3 OBTENCIÓN AUTOMÁTICA DE TÉRMINOS Y SU VALIDACIÓN

La *obtención automática de términos* es el uso de sistemas de extracción de información terminológica para la recuperación de unidades terminológicas de un determinado corpus. La *validación* de estas unidades terminológicas es el proceso que consiste en aprobar o desaprobar su estatus de término y obtener como resultado listas de términos con una alta probabilidad de pertenecer a una determinada área de conocimiento.

A través de los capítulos anteriores se ha dado a conocer el marco teórico que permite establecer la obtención automática de términos y su validación. Asimismo, se han dado a conocer algunos de los extractores terminológicos más representativos que por el tipo de conocimiento que emplean, por su restricción temática o por su falta de validación de términos no pueden ser empleados en todos los casos para llevar a cabo una extracción terminológica. Por lo anterior, es que se decidió desarrollar un extractor terminológico que pueda ser empleado para el español y que la validación de términos no esté acotada a una sola área específica.

A lo largo de este capítulo se dará a conocer el corpus que se empleó para la extracción terminológica. Se explicará el método que se usó para la obtención de términos. Posteriormente, se indicará el procedimiento para la validación de las unidades terminológicas. Finalmente se dará a conocer la arquitectura del sistema desarrollado para esta tesis.

3.1 Corpus de textos científicos en español de México (COCIEM)

Como se vio en la sección 1.1.1, uno de los recursos empleados dentro de PLN son los corpus lingüísticos. En este proyecto se empleará un corpus informatizado llamado el Corpus de textos científicos en español de México (COCIEM).

Este corpus fue creado por la Dra. María Pozzi en el proyecto de investigación "El vocabulario básico científico en México: Una investigación de sus características, componentes y difusión", patrocinado por el Consejo Nacional de Ciencia y Tecnología (CONACyT) con número de proyecto 58923.

La construcción del COCIEM se llevó a cabo para poder identificar y obtener el vocabulario básico científico mexicano, es decir, el vocabulario científico elemental que debería ser conocido por un hablante promedio al final de su educación media superior. Posteriormente, a partir del vocabulario obtenido, será posible crear diccionarios o glosarios, así como también emplearlo como recurso lingüístico en diversas aplicaciones de la ingeniería lingüística, por ejemplo para la traducción automática, la generación automática de texto, etcétera.

El COCIEM está conformado por los libros de texto de mayor uso a nivel nacional por los estudiantes de educación básica y media superior. Dentro de estos libros de texto se encuentran libros de teoría, de prácticas de laboratorio, de ejercicios, etc., esto para tener una representatividad del conocimiento pre-universitario.

Cada uno de los libros que pertenecen al COCIEM fue digitalizado y se almacenó en un archivo de texto plano.

En esta tesis, se consideró al COCIEM como un buen corpus para llevar a cabo la extracción terminológica por su gran variedad de materias y porque por ser de carácter educativo tendría que contener una gran cantidad de términos.

3.1.1 Estructura del COCIEM

El Corpus de textos científicos en español de México está conformado por 92 libros de texto divididos en tres niveles educativos: primaria, secundaria y bachillerato. A su vez, cada uno de los niveles educativos se encuentra dividido por año escolar.

Los 92 libros que forman parte del COCIEM son de materias que tienen un enfoque meramente científico, es decir, no son libros de asignaturas que estén relacionadas con las humanidades, como lo es la historia, la ética, la literatura o geografía.

En la Tabla 9 se muestra la estructura del COCIEM por nivel educativo y por materia en general.

Nivel educativo	Materia	Número de libros de texto	Número de tokens	Número de tipos
Primaria	Ciencias naturales	*6*	*175,240*	*11,437*
	Matemáticas	*9*	*125,723*	*9,377*
	Total	*15*	*300,963*	*20,814*
Secundaria	Biología	*8*	*369,099*	*23,908*
	Matemáticas	*24*	*734,374*	*55,797*
	Física	*9*	*538,042*	*29,759*
	Química	*8*	*382,697*	*23, 031*
	Educación ambiental	*1*	*73,247*	*7,922*
	Total	*50*	*2,097,459*	*140,417*
Bachillerato	Biología	*3*	*133,262*	*8,307*
	Matemáticas	*11*	*499,552*	*32,566*
	Física	*3*	*219,795*	*14,251*
	Química	*5*	*156,192*	*11,162*
	Educación para la salud	*3*	*139,369*	*9,421*
	Ecología	*2*	*124,799*	*7,731*
	Total	*27*	*1,272,969*	*83,438*
	Gran total	*92*	*3,671,391*	*244,669*

Tabla 9. Estructura del COCIEM por niveles educativos y materias

Como se puede observar en la tabla anterior, el número de libros que conforman el corpus es diferente para cada uno de los niveles educativos y materias. Esto se contrapone con lo dicho en la sección 1.1.1, en el cual se indicaba que un corpus lingüístico debía ser lo más equilibrado posible, pero como indica Sierra (2008), en los corpus especializados es de esperarse que existan áreas o categorías con una mayor cantidad de textos que otras.

3.2 Preprocesamiento del COCIEM

Antes de llevar a cabo la extracción de términos dentro del COCIEM, cada uno de los textos recibió diversos tratamientos. Este conjunto de tareas son la revisión y limpieza de los

documentos, la lematización, la tokenización y la creación de n-gramas; estas tareas forman lo que se conoce como preprocesamiento.

3.2.1 Revisión, limpieza y adecuación de los documentos

La primera tarea dentro del preprocesamiento del COCIEM fue la revisión, limpieza y adecuación de cada uno de los documentos para las tareas que vendrían posteriormente, como la extracción terminológica.

Debido a que los libros del COCIEM fueron escaneados y se empleó un *reconocedor óptico de caracteres (OCR)*, existían casos en los cuales el reconocimiento de los caracteres no era el adecuado y por tanto existían errores. Por ello era necesario que se llevara a cabo una revisión, al menos a grandes rasgos, de los documentos y se eliminaran los errores existentes. Por ejemplo, un error concurrente dentro de los documentos del COCIEM era la aparición del símbolo de negación (¬) entre los caracteres de una palabra.

De igual manera, para tratar de obtener términos solamente del texto y eliminar la posibilidad de encontrar cadenas muy largas unidas por punto o diagonales, se eliminaron correos electrónicos y páginas web de los documentos.

Además, cada uno de los textos, fueron guardados con la codificación UTF-8 para tener un manejo estándar entre las diversas codificaciones existentes.

3.2.2 Lematización usando FreeLing

La lematización de documentos es una de las tareas, ciertas veces esenciales, para poder llevar a cabo un procesamiento de lenguaje natural. Dada esta razón se decidió emplear un lematizador para obtener la forma canónica de cada una de las palabras de los documentos que pertenecían al COCIEM.

El objetivo de la lematización del COCIEM es la reducción y agrupamiento de los candidatos a término que se obtendrán más adelante, pero también obtener formas canónicas que, como se dijo anteriormente (sección 1.1.5), es la forma que se emplea en un diccionario. Por ejemplo, con la lematización “ecosistema” y “ecosistemas” se convierten en sólo “ecosistema” y, por consiguiente, dos candidatos a término se convierten en uno solo.

Para llevar esta tarea a cabo se empleó el lematizador FreeLing en su versión 2.2, el cual es una biblioteca de procesamiento de lenguaje multilingüe de código abierto que provee un amplio conjunto de analizadores del lenguaje para varios idiomas (Padró et al., 2010). Actualmente FreeLing soporta los idiomas español, inglés, catalán, gallego, galés, italiano, portugués y asturiano.

FreeLing emplea diversos recursos y módulos para llevar a cabo el procesamiento de lenguaje natural. Uno de los recursos que emplea es un diccionario que, en su versión para el idioma español, tiene más de 550,000 formas que corresponden a más de 76,000 lemas[28]. Algunos de los módulos que forman parte de FreeLing son los siguientes (Padró et al., 2010):

- **Tokenizador:** Es un herramienta que recibe un texto plano y crea un archivo con los tokens encontrados.
- **Morfo:** Esta herramienta recibe oraciones e indica las posibles anotaciones morfosintácticas de cada una de las palabras de la oración. Dentro de este procesamiento se encuentran sufijos, números, fechas, cantidades (como razones, porcentajes, monedas), símbolos de puntuación, nombres propios, entre otros.
- **Etiquetador POS:** Recibe la información del módulo Morfo y desambigua las posibles anotaciones morfosintácticas que se indicaron para cada una de las palabras de las oraciones. Esto se lleva a cabo para obtener la etiqueta POS más probable con base en toda la información otorgada por el módulo Morfo.

Para la lematización del COCIEM empleando FreeLing, se desarrolló un programa que está conformado de tres módulos, en la Figura 7 se muestra su arquitectura. El primer módulo extrae un documento del directorio del corpus a analizar. El segundo módulo llama al programa de FreeLing para que lleve a cabo la lematización. Y el tercero cambia el formato de salida que otorga FreeLing a uno más entendible para el humano. En la Figura 8 se muestra un texto sin lematizar, mientras que en la Figura 9 se muestra un fragmento del texto anterior ya lematizado por FreeLing.

[28] Esta información fue extraída de la página oficial de FreeLing:

http://nlp.lsi.upc.edu/freeling/index.php?option=com_content&task=view&id=23&Itemid=58

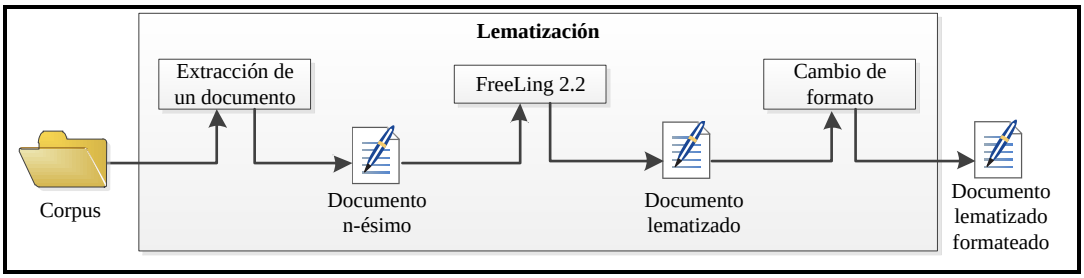

Figura 7. Arquitectura del programa de lematización

María Gaetana Agnesi (1718-1799) fue una matemática italiana que dejó varias contribuciones a la ciencia, entre ellas una curva muy famosa con un nombre sumamente peculiar. Este nombre tan peculiar fue por culpa de un traduttore traditore (traductor traidor) quien confundió la palabra versiera con avversiera que significa bruja. Es por ello que la curva de Agnesi se le conoce como Bruja de Agnesi, no sólo en el español sino en muchos idiomas más.

Figura 8. Un texto que servirá de ejemplo para llevar a cabo la lematización empleando FreeLing

Este este DD0MS0 0.956743	*traidor traidor AQ0MS0 0.509558*
nombre nombre NCMS000 0.97973	*)) Fpt 1*
tan tan RG 1	*quien quien PR0CS000 1*
peculiar peculiar AQ0CS0 1	*confundió confundir VMIS3S0 1*
fue ser VSIS3S0 0.932292	*la el DA0FS0 0.972146*
por por SPS00 1	*palabra palabra NCFS000 1*
culpa culpa NCFS000 0.866667	*versiera versiera VMSI3S0 0.500172*
de de SPS00 0.999919	*con con SPS00 1*
un uno DI0MS0 0.986987	*avversiera avversiera VMSI3S0 0.500172*
traduttore traduttore VMSP3S0 1	*que que CS 0.4375*
traditore traditore VMSP3S0 1	*significa significar VMIP3S0 0.958333*
((Fpa 1	*bruja brujo NCFS000 0.6*
traductor traductor NCMS000 0.490442	*. . Fp 1*

Figura 9. Extracto del archivo de salida generado por FreeLing después de lematizar el texto de la Figura 8

Como se puede observar en la Figura 9, es algo complicado revisar el texto lematizado en el formato de salida propio de FreeLing; es por ello que se desarrolló el tercer módulo, el cual se encarga de cambiar el formato de salida por uno que sea similar al formato original. Para ello, primeramente, se llevó a cabo un análisis de la presentación de salida que otorga FreeLing, el cual se puede observar en la Figura 10.

Palabra original	Lema	Etiqueta POS	Probabilidad

Figura 10. Formato de salida de la lematización realizada por FreeLing

En el área de la palabra original se muestra la palabra o conjunto de ellas de la manera en que se encontraron en el texto. En el área de lema se muestra la forma canónica o un determinado formato en el caso de números, fechas, cantidades, entre otros; cabe aclarar que los lemas se presentan en letras minúsculas. En la etiqueta POS se emplea el formato propuesto por el grupo EAGLES (sección 1.1.4), el cual sirve para todas las lenguas europeas. Finalmente, en la probabilidad, se muestra la probabilidad de que la etiqueta POS elegida para la palabra original sea la correcta.

Para llevar a cabo el cambio de formato se desarrollaron las siguientes reglas:

- Multipalabras: Debido a que FreeLing concatena en algunos casos palabras para formar una multipalabra, como en los nombres propios, fue necesario crear una regla que cambiara los guiones bajos con los que une las palabras por espacios.
- Diagonal (/): La diagonal no solamente expresa un sentido matemático, en ocasiones permite unir palabras que se emplean de manera conjunta. Un ejemplo es el caso de VIH/SIDA el cual al obtener su forma canónica usando FreeLing se separa en tres elementos, “vih”, “/” y “sida”. Por tanto, se creó una regla que uniera estos tres elementos en el cambio de formato para obtener su estructura original.
- Minúsculas: A pesar de que el formato de salida del lema es en minúsculas, existen casos en el cual la primera letra del lema inicia con mayúsculas, esto debido a un error de FreeLing. Por ejemplo, en la frase “Lamentablemente lo que se pide en el examen no se puede realizar.”, la palabra “Lamentablemente” en su forma canónica aparece con mayúscula. Por ello fue necesario en el programa de cambio de formato reconvertir los lemas a minúscula por si ocurrían casos como éste y así mantener uniformemente el formato de salida.
- Saltos de línea: Una de las características de FreeLing es que el análisis de cada una de las oraciones encontradas es separado por un salto de línea extra. Empleando lo anterior, se realizó una regla la cual indica cuándo colocar un salto de línea para indicar que una oración finalizó. En este caso, no se puede saber exactamente la

posición en la que se encontraba la oración en un párrafo, es por ello que no se pueden reconstruir estos como se encontraban en el documento original.

Empleando las reglas anteriores, se permite recuperar hasta cierto punto la estructura o ciertas construcciones que tenía el documento original. En la Figura 11 se muestra el cambio de formato realizado sobre el archivo de salida de FreeLing (Figura 9).

maría gaetana agnesi (1718-1799) ser uno matemática italiano que dejar varios contribución a el ciencia , entre ellos uno curva muy famoso con uno nombre sumamente peculiar .

este nombre tan peculiar ser por culpa de uno traduttore traditore (traductor traidor) quien confundir el palabra versiera con avversiera que significar brujo .

ser por él que el curva de agnesi se le conocer como bruja de agnesi , no sólo en el español sino en mucho idioma más .

Figura 11. Cambio de formato realizado a partir de la salida otorgada por FreeLing

Las razones de llevar el cambio de formato son permitir que se puedan revisar de una manera más sencilla los textos lematizados y se puedan corregir los errores si se desea.

3.2.3 Tokenización del COCIEM

La tokenización es el proceso que consiste en la segmentación de documentos en un conjunto de unidades con significado llamados tokens, como se pudo leer en la sección 1.1.2. Esta tarea forma parte del preprocesamiento que recibió el COCIEM.

Para realizar esta tarea se creó un tokenizador sencillo en Flex (Fast lexical analyzer)[29], que emplea la información otorgada por el programa desarrollado para lematizar el COCIEM (sección 3.2.2), más específicamente del módulo del cambio de formato. Para ello se crearon una serie de expresiones regulares con el objetivo de extraer las construcciones o tokens de los archivos lematizados del corpus y clasificarlos por su tipo (cadena, número, etcétera) al mismo tiempo; esto último se llevó a cabo para que se pueda realizar de forma paralela la creación de n-gramas, del cual se hablará en la siguiente sección (3.2.4).

[29] http://flex.sourceforge.net/

Entre las características que tiene este tokenizador están la detección de palabras unidas por guiones o de palabras unidas por una diagonal; por ejemplo óxido-reducción o ONU/UN. De igual forma, la detección de números, signos de puntuación o de cadenas compuestas por números como lo es H1N1, entre otros.

Asimismo, el tokenizador aprovecha la detección de abreviaturas por parte de FreeLing para tomarlas también como tokens.

3.2.4 Creación de n-gramas

Como se describió en la sección 1.1.3, los n-gramas son uniones de caracteres o de palabras. En el caso de esta tesis lo que se busca es obtener términos multipalabra y, por tanto, es necesario crear n-gramas de palabras y no de caracteres. Con base en la opinión de expertos en el área de terminología es posible obtener una base confiable para la obtención de unidades terminológicas con trigramas. La razón es que a partir de los trigramas ya se pueden visualizar términos comunes muy fácilmente sin emplear tantos recursos computacionales. Por lo tanto, el tamaño máximo de los n-gramas utilizados fue de 3 tokens.

Para llevar a cabo lo anterior, se desarrolló un programa en C que se basa en un ciclo de trabajo para la formación de n-gramas y que emplea la información otorgada por el tokenizador de la sección 3.2.3 para cada uno de los documentos del corpus. El ciclo de trabajo permite crear construcciones de n-gramas con base en los signos de puntuación, los saltos de línea, números y algunos otros marcadores tipográficos, como los paréntesis, las comillas y las llaves. En la Figura 12, se muestra un ejemplo en el cual se generan n-gramas a partir del texto de entrada “A B C D E F. G H I ¶ J K”; donde cada letra representa un token y el símbolo calderón (¶), el fin de un párrafo. Primeramente se genera un unigrama con A (t_1), luego al leer B (t_2) se crea el unigrama de B y el bigrama AB, posteriormente se genera el unigrama C (t_3) al mismo tiempo que el bigrama BC y el trigrama ABC; este proceso se lleva así sucesivamente hasta llegar a t_6, donde se tiene el unigrama F, el bigrama EF y el trigrama DEF. Después, cuando se lee el punto se reinicia el ciclo de trabajo para comenzar desde el principio el ciclo de trabajo. La creación de n-gramas se lleva a cabo hasta que se termina de analizar todo el documento, de esta manera, se puede generar el número exacto de n-gramas siguiendo siempre la estructura del texto.

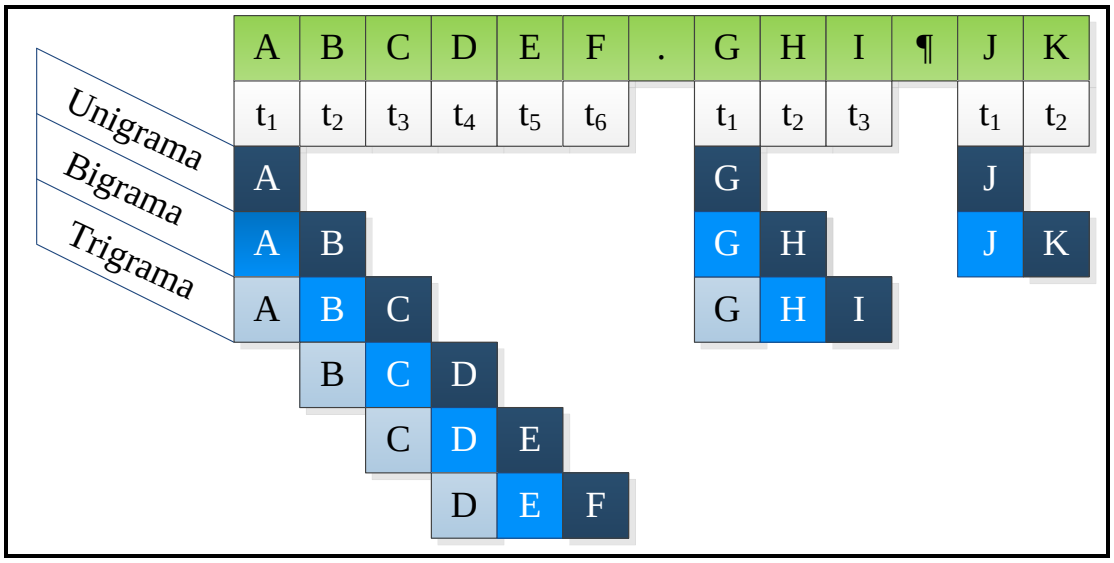

Figura 12. Ejemplo del ciclo de trabajo para la generación de n-gramas. Los cuadros superiores son el texto y el tiempo del ciclo.

El uso de los signos de puntuación, los saltos de línea y otros símbolos para reiniciar el ciclo de trabajo de la creación de n-gramas se basa en la idea de que en teoría ningún término, en el caso de uno multipalabra, tendrá en su interior un signo de puntuación que divida sus partes. De esta manera se pueden disminuir el número de n-gramas a analizar, ya que se reduce el número de formaciones sin sentido o que no existían en el texto original. Por ejemplo, en la frase lematizada "tapar el vaso de precipitado . llenar con agua el matraz erlenmeyer", si no se reiniciara el ciclo con los signos de puntuación se tendría el bigrama "precipitado llenar", aun cuando éste no existe.

Con respecto a los números, estos no son considerados términos, pues el sistema de numeración no es un término en sí, por tanto, con base en la clasificación que otorga el tokenizador desarrollado en la sección 3.2.3, estos se pueden omitir de manera sencilla en los n-gramas y reiniciar asimismo el ciclo de trabajo que los genera. Sin embargo, la única excepción de lo anterior es cuando los números se encuentran unidos por guiones a palabras o forman parte de una palabra, y por tanto de un posible término, por ejemplo el caso de carbono-14 o CH_4; en este caso, estos candidatos son considerados en la generación de n-gramas del ciclo de trabajo como si fueran una cadena de caracteres normal.

Además de la generación de n-gramas, una de las tareas de este módulo es dividir en archivos cada uno de los tipos de n-gramas (unigramas, bigramas y trigramas) de los

documentos del corpus analizado. Esto se realiza para que posteriormente se puedan analizar cada uno de los n-gramas y así extraer de ellos los candidatos a término. En la Figura 13 se muestra un ejemplo del formato de los archivos de salida del generador de n-gramas; en cada línea se encuentra un bigrama encontrado en un documento.

el animal
animal terrestre
punto cardinal
juan tener
energía renovable
localidad rural

Figura 13. Ejemplo de un archivo de n-gramas generado durante el preprocesamiento

3.3 Extracción de candidatos a término

La extracción de candidatos a término del COCIEM se lleva a cabo empleando el método de TF-IDF, el cual fue explicado en la sección 1.2.1. Este método consiste en la asignación de pesos para cada una de las palabras de un documento en corpus, empleando las métricas Term Frequency e Inverse Document Frequency.

Aunque en un principio el método de TF-IDF se empleaba para la creación de listas de palabras claves de sistemas de búsqueda de información, su uso se ha ido ampliando a áreas como la extracción terminológica automática. La razón de lo anterior es que las palabras clave de los sistemas de recuperación de información por lo general son candidatos a término del documento analizado. Por tanto, el método de TF-IDF puede emplearse como método para la extracción terminológica en corpus.

Las ventajas de emplear este método de extracción terminológica es su capacidad de ser independiente de la lengua, su rapidez y sencillez de implementación. De igual manera, TF-IDF tiene como ventaja que permite analizar a la vez una gran cantidad de información de manera paralela y obtener resultados separados, en otras palabras, todos el corpus del COCIEM puede ser analizado por libro al mismo tiempo sin ningún problema y generar una lista de términos por libro. En este caso, TF-IDF se diferencia de otros métodos de extracción

terminológica, como el logaritmo de la verosimilitud donde sólo se pueden comparar dos recursos a la vez, siendo uno de ellos el de referencia y otro el de análisis: un caso práctico es mostrado por Cabrera-Diego et al. (2011); otro caso es el del método C-Value/NC-value donde la extracción terminológica se lleva de manera general a todo un corpus o documento.

La desventaja del método de TF-IDF es que para llevarlo a cabo es necesario que existan al menos dos corpus o documentos, de lo contrario se obtienen pesos iguales a cero, aunque para este caso no es ningún problema por la gran cantidad de libros existentes en el COCIEM. Por ello este método es el adecuado para la extracción terminológica de términos del corpus seleccionado.

En el caso del COCIEM, el método de TF-IDF se aplica a unigramas, bigramas y trigramas, los cuales son generados empleando el método descrito en la sección 3.2.4.

A continuación se mostrarán los algoritmos empleados para llevar a cabo el análisis de TF-IDF y de algunas tareas extras que se realizaron para obtener los candidatos a término.

3.3.1 Cálculo de TF

El primer paso para realizar el método de TF-IDF es el cálculo de la frecuencia relativa de cada uno de los términos (TF) de todos los documentos a analizar. Para ello se creó un programa en C que emplea como entrada cada uno de los tipos de archivos de n-gramas (unigramas, bigramas, trigramas) que se generaron en la sección 3.2.4; su algoritmo se muestra en la Figura 14.

Cálculo de TF
1. Se abre el archivo de n-gramas 2. Se ordena el archivo de n-gramas de manera alfabética 3. Se crea un archivo de salida (S) 4. Se lee la primera línea (L_1) del archivo de n-gramas 5. Mientras L_1!=EOF 6. Se imprime L_1 en S 7. Se imprime en S un tabulador 8. Se imprime en S el número del documento analizado 9. Se imprime en S un tabulador 10. B=0 11. Mientras B!=0

```
12.     Se lee la siguiente línea (L2) del archivo de n-gramas
13.     Si L1==L2
14.         Se aumenta en 1 la frecuencia (F) de L1
15.     De lo contrario
16.         Se imprime F en S
17.         Se imprime en S un salto de línea
18.         B=1 y L1=L2
```

Figura 14. Algoritmo usado para el cálculo de TF

Este programa crea un archivo de salida por cada uno analizado; en la Tabla 10 se muestra un ejemplo de éste, la primera columna es el n-grama, la segunda el identificador (ID) del documento y la tercera el TF de cada uno de los n-gramas analizados, en este caso son bigramas.

N-grama	ID del documento	TF
el animal	1	15
animal terrestre	1	10
punto cardinal	1	12
juan tener	1	6
energía renovable	1	23
localidad rural	1	11

Tabla 10. Ejemplo de un archivo de n-gramas con el identificador de documento y el TF

3.3.2 Limpieza de los n-gramas generados

La limpieza de n-gramas consiste en la eliminación de las construcciones con alta posibilidad de no ser términos. Su objetivo es disminuir el número de candidatos a término obtenidos por el extractor terminológico.

Dado que las unidades terminológicas no empiezan o terminan con palabras vacías como en los ejemplos dados en la sección 2.2.1, la limpieza consiste en la eliminación de los n-gramas que empiecen o terminen con palabras funcionales o números romanos. Asimismo, se eliminan los unigramas que son totalmente palabras funcionales y que no se les haya asignado un peso nulo durante el método de TF-IDF. Con respecto a los números romanos, al ser letras con referente numérico, no se les considera como términos y por tanto es necesario eliminarlos de los archivos de n-gramas; para lo cual se generó una expresión regular que

permitiera la detección de los n-gramas que tuvieran los caracteres de los números romanos (I, V, X, L, C, D, M) o sus combinaciones en los extremos para posteriormente eliminarlos.

La lista de palabras funcionales es un conjunto de palabras comunes que pertenecen al vocabulario general, como preposiciones, conjunciones, artículos, pronombres relativos, entre otros. Asimismo se tienen algunos verbos, como decir, hacer, estar, tener y ser (en este último caso, "ser" como nombre/sustantivo fue diferenciada del verbo, de lo contrario se podrían eliminar candidatos posibles como "ser humano" o "ser vivo"). De igual manera, en la lista de palabras funcionales se encuentran nombres de secciones de libro como capítulo y página. Esta lista de palabras se encuentra en el Anexo A.

Para llevar a cabo esta tarea de limpieza de n-gramas se generó un programa escrito en Perl que permite el uso de distintas listas de palabras vacías; el algoritmo que se empleó para programarlo se muestra en la Figura 15. En la Tabla 11 se muestra el formato de salida de un archivo generado por este programa, donde la primera columna indica el identificador del documento y la segunda el TF del n-grama.

Eliminación de palabras vacías
1. Se abre el archivo que contiene la lista de palabras vacías
2. Se carga la lista de palabras vacías en un hash
3. Se abre el archivo de n-gramas con frecuencias
4. Se crea el archivo de salida (S)
5. Se lee la primera línea (L) del archivo de n-gramas con frecuencias
6. Mientras L!=EOF
7. Se guarda en una cadena (C) la parte del n-grama de L
8. Se extrae de C la primera palabra y se guarda en P
9. Si P no existe en el hash entonces
10. Si P es distinto a un número romano entonces
11. Si C es un unigrama entonces
12. Se imprime L en S
13. De lo contrario se extrae de C la última palabra y se guarda en P
14. Si P no existe en el hash entonces
15. Si P es distinto a un número romano entonces
16. Se imprime L en S

Figura 15. Algoritmo empleado para la limpieza de n-gramas

N-grama	ID del documento	TF
animal terrestre	1	10
punto cardinal	1	12
energía renovable	1	23
localidad rural	1	11

Tabla 11. Ejemplo de un archivo de salida del módulo de limpieza de n-gramas

3.3.3 Cálculo de IDF, TF-IDF y su normalización

Teniendo las listas de n-gramas limpias y con sus valores de TF se procede a realizar el tercer paso para la extracción de candidatos a término. En este paso se realizan tres tareas al mismo tiempo, que es el cálculo de la frecuencia inversa de los documentos (IDF), la multiplicación de las métricas IDF y TF para la obtención de los pesos TF-IDF y la normalización de estos pesos.

Con respecto a la normalización de los pesos de TF-IDF, ésta es frecuente en los sistemas de búsqueda de información para equilibrar los pesos que reciben los documentos largos y cortos, como se observó en la sección 1.2.2. No obstante, en el caso de esta tesis, se emplea la normalización para acotar el valor de los pesos otorgados por el método de TF-IDF entre 1 y 0; esto permite el establecimiento de umbrales que indiquen el peso mínimo para que un término pueda ser considerado como un buen candidato, de manera estática, es decir, que no se tenga que estar calculando según el valor máximo TF-IDF obtenido en cada documento. Por tanto, para realizar la normalización, se eligió el método de normalización de coseno (sección 1.2.2.1), debido a que no tiene una alta complejidad para obtener el factor de normalización, es sencillo de programar y, además, permite obtener pesos de TF-IDF entre 1 y 0.

Para llevar a cabo las tareas dadas a conocer al principio de esta sección se creó un programa escrito en C el cual tiene como entrada los archivos generados por el programa de limpieza de n-gramas (sección 3.3.2). En la Figura 16 se muestra el algoritmo empleado para realizar estas tres tareas.

Cálculo del IDF, TF-IDF y factor de normalización

1. Se unen todos archivos de n-gramas del mismo tipo en uno solo (A).
2. Se ordena A de manera alfabética
3. Se crea un archivo de salida (S)
4. Se crea un arreglo de frecuencias (TF)
5. Se crea un arreglo de TF-IDF (TF_IDF)
6. Se crea un arreglo de columnas (P)
7. P[0]= "B"
8. Se crea un arreglo de factores de normalización (FN)
9. IDF=0, D_1=0, D_2=0, D_3=0, y T=total de documentos
10. Se imprime "GRAMA" en S
11. Para D_3<T; D_3++
12. Se imprime en S un tabulador
13. Se imprime el nombre del libro
14. Se lee la primera línea (L_1) de A
15. Se extrae de L_1 el n-grama (N_1)
16. Se imprime N_1 en S
17. Se extrae de L_1 el número del documento (D_2) donde N_1 se encontraba
18. Se extrae de L_1 la frecuencia (F) de N_1
19. Mientras L_1!=EOF
20. Para D_1 < D_2; D_1++
21. Si D_1!=(D2-1) entonces
22. TF[D_1]=0.0
23. De lo contrario
24. TF[D_1]=F
25. IDF++
26. D_1=D_2
27. Se lee la siguiente línea (L_2) de A
28. Se extrae de L_2 el n-grama (N_2)
29. Se extrae de L_2 el número del documento (D_2) donde N_2 se encontraba
30. Se extrae de L_2 la frecuencia (F) de N_2
31. Si N_1!=N_2
32. Para D_1 < T; D_1++
33. TF[D_1]=0.0
34. IDF=$\log_{10}$(T/IDF)
35. D_3=0
36. Para D_3<T; D_3++
37. Se imprime en S un tabulador
38. TF_IDF[D_3]=TF[D_3]*IDF
39. FN[D_3]+= TF_IDF[D_3]^2
40. Se imprime "="TF_IDF[D_3]/P"$1" en S
41. Si D_3%26==0 entonces
42. P[0]="A"+ (D_3/26)-1

43. P[1]="A"
44. De lo contrario, si D_3>26
45. P[1]+=1
46. De lo contrario
47. P[0]+=1
48. Se imprime en S un salto de línea
49. Se imprime N_2 en S
50. IDF=0, D_1=0, D_2=0, D_3=0
51. L_1=L_2
52. Se crea un archivo para guardar los factores de normalización (C)
53. Se imprime "Factor de normalización" en C
54. Para D_3<T; D_3++
55. FN[D_3]= 1/SQRT(FN[D_3])
56. Se imprime en C un tabulador
57. Se imprime FN[D_3] en C
58. Se imprime en C dos saltos de línea
59. Se concatena el archivo C y S en se guardan en un archivo con formato de hoja de cálculo

Figura 16. Algoritmo para el cálculo de IDF, TF-IDF y del factor de normalización

En la línea 41 de la Figura 16 se verifica si D_3 al dividirlo entre 26 se obtiene un resudio de cero, esto para modificar el valor de P cuando se hayan puesto todas las letras del alfabeto latino, las cuales son 26, en la variable.

Cuando se llega a la línea 52 del algoritmo mostrado en la Figura 16, el archivo de salida tiene el formato mostrado en la Tabla 12, donde la primera columna indica el n-grama analizado, las siguientes son los pesos de TF-IDF dividido por la casilla donde se colocará el factor de normalización para cada libro analizado.

N-GRAMA	LIBRO 1	LIBRO 2	LIBRO 3
animal doméstico	=0/B$1	=0/C$1	=7.63/D$1
animal terrestre	=1.76/B$1	=0/C$1	=6.16/D$1
ecosistema	=0/B$1	=12.88/C$1	=0/D$1
energía renovable	=0/B$1	=0/C$1	=0/D$1
localidad rural	=1.93/B$1	=0/C$1	=3.69/D$1
punto cardinal	=2.11/B$1	=0/C$1	=0.52/D$1

Tabla 12. Ejemplo de un archivo de salida intermedio de la extracción terminológica antes de la normalización

A partir de la línea 52 del algoritmo mostrado en la Figura 16 se procede a llenar las casillas establecidas para el factor de normalización y genera un archivo que está en un formato de hoja de cálculo; al abrir el archivo la normalización se lleva a cabo de manera automática, además de que este formato de archivo tiene la ventaja de que permite analizar las listas n-gramas mucho más fácilmente. En la Tabla 13 se muestra un extracto del archivo de salida de este módulo.

	A	B	C	D
1	**FACTOR DE NORMALIZACIÓN**	3.35	12.88	10.49
2				
3	**N-GRAMA**	**LIBRO 1**	**LIBRO 2**	**LIBRO 3**
4	animal doméstico	0	0	0.72
5	animal terrestre	0.52	0	0.58
7	ecosistema	0	1	0
6	energía renovable	0	0	0
8	localidad rural	0.57	0	0.35
9	punto cardinal	0.62	0	0.04

Tabla 13. Ejemplo de un archivo de salida de la parte de la extracción terminológica

3.4 Validación de los candidatos a término

Aunque frecuentemente las listas que se obtienen de los procesos de extracción terminológicos se consideran las listas de términos finales, en este proyecto de tesis se busca emplear recursos léxicos para la validación de los candidatos a término como lo hace YATE o MetaMap. La razón de llevar a cabo una validación es que ésta permitiría obtener listas de términos mucho más fiables, de menor tamaño y orientadas al área o categorías de análisis.

Para llevar a cabo la validación de los candidatos a término se empleó Wikipedia como recurso léxico por su amplia cobertura en distintas áreas científicas y porque se consideró que tiene la mayoría de los términos que se encuentran dentro del COCIEM. Este proceso consiste primeramente en seleccionar los candidatos a término a validar, a partir de las listas generadas por el análisis de TF-IDF (sección 3.3) con base en los pesos de TF-IDF.

Posteriormente, se determinan las categorías de Wikipedia que correspondan de la manera más cercana a las áreas de las listas de candidatos a término a validar. Finalmente se calcula el coeficiente de dominio, el cual indica qué tan relacionado está el candidato a término con las áreas seleccionadas.

A continuación, se hablará de la estructura de Wikipedia, su conversión a una base de datos, la manera en que se lematizó y del coeficiente de dominio.

3.4.1 Wikipedia para la validación

Uno de los grandes recursos léxicos digitales es Wikipedia, la cual se encuentra en más de 200 idiomas y es la enciclopedia en línea más grande del mundo. Aunque se había dado a conocer su estructura a grandes rasgos en la sección 2.4.3, en esta parte se explicará a mayor profundidad la organización y arquitectura de Wikipedia.

Según Zesch y Gurevych (2007a) la enciclopedia Wikipedia está formada por dos grafos interconectados, el primero de ellos es el de categorías, mientras que el segundo es el de páginas o artículos. Con base en estos autores, la Figura 17 muestra la arquitectura de la enciclopedia Wikipedia.

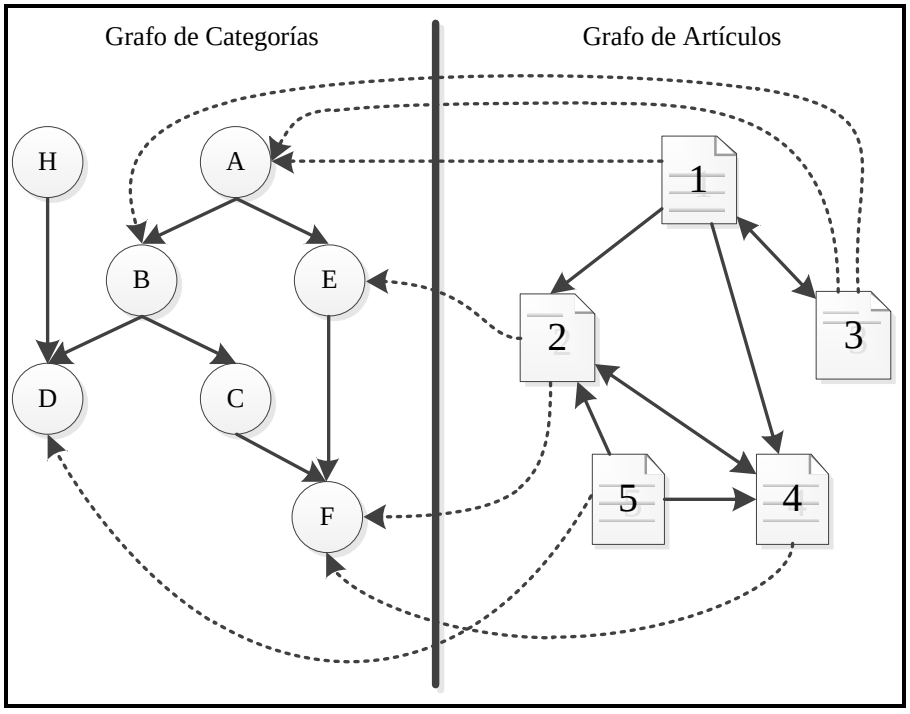

Figura 17. Arquitectura de Wikipedia

El *grafo de categorías* o *Wikipedia Category Graph* (WCG) es una estructura organizada a manera de taxonomía, la cual se maneja, al igual que las taxonomías, mediante una estructura jerárquica; este grafo almacena todas las categorías existentes en Wikipedia. Cada categoría tiene un número arbitrario de subcategorías, donde una subcategoría es típicamente establecida por relaciones de hiponimia[30] y meronimia[31]. Cabe aclarar que Wikipedia no forma, de manera estricta, una taxonomía, debido a que ciertas veces las categorías no cumplen con las relaciones mediante las cuales deberían estar unidas y se generan ciclos o categorías desconectadas de la taxonomía. Por lo anterior, Peters (2009) considera a la Wikipedia como una folksonomía, ya que la gente es quien desarrolla la jerarquización y no los expertos en las materias como ocurre, por ejemplo, con las taxonomías biológicas.

La Wikipedia en español se encuentra dividida en las siguientes partes: Anexos, Artículos, Ayuda, Categorías y Wikipedia. La categoría "Artículos" es la más grande de todas, pues no sólo incluye las categorías de Wikipedia, sino también las páginas de los artículos, y por tanto es la de mayor uso.

El *grafo de artículos* o *grafo de páginas* es una estructura no organizada la cual se genera automáticamente a través de los vínculos que contienen las páginas hacia otros artículos de Wikipedia. Por ejemplo: El artículo "Zeus" tiene un vínculo hacia "Mitología griega", éste a su vez con "Religión de la Antigua Grecia" y finalmente, este artículo tiene un vínculo con la página "Zeus". Como se puede observar en el ejemplo anterior y en la Figura 17, las relaciones entre las páginas pueden ser recíprocas y cíclicas.

Además de los artículos que contienen información, existen algunos artículos de Wikipedia que contienen solamente un redireccionamiento a otra página o forman lo que se conoce como una página desambiguación; estos dos tipos de página funcionan como información extra de la enciclopedia. El redireccionamiento, como se observó en la sección

[30] Las relaciones de hiponimia incluyen a su vez relaciones de hiperonimia, ya que es la relación inversa o que va en sentido contrario.

[31] La *meronimia* es la relación semántica entre un elemento léxico que denota una parte y otro elemento léxico que denota al primer elemento y a otros a su vez (Cruse, 1986). Ejemplo: Bujía es un merónimo de motor, y motor es merónimo de automóvil.

2.4.3, permite incluir, principalmente, variaciones ortográficas, morfológicas o abreviaturas de los nombres de los artículos; de manera menos frecuente, el redireccionamiento se utiliza para pasar de una página que habla de un tema muy específico a una página que expresa un tema más general o de un verbo a un sustantivo, por ejemplo, el verbo "sumar" redirige a "suma", mientras que "DVI-I" dirige a "Digital Visual Interface". En cambio, las páginas de desambiguación son un repositorio de artículos polisémicos, es decir, artículos que representan varios temas pero que tienen un nombre igual. Ya que el nombre de las páginas de Wikipedia debe ser único, a la página de la acepción más común se le deja frecuentemente el nombre del tema, mientras que a las páginas de las acepciones menos comunes se les coloca el nombre del tema más un identificador entre paréntesis que expresa más específicamente la acepción (Zesch et al., 2007b); por ejemplo, en la Wikipedia en español para "Metro" existe una página de desambiguación que permite elegir entre la unidad de longitud (Metro_(medida)), el sistema de transporte metropolitano (Metro_(sistema_de_transporte)), un periódico (Metro_(periódico)) y un canal de televisión argentino (Metro_(canal_de_televisión)), entre otros.

El grafo de artículos y el de categorías están unidos, ya que la gran mayoría de los artículos de Wikipedia están asignados a una o más categorías. En la Figura 17 se muestran con líneas punteadas las uniones entre los artículos y las categorías.

La arquitectura de Wikipedia no es a prueba de errores. Existen casos en los cuales los artículos no están vinculados a la categoría correcta; por ejemplo, en la categoría "Almacenamiento informático" están vinculados de manera correcta los artículos "Unidades de disco", "Caché", entre otros, pero también aparecen las páginas "Quantum Corp." que es una empresa, y "Robocopy" que es un comando de Windows para hacer copias de archivos, ambas que no debieran aparecer. Asimismo, otro de los problemas de Wikipedia es que existen algunos casos en los que los artículos no se encuentran unidos a ninguna categoría, ya que su creador no los unió a alguna de ellas. De igual manera, hay artículos que están unidos a categorías especiales para indicar que existen problemas de edición, de coherencia, de neutralidad, etcétera.

También existen problemas, algunas veces, con la unión entre las páginas de los artículos, es decir, los vínculos dentro de los artículos dirigen a un tema equivocado o a una

acepción errónea. De igual forma, las páginas de redireccionamiento o de desambiguación pueden no estar unidas al artículo correcto y por tanto no hay una coherencia entre lo buscado y el resultado obtenido.

A pesar de que tiene algunas desventajas la arquitectura de la enciclopedia, la Wikipedia ha sido empleada en diversas áreas de PLN, por ejemplo Toral et al. (2006) usan la encliclopedia para la creación de listas de entidades nombradas, Ponzetto y Strube (2008) crean taxonomías de manera automática a partir de Wikipedia. Mientras que Suchanek (2008) desarrolla ontologías usando la información de la enciclopedia y Gabrilovich y Markovitch (2009) emplean Wikipedia para la interpretación semántica de textos en lenguaje natural.

3.4.1.1 Conversión a una base de datos

Para llevar a cabo el proceso de validación de los candidatos a término es necesario tener acceso a Wikipedia, para realizar esto se pueden emplear diversos métodos como los siguientes:

Web crawler: También conocido como *araña web*, es un programa que inspecciona el internet o una determinada página, en este caso Wikipedia, de manera metódica y sistematizada. Estos programas permiten almacenar las páginas web visitadas. Aunque se puede emplear este método, Wikimedia[32] pide que no se empleen arañas web para extraer información de Wikipedia por la sobrecarga que se genera en los servidores.

Bot: Otro de los métodos empleados en la extracción de información de Wikipedia es el uso de *bots*, los cuales son programas informáticos que actúan como si fueran un humano. En este caso sirven como web crawlers pero son mucho más lentos debido a que es necesario que actúen lo más natural posible y no saturaren los servidores.

APIs: Las *Interfaces de programación de aplicaciones* o *APIs* por su acrónimo en inglés son un conjunto de métodos o funciones incluidas en una biblioteca que permiten emplear un determinado software. En la actualidad existen muchos de estos APIs para el manejo de Wikipedia mediante el empleo de copias de las bases de datos que Wikimedia publica

[32] Wikimedia es la organización encargada de manejar la Wikipedia.

frecuentemente para que servidores u otras computadoras tengan acceso a la enciclopedia sin tener que saturar los servidores originales empleando un web crawler.

En el caso de esta tesis se empleará el método descrito por Zesch et al. (2008), el cual se encuentra disponible como una API en el internet[33]. Esta API se llama *Java-based WikiPedia Library (JWPL)*.

La estructura original de las bases de datos de Wikipedia está optimizada para la búsqueda de artículos a través de palabras claves que son hechas por millones de usuarios de Wikipedia cada día (Zesch et al., 2008). Sin embargo, esta estructura no es la adecuada para su uso en proyectos de PLN, debido a que es necesario que se soporten búsquedas iterativas, acceso a gran número de caminos de Wikipedia o a la información dentro de las páginas de los artículos como los vínculos o categorías. Por ello JWPL permite la conversión de la base de datos de Wikipedia en una base de datos optimizada para usarla en tareas del PLN.

La optimización de la base de datos de Wikipedia consiste en convertir la información de redireccionamiento y de otros recursos léxicos y semánticos que se encuentra de forma implícita a una forma explícita; la razón de ello es que una gran parte de la información que contiene Wikipedia está dentro de las páginas de los artículos y no en los archivos de la base de datos de la enciclopedia que se publican; por ejemplo, la página "Capacitor" contiene como información "[[Redirect: Condesador_eléctrico]]", el cual indica que se debe redireccionar a la página "Condensador eléctrico"; por consiguiente, sin la optimización las páginas de los artículos deberían ser analizadas sintácticamente, cada vez que se consulte, para saber si son de redireccionamiento y así obtener la página a la que se redirige (Zesch et al., 2007b). Lo que lleva a cabo una de las herramientas de JWPL es analizar cada una de las páginas de la Wikipedia y separar las que sean de redireccionamiento o que contengan otra información léxica o semántica; estas últimas son analizadas y se les extrae la información implícita y se almacena de manera explícita en tablas de una base de datos.

La base de datos optimizada de Wikipedia está conformada por 11 tablas y son las siguientes:

[33] http://code.google.com/p/jwpl/issues/list

Category: En esta tabla se guardan todas las categorías de Wikipedia con su nombre e identificador.
Category_inlinks: Almacena los identificadores de las categorías superiores para una determinada categoría.
Category_outlinks: Guarda los identificadores de las subcategorías de una categoría específica.
Category_pages: La tabla tiene recopilados los identificadores de los artículos que pertenecen a cada categoría.
MetaData: Contiene la información básica de la base de datos de Wikipedia, como el idioma, número de páginas, nombre de la categoría de desambiguación, etcétera.
Page: Guarda todas las páginas de los artículos de Wikipedia; en esta tabla se encuentra el texto, si es de desambiguación, el nombre de la página, entre otros.
Page_categories: En esta tabla se encuentran los identificadores de todas las categorías que tiene asociado cada artículo de Wikipedia.
Page_outlinks: Esta tabla se genera en la optimización y almacena el identificador de las páginas que salen de un artículo.
Page_inlinks: Al igual que "Page_outlinks", esta tabla se genera en la optimización y se encarga de guardar los identificadores de las páginas que se dirigen hacia un mismo artículo.
Page_redirects: Contiene los identificadores de las páginas que redireccionan a una página en específico. Esta tabla se genera al momento de optimizar la base de datos de Wikipedia.
PageMapLine: Esta tabla contiene el nombre de las páginas de los artículos, su identificador general y el identificador de la página a la que corresponde, que en caso de no ser igual que el identificador general indica que existe un redireccionamiento; en algunos casos aparece el lema y una truncación del nombre de la página.

Entre las ventajas de la optimización de las bases de datos de Wikipedia está tener una eficiencia computacional en tareas de lenguaje natural de gran escala y obtener resultados reproducibles (siempre y cuando se emplee la misma versión de Wikipedia).

Con respecto a la versión de Wikipedia que se empleará en la tesis es la copia de la base de datos de noviembre de 2010.

3.4.1.2 Lematización de Wikipedia

Aunque la lematización de los textos permite obtener algunas ventajas como la reducción y agrupamiento de términos, conlleva un problema con Wikipedia. Este problema está relacionado con los nombres de los artículos de la enciclopedia, pues estos no siempre concuerdan con su forma lematizada; por ejemplo, "Análisis de circuitos" y "Sexualidad humana", tienen como lema "Análisis de circuito" y "Sexualidad humano", respectivamente. Por tanto, no es posible emplear los nombres de los artículos de Wikipedia de manera exacta para realizar la validación de los candidatos a término, ya que los candidatos a término se encuentran lematizados.

Para resolver el problema anterior se decidió lematizar los nombres de los artículos de Wikipedia. Esta tarea se desarrolló con un programa creado en Perl que preparara la Wikipedia y emplea el lematizador FreeLing, este último se usa para que se obtengan resultados similares a los que se llevan a cabo con los textos.

Los pasos seguidos para realizar este proceso fueron, en primer lugar, la extracción de los nombres de los artículos de Wikipedia junto con su identificador[34] para su almacenamiento en un archivo de texto, información que fue obtenida de la tabla "PageMapLine" de la base de datos de Wikipedia. Posteriormente, el archivo fue preprocesado; esto consistió en agregar el símbolo de almohadilla (#) y un espacio en blanco antes del nombre de cada artículo y colocar en minúscula la primera letra del nombre. Lo anterior se llevó a cabo porque sin él la lematización no se realizaba de manera adecuada por falta de contexto; por ejemplo, "Derivada" quedaba como "derivada", mientras que "Sistemas complejos" como "sistemas complejo"; esto porque FreeLing considera en algunos casos que las palabras que inician en mayúscula al principio de una oración son nombres propios y empleando la almohadilla y la conversión a minúscula de la primera letra se corrige este problema. El siguiente paso consistió en la ejecución del programa de FreeLing sobre el archivo de texto preprocesado. Finalmente, la última parte del proceso consistió en convertir

[34] Este identificador o ID es el número secuencial que se crea al agregar un artículo a Wikipedia de manera automática y permite diferenciar cada una de las entradas de la enciclopedia; este ID se encuentra en la tabla Page y PageMapLine (sección 3.4.1.1).

el archivo de texto lematizado a un archivo lo más similar al obtenido en el primer paso (usando un método similar al dado a conocer en la sección 3.2.2), esto para no afectar de manera sistemática la arquitectura de la Wikipedia. Además, dentro de este proceso se eliminó la almohadilla y el espacio en blanco extra y se colocó en mayúscula la primera letra del nombre del artículo.

Una vez teniendo los nombres lematizados, se procedió a actualizar la base de datos de Wikipedia. Para llevar esto a cabo se crearon dos tablas más en la base de datos de Wikipedia, una de ellas para almacenar los nombres de los artículos lematizados y otro para los nombres sin lematizar, ambas como respaldo. Los datos de la tabla con los nombres en su forma canónica actualizaron las tablas "PageMapLine" y "Page", las cuales son las tablas en donde se puede buscar la información por nombre del artículo.

3.4.2 Cálculo del coeficiente de dominio

Para poder llevar a cabo la validación de los candidatos a término que se obtuvieron del extractor automático es necesario acordar la manera en que se determinará si pertenecen a una materia o a otra. Para ello es necesario emplear un método que permita establecer un *coeficiente de dominio*; es decir, una métrica que indique qué tan relacionado se encuentra un término con una determinada categoría o área de Wikipedia.

En esta tesis se empleará el método desarrollado por Vivaldi y Rodríguez (2010) para el cálculo del coeficiente de dominio, el cual se explica a continuación.

Como se había indicado en la sección 3.4.1, Wikipedia está conformada por una serie de categorías y subcategorías. En el cálculo del coeficiente de dominio las categorías y sus divisiones forman lo que llamaremos *fronteras de dominio*, las cuales definen las áreas o materias a las que puede pertenecer un término. En algunos casos la materia a analizar tiene su par exacto en Wikipedia, como "química" o "economía"; en algunos otros no, como "computación" que necesita las categorías de Wikipedia "informática" y "electrónica".

A partir de lo anterior se procede a analizar Wikipedia para cada uno de los candidatos a validar. Este análisis comienza buscando la página del tema que sea igual a la del candidato a término dentro de la tabla "PageMapLine" de la base de datos de Wikipedia.

Luego, si el término es encontrado en la tabla “PageMapLine”, entonces éste se busca en la tabla “Page” para verificar si es una página de desambiguación o no. Si las páginas no son de desambiguación se buscan las categorías a las que está asociada la página del artículo dentro de la tabla “Page_categories”. En el caso de que la página sea de desambiguación, se almacenan los identificadores de las páginas relacionadas al término polisémico y se analizan, una por una, buscando sus categorías como se explicó en arriba. Por ejemplo, en el caso de la Figura 18, se muestra que el término “Oxidrilo”, el cual fue redireccionado a “Grupo hidroxilo”, está asociado a las categorías “Grupos funcionales”, “Compuestos de oxígeno” y “Compuestos de hidrógeno”. Posteriormente, para cada una de las categorías que se encontraron en el paso anterior, se realiza una búsqueda recursiva en sus categorías superiores hasta encontrar las fronteras de dominio o una categoría tope de Wikipedia, en este caso será “Artículos”. Además, esta búsqueda tiene un límite establecido, ya que en ocasiones se pueden encontrar ciclos infinitos y, por tanto, puede llegarse a no terminar el análisis si no existe un límite.

Figura 18. Ejemplo de las categorías a las que pertenece el término Grupo hidroxilo

Posteriormente, con la información obtenida del análisis anterior se calcula el coeficiente de dominio con base en las fórmulas mostradas por Vivaldi y Rodríguez (2010) con algunas modificaciones llevadas a cabo por comunicación expresa con el Dr. Jorge

Vivaldi. A continuación se muestran las fórmulas empleadas para el cálculo del coeficiente de dominio:

Basado en el número de caminos:

$$CDnc(t) = \frac{NCdominio\ (t)}{NCtotal\ (t)} \qquad (20)$$

Donde $CDnc$ es el coeficiente de dominio, t es el candidato a término, $NCdominio$ es el número de caminos al dominio y $NCtotal$ es el número de caminos a la categoría máxima de Wikipedia.

Basado en la longitud de los caminos:

$$CDlc(t) = \frac{LCtotal\ (t) - LCdominio\ (t)}{LCtotal\ (t)} \qquad (21)$$

Donde $CDlc$ es el coeficiente de dominio, t es el candidato a término, $LCdominio$ es la longitud (o número de saltos) de los caminos al dominio y $LCtotal$ es la longitud de los caminos a la categoría máxima de Wikipedia. Además hay que aclarar que $CDlc$ tendrá un valor de 1 cuando $LCtotal$ y $LCdominio$ sean iguales.

Basado en la longitud promedio de los caminos:

$$CDlmc(t) = \frac{LMCtotal\ (t) - LMCdominio(t)}{LMCtotal\ (t)} \qquad (22)$$

Donde $CDlmc$ es el coeficiente de dominio, t es el candidato a término, $LMCdominio$ es la longitud media de los caminos al dominio y $LMCtotal$ es la longitud media a la categoría máxima de Wikipedia. El coeficiente $CDlmc$ será igual a 1 cuando $LMCtotal$ y $LMCdominio$ tengan el mismo valor.

En el caso de de los coeficientes $CDlc$ y $CDlmc$, la longitud de los caminos al tope que pasan por la frontera de dominio se miden hasta ella, es decir, si hay nodos más arriba de ésta no se toman en cuenta. Esto se realiza para que se obtenga un coeficiente de dominio igual a 1 cuando todos los caminos que salen del término se dirigen a la frontera de dominio, de esta manera, la distancia entre la frontera de dominio y la categoría tope no afecta el valor del coeficiente de dominio.

El coeficiente de dominio para los candidatos a término puede tener un valor de -1 si no se encuentran en Wikipedia, entre $(-1, 0]$ si se encuentran en la enciclopedia pero no tienen ninguna relación con las fronteras de dominio, entre $(0,1)$ si tienen una cierta relación con la frontera de dominio y 1 si pertenecen totalmente a la frontera de dominio.

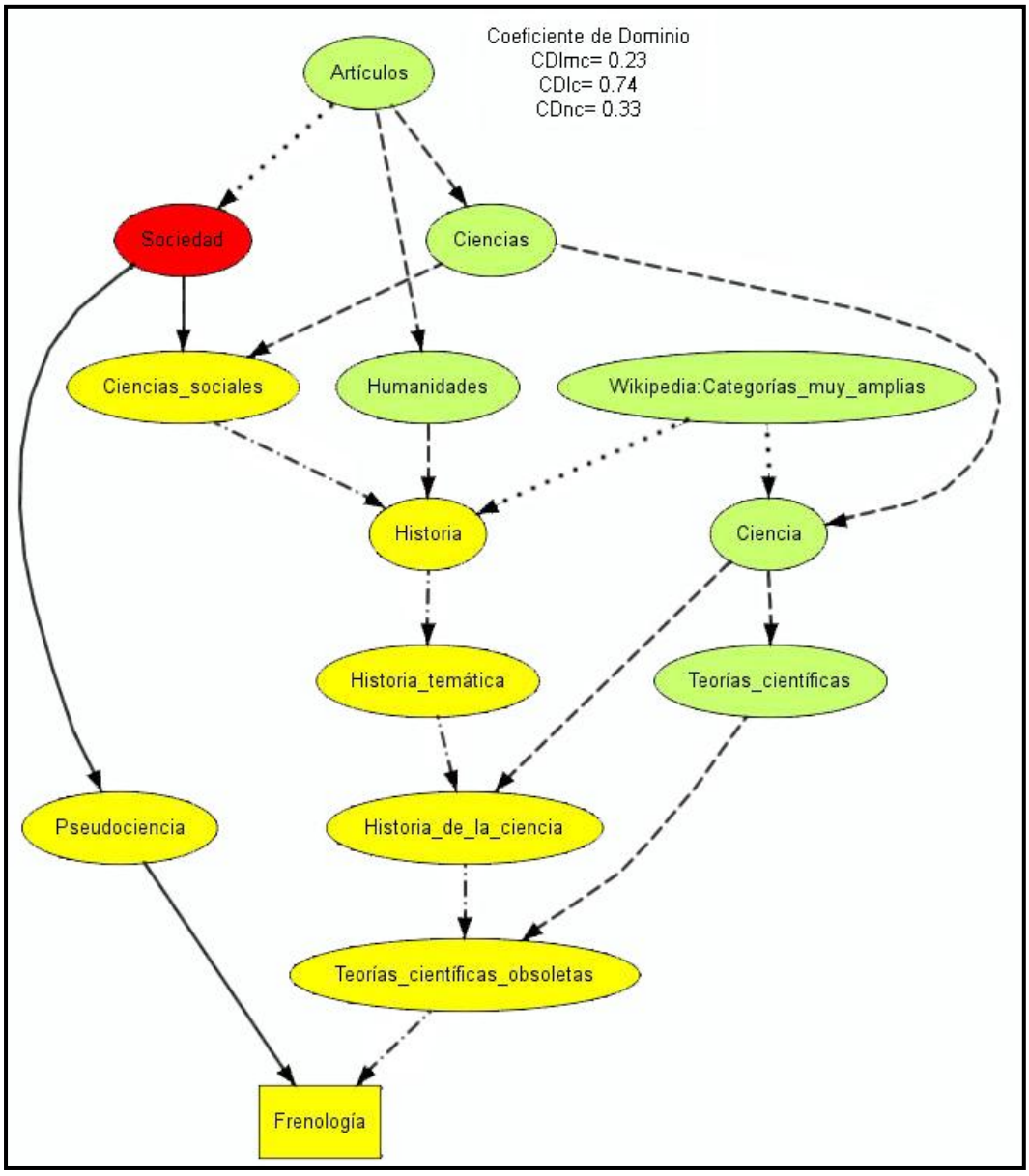

Figura 19. Grafo para el término "Frenología" y sus categorías

En la Figura 19 se muestra un grafo que muestra los datos otorgados por Wikipedia para el término "Frenología". El rectángulo en amarillo indica el término a buscar; los óvalos amarillos indican las categorías que se encuentran entre el término y la frontera de dominio.

El óvalo rojo, es la frontera de dominio, que en este caso es "Sociedad". Finalmente los óvalos verdes, son las categorías que van a la categoría tope pero no pasan por la frontera de dominio. En el caso que existiera en la Figura 19 un óvalo blanco, éste indicaría que es una categoría donde pasa un camino que va únicamente de la frontera de dominio a la categoría tope.

El óvalo verde llamado "Wikipedia: Categorías muy amplias" no se toma en cuenta para este análisis como una categoría tope y sus caminos no son contados en el análisis porque no todos los temas están conectados de alguna manera a esta categoría y se busca que en general todos los temas tengan el mismo tope, que en este caso es la categoría "Artículos".

Asimismo, la Figura 19 indica en la parte superior derecha los coeficientes de dominio calculados a partir de las Ecuaciones (20), (21) y (22). La línea continua indica los caminos del término a la frontera de dominio; la línea punteada es un camino que no se cuenta, ya sea porque va de la frontera de dominio a la categoría tope o porque va a "Wikipedia: Categorías muy amplias". La línea formada por puntos y guiones son los caminos que van tanto al tope como a la frontera de dominio. Y la línea intermitente son los caminos que van solamente a la categoría tope. A continuación se muestran los cálculos de los coeficientes de dominio con base en la Figura 19:

$$CDlmc\ (Variable\ dependiente) = \frac{\frac{31}{6} - \frac{8}{2}}{\frac{31}{6}} = \frac{5.1666 - 4}{5.1666} = \frac{1.1666}{5.1666} = 0.225 \approx 0.23$$

$$CDlc\ (Variable\ dependiente) = \frac{31 - 8}{31} = \frac{23}{31} = 0.74$$

$$CDnc\ (Variable\ dependiente) = \frac{2}{6} = 0.3\overline{3}$$

Para comprender mejor el coeficiente $CDlc$ y el coeficiente $CDlmc$, el cálculo de la distancia con valor igual a 31 se muestra en la Tabla 14.

Distancia del término a la frontera de dominio (suma de segmentos de la línea recta y de la punteada con guiones)	8
Distancia entre el término y la categoría tope sin pasar por la frontera de dominio (suma de la línea punteada con guiones y de la intermitente)	23
Distancia total	31

Tabla 14. Cálculo desglosado para la distancia entre el término y la categoría tope usando como ejemplo la Figura 19

Todo el proceso de validación, como se indicó anteriormente, se lleva a cabo para cada uno de los candidatos a término seleccionados. La validación genera archivos de salida con el formato indicado en la Tabla 15, donde la primera columna indica el coeficiente de dominio, la segunda el candidato a término analizado y la tercera la manera en que se encontró en Wikipedia: "page" si se encontró de manera exacta, "nil" si no se encontró en la enciclopedia, "pagedir" si hubo un redireccionamiento y "pagedesamb" si se encontró una página de desambiguación. En estas dos últimas formas se indica el nombre de la página del artículo a donde se redirigió o desambiguó.

- Bigramas		
- Coeficiente de dominio CDwp_lc		
1.00	corriente eléctrico	(page)
1.00	capacitor	(pagedir: Condensador_eléctrico)
0.96	resistencia	(pagedesamb: Resistecia_eléctrico)
0.00	caucho natural	(pagedir: caucho)
-1.00	Juan Cárdenas	(nil)

Tabla 15. Ejemplo del formato de salida de los archivos de los candidatos a término validados por Wikipedia

3.5 Arquitectura del sistema

Para la obtención y validación de términos se desarrolló un sistema que fue explicado a lo largo de este capítulo. Para finalizar, se mostrará a continuación la arquitectura del sistema creado y se puede observar en la Figura 20.

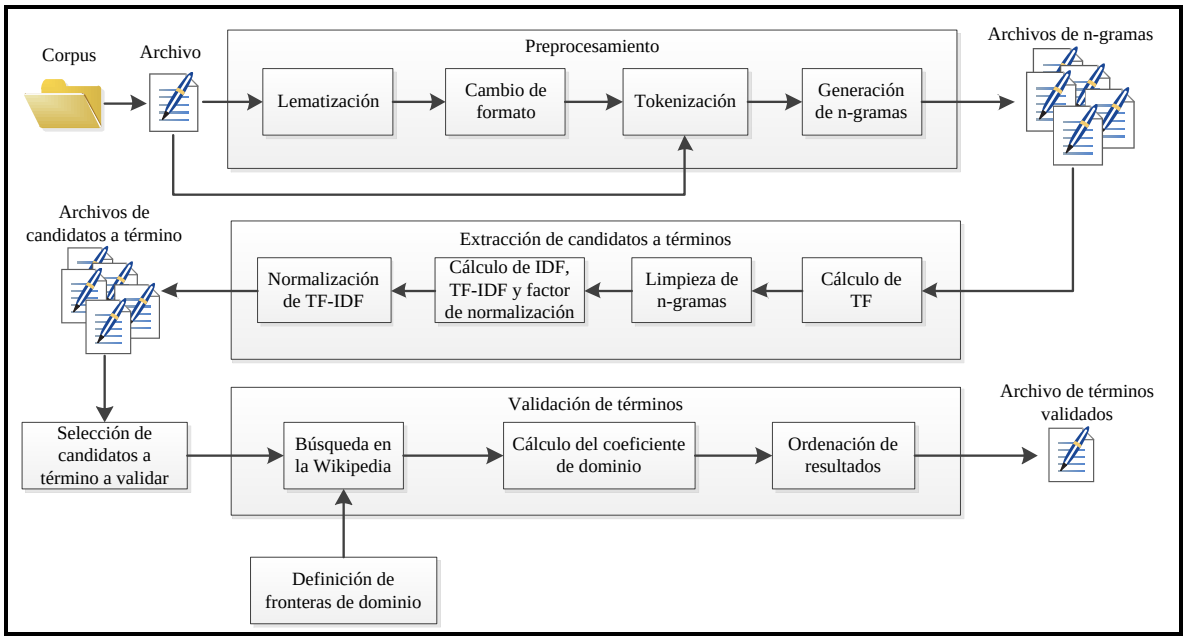

Figura 20. Arquitectura del sistema desarrollado en la tesis

La primera parte es el preprocesamiento de cada uno de los archivos del corpus, en ella se lleva a cabo la lematización, el cambio de formato, la tokenización y la generación de n-gramas (sección 3.2). Los archivos pueden ser directamente tokenizados si ya se llevó a cabo previamente una lematización y un cambio de formato, esto para disminuir los tiempos de preprocesamiento. El preprocesamiento se lleva a cabo para cada uno de los documentos seleccionados del corpus y da como resultado tres archivos por documento; estos tres archivos contienen los tres tipos de n-gramas usados en esta tesis, unigramas, bigramas y trigramas.

La segunda parte es la extracción de candidatos a término (sección 3.3), la cual se lleva a cabo, uno por uno, a los archivos generados por el preprocesamiento. Está conformado por 4 módulos; el primero de ellos se encarga de calcular el TF de los n-gramas que se encuentren en los archivos generados por el preprocesamiento. El segundo módulo quita los n-gramas de los archivos de salida del primer módulo que tengan en sus extremos palabras funcionales o números romanos, y que por tanto tienen alta probabilidad de no ser términos. El tercer módulo, primeramente, une todos los archivos creados en el segundo módulo por tipo de n-gramas (unigramas, bigramas, trigramas), y posteriormente para cada archivo de tipo de n-gramas realiza tres tareas, las cuales son, el cálculo del IDF, del TF-IDF y del factor de normalización. El último módulo de esta parte crea archivos en formato de

hoja de cálculo de las listas generadas por el tercer módulo y aplica el factor de normalización a los pesos de TF-IDF calculados.

Después de la extracción de candidatos a término, se procede a seleccionar los candidatos que serán validados por Wikipedia en función de los pesos recibidos por el análisis TF-IDF.

Finalmente, la cuarta parte consiste en la validación de los candidatos a término seleccionados en la parte anterior. Para llevar a cabo el proceso de esta parte se seleccionan las fronteras de dominio, las cuales deben coincidir con categorías existentes en Wikipedia y abarcar las de las listas de candidatos a término a analizar. Teniendo lo anterior determinado, se comienza la búsqueda de cada uno de los candidatos a término en la base de datos de Wikipedia y se realiza el cálculo del coeficiente de dominio. Finalmente, se ordena la lista de los términos validados, es decir, los resultados de mayor a menor coeficiente de dominio y se almacenan en un archivo de texto plano.