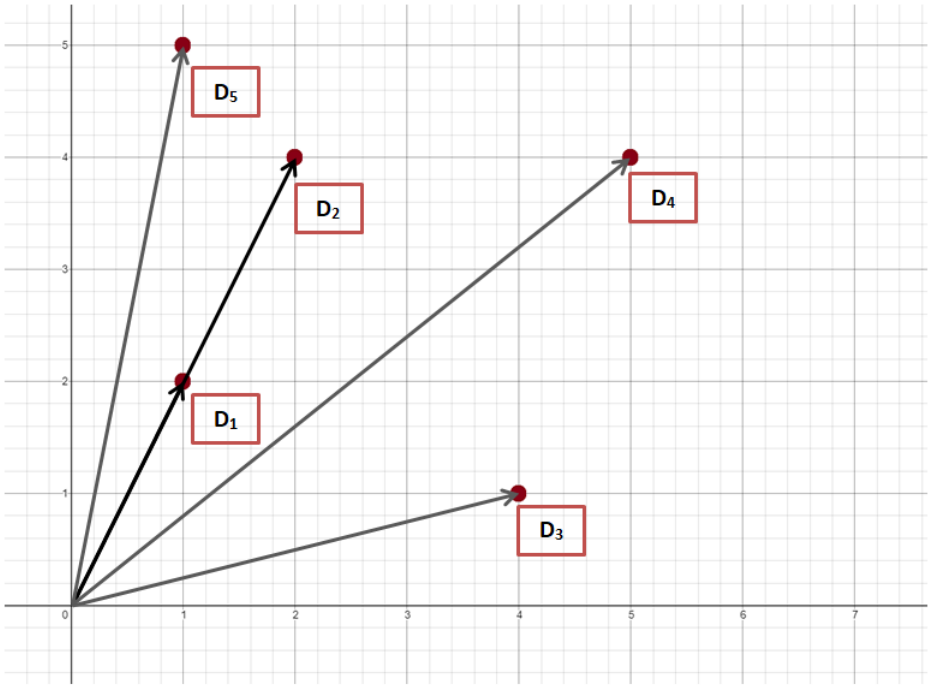


FIGURA 2.4: Ejemplo de representación vectorial con medida de similitud coseno

D-COV-C: Esta métrica, como se mencionó anteriormente, se basa en el ángulo entre dos vectores. En este caso uno de los vectores es una sentencia del documento y el otro es el resto del documento. Para poder explicar de una manera más sencilla, se optará por mostrar mediante un ejemplo cómo es que se calcula esta medida.

Si se tienen dos términos distintos, "Rosario" y "Villa María", un conjunto de documentos D_i, y se calcula cuántas veces aparece cada término en cada documento.

Si se representa lo calculado como un punto en el plano xy donde el documento 2 D_2 posee la palabra "Rosario" dos veces y "Villa María" cuatro veces, un punto (2,4) representaría este documento o, en términos de vectores, este documento sería un vector que iría del origen al punto en cuestión. Esto se puede observar en el gráfico 2.4, donde se representaron, a modo de ejemplo, cinco documentos distintos.

Como se puede ver D_1 y D_2 son documentos similares, ya que sus vectores se superponen. Esto se debe a que la cantidad de referencias a las palabras "Rosario" y "Villa maría" es proporcional entre ambos documentos. Probablemente D_2 sea un texto más largo o solo repite más veces los términos y por esto es representado con un punto en (2,4).

El coseno del ángulo, es un valor entre 0 a 1 (o -1 a 1, dependiendo de cómo se contabilice) y resulta un indicativo de la similitud entre documentos. Cuanto menor sea el ángulo entre vectores, mayor será el valor del coseno (más cercano a uno) y también mayor será la similitud.

Por otro lado, se puede observar que D_3 y D_5 no tienen mucho en común, son los vectores con mayor ángulo entre ellos, por lo que el valor del coseno del ángulo será cercano a cero y la similitud tiende a ser nula. Sea D el documento y S la oración,

$$d\_cov\_c(S_i) = \frac{\vec{S_i} \times (D \vec{-} S_i)}{| \vec{S_i} | \times | D \vec{-} S_i |} \tag{2.17}$$

2.4.5. Métricas basadas en grafos

Este es un conjunto de métricas donde se presentan a los documentos como grafos para realizar los cálculos.

Primero se debe representar al documento como grafo, donde los nodos representan una unidad textual del documento, elegida según conveniencia.

Un nodo puede tener enlaces a otros nodos dependiendo de la similitud entre ellos, es decir, si la unidad elegida es la oración, dos nodos estarán interconectados si las sentencias tienen palabras en común. A su vez, cada enlace tiene un valor numérico denominado *peso*, que es la cantidad de palabras que comparten las dos oraciones. Como se puede observar en la figura 2.5, los enlaces representan la existencia de palabras en común entre sentencias y el peso de cada enlace es la cantidad de palabras compartidas.

En la presente tesina se utilizaron grafos no dirigidos y las sentencias como unidades textuales representadas por los nodos del grafo. Esta representación permitió

utilizar distintas heurísticas basadas en grafos, las cuales se desarrollan a continuación.

Métrica GRAPH-E

El grado de un nodo se define como el número de enlaces que el nodo posee. Esta métrica se basa en que, cuantos más enlaces tenga un nodo, mayor es la incidencia de la oración (nodo) en el documento (grafo). Por consiguiente, los nodos con menor número de enlaces muestran su poca incidencia en el documento.

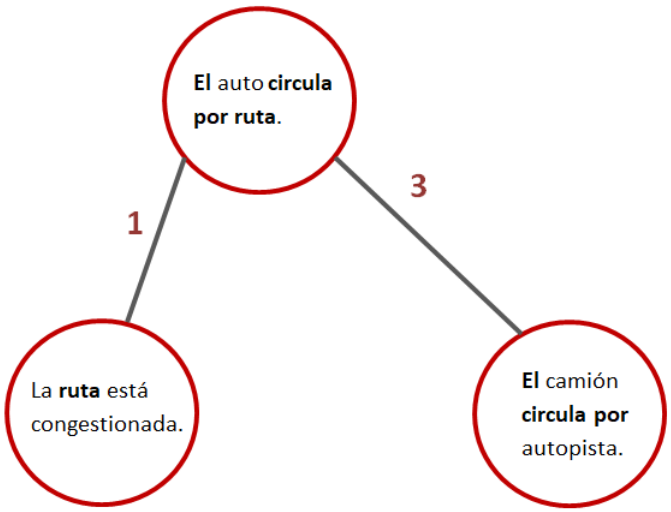


FIGURA 2.5: Ejemplo de representación basada en grafos.

Métrica GRAPH-S

Como se mencionó anteriormente, los enlaces entre nodos poseen un peso. Este valor es determinado por la similitud entre los nodos participativos. Esta métrica se basa en sumar todos los pesos de los enlaces que tiene el nodo en cuestión.

LUHN-DEG, COV-DEG y KEY-DEG

Estas métricas son una extensión de las métricas LUHN (2.4.3), COV (2.4.3) y KEY (2.4.3) respectivamente, pero basadas en grafos. Se contabiliza el grado del nodo en lugar de la frecuencia de la palabra, como en las medidas originales.

Por ejemplo, en LUHN-DEG el proceso sería el siguiente:

- Se arma el grafo con las palabras como nodos, donde cada nodo tiene un enlace al nodo que contiene su palabra vecina.

- Una vez armada la topología, se contabilizan la cantidad de enlaces (grado del nodo) y las palabras de los nodos que posean mayor grado, pasan a ser palabras clave.
- En base a estas palabras se continúa con el cálculo original de la métrica 2.4.3.

Las otras dos medidas, COV-DEG y KEY-DEG, utilizan como conjunto de palabras clave al conjunto calculado en LUHN-DEG.

2.5. Conclusión

En este capitulo se presentó la clasificación de los resúmenes automáticos de texto y sus características principales. Entre ellos los más importantes, extractivo y abstractivo, fueron desarrollados con mayor profundidad.

Dentro del proceso de resumir automáticamente se encuentran las fases de *Interpretación*, *Transformación* y *Generación*. En el primer capítulo se desarrolló parte de la primer fase de este proceso: el preprocesamiento del texto. Por este motivo, en este capítulo se continuó con esa fase y con la de *Transformación*. Se realizó la extracción del texto fuente, la obtención de información de un documento y cómo persistirla. Se definió la manera de representación de los documentos en base a la caracterización de sus sentencias, utilizando un conjunto de métricas.

Una vez obtenida la representación numérica del documento correspondiente al problema que se desea resolver, se puede comenzar con la etapa de análisis de esta información. Este análisis parte de un conjunto de oraciones etiquetadas según la definición de un usuario de que formen parte o no del resumen de un documento. El hecho de tener caracterizadas todas las sentencias, etiquetadas y no etiquetadas, con medidas de importancia permite compararlas entre sí y hacer un análisis de los datos donde la semejanza de los valores de las sentencias etiquetadas con las no etiquetadas indica si las últimas serían elegidas por el mismo usuario como parte del resumen.

Como se mencionó anteriormente, dentro de la minería de datos se encuentran las herramientas capaces de realizar este análisis para la búsqueda de patrones dentro de los datos.

En el siguiente capítulo, se describirá con mayor detalle una herramienta sumamente importante dentro de esta disciplina, las redes neuronales, ya que constituyen la base del método propuesto en esta tesina.

Capítulo 3

Redes neuronales

3.1. Introducción

En base a las métricas definidas en el capítulo anterior en donde se explicó cómo calcular, para cada sentencia, un conjunto de valores numéricos capaces de ponderar su importancia dentro de un documento. Luego, ubicando cada métrica en una posición fija dentro de un vector puede transformarse cada oración en un vector numérico.

En este capítulo se presenta un modelo de la Minería de Datos que recibe como entrada un conjunto de oraciones. Un subconjunto de estas se encuentran etiquetadas según el criterio de importancia que les dio un usuario como parte del resumen del documento al que pertenecen. En base a estos ejemplos etiquetados, se pretende clasificar las restantes oraciones según lo haría ese usuario en particular. Este modelo se clasifica dentro de los sistemas de *redes neuronales*.

Primero se definen de manera general los modelos de redes neuronales y luego se pasa a desarrollar con mayor detalle el modelo utilizado en el presente trabajo.

3.2. Redes Neuronales Artificiales

Las redes neuronales artificiales (RNA) son un modelo con una estructura basada en elementos neuronales de procesamiento paralelo cuyo comportamiento global busca "emular" los sistemas neuronales de los humanos (Isasi y Galván, 2004).

Las RNA son sistemas que se encuentran dentro del campo de la *inteligencia artificial* (IA). Dentro de este campo se pueden distinguir dos grandes áreas. Una de ellas es la denominada *inteligencia artificial simbólica*, la cual se ocupa de la construcción de sistemas con características que se pueden definir como inteligentes. Donde se

establece el problema a resolver y se diseña el sistema capaz de resolverlo siguiendo esquemas prefijados por la disciplina. Frente a esta perspectiva se encuentra la otra gran área de la IA, la *inteligencia artificial subsimbólica*. En este caso no se realizan diseños de sistemas capaces de resolver los problemas, sino que se parte de sistemas genéricos que van adaptándose, aprendiendo y construyéndose hasta formar por sí mismos un sistema capaz de resolver el problema.

La perspectiva subsimbólica estudia los mecanismos físicos que nos capacitan como seres inteligentes, frente a los programas de computadoras clásicos que son simples autómatas que obedecen órdenes muy concretas.

El mecanismo fundamental que capacita a los seres vivos para la realización de tareas sofisticadas no preprogramadas directamente es el sistema nervioso. Desde este punto de vista la perspectiva subsimbólica estudia los mecanismos de los sistemas nerviosos, del cerebro, así como su estructura, funcionamiento y características lógicas, con la intención de diseñar programas basados en dichas características que se adapten y generen sistemas capaces de resolver problemas. En este campo se encuadran las redes neuronales artificiales.

Se define a las redes neuronales como colecciones de procesadores paralelos conectados entre si en forma de grafo dirigido, organizado de tal manera que su estructura permita resolver un problema. La resolución se hace en base a una aproximación en donde no es necesario tener un algoritmo definido de traducción de entradas en salidas de la red, sino que lo que se necesita es un conjunto de problemas resueltos. En base a estos, la red se adapta para producir la salida deseada de la entrada actual. La potencia de estos modelos reside en su generalidad, es capaz de hallar sus propias soluciones para problemas concretos, dándosele simplemente ejemplos del comportamiento que se pretende (Freeman y Skapura, 1993).

3.2.1. Arquitectura y funcionamiento

Los elementos individuales que forman a la mayoría de los modelos de RNA suelen denominarse nodos, neuronas, unidades o elementos de procesamiento. Cada neurona puede tener muchas entradas pero solo una salida, que se puede aplicar a muchas otras neuronas de la red.

La figura 3.1 representa el modelo de neurona donde se introduce un conjunto de entradas $x_1, x_2, \ldots, x_n$. La entrada que recibe la *j-ésima* neurona procedente de la *i-ésima* neurona se indica como x_i. Cada conexión de entrada con la *j-ésima* neurona

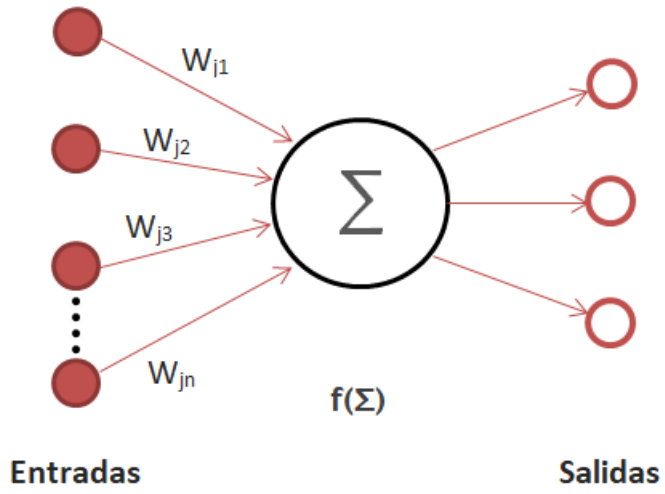


FIGURA 3.1: Esquema de una unidad de procesamiento típica.

tiene asociada una magnitud llamada peso. El peso de la conexión procedente del *i-ésimo* nodo, que llega al *j-ésimo* nodo se denota mediante w_{ji}. Si el peso entre dos neuronas conectadas es positivo, el efecto producido es de excitación, por lo cual la conexión de entrada es excitatoria. Por el contrario, si el peso es negativo, este efecto es de inhibición y la conexión es inhibitoria.

Una única neurona es una unidad de procesamiento relativamente simple; el potencial del modelo de RNA proviene de la unión de un conjunto de neuronas actuando en paralelo. En la figura 3.2 se puede observar una RNA que ejemplifica una arquitectura de cuatro capas de neuronas, donde hay un conjunto de entradas accediendo a la red desde el lado izquierdo de la misma y se propaga hasta que alcanza la salida. A las capas intermedias se las llama *capas ocultas* debido a que no son accesibles desde fuera de la red.

Resulta útil describir ciertas magnitudes de las RNA como vectores. Considerando una red neuronal compuesta por varias capas, como la de la figura 3.2, de elementos de procesamiento idénticos, una capa que contiene n unidades va a tener como salida un vector n-dimensional, $\mathbf{x} = (x_1, x_2, ..., x_n)$.

Suponiendo que ese vector n-dimensional proporciona los valores de entrada de todas las unidades de una capa m-dimensional. Cada una de estas unidades poseerá n pesos asociados procedentes de la capa anterior. De esta manera hay m vectores de pesos n-dimensionales asociados a esta capa, un vector n-dimensional de pesos para cada una de las m unidades. El vector de pesos de la *i-ésima* unidad se puede escribir como $W_i = (w_{i1}, w_{i2}, \ldots, w_{in})$ (Freeman y Skapura, 1993). Así es como cada

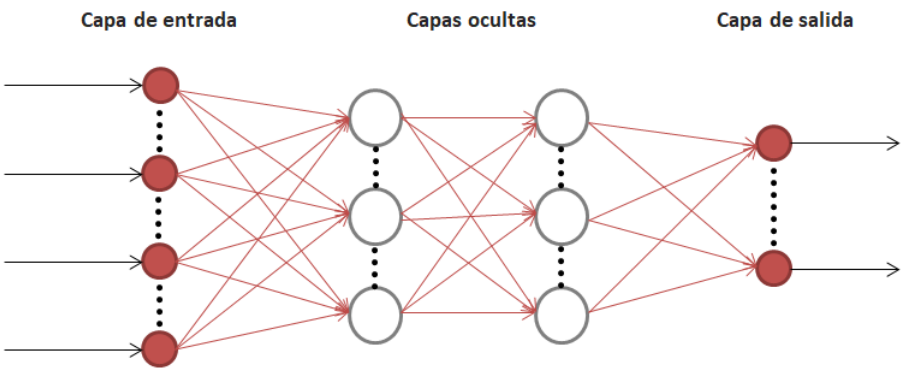


FIGURA 3.2: Ejemplo de red neuronal con cuatro capas de neuronas.

unidad determina su valor de entrada neto basándose en todas sus conexiones de entrada. En general, se calcula sumando esas entradas, ponderadas (multiplicadas) por sus correspondientes pesos. Así, la entrada neta de la *i-ésima* unidad se puede escribir de la siguiente forma

$$neta_i = \sum_j x_j w_{ij} \tag{3.1}$$

donde el índice j recorre todas las conexiones de entrada que posea la neurona. Obsérvese que la excitación y la inhibición se tienen en cuenta automáticamente mediando el signo de sus pesos.

Una vez calculada la entrada neta, se puede determinar el valor de salida aplicando la función de salida:

$$y = f(neta_i) \tag{3.2}$$

Resulta útil visualizar la colección de valores de los pesos como un sistema dinámico. Estos sistemas son los que evolucionan a lo largo del tiempo.

3.2.2. El aprendizaje en las redes neuronales

La parte más importante de una RNA es el aprendizaje, ya que es lo que determina el tipo de problema que la red será capaz de resolver. Aprenden en base a ejemplos, por lo que las RNA resolverán problemas en base a los ejemplos que se le otorguen durante el proceso de aprendizaje (Isasi y Galván, 2004).

Retomando la analogía de las RNA con las neuronas del cerebro humano, el sistema nervioso del cerebro debe poseer plasticidad para adaptarse al entorno. Esta

plasticidad es esencial para el funcionamiento de la neuronas como unidades de procesamiento de información, lo mismo sucede con las neuronas de las RNA. La capacidad de adaptación de las mismas se basa en los cambios sobre los pesos de las conexiones entre ellas y estos cambios son los que dan lugar al aprendizaje de la red.

El proceso de aprendizaje consiste en hallar los pesos que codifican el conocimiento que se desea que aprenda el sistema. En el caso de las Redes Neuronales, se utilizan procesos iterativos que inspeccionan reiteradamente los ejemplos suministrados hasta alcanzar un estado de estabilidad. Una vez alcanzado este punto, se considera que la red ha "aprendido" el comportamiento esperado. (Freeman y Skapura, 1993).

Tipos de aprendizaje

Hay dos tipos principales de procesos de aprendizaje en RNA:

- Aprendizaje supervisado: Se le proporciona a la red un vector de datos de entrada y la respuesta esperada. El conjunto de datos de entrada se propaga hasta que la activación alcanza la salida. En ese momento se compara la respuesta calculada por la red con la respuesta correcta y se ajustan los pesos para producir la adaptación de la red.

- Aprendizaje no supervisado: Como entrada solo se le proporcionan los datos de entrada y la red debe auto-adaptarse, sin conocer la respuesta esperada. Para poder realizar este proceso la red debe aprender de algún tipo de estructura dentro de los datos proporcionados. Esta estructura suele deberse a la redundancia o agrupamiento en el conjunto.

Tiempo de aprendizaje

El tiempo de aprendizaje de la red no puede ser conocido a priori. No se puede determinar el número de veces que será necesario introducir todo el conjunto de datos para que la red aprenda. Se puede determinar utilizando alguno de los siguientes criterios:

- Mediante un número fijo de ciclos. Se decide arbitrariamente cuántas veces será introducido el conjunto de datos y una vez alcanzado el número de ciclos, se da por aceptada la red resultante.

- Cuando el error desciende por debajo de un número establecido. Este criterio necesita que se defina previamente una función de error y un valor aceptable para dicho error. El proceso de aprendizaje se detiene cuando la RNA produce un valor de error por debajo del prefijado. Esto puede ser un problema, ya que la red puede no bajar nunca de ese valor de error preestablecido, para esto es necesario tener un método adicional de parada. Se puede utilizar como segundo criterio, una cantidad de ciclos.

- Cuando la modificación de los pesos sea irrelevante. Este esquema se utiliza en el tipo de modelo que hace que las conexiones vayan modificándose cada vez con menor frecuencia. Llegará un momento que ya no se producirán variaciones en los pesos y ahí es donde finaliza el proceso de aprendizaje.

La finalidad del proceso de aprendizaje no es que la RNA prediga bien los ejemplos etiquetados, sino que si a la red le ingresan datos no etiquetados se obtenga una salida correcta. Para poder determinar si la red produce salidas adecuadas, se divide al conjunto de datos en dos: por un lado *el conjunto de entrenamiento* y por otro *el conjunto de validación*.

El primero se utiliza para adaptar los pesos mediante el proceso de aprendizaje descripto anteriormente, pero para medir el error se utiliza el conjunto de validación. Así es como, para medir la eficacia de la red para resolver el problema, se utilizarán datos con los que no se entrenó a la misma. Si el error es pequeño, queda garantizada la capacidad de generalización de la red.

En la siguiente sección se describirá con detalle el modelo de Red Neuronal utilizado en el método propuesto en esta tesina. Se trata de una arquitectura mínima formada por una única neurona por lo que se comenzará analizando la arquitectura más simple que presenta esta característica, el Perceptrón, para luego pasar a describir la utilizada en este documento.

3.3. Perceptrón

Los primeros estudios sobre redes neuronales datan de los años 50, con la aparición del perceptrón, el cual fue inventado por Frank Rosenblatt (Rosenblatt, 1958) en su intento de "ilustrar" algunas de las propiedades fundamentales de los sistemas inteligentes en general.

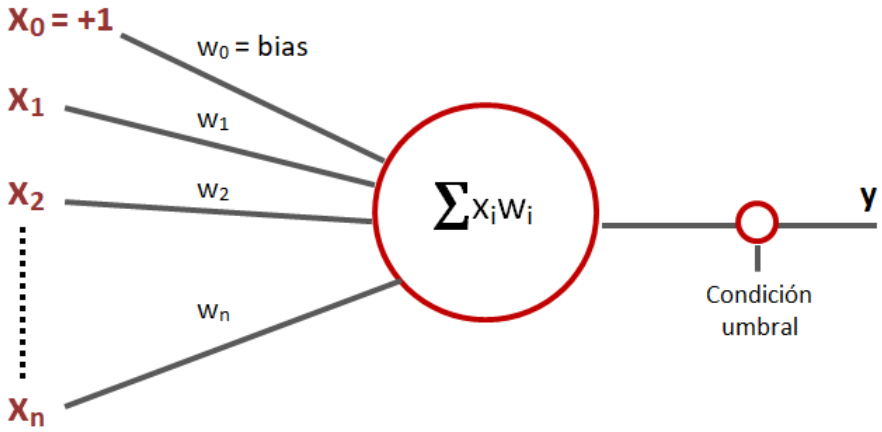


FIGURA 3.3: Arquitectura de un perceptrón.

Este modelo se concibió como un sistema capaz de realizar tareas de clasificación de forma automática. Es un dispositivo de aprendizaje, por lo que en su configuración inicial no es capaz de clasificar ejemplos pero al final del proceso el sistema puede determinar para cualquiera de ellos a qué clase pertenece.

Una de las características principales es que para que el perceptrón pueda clasificar los ejemplos, las clases deben ser linealmente separables (Freeman y Skapura, 1993).

Se dice que dos conjuntos A y B son **linealmente separables** en un espacio $n-$ *dimensional* si existen $n+1$ números $w_1, w_2, \ldots, w_n, \theta$ tal que cada punto $(x_1, x_2, \ldots, x_n) \epsilon A$ satisface $\sum_{i=1}^{n} w_i x_i \geq \theta$ y cada punto $(x_1, x_2, \ldots, x_n) \epsilon B$ satisface $\sum_{i=1}^{n} w_i x_i < \theta$.

3.3.1. Arquitectura del modelo

La arquitectura de la red es simple, se trata de una estructura monocapa en la que hay un conjunto de células de entrada y una o varias células de salida.

En la Figura 3.3 se puede observar una neurona con $n+1$ entradas $(x_0, x_1, x_2, \ldots, x_n)$, un vector de pesos $(w_0, w_1, w_2, \ldots, w_n)$, una condición umbral y una única salida y.

La entrada x_0 es una entrada fija con valor $+1$ y el peso de conexión w_0, que la vincula con la neurona, permite calcular el valor del término independiente, de la función discriminante lineal, junto con el resto de los pesos.

En este esquema la salida de la red se obtiene de la siguiente forma, primero se calcula el total del estímulo recibido por la neurona, llamada entrada *neta*. Su valor se calcula como el producto interior, o producto escalar, del vector de entrada por el vector de pesos tal como se indica en la ecuación (3.3). Para vectores de iguales

dimensiones, el producto interior se define como la suma de los productos de los componentes de ambos vectores:

$$neta = \sum_{j=0}^{n} w_j x_j \tag{3.3}$$

De manera resumida, se puede representar a la ecuación anterior en notación vectorial en la forma

$$neta = X \cdot W \tag{3.4}$$

En el caso de tener dos entradas, se puede representar una recta en el plano en base a la siguiente ecuación:

$$w_0 + x_1 w_1 + x_2 w_2 = 0 \tag{3.5}$$

recuerde que $x_0 = 1$.

La función F, función de salida, es una función umbral que utiliza el valor de la entrada *neta* para separar los ejemplos en dos clases, según de qué lado de la función discriminante lineal se encuentren.

$$y = F(neta) = \begin{cases} 1 & \text{si } X \cdot W \geq 0 \\ 0 & \text{si } X \cdot W < 0 \end{cases} \tag{3.6}$$

De esta forma, la función F produce una salida binaria que puede ser fácilmente traducible a una clasificación en dos categorías.

La figura 3.4 ejemplifica esta situación para un problema cuyos ejemplos poseen dos atributos numéricos. De esta forma, los datos de entrenamiento podrían representarse como puntos en un espacio bidimensional y si además, dichos puntos, pertenecen a una de dos categorías, la separación de las mismas podrá hacerse mediante una recta.

Si las clases están linealmente separadas puede demostrarse que la recta discriminante existe. El perceptrón determina dicha recta en su proceso de aprendizaje mediante los datos de entrenamiento. Siguiendo este razonamiento, para un conjunto de ejemplos de entrada de dimensión 3 la función discriminante será un plano y para dimensiones mayores, será un hiperplano.

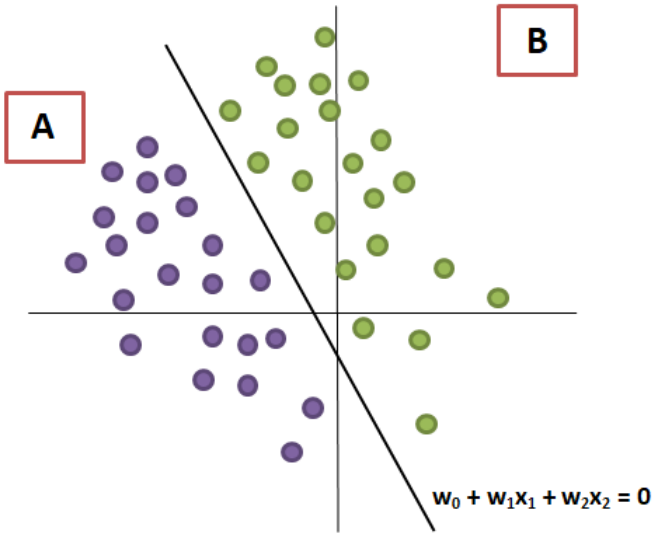


FIGURA 3.4: Separación en dos clases mediante un perceptrón

3.3.2. Regla de aprendizaje y teorema de convergencia

El proceso para determinar la ecuación de la función discriminante lineal (una recta en dos dimensiones, un plano en tres dimensiones, etc.) que clasifica los ejemplos, es el *proceso de aprendizaje del perceptrón*. Es un algoritmo iterativo en el que paulatinamente se van modificando los valores de los pesos de las conexiones, hasta encontrar los valores que determinan las ecuaciones discriminantes.

Sea E el conjunto de ejemplos a utilizar para entrenar el perceptrón de la forma $\{(X_1, d_1), (X_2, d_2), \ldots, (X_n, d_n)\}$; es decir que para cada vector de entrada X_i se conoce su valor de salida esperado d_i. Además, como el perceptrón solo puede distinguir entre dos casos posibles, los valores para d_i tienen generalmente valor 0 ó 1. Por su parte, cada X_i es de la forma $(x_{i1}, x_{i2}, \ldots, x_{in})$. Luego, la actualización realizada sobre el vector de pesos al ingresar cada uno de los ejemplos estará dada por

$$w_j = w_j + \alpha * (d_i - y) * x_{ij} \tag{3.7}$$

donde α es la velocidad de aprendizaje y corresponde a un valor real perteneciente al intervalo $(0, 1]$ e y es el valor de respuesta obtenido por el perceptrón al ingresar el ejemplo X_i. Nótese que en caso de que haya coincidencia entre el valor esperado d_i y el valor obtenido y, no se realizará ninguna modificación.

Con respecto al peso w_0 o *bias* indicado en la figura (3.3) se aplicará la misma actualización según se indica en la ecuación (3.7) utilizando como X_0 el vector nulo.

En base a lo definido previamente, el algoritmo de entrenamiento del perceptrón posee dos parámetros: la velocidad de aprendizaje α la cantidad máxima de iteraciones $MaxITE$ que se utilizarán para hallar la ecuación del hiperplano que separe correctamente los ejemplos.

El algoritmo 2 resume lo antes dicho.

Algoritmo 2: Algoritmo de entrenamiento del Perceptrón

Data: $E = \{(X_1, d_1), \ldots, (X_n, d_n)\}$, α, $maxITE$
Result: Vector de pesos W
Inicializar el vector de pesos W con valores aleatorios
$ite = 0$
repeat
  for $i = 1$ **to** n **do**
    $neta = W, X_i$
    if $neta \geq 0$ **then**
      $y = 1$
    else
      $y = 0$
  $W = W + \alpha * (d_i - y) * X_i$
  $ite = ite + 1$
until *(ite* $\geq$ *maxITE) or (el mismo W clasifique bien a todos los ejemplos)*

Teorema de convergencia

Rosenblatt demostró que si los datos que se utilizan para entrenar al perceptrón pertenecen a dos clases linealmente separables, entonces el algoritmo converge y posiciona la superficie de decisión (hiperplano) como una división entre las dos clases. La convergencia de este algoritmo es conocida como *el teorema de convergencia del perceptrón*.

Este teorema permite afirmar que si las clases son linealmente separables, es decir que existe un vector de pesos W que puede hacer que la salida y sea igual al valor esperado, la regla de aprendizaje del perceptrón llegará a una solución en un número finito de pasos para cualquier elección inicial de pesos. Es decir, podrá clasificar correctamente los ejemplos.

Dado que hay muchos problemas de clasificación que no son linealmente separables, esta condición limita bastante a la aplicabilidad del modelo (Freeman y Skapura, 1993).

Como se mencionó anteriormente, el perceptrón es un modelo de clasificación y existe un gran número de problemas que no son abordables desde esta perspectiva. La naturaleza de clasificador que tiene el perceptrón viene dada por su salida, ya que la función de salida es una función escalón, que transforma valores reales en una salida binaria. Al ser binaria solo se puede codificar un conjunto discreto de estados (Isasi y Galván, 2004).

Si las salidas fueran valores reales se podría codificar cualquier tipo de salida y se convertiría en un sistema de resolución de problemas generales. Esto no es posible con el perceptrón pero si con modelos como el de la siguiente sección.

3.4. Adaline

El término *Adaline* es la sigla de ADAptive LInear NEuron (Neurona Lineal Adaptativa); sin embargo su significado ha cambiado ligeramente con el paso de los años y pasó a ser el ADAptive LInear Element (Elemento Lineal Adaptativo).

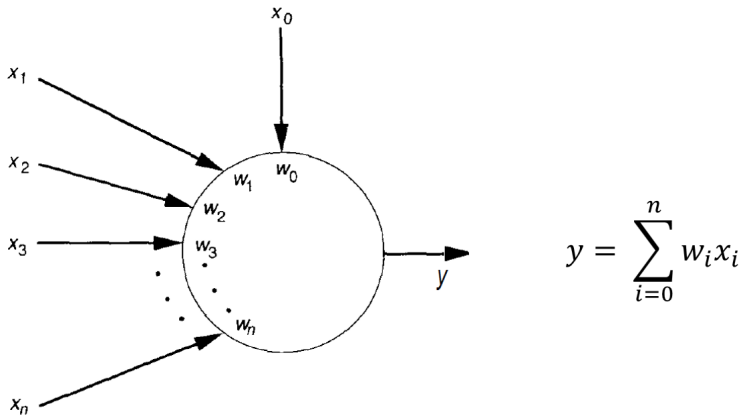


FIGURA 3.5: Combinador lineal adaptativo

El Adaline es un dispositivo que, al igual que el perceptrón, posee un único elemento de procesamiento. Su arquitectura completa consta de un combinador adaptativo lineal (ALC - Adaptative Linear Combinar 3.5) y de una función bipolar de salida. Recibe un conjunto de entradas y las combina para producir como salida una función lineal. Esta salida puede transformarse en binaria mediante el conmutador bipolar.

Su estructura es semejante a la de una unidad de procesamiento general, salvo por dos características. La primera consiste en añadir una conexión de peso w_0 a la cuál se la suele llamar *tendencia (o bias)*. Esta conexión siempre tiene un valor de entrada igual a 1. La segunda modificación consiste en una condición bipolar a la salida de la unidad de procesamiento. Si la salida del ALC es positiva, la salida del Adaline es +1 y si la salida del ALC es negativa, la salida del Adaline es -1.

Dado que el desarrollo del presente trabajo se realizó utilizando solo el ALC de esta arquitectura, se desarrollará principalmente esa parte del funcionamiento del Adaline.

3.4.1. Combinador adaptativo lineal (ALC)

El ALC lleva a cabo el cálculo de la suma de los productos con los vectores de entrada y peso, y es quien aplica la función de salida:

$$y = w_0 + \sum_{j=1}^{n} w_j x_j \tag{3.8}$$

donde w_0 es el *peso tendencia*, W es el vector de pesos y X el vector de entradas. Identificando que $x_0 = 1$, se puede reescribir la función de la siguiente manera

$$y = \sum_{j=0}^{n} w_j x_j \tag{3.9}$$

El ALC es adaptativo en el sentido de que existe un procedimiento definitivo para calcular los pesos correctos para llegar al valor de salida esperado para la entrada dada. Este procedimiento de entrenamiento fue presentado por Widrow and Hoff como *regla de aprendizaje de mínimos cuadrados* (B. Widrow y Hoff, 1960).

3.4.2. Regla de aprendizaje mínimos cuadrados

Dado un vector de entrada X, se pueden determinar un conjunto de pesos W que dé lugar a un valor de salida concreto y. Si disponemos de un conjunto de vectores de entrada $(x_1, x_2, \ldots, x_n)$ donde cada uno posee su propio valor esperado $d(x)$, el problema estará en encontrar un único vector de pesos que pueda asociar correctamente cada vector de entrada con el valor de salida correcto. Para esto se utiliza la regla de aprendizaje de mínimos cuadrados o LMS, por sus siglas en inglés de "Least mean square".

Esta regla puede ser incorporada al propio dispositivo ALC, el cual se irá ajustando a medida que vayan cambiando las entradas y las salidas deseadas. Se realizan pequeños ajustes en el vector de pesos cada vez que se procesa una combinación entrada-salida y esto se repite hasta que el ALC acierta la salida esperada. La regla LMS es en definitiva el *proceso de entrenamiento del ALC*.

Procedimiento de la regla de aprendizaje

Sea γ una razón de aprendizaje utilizada para regular cuánto afecta cada modificación de los pesos y para determinar la estabilidad y la velocidad de convergencia del vector de pesos hacia el valor del error mínimo:

- 1: Inicializar los pesos de manera aleatoria.
- 2: Introducir a la red un vector con valores de entrada.
- 3: Calcular la salida de la red y compararla con la deseada para obtener el error cometido.
- 4: Calcular el vector gradiente para el valor actual de W.
- 5: Modificar el peso restando el valor obtenido en punto 4 ponderado por la velocidad de aprendizaje α.
- 6: Regresar al punto 2 hasta que se hayan procesado todos los ejemplos. Luego verificar el criterio de convergencia y de ser necesario volver al punto 2.

En cada iteración, los cambios al vector de pesos deberían ser relativamente pequeños, ya que, si son demasiado grandes, el vector podría no encontrar nunca el mínimo, o podría alcanzarlo solo por accidente. La misión del parámetro γ es evitar esa búsqueda sin sentido.

Como se puede observar en el punto 3 del algoritmo anterior (3.4.2), el aprendizaje incluye el cálculo de la diferencia entre el valor real y_i obtenido como salida para un ejemplo de entrada x_i y el valor que debería haber producido dicho ejemplo, es decir, su salida esperada $d(x_i)$. Esta diferencia está definida por $(d(x_i) - y_i)$. No es posible conseguir una salida exacta, pero si aproximarse minimizando las diferencias, es decir, minimizando el error cometido por la red para la totalidad del conjunto, a lo cual se le da el nombre de *error cuadrático medio*.

Al ser esta una medida de error global, la regla buscará minimizar este valor para todos los elementos del conjunto de entrada de aprendizaje. Para ello se deben modificar los parámetros de la red, es decir los pesos de las conexiones, y esto se realiza mediante la *regla del descenso del gradiente* (Isasi y Galván, 2004).

3.4.3. Gradiente Descendente

El gradiente descendente es un algoritmo de optimización utilizado para minimizar funciones continuas.

En el caso del Combinador Lineal, el error que efectivamente se busca minimizar es el que comete, en promedio, la neurona al predecir la respuesta para todos los ejemplos de entrada, elevado al cuadrado; tal como se indica en la ecuación 3.10

$$Error = \frac{\sum_{i=1}^{n} error_i^2}{n} \tag{3.10}$$

siendo $error_i$ el error cometido en el ejemplo X_i, elevado al cuadrado, tal como se indica en 3.11

$$error_i = \left(d_i - \sum_{j=0}^{n} w_j * x_{ij} \right) \tag{3.11}$$

Si bien es posible minimizar el error indicado en 3.10, su cálculo implica disponer de todos los ejemplos en forma simultánea. De ser así, siguiendo la dirección del gradiente negativo, podría calcularse una trayectoria como la representada en la figura 3.6.

Sin embargo, la técnica de minimización del error utilizando la dirección del gradiente negativo propone modificar la función objetivo y en lugar de calcularlo para la ecuación (3.10) lo calculará sobre la ecuación (3.11). Si bien la trayectoria no será tan suave como la que se observa en la figura 3.6, por tratarse de un proceso iterativo aplicado sobre una función que posee un único mínimo global, ofrecerá buenos resultados.

Calculando las derivadas parciales de la expresión del error cuadrático cometido en el ejemplo X_i se obtiene lo siguiente:

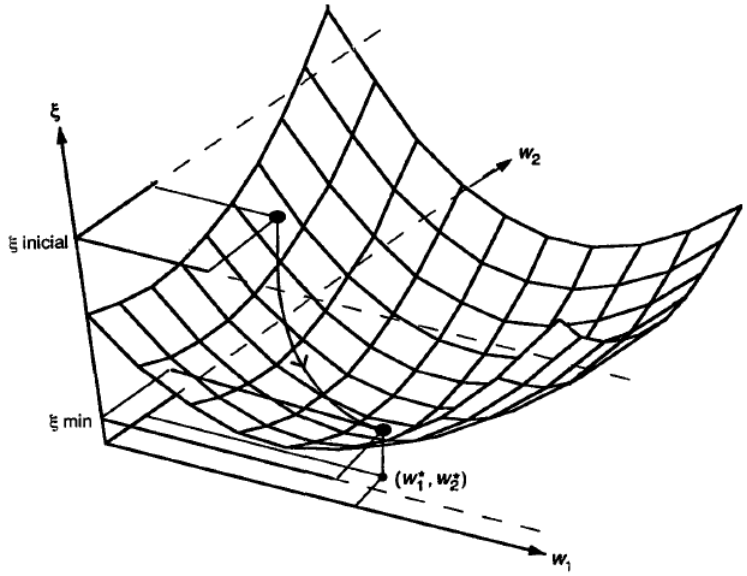


FIGURA 3.6: Minimización de la función de error utilizando todos los ejemplos de entrada (Freeman y Skapura, 1993).

$$\begin{aligned}
\nabla error_i^2 &= \frac{\partial error_i^2}{\partial W} = \left[\frac{\partial (d_i - y_i)^2}{\partial w_0}; \ldots; \frac{\partial (d_i - y_i)^2}{\partial w_n}\right] \\
\nabla error_i^2 &= \frac{\partial error_i^2}{\partial W} = \left[-2(d_i - y_i)\frac{\partial y_i}{\partial w_0}; \ldots; -2(d_i - y_i)\frac{\partial y_i}{\partial w_n}\right] \\
\nabla error_i^2 &= \frac{\partial error_i^2}{\partial W} = -2error_i\left[\frac{\partial y_i}{\partial w_0}; \ldots; \frac{\partial y_i}{\partial w_n}\right] \\
\nabla error_i^2 &= \frac{\partial error_i^2}{\partial W} = -2error_i[x_{i0}, x_{i1}, \ldots, x_{in}]
\end{aligned}$$

Por lo tanto, el vector gradiente que se obtiene al derivar de manera parcial la expresión del error cometido al utilizar solo el vector de entrada X_i queda determinado por la siguiente ecuación:

$$\nabla error_i^2 = -2 * error_i * X_i \tag{3.12}$$

Solo resta ahora, tomar la dirección de este vector gradiente para modificar el vector de pesos en cada iteración.

Dado que las derivadas parciales solo se calculan sobre la expresión de error de un ejemplo en particular, el descenso sobre la superficie de error tendrá un recorrido similar al que se muestra en la Figura 3.7. Nótese que en dicha figura el vector de pesos W es de la forma (W_0, W_1) y su valor actual se representa sobre el plano

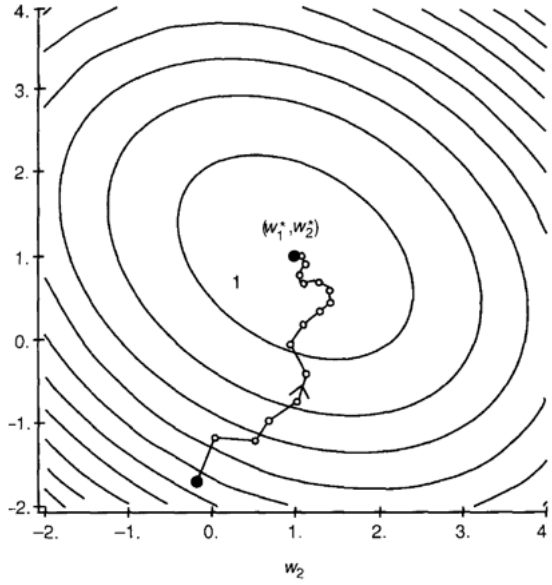


FIGURA 3.7: Minimización de la función de error utilizando el gradiente calculado sobre X_i (Freeman y Skapura, 1993).

XY mientras el error, que debería estar representado en el eje Z, ha sido indicado mediante curvas de nivel. El descenso en la dirección al mínimo se presenta algo "errático" dado que la forma del paraboloide depende del valor de W y del ejemplo utilizado para calcularlo. Sin embargo, dado que se trata de una función que solo posee un minimo global, si las modificaciones no resultan excesivamente grandes, la convergencia estará asegurada.

El entrenamiento del combinador lineal consiste en un algoritmo iterativo que ingresa los ejemplos uno a la vez y modifica, para cada uno de ellos, el vector de pesos W. Comienza asignando valores arbitrarios al vector de pesos W y en base al primer ejemplo ingresado, se establece la dirección de la pendiente más pronunciada para efectuar el descenso, es decir, se calcula el vector gradiente negativo. Luego se modifica el vector W utilizando una fracción del vector gradiente antes calculado. De esta forma, si se volviera a calcular el error en el mismo ejemplo, el valor sería menor pero como se trata de un proceso iterativo que ingresa los ejemplos uno a la vez, en la próxima iteración el ejemplo habrá cambiado. Tanto el cambio producido en W como el ingreso del nuevo ejemplo dan lugar a una nueva función de error. En este punto se vuelve a calcular el vector gradiente y el proceso se repite. Nótese que las curvas de nivel representadas en la figura 3.7 corresponden al paraboloide generado a partir de todos los ejemplos pero el recorrido en la dirección al mínimo muestra el descenso utilizando el vector gradiente calculado como se indica en la

ecuación (3.12)

Dado que el vector de pesos es variable, se tiene en cuenta un paso temporal t. Sea $w(0)$ el vector inicial de pesos, $w(t)$ el vector de pesos en el instante t, el próximo vector de pesos se calcula de la siguiente manera:

$$w(t+1) = w(t) + \Delta w(t) \tag{3.13}$$

donde $\Delta w(t)$ es el cambio que sufre w en el instante t.

Para poder encontrar la dirección del descenso más pronunciado, es necesario calcular el gradiente de la superficie (ecuación 3.12), el cual indicará la dirección de la pendiente ascenso más pronunciada. Por este motivo, para ir en dirección al mínimo debe cambiarse el signo a cada componente del vector. Esto es lo que le da el nombre de *gradiente descendente negativo*.

La modificación de los coeficientes se realiza según la ecuación 3.14, sea α un valor entre cero y uno:

$$w(t+1) = w(t) - \alpha * \nabla error_i^2 \tag{3.14}$$

En cada iteración, la *corrección* entre $w(t)$ y $w(t+1)$ se hace aplicando la siguiente ecuación

$$\begin{aligned} \Delta w(t) &= w(t+1) - w(t) \\ &= -\alpha * \nabla error_i^2 \end{aligned} \tag{3.15}$$

Con cada nuevo conjunto de valores de parámetros ingresado a la red se espera que la función de error se reduzca, modificando los coeficientes luego de cada cálculo. Aunque se puede admitir un aumento del error, siempre y cuando, esto conduzca a una disminución posterior mayor. El criterio de finalización está dado por una cota en la diferencia de dos errores promedio consecutivos o por una cantidad máxima de iteraciones; lo que ocurra primero.

El algoritmo 3 resume lo antes dicho.

Como se puede observar en la ecuación del gradiente descendente (ecuación 3.14) para obtener la magnitud del cambio, se multiplica al gradiente por la constante positiva α (Freeman y Skapura, 1993). A continuación se darán detalles sobre este parámetro que si bien ya se utilizó en el perceptrón, en el caso del Combinador Lineal es sumamente importante.

Algoritmo 3: Algoritmo de entrenamiento del Combinador lineal

Data: $E = \{(X_1, d_1), \ldots, (X_n, d_n)\}$, α, $maxITE$, $cotaError$
Result: Vector de pesos W
Inicializar el vector de pesos W con valores aleatorios
$ite = 0$
repeat
  $errorAVG = 0$
  for $i = 1$ **to** n **do**
    $y = W, X_i$
    $error_i = d_i - y$
    $gradiente = -2 * error_i * X_i$
    $W = W + \alpha * gradiente$
    $errorAVG = errorAVG + error_i$
  $errorAVG = errorAVG/n$
  $ite = ite + 1$
until *(ite* $\geq$ *maxITE) or (errorAVG <cotaError)*

Selección del tamaño del paso

Se suele referir al coeficiente α como *paso* o *velocidad de aprendizaje*. Representa el tamaño del paso que se utiliza en cada iteración para acercarse al punto mínimo (Haykin, 2009).

La elección de α es muy importante ya que tiene una gran influencia en la convergencia del algoritmo:

- Si el tamaño del paso es muy chico tardará mucho tiempo en converger, lo cuál hace al algoritmo costoso en tiempos computacionales.
- Si el paso es demasiado grande, puede causar oscilaciones muy grandes en los pesos aprendidos y no llegar a converger, independientemente del tiempo de entrenamiento. Así también, puede saltearse el valor mínimo.

Existen varios métodos de selección del coeficiente. Si se conocen las estadísticas del vector de entrada se podría utilizar un α que toma valores dentro del siguiente intervalo

$$\frac{1}{\lambda_{\max}} > \alpha > 0 \tag{3.16}$$

donde λ_{max} es el valor máximo del vector de entrada. Esta metodología es buena, pero no siempre se cuenta con esta información, aunque hay algunos casos en donde se la puede estimar.

En el texto de Widrow y Stears (Bernard Widrow y Stearns, 1985) se propone una aproximación un poco más heurística. Se toma un valor α tal que los pesos no varíen más que cierta fracción de su valor actual.

Sin embargo, todo indica que a la hora de seleccionar un valor adecuado de α la experiencia suele ser lo más importante. A medida que el entrenamiento avanza, el error va disminuyendo y los cambios son cada vez más pequeños haciendo más lenta la convergencia. En esos momentos, puede resultar útil aumentar el valor de α y así acelerar la velocidad de convergencia. Esto se puede realizar siempre teniendo en cuenta que un valor demasiado grande tampoco sirve.

En conclusión, seleccionar un α adecuado depende de la experiencia, probando y ajustando, hasta encontrar el óptimo para el problema a resolver.

3.5. Técnicas descriptivas

Como bien se explicó en el capítulo anterior, dentro de la Minería de Datos existen herramientas para resolver dos tipos de problemas: predictivos y descriptivos. En las secciones anteriores se desarrolló en profundidad una técnica de tipo predictiva: las redes neuronales artificiales.

En esta sección se describirán los conceptos básicos de las técnicas de agrupamiento partitivo y en particular del algoritmo k-medias, ya que fue utilizado en el análisis inicial de la representación numérica dada a cada oración del documento.

Disponer de herramientas que representan de una manera descriptiva la información resulta sumamente útil para elegir el camino a seguir de una manera consciente. Este tipo de herramientas tiene que ver con hallar una representación que explique las características relevantes de los distintos grupos presentes en los datos.

3.5.1. Agrupamiento

El agrupamiento o clustering de un conjunto de datos es una operación muy utilizada cuando lo que se busca es explicar cómo están organizados entre ellos, identificando similitudes y diferencias.

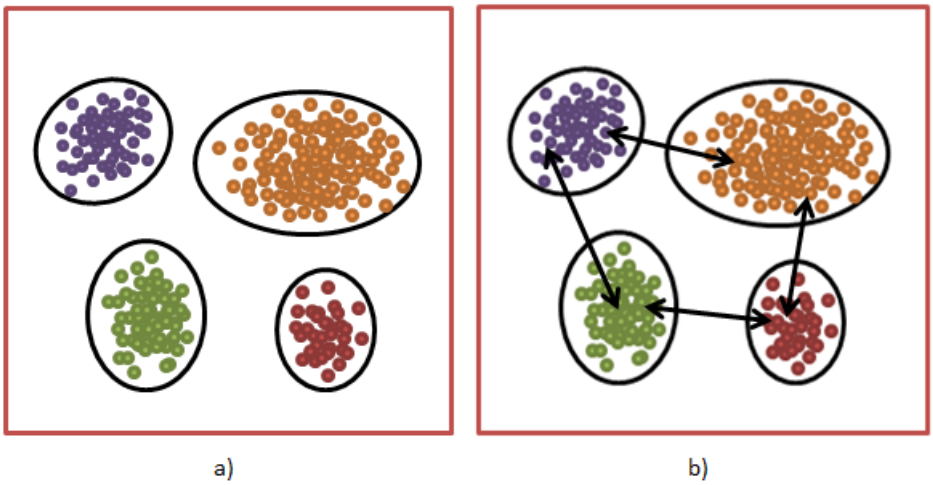


FIGURA 3.8: Agrupamiento de ejemplos en 4 clusters. El color indica a qué grupo pertenece cada ejemplo. En a) se observa que los grupos son compactos y en b) que la separación en ellos es marcada. Es decir que la distancia intracluster es menor que la intercluster.

Las técnicas de clustering tienen por objetivo reunir ejemplos o casos similares sin utilizar ninguna información de etiquetado o clasificación previa. En su lugar requieren de una métrica que pondere el parecido, cercanía o similitud entre los ejemplos.

Un mismo conjunto de datos puede agruparse de distintas formas. Las distintas técnicas de clustering tienen por objetivo agrupar los ejemplos de manera tal que los elementos de un grupo sean lo más parecidos entre si y diferentes de los ejemplos de otros grupos. Esto da como resultado la formación de grupos bien distintos entre si y con poca dispersión entre los elementos de un mismo grupo. A cada uno de los grupos se los denomina "cluster".

Como se dijo previamente, para poder generar los grupos, los algoritmos de clustering utilizan una medida de similitud o distancia que indica de manera unívoca qué tan distantes están entre sí cada uno de los ejemplos.

Independientemente del método de agrupamiento que se utilice, el objetivo será maximizar la distancia entre los grupos o distancia intercluster y minimizar la distancia entre los elementos de un mismo grupo o distancia intracluster. Véase figura 3.8.

La manera en que los ejemplos queden agrupados depende de varios factores: el algoritmo de agrupamiento seleccionado, la medida de similitud utilizada para comparar los ejemplos y el conjunto de datos disponible. Esto último no solo depende de las características relevadas y de la escala en la que las mismas se encuentren sino

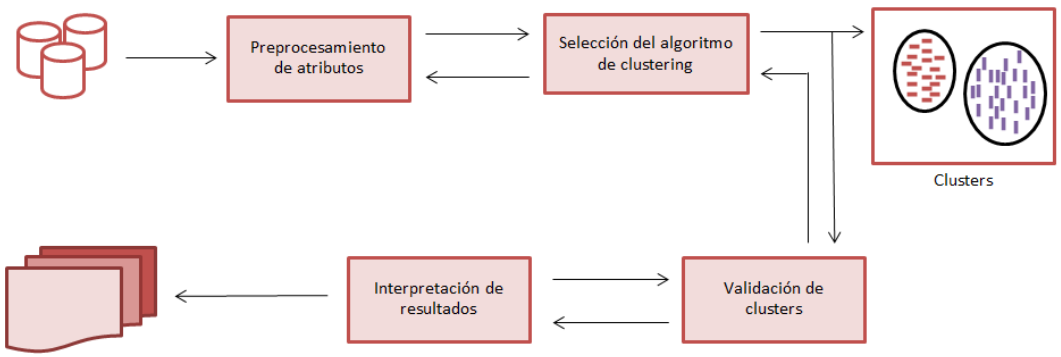


FIGURA 3.9: Fases de un proceso de clustering

que la dimensión del espacio de entrada juega un papel fundamental en el desempeño de la medida de similitud.

La figura 3.9 ilustra un proceso de clustering clásico.

3.5.2. Medidas de distancia o similitud

Como se dijo anteriormente, el objetivo de las técnicas de agrupamiento es reunir ejemplos con características parecidas. Por lo tanto, es necesario definir el criterio con el cual se compararán dos ejemplos y se decidirá si se parecen o no; es decir, si se encuentran cerca en el espacio de la representación del problema o no.

A partir de esta idea surge la necesidad de determinar qué se entiende por "similitud". Para formalizar este concepto se utilizan diferentes medidas de distancia. Si se quiere conocer la similitud entre dos instancias o ejemplos, es necesario elegir una función de distancia y calcular con ella la misma entre los dos individuos. Si se denomina $d(i, j)$ a la distancia que separa los ejemplos X_i y X_j, la notación $d(i, j) > d(i, k)$ nos indica que el objeto X_i es más parecido a X_k que a X_j.

La definición de la métrica de similitud/distancia será distinta en función del tipo de dato y de la interpretación semántica que se realice. Esto implica que un mismo método puede funcionar de maneras diferentes, variando la medida de distancia.

El conjunto de datos que se va a agrupar puede considerarse como una colección de vectores n-dimensionales. En particular, los vectores en un espacio Euclídeo están formados por números reales. En el caso de esta tesina, dichos vectores corresponden a los valores de las métricas de cada oración.

Calculando la distancia entre dos ejemplos (individuos del conjunto) se puede determinar la similitud entre los mismos. Hay diferentes funciones para calcular la

distancia entre dos vectores de números reales (Rajaraman y Ullman, 2011), algunas de las más utilizadas son:

- *Distancia de Manhattan*, o distancia por cuadras, que recorre un camino zigzagueando, es la suma de las diferencias entre dimensiones:

$$d(x,y) = \sum_{i=1}^{n} \mid x_i - y_i \mid \tag{3.17}$$

- *Distancia de Chebychev*, que calcula la discrepancia más grande en alguna de las dimensiones. Se usa cuando se quiere considerar que los individuos son diferentes si lo son en una de las dimensiones:

$$d(x,y) = \max_{i=1..n} \mid x_i - y_i \mid \tag{3.18}$$

- *Distancia coseno:* la distancia es el coseno del ángulo que forman los vectores:

$$d(x,y) = \arccos\left(\frac{x^t\, y}{||x||\,.\,||y||}\right) \tag{3.19}$$

- *Distancia Euclídea:* se define como la longitud de la recta que une dos puntos en el espacio euclídeo, es una de las más utilizadas:

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{3.20}$$

- *Distancia de Mahalanobis*, que generaliza la distancia euclídea admitiendo escalas lineales arbitrarias y rotaciones del espacio de características , tiende a eliminar información redundante:

$$d(x,y) = \sqrt{(x-y)^T S^{-1}(x-y)} \tag{3.21}$$

Cuando se trabaja con vectores numéricos es muy importante normalizarlos ya que, de no ser así, algún elemento (métrica) podría tener un valor con una magnitud media superior al resto convirtiéndose en un término dominante al momento de calcular las distancias. La detección de valores anómalos también es importante, ya que la normalización puede verse muy afectada por estos valores.

A continuación se describirá con más detalle la técnica de agrupamiento que se utilizó para realizar el análisis inicial de las oraciones de los documentos luego de haberlas representado como vectores numéricos mediante las métricas.

3.5.3. Algoritmo K-medias

El algoritmo K-medias o *K-means*, es uno de los algoritmos de agrupamiento basados en centroides más conocido. La versión estándar fue propuesta por primera vez por Stuart Lloyd en 1957 como una técnica para modulación por impulsos codificados, aunque no se publicó fuera de los laboratorios Bell hasta 1982 (Lloyd, 2006). Es un método de agrupamiento adaptativo que tiene por objetivo minimizar la distancia interna de cada grupo; esta distancia se calcula como la suma de las distancias de los ejemplos asignados a un grupo con respecto al centro del mismo.

Su funcionamiento requiere conocer de antemano la cantidad k de grupos a formar. Inicialmente toma k ejemplos como centros o prototipos iniciales. Estos prototipos se irán modificando a lo largo del proceso iterativo hasta alcanzar una ubicación estable. En ese momento es cuando se considera que el proceso de agrupamiento ha finalizado.

Luego de establecer los centros iniciales, el algoritmo calcula para cada ejemplo X_n el prototipo más próximo C_j e incluye el ejemplo en la lista de dicho prototipo. Una vez introducidos todos los ejemplos, cada prototipo C_j tendrá un conjunto de ejemplos a los que representa. Se desplaza el prototipo hacia el centro de masa de su conjunto de ejemplos y se repite el procedimiento hasta que los prototipos ya no se desplacen. En ese momento los ejemplos de entrada quedan divididos en k grupos y el prototipo correspondiente se encuentra en el centro del mismo, por lo que también es denominado *centroide*. Estos centros minimizan las distancias cuadráticas euclídeas entre los ejemplos de entrada y el centro más cercano, es decir, minimizan el valor de J indicado en la ecuación 3.22

$$J = \sum_{j=1}^{k} \sum_{n=1}^{m} M_{x_n, C_j} D(x_n - C_j)^2 \tag{3.22}$$

$$M(x_n, C_j) = \begin{cases} 1 & si\ D(x_n - C_j) < D(x_p - C_j)\ \ \forall p\ ;\ n \neq p \\ 0 & sino \end{cases} \tag{3.23}$$

donde m es el tamaño del conjunto de ejemplos, D es una medida de distancia, X_n es el n-ésimo ejemplo de entrada, C_j es el prototipo de la clase j y $M(j,n)$ es la función de pertenencia del ejemplo n a la clase j indicada en 3.23

El algoritmo 4 representa el proceso descripto previamente.

Algoritmo 4: Algoritmo K-Medias

Data: $E = \{X_1, \ldots, X_n\}, k$
Result: Vector de centros C
begin
  $C \leftarrow$ Tomar al azar k ejemplos como centros iniciales
  $Asignaciones \leftarrow$ Asignar cada ejemplo a su centro más cercano
  repeat
    $C \leftarrow$ Recalcular los centros promediando los ejemplos asignados a cada uno
    $AsigAnteriores \leftarrow Asignaciones$
    $Asignaciones \leftarrow$ Asignar cada ejemplo a su centro más cercano
  until $Asignaciones = AsigAnteriores$
  return C

La ventaja principal del algoritmo k-medias es su simplicidad y eficiencia; es fácil de entender y de implementar. Existen mejoras que no requiere conocer de antemano la cantidad k de grupos a formar como por ejemplo el algoritmo ISODATA (Ball y Hall, 1965) (Venkateswarlu y Raju, 1992). El resultado a obtener depende de la elección de los centros iniciales. Sobre este tema puede consultarse (S. S. Khan y Ahmad, 2004) (Erisoglu, Calis y Sakallioglu, 2011).

3.6. Conclusiones

En el marco de la tesina, se analizaron los resultados del cálculo de las métricas (2.4) mediante las herramientas mencionadas en esta sección y en base a esto se pudieron tomar decisiones teniendo un conocimiento extra sobre la estructura de datos que se tenía.

Se trató principalmente el tema redes neuronales, introduciendo al lector desde el surgimiento de las redes hasta dos modelos de redes simples: Perceptrón y ADALINE. Ambos modelos se desarrollaron, junto con la arquitectura y el algoritmo de aprendizaje particular de cada uno.

Se describió una de las técnicas descriptivas de la Minería de Datos, el agrupamiento. En particular, el agrupamiento K-medias fue el desarrollado debido a que se lo utilizó en este trabajo como método de descripción de los valores de las métricas obtenidas en el capítulo anterior.

En el siguiente capitulo se realiza la aplicación de todos los conceptos desarrollados en los capítulos anteriores.

Capítulo 4

El método propuesto

4.1. Introducción

En base al marco teórico expuesto en los anteriores capítulos, se pasa a explicar cómo se aplican las diferentes herramientas seleccionadas para llegar al aporte central de esta tesina, aprender el criterio de selección utilizado por una persona al resumir un texto. Esto se logrará por medio de la selección del subconjunto de métodos de puntuación de sentencias que mejor aproximan al resumen humano y en qué proporción incide cada uno de ellos al momento de ponderar la importancia de una oración. Una vez obtenido dicho criterio, podrá aplicarse sobre otros documentos para obtener como respuesta un resumen similar al que hubiera realizado el usuario en forma manual pero en un tiempo de respuesta mucho menor.

4.2. Propuesta metodológica

Como se mencionó en el capítulo 1, los resúmenes automáticos pueden clasificarse principalmente entre resúmenes extractivos o abstractivos. Dependiendo del texto que se quiera resumir, los metadatos que se posean de él y del resumen que se quiera obtener, se elige el tipo de resumen.

En el presente trabajo se seleccionó la metodología extractiva, realizando el resumen en base a un solo documento con propósito general.

Es importante recordar que el enfoque extractivo no garantiza la coherencia narrativa de las sentencias seleccionadas. Sin embargo, se consigue reducir el tamaño del documento dejando únicamente el contenido relevante. Este enfoque tiene tres ventajas: (I) se puede controlar el tamaño del resumen, (II) el contenido del resumen

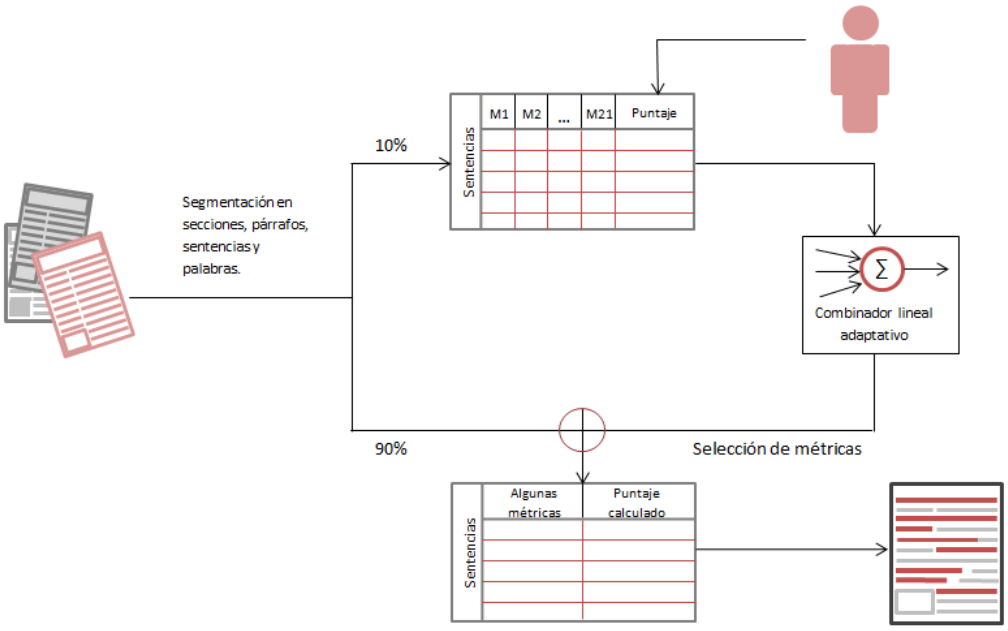


FIGURA 4.1: Esquema de la metodología utilizada para la generación de resúmenes de documentos

se obtiene de forma precisa y (III) la relación entre el resumen y el texto original es inmediata.

En la figura 4.1 se muestra un diagrama que resume la metodología propuesta, la cual se desarrollará a continuación.

4.2.1. Preprocesamiento del texto

Como se mencionó en el primer capítulo, toda tarea de *Minería de Textos* comienza con la etapa de preprocesamiento. Dentro de esta etapa se realizan los procesos de segmentación del texto, tokenización, extracción de las stopwords y stemming de las palabras. Dichos procesos se encuentran explicados de manera detallada en la sección 1.4.2.

La segmentación es el proceso en donde se divide al texto en porciones más pequeñas tal como aparecen en el documento, utilizando delimitadores de corte. Generalmente a estas porciones del texto se las llama "sentencias" en forma genérica.

Una vez que se dispone de las sentencias se procede a dividirlas en *términos*, este es el proceso de tokenización. Los términos son las porciones de texto sobre las cuales se va a trabajar. En esta tesina se seleccionaron la palabras como términos.

A su vez es necesario seleccionar las palabras que deben influir en los cálculos de las métricas 2.4 y las palabras que no. Como se mencionó en capítulos anteriores,

hay palabras que se utilizan para conectar términos o conceptos, como por ejemplo "entre", "al", "desde". Estas palabras no son representativas del documento a resumir, sirven para dar coherencia al texto, pero no deberían influir en los valores de las sentencias. Estas y muchas otras más, se denominan palabras vacías o stopwords y se suelen quitar del texto utilizado para realizar los cálculos.

Una vez obtenidas las palabras, se aplicó la técnica de stemming para extraer las raíces correspondientes. Se considera que esta representación posee un valor mayor ya que, en general, las palabras que comparten una misma raíz son palabras que poseen el mismo significado o parecido.

4.2.2. Representación de los documentos

Existen muchas maneras de caracterizar las sentencias de un documento. En la presente tesina se seleccionaron de la literatura 21 métricas para calificar cada sentencia y representar con ellas a cada documento como un conjunto de vectores numéricos. Estos valores están basados en cantidades calculadas a partir de diferentes aspectos que tienen las sentencias en el texto. El proceso de cálculo de las mismas se encuentra detallado en el capítulo 2 sección 2.4. De esta manera cada documento estará representado por una matriz con tantas filas como sentencias tenga el documento y tantas columnas como métricas se utilicen para representar las sentencias (en este caso 21).

Cada uno de estos valores se escaló linealmente mediante una *normalización Z* la cual se describe brevemente a continuación.

Dado que la unidad de medida utilizada para representar los datos puede afectar significativamente el análisis de los mismos, fue necesario expresarlos en una unidad con un rango o escala particular. Esto se llevó a cabo mediante la normalización de los datos, transformando la distribución de los valores originales en un conjunto de nuevos valores con las propiedades necesarias para su correcto análisis (García, Luengo y Herrera, 2014).

Cuando se desconocen los valores máximo y mínimo del conjunto, o dentro del conjunto de datos hay *datos fuera de rango o "outliers"*, se suele utilizar la normalización Z también conocida como estandarización. La normalización Z es una transformación que reemplaza el valor original de una variable en la cantidad de desvíos que lo separan del valor promedio. El signo indica si esa cantidad se encuentra por encima o por debajo de la media según sea positiva o negativa respectivamente. Es

decir que un puntaje z indica la dirección y el grado en que un valor individual obtenido se aleja de la media, en una escala de unidades de desviación estándar.

Si μ_A es la media de los valores del atributo A y σ_A es la desviación estándar, un valor v es normalizado a v' usando la ecuación (4.1). Al aplicarle esta transformación, los valores presentan una media igual a cero y una desviación estándar igual a uno.

$$v' = \frac{v - \mu_A}{\sigma_A} \tag{4.1}$$

Análisis descriptivo de las métricas

A partir de las métricas almacenadas para cada sentencia de los documentos, se buscó analizar de una manera preliminar la existencia de distintas relaciones entre las mismas. Con este fin, se desarrollaron e implementaron algunas herramientas para realizar un análisis comparativo.

En primer lugar se desarrolló y aplicó el algoritmo K-medias descripto en la sección 3.5.3 con el objetivo de identificar la presencia de distintos tipos de oraciones. Se hicieron pruebas con distintos valores de k y, a través del índice de Davies Bouldin (Davies y Bouldin, 1979), se logró determinar que un valor de $k = 4$ ofrecía buenos resultados en la cohesión y separación de los grupos.

Dado que el algoritmo K-medias ofrece distintos resultados según la manera en que se seleccionen sus centros iniciales, se realizaron 30 ejecuciones independientes. En cada ejecución, los k centros iniciales fueron seleccionados de manera aleatoria. Los resultados de las 30 ejecuciones fueron similares.

La figura 4.2 muestra el resultado de promediar los centros obtenidos como resultado de las 30 ejecuciones. Nótese que si bien cada ejecución dará como resultado 4 vectores de longitud 21 no necesariamente corresponden a los mismos agrupamientos. Esto llevó a la necesidad de ordenarlos previamente utilizando una de las métricas. En este caso se utilizó el valor de $LUHN$ para tal fin.

La figura 4.2 es un gráfico de coordenadas paralelas (Inselberg y Dimsdale, 1990) en el cual puede apreciarse el valor del centro de cada grupo para cada métrica. Lo importante del gráfico es ver que algunas métricas toman valores parecidos en los 4 grupos mientras que otras son llamativamente diferentes. Son estas últimas las que se espera que determinen el agrupamiento.

Este análisis permitió identificar la existencia de 4 grupos de oraciones caracterizados por menos del 50 % de las métricas.

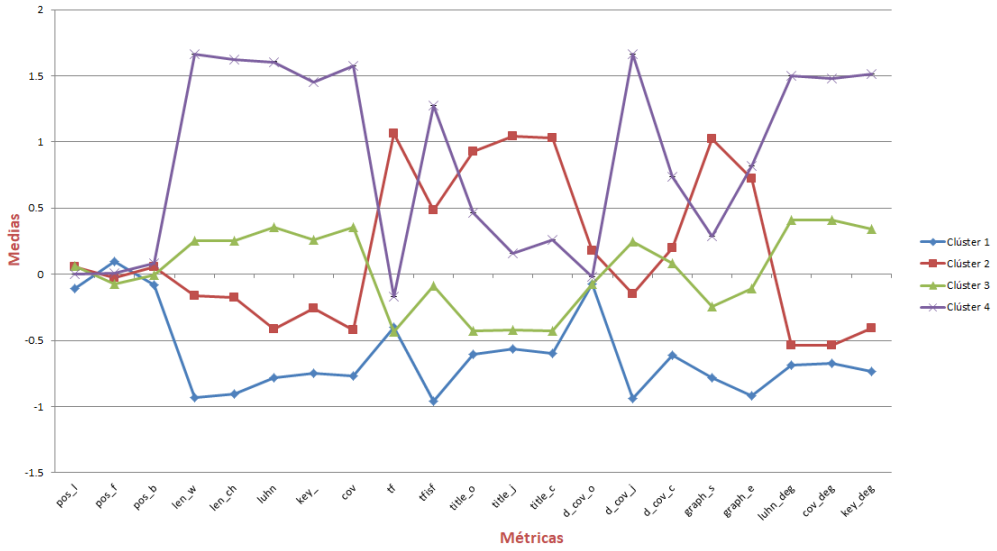


FIGURA 4.2: Agrupamiento promedio de las oraciones de un documento utilizando k-medias con $k = 4$.

Como otra estrategia de selección se calculó el coeficiente de correlación lineal entre los distintos pares de métricas a fin de determinar su relación. Aquellos pares de atributos con correlación superior a 0,85 o inferior a $-0{,}85$ fueron considerados "fuertemente correlacionados" pudiendo prescindirse de uno de ellos. La matriz de la figura 4.3 muestra los atributos que no poseen entre sí una correlación lineal fuerte, es decir, que serían los seleccionados para trabajar.

Métricas	pos_f	pos_b	luhn	tf	tfisf	title_c	d_cov_o	d_cov_c	graph_e
pos_f	1	-0.037	-0.049	-0.043	-0.056	0.127	-0.007	0.066	0.198
pos_b	-0.037	1	0.040	0.034	-0.028	0.166	0.019	0.100	0.008
luhn	-0.049	0.040	1	0.160	-0.160	0.003	0.038	0.311	0.280
tf	-0.043	0.034	0.160	1	0.097	-0.121	-0.052	-0.078	-0.151
tfisf	-0.056	-0.028	-0.160	0.097	1	-0.165	-0.037	-0.235	-0.294
title_c	0.127	0.166	0.003	-0.121	-0.165	1	0.153	0.428	0.401
d_cov_o	-0.007	0.019	0.038	-0.052	-0.037	0.153	1	0.242	0.201
d_cov_c	0.066	0.100	0.311	-0.078	-0.235	0.428	0.242	1	0.593
graph_e	0.198	0.008	0.280	-0.151	-0.294	0.401	0.201	0.593	1

FIGURA 4.3: Matriz de correlación de métricas con valores absolutos menores a 0.85

Si se observa la matriz de correlación de la figura 4.3, se pueden ver las métricas que al ser aplicadas sobre las sentencias producirían resultados distintos. Las métricas que faltan, deberían ofrecer un resultado similar a alguna de estas por presentar una correlación lineal fuerte.

Como se mencionó anteriormente, este tipo de análisis es puramente descriptivo. A través de estas metodologías es posible identificar cuáles son las métricas que determinan los tipos de oraciones o cuáles presentan relaciones lineales significativas. Sin embargo, en ninguno de los dos casos es posible establecer el grado de participación de dichas métricas en el criterio del usuario al momento de ponderar una sentencia.

Por esta razón es que el método propuesto se realizó con una de las técnicas predictivas definidas en el capítulo 3, el *combinador lineal adaptativo.*

4.2.3. Aprendizaje del criterio del usuario

La preferencia del usuario por una sentencia viene dada por la puntuación que le asigne. Esta se trata de un valor entero positivo proporcional al grado de importancia estimado. Aquellas sentencias que reciban valor 0 como puntaje serán interpretadas como irrelevantes mientras que las que reciban los valores más altos serán las más significativas.

Dado que se cuenta con varias métricas calculadas para cada sentencia del documento se espera que una combinación lineal de ellas represente el criterio del usuario. Por lo tanto, el problema a resolver consiste en hallar los coeficientes $c_1, c_2, \ldots, c_k$ tales que aplicados a las métricas de la sentencia S_i, $m_{i1}, m_{i2}, \ldots, m_{ik}$, permitan aproximar el puntaje indicado por el usuario.

Para resolver esto se utilizó un Combinador Adaptativo Lineal (*Adaptive Linear Combiner - ALC*) tal como fue explicado en 3.

En este caso, el ALC es una red neuronal formada por una única neurona con tantas entradas como métricas se utilicen para representar cada sentencia y una única salida cuyo valor corresponde al puntaje predicho y viene dado por lo indicado en la ecuación 4.2. El coeficiente c_0 funciona como término independiente y la entrada m_{i0} vale 1 siempre. Para una mejor comprensión del método véase la figura 4.4.

$$puntaje_i = \sum_{j=0}^{k} (c_j * m_{ij}) \tag{4.2}$$

Los pesos de los arcos que unen cada métrica con la neurona funcionan como factores de escala y regulan la importancia o participación de cada métrica en el puntaje.

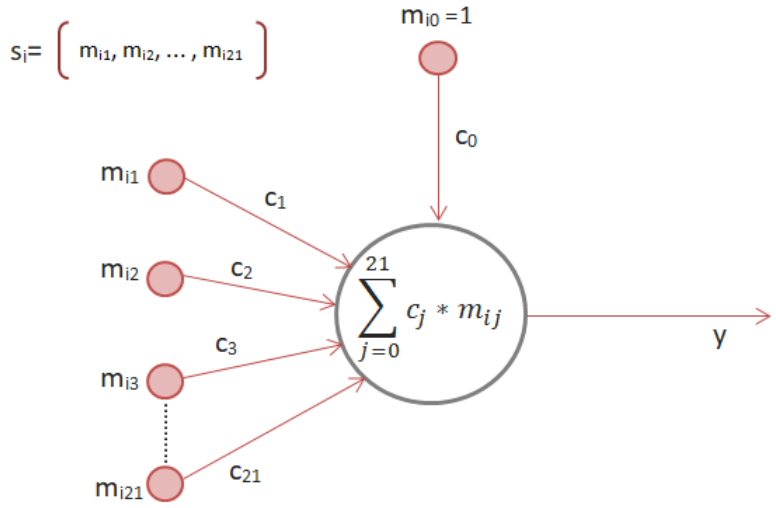


FIGURA 4.4: Representación general del combinador lineal utilizado en esta tesina.

El aprendizaje por parte de la neurona consiste en establecer los coeficientes asociados a cada métrica representados por los pesos de los arcos que la unen con la información de entrada. Para lograrlo, se debe contar con un conjunto de ejemplos etiquetados; en este caso, un conjunto de sentencias para las cuales se indique el puntaje esperado.

A partir del combinador adaptativo lineal ya entrenado, es posible identificar las métricas que tienen mayor participación en la predicción del puntaje de las sentencias. Para ello, debe tenerse en cuenta la proporción que guarda el valor de la métrica, al ser ponderado por su coeficiente correspondiente, con respecto al puntaje predicho para la sentencia.

Considerando la sentencia S_i, puede decirse que la contribución total suministrada por todas las métricas al valor $puntaje_i$ viene dada por la suma de sus valores ponderados absolutos, tal como se indica en la ecuación 4.3.

Por lo tanto, puede usarse la proporción indicada en la ecuación 4.4 para estimar la participación de la j-ésima métrica de la sentencia i en esta suma.

$$acumulado_i = \sum_{j=1}^{k} abs(c_j * m_{ij}) \tag{4.3}$$

$$participa_{ij} = \frac{abs(c_j * m_{ij})}{acumulado_i} \tag{4.4}$$

Repitiendo esto para cada sentencia del conjunto de entrenamiento y promediando los valores obtenidos para cada métrica se obtendrá un vector $P = (p_1, p_2, \ldots, p_k)$ con los valores de los grados de participación de cada métrica en la estimación de los puntajes. La ecuación 4.5 indica la forma en que debe realizarse el promedio para un conjunto de sentencias de cardinalidad M.

$$p_j = \frac{\sum_{i=1}^{M} participa_{ij}}{M} \tag{4.5}$$

Las métricas cuyo grado de participación supere el valor promedio ($(\sum_{j=1}^{k} p_j)/k$) serán consideradas relevantes para la estimación del puntaje.

4.3. Resultados obtenidos

Para comprobar el funcionamiento de la metodología propuesta se utilizaron artículos científicos extraídos de la revista *PLOS Medicine* (ONE, 2004) publicados entre enero de 2004 y diciembre de 2016. La elección de la revista estuvo condicionada no solo por su importancia desde el punto de vista médico sino porque brinda la posibilidad de descargar los archivos en distintos formatos. En este caso se obtuvieron documentos etiquetados en formato XML; este aspecto resulta fundamental para automatizar la extracción de la información para su posterior carga a la base de datos.

Como se mencionó en el capítulo 1, el beneficio de los documentos con formato XML es que son textos estructurados en donde se pueden identificar fácilmente las partes de los documentos ya que se encuentra segmentado por etiquetas. Así también, se identifican los metadatos, si es que el documento los posee, y esto hace posible guardar la mayor cantidad de información de los documentos. En el anexo A se detalla la estructura de este tipo de documentos; se recomienda su consulta para mayor información.

Se extrajeron 223 artículos científicos, donde cada uno estuvo formado aproximadamente por 30 párrafos de 4 o 5 oraciones cada uno y cada oración tuvo un promedio de 20 palabras.

El lenguaje de programación utilizado para realizar los algoritmos de las etapas de preprocesamiento de los documentos (1.4.2) y cálculo de las métricas (2.4) fue *Python 2.7*.

Se diseñó una base de datos para poder persistir los documentos de una manera adecuada para su posterior procesamiento (anexo B). Antes de almacenarlos en la estructura, se los preprocesó utilizando la librería NLTK (*Natural Language Toolkit*) de Python que permite manipular el lenguaje natural y simplifica los procesos de tokenización, stopwords y stemming.

Dentro de estos procesos se debieron tomar ciertas consideraciones para persistir los documentos de la mejor forma posible para su posterior manipulación. Una vez obtenidos los términos del proceso de tokenización, se realizó un proceso de normalización del texto. Estando los documentos estructurados con XML, el corte de los textos llevó a obtener algunos términos compuestos por etiquetas y texto, interpretables por un navegador para representar el texto de alguna manera particular. Esto no es algo necesario de almacenar, por lo que se procedió a seleccionar solo los términos que eran pertinentes a los cálculos.

Al realizar los cortes en palabras se obtuvieron algunas que solo consistían en etiquetas HTML, como por ejemplo <sup> o <sub>, o números, como *[1,2,3]*, que no son palabras pero al extraerlas de los documentos quedaban como una unidad aparte. Si se dejaran en la base como "tokens" estas palabras influirían en los cálculos, lo cual no sería correcto.

Es importante recordar en este punto que el objetivo de este trabajo es aprender el criterio utilizado por el usuario para seleccionar las sentencias que dan lugar al resumen extractivo. De esta forma, contando con un número reducido de documentos resumidos será posible reproducir el criterio empleado en los documentos restantes. Sin embargo, para poder medir el desempeño del combinador lineal propuesto es preciso conocer el puntaje de todas las sentencias del corpus. El etiquetado manual resulta una tarea sumamente tediosa para el usuario y por tal motivo se optó por resolverlo en forma automática utilizando una aplicación web.

Luego de analizar varias aplicaciones que realizan resúmenes online a partir de un texto se decidió utilizar (*Online summarize tool.* 2019) ya que de las disponibles en Internet fue la única que cumplió todos los requisitos: (1) cada sentencia corresponde a una oración del texto del documento, (2) permite ordenar según criterio de importancia la totalidad de las sentencias del documento, (3) establece el orden de importancia de las sentencias asignándoles un puntaje a cada una y (4) dispone de una interfaz Web que pudo integrarse al desarrollo realizado.

Una vez etiquetado todo el corpus se procedió a entrenar el combinador lineal

descripto en la sección 4.2.3 a través de una validación cruzada de 10 partes utilizando en cada caso el 10 % de los documentos para aprender el criterio y el 90 % restante para testear si dicho criterio permite obtener el resumen esperado.

Dado que el valor de los coeficientes obtenidos a través del combinador lineal depende de los valores aleatorios con los que se los inicialice, se realizaron 40 ejecuciones independientes de cada caso y se promediaron los resultados obtenidos.

Se tuvieron en cuenta cuatro variantes del combinador lineal:

- **CL-All**: Se entrenó el combinador lineal utilizando las 21 métricas. Esta neurona es la que recibió la mayor información.
- **CL-Corr**: Se entrenó el combinador lineal utilizando solo las 14 métricas que presentaban una correlación inferior a *0.85*. Este umbral fue establecido empíricamente.
- **CL-Pruned**: Para cada partición se aplicó el criterio de selección de métricas según su grado de participación indicado en la sección 4.2.3. Con esta información se quitaron las entradas correspondientes a las métricas no seleccionadas del combinador usado en CL-All y se lo aplicó al conjunto de sentencias de testeo correspondiente. Nótese que el proceso de evaluación del combinador lineal está formado por 40 ejecuciones independientes de 10 partes cada una. Es decir que este proceso de selección se realizó 400 veces.
- **CL-Sel-Train**: Se efectuó la misma selección que en CL-Pruned pero en lugar de usar los pesos de CL-All asociados a las métricas seleccionadas, se reentrenó un nuevo combinador lineal con dichas entradas.
- **CL-Selecc**: Se promediaron los grados de participación de todas las sentencias del corpus en cada una de las ejecuciones y se realizó una selección única de métricas. Con esta información se entrenó un único combinador lineal que luego fue aplicado a todas las particiones. Esta es la mejor solución pero su uso solo es posible si se dispone del resumen de todo el corpus por lo que solo se evaluó a modo comparativo.

La tabla 4.1 resume los resultados obtenidos luego de efectuar una validación cruzada de 10 partes. Los valores mostrados corresponden al promedio de 40 corridas independientes para cada uno de los umbrales de corte.

TABLA 4.1: Tasa de acierto promedio obtenida por 4 variantes del método propuesto

Corte	Métricas utilizadas				
	CL-All	CL-Corr	CL-Pruned	CL-Sel-Train	CL-Selecc
5 %	$0{,}9481_{\pm 0{,}0021}$	$0{,}9472_{\pm 0{,}0015}$	$0{,}9446_{\pm 0{,}0056}$	$0{,}9483_{\pm 0{,}0014}$	$0{,}9489_{\pm 0{,}0010}$
10 %	$0{,}9086_{\pm 0{,}0038}$	$0{,}9064_{\pm 0{,}0022}$	$0{,}9026_{\pm 0{,}0105}$	$0{,}9097_{\pm 0{,}0022}$	$0{,}9107_{\pm 0{,}0015}$
15 %	$0{,}8751_{\pm 0{,}0054}$	$0{,}8729_{\pm 0{,}0033}$	$0{,}8669_{\pm 0{,}0152}$	$0{,}8768_{\pm 0{,}0028}$	$0{,}8784_{\pm 0{,}0018}$
20 %	$0{,}8475_{\pm 0{,}0068}$	$0{,}8452_{\pm 0{,}0041}$	$0{,}8373_{\pm 0{,}0194}$	$0{,}8498_{\pm 0{,}0033}$	$0{,}8513_{\pm 0{,}0021}$
25 %	$0{,}8267_{\pm 0{,}0079}$	$0{,}8249_{\pm 0{,}0044}$	$0{,}8150_{\pm 0{,}0226}$	$0{,}8290_{\pm 0{,}0033}$	$0{,}8304_{\pm 0{,}0022}$
30 %	$0{,}8096_{\pm 0{,}0088}$	$0{,}8084_{\pm 0{,}0053}$	$0{,}7963_{\pm 0{,}0258}$	$0{,}8124_{\pm 0{,}0037}$	$0{,}8138_{\pm 0{,}0022}$
35 %	$0{,}7978_{\pm 0{,}0099}$	$0{,}7960_{\pm 0{,}0057}$	$0{,}7829_{\pm 0{,}0287}$	$0{,}8011_{\pm 0{,}0042}$	$0{,}8030_{\pm 0{,}0025}$
40 %	$0{,}7880_{\pm 0{,}0103}$	$0{,}7864_{\pm 0{,}0058}$	$0{,}7725_{\pm 0{,}0301}$	$0{,}7913_{\pm 0{,}0042}$	$0{,}7932_{\pm 0{,}0024}$

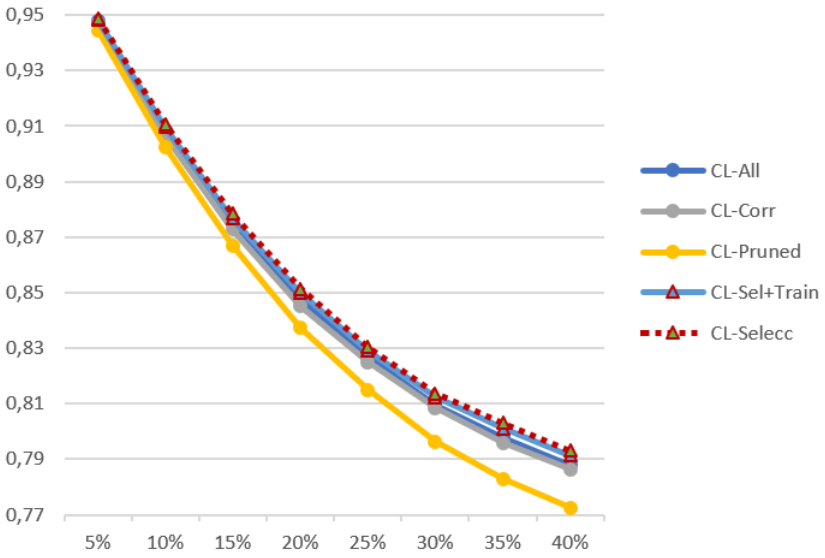


FIGURA 4.5: Tasa de acierto promedio obtenida por el método propuesto durante la validación cruzada

Tanto la tabla 4.1 como en la figura 4.5 detallan el desempeño de las distintas variantes del combinador lineal. Puede verse que salvo la opción CL-Pruned, el resto ofrece resultados similares.

Para analizar la existencia de diferencias significativas entre los resultados de la tabla 4.1 se realizó un test ANOVA de diferencia de medias con un nivel de confianza de 95 %. Los resultados obtenidos se encuentran representados en la tabla 4.2 y permiten afirmar que no hay diferencias significativas entre los métodos CL-All, CL-Corr, CL-Sel-Train, CL-Selecc. El único método que es significativamente diferente del resto es CL-Pruned y corresponde a la selección de las métricas según su grado de participación (según 4.5) a partir del entrenamiento del combinador CL-All.

TABLA 4.2: Resultado del test ANOVA para $p - value < 0{,}05$. Se indica con $-$ cuando no se encuentra diferencia significativa entre las precisiones promedio de ambos combinadores. Los símbolos $\triangle$ y $\triangledown$ indican que hay diferencia siendo el combinador indicado en la fila mejor o peor que el indicado en la columna respectivamente.

	CL-Pruned	CL-Corr	CL-Sel-Train	CL-Selecc
CL-All	$\triangle$	–	–	–
CL-Pruned	–	$\triangledown$	$\triangledown$	$\triangledown$
CL-Corr		–	–	–
CL-Sel-Train			–	–

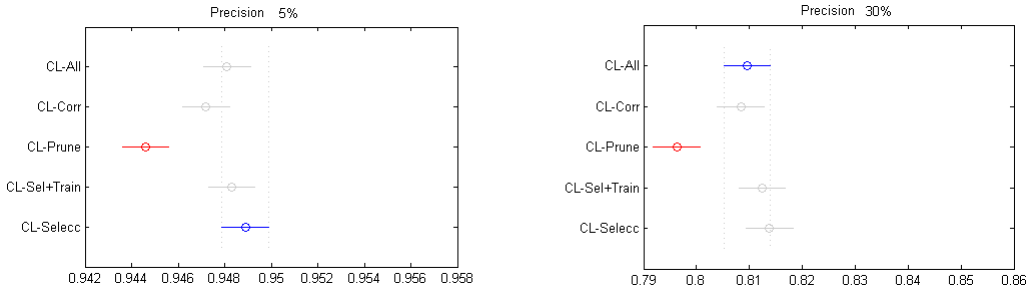


FIGURA 4.6: Intervalos de confianza correspondiente al test ANOVA de diferencia de medias con un nivel del 95 %

La figura 4.6 muestra los intervalos de confianza correspondientes al test ANOVA realizado con un nivel de confianza del 95 %. Si bien las gráfica corresponden al corte del 5 %, los resultados no presentan variantes en las restantes opciones. Aquí puede verse claramente que solo CL-Pruned es distinto del resto.

Con respecto a la selección de las métricas, si bien se observa que depende del conjunto de documentos que se tomen para realizar el entrenamiento, se identifica un núcleo común formado por las métricas: pos_l, pos_f, len_w, len_ch, tfisf, d_cov_j, graph_s, graph_e y luhn_deg.

Estas métricas estuvieron presentes en la mayoría de los conjuntos resultantes.

La figura 4.7 ilustra el grado de participación promedio de cada métrica calculado según la ecuación 4.5.

La figura 4.8 muestra la cantidad de veces que cada métrica fue seleccionada considerando cada una de las 10 particiones de las 40 ejecuciones. En ella puede verse, por ejemplo, que las métricas, pos_l, pos_f y d_cov_j fueron seleccionadas casi siempre mientras que pos_b no fue seleccionada o lo fue una cantidad de veces muy pequeña como para aparecer en la figura.

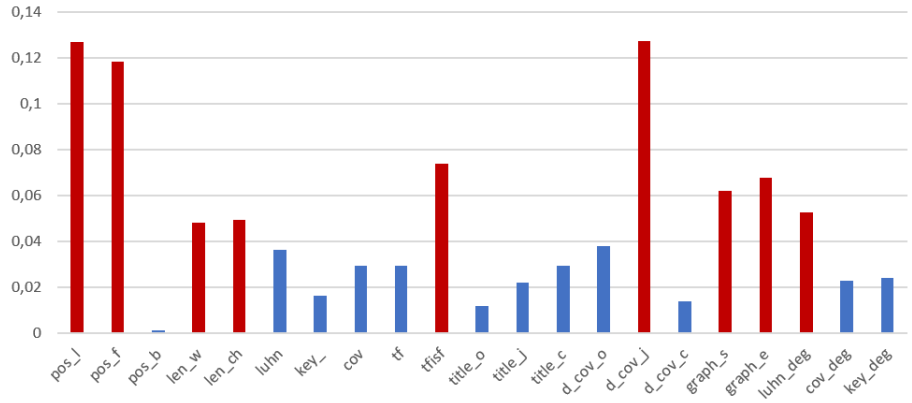


FIGURA 4.7: Grado de participación de cada métrica. Se seleccionan las que posean un valor superior al promedio (en este caso el promedio vale 0.0476)

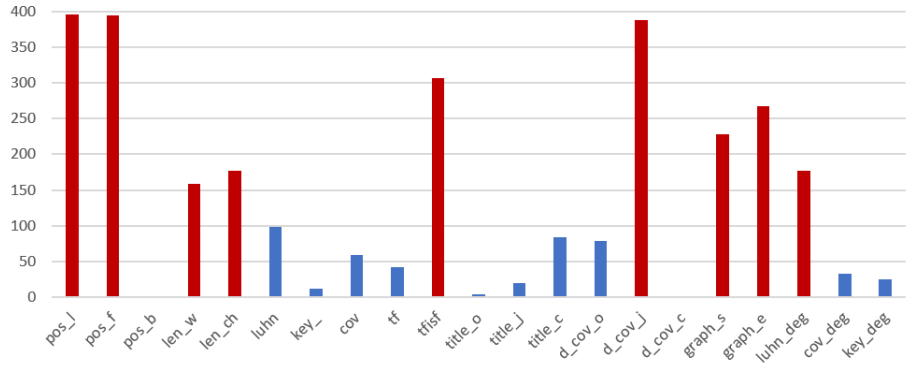


FIGURA 4.8: Cantidad total de veces que cada métrica fue seleccionada considerando cada una de las 10 particiones de las 40 ejecuciones. En cada caso se seleccionaron las que tuvieron un grado de participación superior al promedio (color rojo)

En cada caso se seleccionaron las que tuvieron un grado de participación superior al promedio. La cantidad promedio de métricas seleccionada fue 9, es decir, menos del 50 % de la cantidad total de métricas. Estas métricas se encuentran representadas en la figura con color rojo.

Por lo tanto, puede afirmarse que no hay diferencia significativa entre las alternativas CL-All, CL-Corr, CL-Sel-Train y CL-Selecc en lo que se refiere a la tasa de acierto; sin embargo, la cantidad de métricas utilizadas en cada caso fueron 21, 14, aproximadamente 9 y 9 respectivamente. La solución ofrecida por CL-Pruned es la de menor desempeño aunque su diferencia promedio con respecto a CL-All es $0,0018$ y considerando que su uso implica no tener que reentrenar el combinador lineal antes de usarlo, ya que es una poda directa del generado en CL-All, no se la considera una opción tan mala.

4.4. Conclusión

Se ha presentado una técnica capaz de aprender el criterio utilizado por un usuario para seleccionar las sentencias más representativas de un documento. Es decir que, a través de un umbral es posible realizar un resumen extractivo con las partes más relevantes según un criterio dado.

Los resultados experimentales muestran que el método propuesto es efectivo. La combinación de métodos de puntuación con un combinador lineal permite identificar las métricas más utilizadas por la persona al momento de resumir.

Es importante destacar que la calidad del resumen obtenido depende de la combinación de los métodos de puntuación de sentencias que se utilicen.

La comparación de performance entre las métricas seleccionadas y el conjunto de métricas completo arroja resultados casi iguales en lo que se refiere a identificar hasta el 20 % de las sentencias más significativas según el criterio del usuario. Esto permite afirmar que la selección realizada ha sido correcta.

Capítulo 5

Conclusiones y trabajos futuros

Son numerosos los contextos en los que contar con la posibilidad de resumir texto en forma automática es una tarea de interés. Independientemente de que se trate de un marco educativo, comercial, social, político u otro, todos se ven beneficiados por el resumen de la enorme cantidad de documentos disponibles. Ya sea que se trate de un proceso manual relacionado con la lectura por parte de un lector humano o el manejo automático de documentos (clasificación, almacenamiento, acceso, etc), contar con un resumen es una ventaja importante.

Es en este punto en el que el primer aspecto a resolver tiene que ver con el tipo de resumen a generar. La clasificación más característica es entre resumir de manera extractiva o de manera abstractiva. En la primer variante el proceso implica una selección textual de unidades del documento como partes representativas del mismo. En cambio, la variante abstractiva involucra la creación de contenido para formar parte del resumen del documento. Esto implica una interpretación del texto fuente y un tipo de representación del mismo que permita componer el resumen, lo que lo hace más complejo que el método extractivo. Como se mencionó en los primeros capítulos, el enfoque abstractivo trabaja mejor en resúmenes automáticos de múltiples documentos, pero en resúmenes de un solo documento la mejor elección sigue siendo el acercamiento extractivo. Este último se caracteriza por su simplicidad y su capacidad para mantener la idea original del autor sin necesidad de interpretar el texto que formará parte del resumen. Por estos motivos, en la presente tesina se optó por el resumen automático extractivo.

Este tipo de resumen pondera la importancia de las sentencias del documento y ofrece como resultado un subconjunto formado por las más representativas. Antes de comenzar con el análisis es preciso definir qué es una sentencia, es decir, cuál será la unidad de texto del documento que se busca ponderar. Se pueden elegir secciones

completas, párrafos, oraciones o palabras del texto original. En base a esta elección es que se podrá determinar un listado ordenado de unidades según cuán significativas sean para formar parte del resumen.

En esta tesina las unidades textuales seleccionadas fueron las oraciones y se trabajó con un CORPUS de documentos de texto formado por 223 artículos científicos, obtenidos de la revista PLOS Medicine. Su elección no solo se hizo teniendo en cuenta la importancia dentro del ámbito de la investigación en medicina, sino también en la posibilidad que brinda la revista de extraer cada uno de sus artículos en un formato de texto estructurado. De no ser así, la identificación de las distintas partes que componen el documento sería sumamente compleja y degradaría notablemente los resultados de todo el proceso.

A lo largo de esta tesina se trabajó con un número importante de métricas existentes en la literatura. El procedimiento habitual consiste en utilizar el resultado de aplicar una de ellas y, utilizando su valor como ponderación de cada sentencia, establecer un orden de importancia entre las distintas unidades textuales que componen el documento. En base a esto, cada artículo de la revista fue debidamente preprocesado, descompuesto en oraciones y almacenado en una base de datos. Para cada oración se calcularon y almacenaron los valores de las 21 métricas mencionadas en la sección 2.4

Una vez ordenadas las sentencias sólo queda definir el tamaño del resumen a generar (generalmente expresado como un porcentaje del tamaño del documento) y seleccionar las mejor puntuadas. Dicho de esta forma parece un proceso sencillo cuya única complejidad radica en el tiempo de cálculo de las métricas de cada sentencia. Sin embargo queda por definir el aspecto subjetivo del resumen que se encuentra relacionado con el "criterio" del lector. Sentencias que pueden ser significativas para una persona pueden no serlo en la misma medida para otra y es aquí donde el método propuesto en esta tesina cobra valor. La pregunta es ¿cómo aprender dicho criterio?

La respuesta a esta pregunta fue planteada a partir de un modelo predictivo: el combinador lineal. Este modelo posee algunas ventajas y desventajas. Como ventajas debe decirse que es fácil de programar y tanto su construcción como su aplicación tienen un tiempo de cálculo muy bajo. Ambas características son ideales a la hora de seleccionar un modelo. Su característica desfavorable es compartida por todos los modelos predictivos y tiene que ver con el uso del aprendizaje supervisado, es decir

que para los ejemplos de entrada debe conocerse la respuesta esperada. En el caso de esta tesina, para cada sentencia debe disponerse de un valor numérico que represente su nivel de importancia. Esto parece contradictorio. Para aprender a resumir, es preciso contar con un resumen. En realidad, no es un problema tan grave ya que no se requiere de un resumen demasiado largo aunque de todas formas, debe ser generado de alguna manera. En el presente trabajo se utilizó la ponderación dada por un generador de resúmenes web pero bien podría haber sido generado por el futuro lector. Es importante pensar que esto se realiza una única vez y el modelo generado podrá ser aplicado muchas veces, siempre que la temática no cambie.

En este punto, cada oración de los documentos (artículos) del corpus tiene calculadas las 21 métricas y la ponderación dada por el lector (o la aplicación web).

Una suposición muy fuerte pero que ha arrojado resultados muy alentadores fue considerar que la ponderación indicada para cada sentencia podría ser predicha utilizando una combinación lineal de métricas. Esto llevó a medir distintas alternativas generadas a partir de un combinador lineal seleccionando distintas métricas como atributos de entrada. Como resultados importantes cabe mencionar que salvo la versión en la que las métricas fueron seleccionadas a través del combinador podado, el resto arrojaron muy buenos resultados llegando a superar el 90 % de acierto cuando se trata de generar un resumen con el 10 % de las oraciones y superior al 80 % para un tamaño del 25 %. Esto confirma la factibilidad de predecir la importancia de las sentencias de manera similar a como lo haría el lector.

Con respecto a la selección de las métricas, si bien se observa que depende del conjunto de documentos que se tomen para realizar el entrenamiento del combinador lineal, se identifica un núcleo común formado por las métricas: pos_l, pos_f, len_w, len_ch, tfisf, d_cov_j, graph_s, graph_e y luhn_deg.

A futuro se propone medir la relación entre las métricas seleccionadas y el tipo de documento o la temática del CORPUS. En este caso, los artículos utilizados poseen una estructura bien marcada e impuesta por el formato de la revista PLOS medicine. Esto provoca que métricas como pos_l y pos_f se encuentren entre las más seleccionadas. Esto podría cambiar si se tratara de un cuento infantil donde una parte importante de las oraciones serán relevantes al momento de interpretar la historia narrada.

Además, se espera ampliar el conjunto de métricas, sin descartar mejoras sobre

las ya implementadas. Dentro de la ampliación se considera la posibilidad de implementar algunas con enfoque abstractivo. Esto implicaría a su vez, modificar la forma de representación de los documentos. También es de interés considerar la utilización del método propuesto en base a resúmenes de múltiples documentos.

Una vez completados los puntos anteriores, sería interesante analizar la incidencia del idioma en el que está escrito el documento en el resumen final obtenido ya que la obtención de las raíces de las palabras cambia. Esto modificará el resultado de las métricas y por consiguiente su participación en la ponderación de las sentencias. De todas formas, este aspecto no invalida el funcionamiento general del método propuesto en esta tesina. De todas formas, es un aspecto que aún no se ha analizado en profundidad.

Finalmente se repetirán estas mismas pruebas utilizando otras marcaciones para los documentos, incluyendo no sólo las nuevas que se puedan generar en forma automática sino también otras construidas manualmente.

Finalmente sería importante considerar en el análisis el estilo utilizado por el autor; aunque este aspecto en una primera etapa estaría resuelto por la ponderación de las sentencias por parte del lector al momento de construir el conjunto de oraciones que se utilizarán para entrenar el combinador lineal.

Apéndice A

Recuperación desde documentos XML

Un documento XML (*Extensible Markup Language*) es como un árbol ordenado y etiquetado, donde cada hoja del árbol es un elemento XML escrito entre etiquetas, una de apertura y una de cierre, y a su vez, cada elemento puede tener más de un atributo XML (Manning, Raghavan y Schütze, 2008).

En el siguiente extracto de documento con formato XML, el elemento *biblioteca* tiene una etiqueta de apertura <biblioteca>y una de cierre </biblioteca>, al igual que todos los elementos que contiene *biblioteca*:

```
<?xml version="1.0" encoding="UTF-8"?>
<biblioteca>
        <libro>
            <titulo>"Cien años de soledad"</titulo>
            <autor AñoNac="1927">Gabriel García Márquez</autor>
            <FechaPublicacion>1967</FechaPublicacion>
            <Pais>Colombia</Pais>
        </libro>
        <libro>
            <titulo>"1984"</titulo>
            <autor>George Orwell</autor>
            <FechaPublicacion>1949</FechaPublicacion>
            <Pais>Reino Unido</Pais>
        </libro>
</biblioteca>
```

Las hojas del árbol que se forma de la estructura son texto plano, como *Colombia* o *George Orwell*. En cambio, los nodos internos del árbol representan tanto la estructura del documento, en el ejemplo titulo, autor, fecha de publicación y país, como los metadatos, tales como la fecha de nacimiento del autor García Márquez.

Tener la colección de documentos en este formato ayuda a los sistemas de recuperación de información a sacar ventaja de la estructura recuperando los componentes individuales del documento por separado. De los documentos de texto con estructura XML, algunos de los componentes que se pueden extraer son párrafos, secciones, artículos, bibliografías, entre otros elementos (Büttcher, Clarke y Cormack, 2010). Cada uno de estos elementos se pueden tomar como la parte esencial a recuperar de los documentos, así como también se pueden utilizar para el guardado en un modelo de bases de datos relacional. Se extrae la información desde los documentos XML en base a la estructura de sus componentes y se los guarda manteniendo esta estructura pero en una base de datos.

Otra de las características importantes de los documentos XML que beneficia a la extracción de información, es que soportan una estructura recursiva. En el siguiente fragmento de código se puede observar un documento XML que representa un artículo, el cual posee secciones y subsecciones.

```
<?xml version="1.0" encoding="UTF-8"?>
<body>
    <sec id="sec008" sec-type="intro">
        <title>Introducción</title>
        <p>...</p>
    </sec>
    <sec id="sec009" sec-type="Materiales|Métodos">
        <title>Métodos</title>
        <sec id="sec009a">
            <title>Protocolo y plan de análisis</title>
            <p>...</p>
        </sec>
        <sec id="sec009b">
            <title>Participantes, configuraciones y ética</title>
            <p>Se seleccionó una población de personas
            entre los 18 y 70 años de edad.</p>
```

```
        </sec>
    </sec>
    <sec id="sec010" sec-type="resultados">
        <title>Resultados</title>
        <p>...</p>
    </sec>
</body>
```

Como se puede observar, la sección *"sec009"* tiene dos subsecciones en el ejemplo, *"sec009a"* y *"sec009b"*, las cuales a su vez podrían tener otras subsecciones.

Una vez extraído el texto y los meta-datos de los documentos, se deben guardar de manera que se puedan recuperar, tanto para rearmar el documento en su orden original como para poder calcular y analizar para realizar el resumen. Por tal motivo, toda la información se persiste de manera estructurada en una base de datos.

Apéndice B

Almacenamiento de documentos de manera estructurada

Para poder alcanzar el método propuesto en el presente trabajo, fue necesario estructurar el conjunto de documentos con los que se trabajó. Para ello se utilizó un diseño de base de datos presentado en un trabajo previo (Villa-Monte, Corvi y col., 2019).

En el presente anexo se desarrollará la metodología llevada a cabo para almacenar el conjunto de documentos en dicha base, junto con los recursos y herramientas que se utilizaron para estructurar el texto y así poder manipularlo.

El diseño de esta base fue realizado especialmente para artículos científicos considerando las necesidades de los resúmenes extractivos. En general los documentos científicos cuentan con una estructura común donde primero tienen el título del artículo, luego el listado de autores e información adicional de los mismos, esto seguido del resumen del artículo, las palabras claves y finalmente el cuerpo del documento.

En el cuerpo del artículo el contenido se estructura generalmente en secciones y subsecciones, donde cada sección es un conjunto de párrafos y cada párrafo un conjunto de oraciones. Al final del cuerpo se pueden observar la mayoría de las veces las secciones de resultados obtenidos y conclusiones, así como la sección de la bibliografía utilizada en el documento. Los documentos utilizados en el presente trabajo poseen formato XML. Este formato y la manera de extraer de los documentos la información textual, se explica en el Apéndice A.

Una vez obtenida la información de los artículos científicos, se debe almacenar la misma en alguna estructura que permita rearmar los documentos manteniendo su

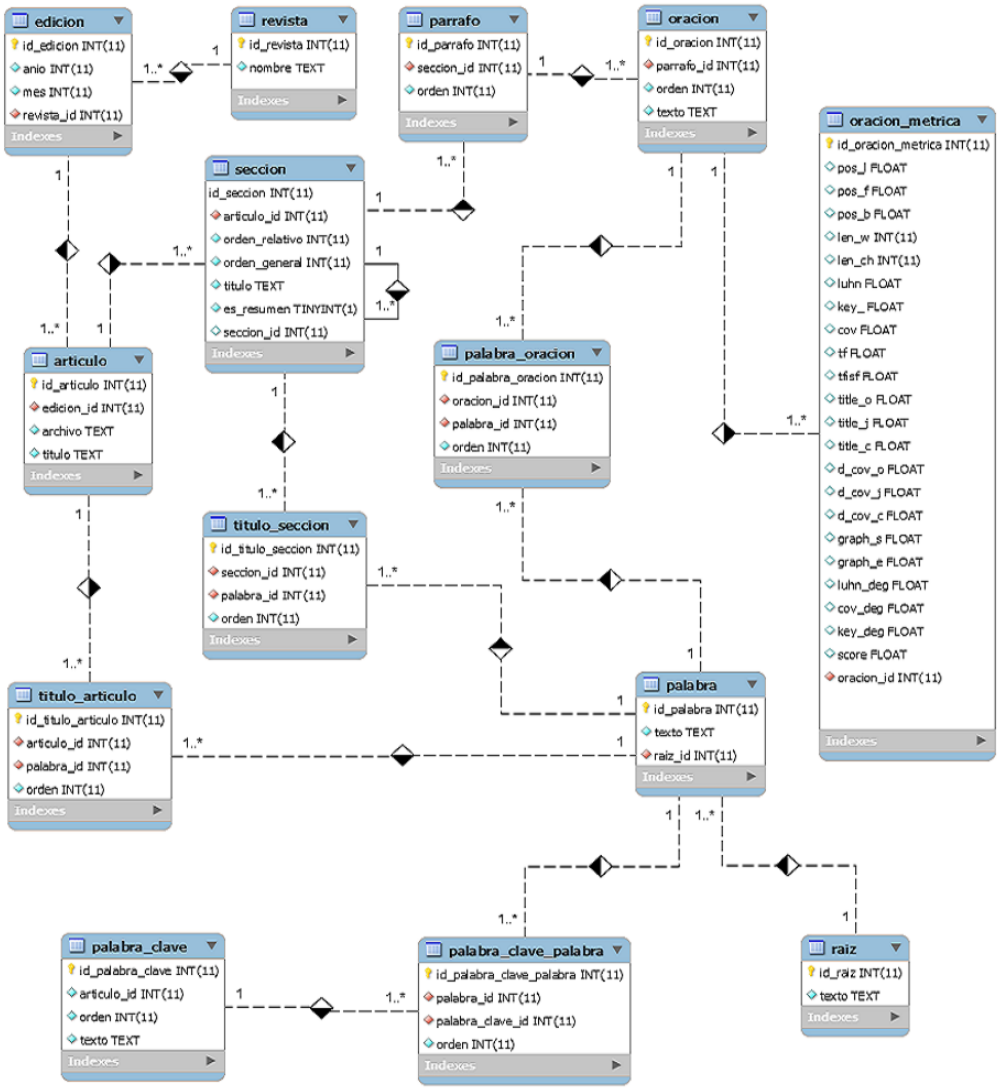


FIGURA B.1: Modelo de base de datos utilizado en esta tesina para almacenar los documentos de los artículos científicos.

forma original en cualquier momento. En la figura B.1 se puede observar en detalle el modelo utilizado en esta tesina.

El diseño consta de catorce tablas en donde se almacenó toda la información de los artículos científicos extraídos de la revista *PLOS Medicine* (ONE, 2004). El modelo tiene en cuenta la revista, la edición de la misma y los artículos publicados en cada edición (tablas *revista*, *edicion* y *articulo*).

Todo el texto del documento se almacena dividido según la estructura antes mencionada. Se posee una tabla *seccion* para almacenar a la mayor subdivisión de texto dentro del cuerpo del documento. Cada *seccion* posee un atributo orden general, teniendo en cuenta el orden dentro del documento completo y un orden relativo. El

orden relativo se refiere a que una sección puede contener una o más secciones dentro de ella, por lo que se debe almacenar un orden relativo dentro de la recursividad de secciones.

Así mismo, las secciones pueden contener otras secciones o párrafos. Los párrafos poseen oraciones y las oraciones están comprendidas por un conjunto de palabras. Toda esta información se almacena en las tablas *parrafo*, *oracion* y *palabra* respectivamente.

Cada palabra del texto tiene una raíz, la cual se obtiene mediante un algoritmo de *stemming* desarrollado en el preprocesamiento del texto (para más información volver al capítulo 1 sección 1.4.2). Estas raíces son almacenadas en una tabla *raiz*, la cual posee una relación directa con las palabras que la poseen.

Habiendo descrito la estructura utilizada, se presentarán las consideraciones que se tuvieron a la hora de almacenar los documentos en esta base.

Una vez obtenidos los artículos del sitio web de la revista (ONE, 2004), se llevaron a cabo las siguientes acciones para cada uno de ellos:

- Se extrajo el título y el resumen de autor.
- Se procesaron cada una de las secciones, almacenando sus títulos en la tabla *titulo_seccion*. Por cada sección debió verificarse que si poseían otra sección, esta también tuviera el procesamiento adecuado, debido a la estructura recursiva de los documentos. La secciones como *"Bibliografía"* o las que poseían sólo imágenes, tablas o ecuaciones fueron descartadas en esta parte del proceso.
- Si la sección obtenida poseía un párrafo, se debía trabajar el mismo para obtener todas sus partes hasta llegar a las palabras y sus raíces. Las oraciones dentro de los párrafos se obtuvieron haciendo el corte en el punto final delimitador, considerando los casos en los que el punto fue utilizado como separador decimal o como parte de una abreviatura, entre otros casos.
- Tanto el título del artículo como el título de cada sección se almacenaron en tablas específicas y a su vez, se almacenaron las palabras que contenían dichos títulos. Esto en base a un tipo de medida de los resúmenes extractivos que se relaciona con los títulos de los textos.

El resultado final de todo el proceso de almacenado de los documentos permite que se pueda continuar con el proceso de análisis de los datos. En este punto, dentro

del desarrollo del método propuesto para la tesina, se pudo comenzar con el cálculo de las métricas definidas en 2.4.

A partir del conjunto de documentos almacenado, por cada oración se calcula cada métrica de puntuación. Cada una de estas medidas se almacenan en la tabla *oracion_metrica*, la cual posee una entrada por cada oración de los documentos y tiene una columna por cada tipo de métrica. Formando así una matriz donde las filas son las oraciones y los valores de las métricas se almacenan en la celda de esa fila, columna de la métrica calculada.

Bibliografía

Ball, G.H. y D.J. Hall (1965). *Isodata: A Method of Data Analysis and Pattern Classification*. Stanford Research Institute.

Baumel, Tal, Raphael Cohen y Michael Elhadad (2014). «Query-Chain Focused Summarization». En: vol. 1, págs. 913-922. DOI: 10.3115/v1/P14-1086.

Baxendale, P. B. (1958). «Machine-made Index for Technical Literature: An Experiment». En: *IBM J. Res. Dev.* 2.4, págs. 354-361. ISSN: 0018-8646. DOI: 10.1147/rd.24.0354.

Büttcher, Stefan, Charles Clarke y Gordon V. Cormack (2010). *Information Retrieval: Implementing and Evaluating Search Engines*. The MIT Press. ISBN: 0262026511.

Carenini, Giuseppe, Jackie Chi y Kit Cheung (2008). «Extractive vs. NLG-based Abstractive Summarization of Evaluative Text: The Effect of Corpus Controversiality». En:

Davies, D. L. y D. W. Bouldin (1979). «A Cluster Separation Measure». En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.2, págs. 224-227. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1979.4766909.

Edmundson, H. P. (1969). «New Methods in Automatic Extracting». En: *J. ACM* 16, págs. 264-285.

Erisoglu, Murat, Nazif Calis y Sadullah Sakallioglu (2011). «A new algorithm for initial cluster centers in k-means algorithm». En: *Pattern Recognition Letters* 32.14, págs. 1701-1705. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2011.07.011.

Erkan, Günes y Dragomir R. Radev (2004). «LexRank: Graph-based Lexical Centrality As Salience in Text Summarization». En: *J. Artif. Int. Res.* 22.1, págs. 457-479. ISSN: 1076-9757. URL: http://dl.acm.org/citation.cfm?id=1622487.1622501.

Fayyad, Usama, Gregory Piatetsky-Shapiro y Padhraic Smyth (1996). «The KDD process for extracting useful knowledge from volumes of data». En: *Commun. ACM* 39.11, págs. 27-34. ISSN: 0001-0782.

Feldman, Ronen y James Sanger (2006). *Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. New York, NY, USA: Cambridge University Press. ISBN: 9780521836579.

Freeman, James A. y David M. Skapura (1993). *Redes neuronales : algoritmos, aplicaciones y técnicas de programación*, pág. 431. ISBN: 978-0-201-60115-2.

García, Salvador, Julin Luengo y Francisco Herrera (2014). *Data Preprocessing in Data Mining*. Springer Publishing Company, Incorporated. ISBN: 9783319102467.

Gruber, Thomas R. (1993). «A Translation Approach to Portable Ontology Specifications». En: *Knowl. Acquis.* 5.2, págs. 199-220. ISSN: 1042-8143. DOI: 10.1006/knac.1993.1008. URL: http://dx.doi.org/10.1006/knac.1993.1008.

Hahn, Udo e Inderjeet Mani (2000). «The Challenges of Automatic Summarization». En: *Computer* 33.11, págs. 29-36. ISSN: 0018-9162. DOI: 10.1109/2.881692. URL: http://dx.doi.org/10.1109/2.881692.

Haykin, Simon S. (2009). *Neural networks and learning machines*. Third. Upper Saddle River, NJ: Pearson Education.

Hernández Orallo, José, María José Ramírez Quintana y César Ferri Ramírez (2004). *Introducción a la Minería de Datos*. Pearson Educación, pág. 680.

Inselberg, A. y B. Dimsdale (1990). «Parallel coordinates: a tool for visualizing multi-dimensional geometry». En: *Proceedings of the First IEEE Conference on Visualization: Visualization '90*, págs. 361-378. DOI: 10.1109/VISUAL.1990.146402.

Isasi, Pedro e Inés Galván (2004). *Redes neuronales artificiales: un enfoque práctico*.

Jones, Karen Spärck (2007). «Automatic summarising: The state of the art». En: *Information Processing & Management* 43.6. Text Summarization, págs. 1449-1481. ISSN: 0306-4573. DOI: https://doi.org/10.1016/j.ipm.2007.03.009. URL: http://www.sciencedirect.com/science/article/pii/S0306457307000878.

Kallel F. J. Jaoua M., Hadrich L. B. y Hamadou A. B. (2004). «Summarization at LARIS Laboratory. In Proceedings of the document understanding conference.» En:

Khan, Atif y Naomie Salim (2014). «A review on abstractive summarization methods». En: *Journal of Theoretical and Applied Information Technology* 59.1, págs. 64-72. ISSN: 1992-8645.

Khan, Shehroz S. y Amir Ahmad (2004). «Cluster center initialization algorithm for K-means clustering». En: *Pattern Recognition Letters* 25.11, págs. 1293-1302. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2004.04.007. URL: http://www.sciencedirect.com/science/article/pii/S0167865504000996.

Kumar, Yogan Jaya, Ong Sing Goh y col. (2016). «A Review on Automatic Text Summarization Approaches». En: *JCS* 12.4, págs. 178-190. DOI: 10.3844/jcssp.2016.178.190. URL: https://doi.org/10.3844/jcssp.2016.178.190.

Kumar, Yogan Jaya y Naomie Salim (2012). «Automatic Multi Document Summarization Approaches». En: *K.S. Gayathri, Received B.E degree in CSE from Madras University in 2001 and M.E degree from Anna University, Chennai. She is doing Ph.D. in the area of Reasoning in Smart*.

Kupiec, Julian, Jan Pedersen y Francine Chen (1995). «A Trainable Document Summarizer». En: *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '95. Seattle, Washington, USA: ACM, págs. 68-73. ISBN: 0-89791-714-6. DOI: 10.1145/215206.215333. URL: http://doi.acm.org/10.1145/215206.215333.

Larocca Neto, Joel y col. (2000). «Generating Text Summaries through the Relative Importance of Topics». En: *Advances in Artificial Intelligence. IBERAMIA 2000, SBIA 2000. Lecture Notes in Computer Science, vol 1952*. Ed. por Maria Carolina Monard y Jaime Simão Sichman. Berlin, Heidelberg: Springer Berlin Heidelberg, págs. 300-309. ISBN: 978-3-540-44399-5.

Liang, Yanhong y Runhua Tan (2007). «A Text-Mining-based Patent Analysis in Product Innovative Process». En: *IFIP CAI*.

Litvak, Marina y col. (2010). *Towards multi-lingual summarization: A comparative analysis of sentence extraction methods on English and Hebrew corpora*.

Liu, Ling y M. Tamer Özsu, eds. (2009). *Encyclopedia of Database Systems*. Springer US. ISBN: 978-0-387-35544-3. DOI: 10.1007/978-0-387-39940-9. URL: https://doi.org/10.1007/978-0-387-39940-9.

Lloyd, S. (2006). «Least Squares Quantization in PCM». En: *IEEE Trans. Inf. Theor.* 28.2, págs. 129-137. ISSN: 0018-9448. DOI: 10.1109/TIT.1982.1056489. URL: http://dx.doi.org/10.1109/TIT.1982.1056489.

Luhn, H. P. (1957). «A Statistical Approach to Mechanized Encoding and Searching of Literary Information». En: *IBM J. Res. Dev.* 1.4, págs. 309-317. ISSN: 0018-8646. DOI: 10.1147/rd.14.0309. URL: http://dx.doi.org/10.1147/rd.14.0309.

— (1958). «The Automatic Creation of Literature Abstracts». En: *IBM J. Res. Dev.* 2.2, págs. 159-165. ISSN: 0018-8646. DOI: 10.1147/rd.22.0159. URL: http://dx.doi.org/10.1147/rd.22.0159.

Manning, Christopher D., Prabhakar Raghavan e Hinrich Schütze (2008). *Introduction to Information Retrieval*. Cambridge, UK: Cambridge University Press. ISBN: 978-0-521-86571-5. URL: http://nlp.stanford.edu/IR-book/information-retrieval-book.html.

McKeown, Kathleen y Dragomir R. Radev (1995). «Generating Summaries of Multiple News Articles». En: *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '95. Seattle, Washington, USA: ACM, págs. 74-82. ISBN: 0-89791-714-6. DOI: 10.1145/215206.215334. URL: http://doi.acm.org/10.1145/215206.215334.

Naderi, Nona (2012). *Mutation Impact Analysis System: Automated Extraction of Protein Mutation Impacts from the Biomedical Literature*. Germany: LAP Lambert Academic Publishing. ISBN: 9783659141164.

Nenkova, Ani y Kathleen McKeown (2012). «A Survey of Text Summarization Techniques». En: *Mining Text Data*. Ed. por Charu C. Aggarwal y ChengXiang Zhai. Springer, págs. 43-76. ISBN: 978-1-4614-3223-4.

Neto, Joel Larocca y col. (2000). *Document Clustering and Text Summarization*.

Nobatay, Chikashi y col. (2001). «Sentence extraction system assembling multiple evidence». En: *In Proceedings of the Second NTCIR Workshop Meeting*, págs. 5-213.

«PLOS Medicine 2004-2016» (2004). En: *Revista de investigación en el campo de la medicina con licencia Creative Commons Attribution (CC BY) de PLOS (Public Library of Science), una organización sin fines de lucro.* Ed. por PLOS ONE. URL: https://journals.plos.org/plosmedicine/.

Online summarize tool. (2019). URL: https://www.tools4noobs.com/summarize/ (visitado 10-03-2019).

Porter, M.F. (1980). «An algorithm for suffix stripping». En: *Program* 14.3, págs. 130-137. DOI: 10.1108/eb046814. eprint: http://www.emeraldinsight.com/doi/pdf/10.1108/eb046814. URL: http://www.emeraldinsight.com/doi/abs/10.1108/eb046814.

Rajaraman, A. y J.D. Ullman (2011). *Mining of Massive Datasets*. Mining of Massive Datasets. Cambridge University Press. ISBN: 9781107015357.

Rosenblatt, Frank (1958). «The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain». En: *Psychological Review* 65, págs. 386-408.

Saggion, Horacio y Thierry Poibeau (2013). «Automatic Text Summarization: Past, Present and Future». En: *Multi-source, Multilingual Information Extraction and Summarization*. Ed. por Thierry Poibeau y col. Berlin, Heidelberg: Springer Berlin Heidelberg, págs. 3-21. ISBN: 978-3-642-28569-1. DOI: 10.1007/978-3-642-28569-1_1. URL: https://doi.org/10.1007/978-3-642-28569-1_1.

Sparck Jones, K. (1999). «Automatic summarising: Factors and directions.» En: *Advances in Automatic Text Summarization*. Ed. por Mark T. Maybury. Cambridge, MA, USA: MIT Press, págs. 1-14. ISBN: 0262133598.

Tauchmann, Christopher y col. (2018). «Beyond Generic Summarization: A Multi-faceted Hierarchical Summarization Corpus of Large Heterogeneous Data». En: *Proceedings of the 11th Language Resources and Evaluation Conference*. Miyazaki, Japan: European Language Resource Association. URL: https://www.aclweb.org/anthology/L18-1503.

Vanderwende, Lucy y col. (2007). «Beyond SumBasic: Task-focused Summarization with Sentence Simplification and Lexical Expansion». En: *Inf. Process. Manage.* 43.6, págs. 1606-1618. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2007.01.023. URL: http://dx.doi.org/10.1016/j.ipm.2007.01.023.

Venkateswarlu, N.B. y P.S.V.S.K. Raju (1992). «Fast isodata clustering algorithms». En: *Pattern Recognition* 25.3, págs. 335-342. ISSN: 0031-3203. DOI: https://doi.org/10.1016/0031-3203(92)90114-X. URL: http://www.sciencedirect.com/science/article/pii/003132039290114X.

Villa-Monte, A., J. Corvi y col. (2019). «Text pre-processing tool to increase the exactness of experimental results in summarization solutions». En: *XXIV CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACION (CACIC 2018)*. CACIC 2018. Tandil, Buenos Aires, Argentina, págs. 481-490. ISBN: 978-950-658-472-6. DOI: http://cacic2018.exa.unicen.edu.ar/pdf/LibroDeActasCACIC2018.pdf.

Villa-Monte, A., L. Lanzarini y col. (2016). «Document summarization using a scoring-based representation». En: *2016 XLII Latin American Computing Conference (CLEI)*, págs. 1-7. DOI: 10.1109/CLEI.2016.7833396.

Wang, William Yang y Kathleen R. McKeown (2010). «"Got You!": Automatic Vandalism Detection in Wikipedia with Web-based Shallow Syntactic-semantic Modeling». En: *Proceedings of the 23rd International Conference on Computational Linguistics*. COLING '10. Beijing, China: Association for Computational Linguistics, págs. 1146-1154. URL: http://dl.acm.org/citation.cfm?id=1873781.1873910.

Widrow, B. y M. E. Hoff (1960). «Adaptive Switching Circuits». En: *1960 IRE WESCON Convention Record*. Reprinted in *Neurocomputing* MIT Press, 1988 ., págs. 96-104.

Widrow, Bernard y Samuel D. Stearns (1985). *Adaptive Signal Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc. ISBN: 0-13-004029-0.

Witten, Ian H. y Eibe Frank (2005). *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN: 0120884070.

Wu, Chia-Wei y Chao-Lin Liu (2003). «Ontology-based Text Summarization for Business News Articles». En: *Proceedings of the ISCA 18th International Conference Computers and Their Applications, Honolulu, Hawaii, USA, March 26-28, 2003*, págs. 389-392.

Yao, Jin-Ge, Xiaojun Wan y Jianguo Xiao (2017). «Recent Advances in Document Summarization». En: *Knowl. Inf. Syst.* 53.2, págs. 297-336. ISSN: 0219-1377. DOI: 10.1007/s10115-017-1042-4. URL: https://doi.org/10.1007/s10115-017-1042-4.