DECSAI

ugr Universidad de Granada

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN E INTELIGENCIA ARTIFICIAL

E.T.S. de Ingenierías Informática y de Telecomunicación
Programa de Doctorado en Tecnologías de la Información y la Comunicación

NUEVAS TÉCNICAS DE MINERÍA DE TEXTOS: APLICACIONES

MARÍA DEL CONSUELO JUSTICIA DE LA TORRE

Directores
DANIEL SÁNCHEZ FERNÁNDEZ y MARÍA JOSÉ MARTÍN BAUTISTA

Imagen de portada tomada de la web de la Biblioteca Británica (*The British Library — Flickr - Photo Sharing [TheBritishLibrary, 2007]*)

Editor: Universidad de Granada. Tesis Doctorales
Autora: María Consuelo Justicia de la Torre
ISBN: 978-84-9163-243-6
URI: http://hdl.handle.net/10481/46975

ugr Universidad de Granada

DECSAI

Nuevas Técnicas de Minería de Textos: Aplicaciones

memoria que presenta

María del Consuelo Justicia de la Torre

para optar al grado de

Doctor en Informática

2017

DIRECTORES

Daniel Sánchez Fernández María José Martín Bautista

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
E INTELIGENCIA ARTIFICIAL

La doctoranda *María del Consuelo Justicia de la Torre* y los directores de la tesis *Daniel Sánchez Fernández* y *María José Martín Bautista* garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por el doctorando bajo la dirección de los directores de la tesis y hasta donde nuestro conocimiento alcanza, en la realización del trabajo, se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

Granada, a 29 de marzo de 2017

Directores de la Tesis:

Daniel Sánchez Fernández

Fdo.:

María José Martín Bautista

Fdo.:

Doctoranda:

María del Consuelo Justicia de la Torre

Fdo.:

Resumen

Una diferencia fundamental entre Minería de Datos y Minería de Textos es la naturaleza de la información de partida. Los datos tienen una estructura explícita y contienen conocimiento de forma implícita, mientras que con el texto ocurre todo lo contrario, es decir, contiene un conocimiento explícito (aunque por lo general dependiente de un contexto que no está del todo hecho expresado en el documento) y una estructura que se puede sobrentender.

Esta diferencia puede interpretarse desde el punto de vista de la Ingeniería del Conocimiento, según el cual los datos constituyen colecciones de hechos simples, representados mediante un modelo de datos que facilita su tratamiento, mientras que el texto contiene una base de conocimiento completa que incorpora, además de hechos, relaciones de diversos tipos, con la desventaja de estar expresados en un lenguaje de representación (el lenguaje natural) que, aunque de una expresividad máxima, resulta totalmente inadecuado para llevar a cabo procesos automáticos de inferencia.

Dichas diferencias plantean nuevos retos a la investigación en Minería de Textos y, al mismo tiempo, abren la puerta a un importante paso adelante desde el estado actual de la técnica, con nuevas capacidades y aplicaciones.

En este trabajo se muestra una retrospectiva de los principales artículos científicos que han tratado el tema de la Minería de Textos, se revisa su concepto, definición y proceso y se exponen las diferentes formas de representación intermedia del texto que pueden resultar "adecuadas" para trabajar con el lenguaje natural en el que está escrito. Igualmente, se repasan las técnicas de minería existentes y se enumeran algunas de las aplicaciones más frecuentes de la Minería de Textos.

Por otro lado, se discuten una serie de retos de futuro dentro de este área de investigación y se presenta el innovador paradigma *text knowledge mining* como complemento a la Minería de Textos tradicional o "basada en datos". Sustentaremos nuestro estudio aportando una técnica de Minería de Textos basada en la *localización de contradicciones en textos*, atenderemos a su implementación, y a la demostración experimental de que es una técnica factible y con requisitos computacionales razonables.

A partir de aquí, abrimos un área de investigación apasionante de la que se pueden esperar muchas contribuciones importantes en un futuro próximo.

Agradecimientos

Hace muchos años llamé a la puerta del despacho de Dani y de María José y les pedí que fueran mis tutores para el programa de doctorado. Y accedieron. En ese momento yo no era muy consciente de las andanzas que depararían esta decisión, puede que ellos sí, pero les agradezco, de corazón, que aceptaran.

Ha sido un camino muy largo el andado desde entonces y no siempre ha seguido un trazado continuo pero su trayectoria me ha ayudado a crecer en diferentes aspectos, no sólo en el ámbito científico. Quienes han estado siempre ahí lo saben.

Fueron mis directores quienes me sugirieron que Nacho podría ayudarme en el recorrido y estaban en lo cierto: mi querido compañero de promoción ha sido un pilar importante de este trabajo.

Afortunadamente he estado acompañada, arropada y animada durante toda esta aventura por personas, sin las cuales, este proyecto no habría concluido.

A mi familia, que me alentó a comenzar la andadura, a mi marido que me ha apoyado en cada momento y a mis amigos que han creído en mí cuando yo no lo hacía,

Gracias.

Lista de Publicaciones Originales

Este trabajo es el resultado del estudio del campo de la Minería de Textos. A lo largo de este proceso se ha generado diversa producción científica, de la que se puede destacar la siguiente:

- C. Justicia de la Torre, D. Sánchez, I. Blanco and M.J. Martín-Bautista. (2017). *Text Mining Techniques, Applications, and Challenges*. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. Aceptado 2017.
- D. Sánchez, M.J. Martín-Bautista, I. Blanco and C. Justicia de la Torre. (2008). *Text knowledge mining: an alternative to text data mining*. In IEEE International Conference on Data Mining Workshops. ICDMW 2008.
- C. Justicia de la Torre, M.J. Martín-Bautista, D. Sánchez and I. Blanco. (2008). *Text knowledge mining: An approach to text mining*. Actas del Congreso Español sobre Tecnologías y Lógica Fuzzy. ESTYLF 2008.
- C. Justicia de la Torre, M.J. Martín-Bautista, D. Sánchez and M.A. Vila. (2006). *Un enfoque deductivo para la minería de textos*. Actas del Congreso Español sobre Tecnologías y Lógica Fuzzy. ESTYLF 2006.
- C. Justicia de la Torre, M.J. Martín-Bautista, D. Sánchez and M.A. Vila. (2005). *Text Mining: Intermediate Forms on Knowledge Representation*. Proceedings de la European Society for Fuzzy Logic and Technology. EUSFLAT 2005.
- C. Justicia de la Torre, M.J. Martín-Bautista and D. Sánchez. (2004). *Minería de Textos: Aplicaciones con Lógica Difusa*. Proceedings del XII Congreso Español sobre Tecnologías y Lógica Fuzzy. ESTYLF 2004.

Lista de Abreviaturas

CWA	*Closed World Assumption*
DL	*Description Logic*
DM	*Data Mining*
KB	*Knowledge Base*
KDD	*Knowledge Discovery in Data*
KDT	*Knowledge Discovery in Texts*
KR	*Knowledge Representation*
IE	*Information Extraction*
IF	*Intermediate Form*
IR	*Information Retrieval*
LDO	*Linked Data Ontologies*
NLP	*Natural Language PreProcessing*
OWA	*Open World Assumption*
OWL	*Web Ontology Language*
RDF	*Resource Description Framework*
RIF	*Rule Interchange Format*
TDT	*Topic Detection y Tracking*
TM	*Text Mining*
URI	*Uniform Resource Identifier*

The heaviness of being successful was replaced by the lightness of being a beginner again, less sure about everything. (Steve Jobs, discurso en la graduación en la Universidad de Stanford, 12 de junio de 2005)

Cuando se elevó por encima del poste, esperabas ver a los ángeles que habían guiado esa pelota. (Desmond Tutu, documental 'Nelson Mandela: El jugador nº 16')

Contenidos

Figuras

Tablas

Algoritmo

Capítulo 1 Introducción

Entre la nutrida enumeración de los derechos del
hombre que la sabiduría del siglo XIX repite tan
a menudo y con tanta complacencia, se han olvidado
dos asaz importantes, que son:
el derecho a contradecirse y el derecho a marcharse
(Baudelaire [Baudelaire, 1988])

Las bases de datos, hasta hace poco tiempo, sólo trataban con datos, tablas y relaciones y podían responder a preguntas sobre información almacenada en su estructura. Además, los métodos para analizar la información, muchos de ellos manuales, no respondían a necesidades actuales de eficiencia y rapidez. De este modo, surge la necesidad de automatizar el proceso de análisis de la información para obtener algo más: conocimiento. Es el comienzo de la Minería de Datos, que no sólo aporta la posibilidad de encontrar patrones y tendencias desconocidas en las bases y conjuntos de datos, sino que permite trabajar con grandes cantidades de información. Gracias a este proceso, han subido las ventas en empresas, se han reorientado negocios con éxito, ha mejorado la producción de determinados productos y se han detectado fraudes económicos, entre otras aplicaciones.

Años más tarde, una vez que la labor de la **Minería de Datos** está más que reconocida, se hace notable que no sólo existe información almacenada en bases de datos numéricos dentro de las empresas, sino que existe una considerable cantidad de documentos que, estudiados y tratados convenientemente, nos pueden aportar más información que la meramente escrita. A este ámbito de trabajo lo llamaremos Minería de Textos.

Cuando hablamos de Minería de Textos (del inglés ***Text Mining***), intentamos abarcar un problema nombrado de muchas formas a lo largo de la historia: *Minería*

de la Información (Information Mining), Arqueología de la Información (Information Arqueology), Gestión del Conocimiento (Knowledge Management), Minería de Datos (Data Mining), Destilación de Conocimiento (Knowledge Distillation), Minería de Documentos (Document Mining), Minería de Datos Textuales (Text Data Mining), Extracción de Conocimiento en Texto (Knowledge Discovery in Text) o *Minería de Documentos Web (Web Document Mining).* Hemos elegido el término Minería de Textos para referirnos al proceso de extracción de información interesante que no está presente explícitamente en la colección de documentos objeto de estudio.

La Minería de Textos surge ante la necesidad de darle un valor adicional a la información documental que se acumula en las empresas. Esta información puede proceder de diversas fuentes externas, de naturaleza heterogénea y distribuida, que alimentan los repositorios y que no se están aprovechando en toda su dimensión. Son documentos procedentes de Intranets o de Internet, almacenados en las bases de datos documentales o en el servidor de correo electrónico, que pueden estar en formato texto o escritos en un procesador de textos, en diversos idiomas o en uno sólo pero todos están pidiendo un trato específico, un aprovechamiento adecuado a su condición.

Son muchos los autores que estudian el tema y muchos también los enfoques que se dan sobre el mismo. Veremos a lo largo de este trabajo que, si ya con el nombre no existe acuerdo, mucho menos a la hora de definirlo o de encontrar herramientas para aplicarlo.

Cuando nos planteamos la opción de extraer conocimiento, desconocido a priori, pensamos en aplicar las técnicas de Minería de Datos a la información textual pero nos encontramos con que este tipo de minería trata sobre información estructurada, almacenada generalmente en bases de datos relacionales, que puede ser gestionada por un algoritmo o un programa específico de ordenador. El texto no tiene estructura, o mejor dicho, su estructura implícita es demasiado complicada y rica como para ser tratada computacionalmente sin un procesamiento previo.

Esta aparente ausencia de estructura es el mayor problema de la Minería de Textos: la necesidad de preparar los documentos, pasarlos a una **Forma Intermedia** para que se les pueda aplicar algún algoritmo automático es una de las principales diferencias de la Minería de Textos con la Minería de Datos.

Los autores detallados en este trabajo no se ponen de acuerdo a la hora de dar una definición de este proceso y no todos plantean la necesidad de colocar el documento en esta Forma Intermedia antes de proceder a la extracción de conocimiento. Podemos agrupar a los autores dependiendo de la comparación que hacen

de Minería de Textos; así, hay quienes ven la Minería de Textos como un avance de las técnicas de Recuperación de Información; otros estiman que su función principal es localizar temas o conceptos en un documento, labor que se lleva a cabo por las técnicas de Extracción de Información; algunos investigadores siguen equiparándolo con Minería de Datos o finalmente, hay quienes diferencian ambos procesos completamente basándose en que el texto no tiene una estructura predefinida, mientras que los datos generalmente sí la tienen.

Las técnicas de Minería de Textos existentes se basan en técnicas de razonamiento inductivo a partir de datos extraídos de los textos. Con este tipo de técnicas se pierde gran parte de la carga semántica del texto ya que éstos contienen conocimiento más complejo con el que se puede realizar otro tipo de razonamiento y pueden proporcionar conocimiento que las técnicas actuales no pueden proporcionar.

Los trabajos realizados hasta el momento están basados en su mayoría en la primera aproximación, probablemente por la tendencia a extrapolar las técnicas usadas en Minería de Datos a texto. Sin embargo, no hay que olvidar que la semántica del texto no se trata convenientemente con estas técnicas y es una parte muy importante del documento a considerar.

1.1 Objetivos de la Tesis

La meta principal de esta disertación es arrojar luz y clarificar el concepto de Minería de Textos desde diferentes puntos de vista, siendo el *objetivo general* proponer y estudiar un nuevo marco de Minería de Textos.

Entre los objetivos particulares perseguidos tanto desde el punto de vista teórico como el práctico, destacamos

i. Hacer una revisión del concepto actual de *Minería de Textos*, sus sinónimos y sus variantes, las etapas del proceso, técnicas existentes, y las formas intermedias empleadas.

ii. Plantear un paradigma novedoso de *Minería de Textos* alternativo al basado en razonamiento inductivo sobre datos, como complemento a las técnicas y aplicaciones existentes.

iii. Proponer una técnica de *Minería de Textos* dentro del nuevo paradigma, así como su implementación y la demostración experimental de que es una técnica factible y con requisitos computacionales razonables.

iv. Realizar una experimentación con baterías de ejemplos sintéticos aplicando la técnica de *Minería de Textos* propuesta que demuestre la fortaleza de la técnica.

1.2 Organización de la Tesis

La estructura y contenido de este trabajo se desglosa en los distintos capítulos que se detallan a continuación:

- El *Capítulo 2, Minería de Textos: Antecedentes*: presenta una retrospectiva de los principales artículos científicos que han tratado el tema de la Minería de Textos, proceso que se sitúa dentro del marco general del Descubrimiento de Conocimiento en Textos y se analizan tanto sus fases como las técnicas que, de forma frecuente, se pueden encontrar en la literatura, y que sirven como instrumento de extracción de conocimiento, previamente desconocido, a partir del texto escrito en lenguaje natural. Establece la búsqueda de una Forma Intermedia de representación del texto como una fase fundamental de la Minería de Textos, y explora los autores que trabajan buscando conocimiento en textos, presentando las Formas Intermedias que ellos utilizan y las aplicaciones de las Formas Intermedias más frecuentes en la Minería de Textos y extrae una recopilación de las técnicas más destacadas para realizar la Minería de Textos, en cuanto a frecuencia de utilización, complejidad de uso y tipo de conocimiento obtenido.
- El *Capítulo 3, Una Nueva Visión de la Minería de Textos*: aporta la idea de que el descubrimiento de conocimiento potencialmente útil, no trivial y previamente desconocido a partir de una colección de documentos presenta una distinción claramente diferente de la Minería de Datos: el texto, escrito en lenguaje natural, se caracteriza por una estructura implícita muy compleja con muchas posibilidades de representación y una semántica muy rica. Existen muchos autores que no están de acuerdo con aplicar técnicas de Minería de Datos al texto porque han sido diseñadas para trabajar con estructuras de datos bastante simplificadas y por lo tanto, con una limitada capacidad expresiva. Si utilizamos dichas estructuras para proporcionar una representación intermedia al texto, estaremos perdiendo una gran cantidad semántica, sólo explotaríamos una parte muy pequeña de su poder expresivo. En este capítulo se expondrán las razones que avalan dicha idea.
- El *Capítulo 4, Una aplicación en el marco de la Minería de Textos: Búsqueda de Contradicciones en Texto*: una vez elegidas las ontologías como Forma

Intermedia de Representación textual, se presenta una aplicación real de la Minería de Textos que puede llevar, en otras utilidades, a demostrar la congruencia de una colección de documentos. Esta aplicación es la *Búsqueda de Contradicciones en Textos*. Para apoyar la validez de la hipótesis, se presenta tanto el algoritmo de búsqueda como una ejemplificación del mismo sobre un caso real.

- El *Capítulo 5, Experimentación*: describe una serie de experimentos controlados de Minería de Textos con el fin de localizar contradicciones en los textos. A partir de la elección de las ontologías como Forma Intermedia de Representación de Corpus Textuales, se revisa la literatura buscando una definición adecuada de ontología, se opta por un tipo de ontología que será el más conveniente para este trabajo y se enumeran los elementos involucrados en la creación de una ontología como son el lenguaje de marcado, la expresividad del lenguaje y la estructura de una ontología, entre otros.

- El *Capítulo 6, Conclusiones*: resume el problema de la Minería de Textos desde diferentes puntos de vista con el fin de que sirva tanto como revisión del estado del arte de este área como para clarificar las diferentes fases y aplicaciones del mismo. También presenta las principales conclusiones que se han alcanzado, y propone posibles líneas de trabajo futuro.

Capítulo 2 Minería de Textos: Antecedentes

En los últimos años del siglo XIX nadie habría creído que los asuntos humanos eran observados aguda y atentamente por inteligencias más desarrolladas que la del hombre y, sin embargo, tan mortales como él (Orson Wells [Wells, 2005])

2.1 Revisión de los Procesos de Descubrimiento de Conocimiento en Datos

En este primer capítulo damos una visión global sobre la evolución de procesos de extracción de conocimiento en datos y enumeramos sus fases principales. Entre ellas encontraremos la Minería de Datos, proceso que descubre información nueva a partir de los datos y que constituye la fase central del proceso de descubrimiento.

2.1.1 Descubrimiento de Conocimiento a partir de Bases de Datos (KDD, *Knowledge Discovery from Databases*)

Desde los años 60, la evolución de los sistemas de soporte a la decisión ha sido completamente exponencial, motivada por la necesidad de las organizaciones de tener un estado diario de sus principales departamentos y poder visualizar cambios de comportamiento en los datos que faciliten a los directivos tomar decisiones oportunas y, a veces, vitales para la empresa. Desde los primeros informes *batch* hasta las

actuales herramientas de Minería de Datos, existe casi medio siglo de evolución en las técnicas de extracción de información en los datos.

La información almacenada en las bases de datos de las organizaciones se ha venido analizando de forma manual por expertos para obtener conclusiones a partir de dichos almacenes de datos. Sin embargo, en la actualidad y, debido a que es demasiado grande el volumen de información que se acumula en las bases de datos y, quienes deben tomar las decisiones no tienen por qué ser expertos en análisis de datos, se hace necesario que el proceso de extracción de conocimiento a partir de los datos sea automatizado total o parcialmente.

De este modo, surge un área de estudio, que denominamos Descubrimiento de Conocimiento en Datos, en inglés *Knowledge Discovery from Databases (KDD)* (gráficamente reflejado en la Figura 1 ([Fayyad et al., 1996]), que se basa en las disciplinas tradicionales para obtener información a partir de los datos, como son: la Estadística, los Sistemas de Información, el Aprendizaje Automático o la Computación Paralela y que engloba al proceso que pretende extraer conocimiento a partir de los datos. Este proceso podemos definirlo como:

Proceso no trivial de identificar patrones válidos, novedosos, potencialmente útiles y en última instancia comprensibles a partir de los datos ([Fayyad et al., 1996]).

El tratamiento de los datos almacenados en las bases de datos será el que nos proporcione nuevo conocimiento. Para este fin, hay que realizar un estudio de la naturaleza y origen de los datos, bien sea determinando las fuentes de información que pueden ser útiles, o estudiando la viabilidad de construir un almacén de datos para unificar la diversidad de fuentes de la empresa,(*DataWarehouse* si las bases de datos de las que disponemos son muy amplias o bien *Data Mart*, para aspectos concretos de la organización, por ejemplo, el departamento de deportes de unos grandes almacenes), y puede resultar de utilidad visualizar los datos para concretar qué aspectos pueden ser más útiles y centrar el estudio sobre ellos.

Cada una de las fases del proceso KDD se sirve de una serie de herramientas para conseguir llevar a cabo su labor. Veamos cuáles son esas fases ([Han and Kamber, 2000]):

1. **Limpieza de Datos**: fase del proceso en la que se elimina el ruido y los datos inconsistentes obtenidos de las fuentes externas. Para ello se utilizan casi exclusivamente métodos estadísticos: histogramas (detección de datos anómalos),

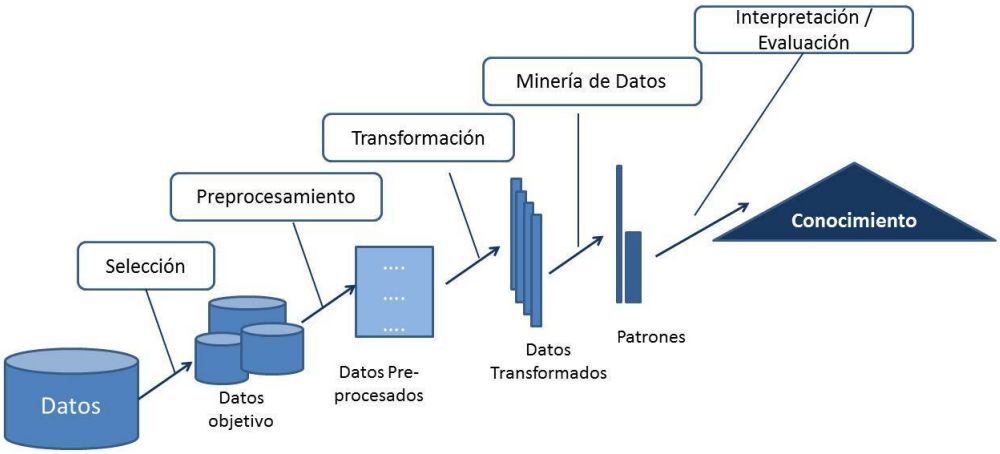

Figura 1: Fases del Proceso KDD ([Fayyad et al., 1996])

selección de datos (muestreo, ya sea verticalmente, eliminando atributos, u horizontalmente, eliminando tuplas), redefinición de atributos (agrupación o separación).

Hay dos situaciones irregulares a considerar en esta fase debifo a datos anómalos o datos que faltan. Las actuaciones ante el primer tipo de datos pueden ser: ignorarlos, filtrar la columna (puede ser útil no eliminarla sino sustituirla por una columna con datos discretos donde el valor anómalo se identifique como tal), filtrar la fila o reemplazar el valor. Cuando se está ante una situación de falta de datos, una posible solución puede ser esperar a que estos datos estén disponibles.

2. **Integración de datos**: proceso en el que se combinan múltiples fuentes de datos externas y heterogéneas. Uno de los principales caballos de batalla con los que se topa el analista cuando intenta realizar KDD es el de las fuentes externas, normalmente heterogéneas, que nutren los almacenes de datos y que pueden ser tan diversas como:

   - Bases de datos relacionales de trabajo diario dentro de la empresa
   - Datos del área geográfica, de economía ...
   - Bases de datos externas de otras organizaciones
   - Censos, información extraída de Internet, directorios

3. **Selección de Datos**: no todos los datos almacenados en bases de datos serán

útiles para extraer información. En esta fase se recuperan de la base de datos los datos relevantes para la tarea de análisis.

4. **Transformación de Datos**: los datos se transforman o se consolidan en formas apropiadas para la minería. Este proceso implica perder las relaciones de integridad y la normalización. Cuando se trabaja con *Data Marts*, se puede optar por un modelo en estrella o un modelo de copo de nieve, que agilizan las consultas posteriores y permiten buscar sobre información resumida.

5. **Minería de Datos**: fase principal del proceso KDD. Se aplican métodos inteligentes para extraer patrones de datos. Habrá que elegir los algoritmos de minería adecuados en fucinón de los datos y del tipo de información que se quiere descubrir.

6. **Evaluación de los Patrones extraídos en la fase anterior**: se identifican los patrones interesantes.

7. **Representación del conocimiento**: visualización del conocimiento obtenido en la minería.

KDD es un proceso iterativo e interactivo, de tal manera que se pueden realizar cambios y repetir cada paso para conseguir los mejores resultados ([Paralic, 2001]). Por ejemplo, el proceso de selección de datos se realizará las veces que se estimen oportunas hasta eliminar ruidos e incongruencias.

Hasta aquí, nos hacemos una idea del proceso global de Descubrimiento de Conocimiento en Datos que nos servirá como germen para definir el Descubrimiento de Conocimiento en Textos. No obstante, para que dicho conocimiento sea extraído, tendremos que centrarnos en la fase principal de este proceso: la Minería de Datos. En la siguiente sección hablaremos de ella más detenidamente.

2.1.2 Minería de Datos (DM, *Data mining*)

El descubrimiento se puede definir como el proceso significativo que encuentra nuevas correlaciones, patrones y tendencias a través de amplias cantidades de datos, usando técnicas matemáticas, estadísticas y de reconocimiento de patrones ([Thuraisingham, 1999]) . Si lo focalizamos en decisiones a partir de datos y en bases de datos estructuradas, hablamos del proceso KDD estudiado con anterioridad pero, ¿cómo se extrae el conocimiento?. La respuesta a esta pregunta la encontramos en el núcleo del proceso de KDD: la Minería de Datos.

La meta de la Minería de Datos es descubrir o derivar nueva información de los datos, encontrando patrones a través de los conjuntos de datos y / o separando señales de ruido. Podemos definir Minería de Datos como:

Proceso de descubrimiento eficiente de patrones, desconocidos a priori, en grandes bases de datos. ([Agrawal and Shafer, 1996])

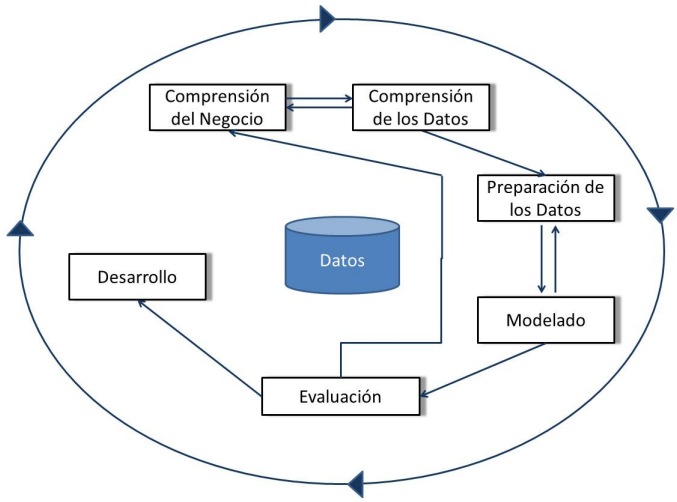

Figura 2: Proceso de Minería de Datos ([ZenTut, 2015])

La necesidad de la Minería de Datos surge ante una serie de características de los datos y de los resultados esperados con el análisis de los mismos:

- La naturaleza heterogénea y distribuida de los datos: La cantidad de datos, la variedad de información que almacenan las empresas y los distintos orígenes de la misma (bases de datos tradicionales, archivos, *DataWarehouse*,...) hacen necesario darle alguna utilidad adicional.
- La alta dimensionalidad de los datos (número de columnas / número de atributos) que no es manejable para los algoritmos clásicos.
- La toma de decisiones en la empresa se suele basar en experiencias ya ocurridas en la misma; es útil usar esa "información histórica" para predecir "información futura".

- La necesidad de crear un modelo automático frente a las "técnicas clásicas" (estadística) que necesitan ser manejadas por un experto en estadística. Aunque los resultados, a menudo, siguen siendo interpretados por un analista de datos, se intenta orientarlos al usuario final, que no tiene por qué ser un experto.
- Trabaja con datos estructurados ([Delgado et al., 2002a]) , es decir, una colección de casos donde cada uno de ellos se describe por un conjunto de variables.
- Las técnicas de DM nos permiten trabajar con un gran número de ejemplos.
- Permite realizar tareas tanto predictivas como descriptivas ([Kantardzic, 2011]).
- Es independiente del lenguaje porque trabaja con datos numéricos o simbólicos ([Tan, 1999]).

Desde los años 90 en los que comienza su estudio y expansión, la Minería de Datos se ha llamado de diversas formas: *Data dredging, Data harvesting, Data archeology, Knowledge extraction, data/pattern analysis, data archeology, information harvesting.*

Hearst ([Hearst, 1999b]) menciona el concepto de *Information Archeology* (también tratado por otros autores ([Brachman et al., 1993])) como sinónimo de DM definido del siguiente modo:

Proceso de derivar nuevo conocimiento de una base de datos. Es una tarea humana que no puede ser realizada con simples métodos automáticos. El análisis consiste en la segmentación de clases con diferentes atributos y con diferentes parámetros, visualizando y comparando los datos y guardando y reutilizando el trabajo, incluyendo los resultados de análisis y las técnicas usadas para generar los resultados. La información útil se descubre a través del proceso dinámico de interactuar con los datos como hipótesis, que son formados, investigados y revisados.

Hearst señala la relevancia de la palabra *minería* como la extracción de "trozos de mineral" de una roca que, de otro modo, no tendría valor. En la actualidad, la Minería de Datos no realiza este proceso, sino que descubre tendencias y patrones en grandes conjuntos de bases de datos para la toma de decisiones. Sin embargo, si la Minería de Datos realizara el proceso de minería tal y como debe ser entendido, descubriría nuevos hechos dentro de las bases de datos de las empresas. En este sentido, Hearst vaticina que este proceso lo podrá llevar a cabo la Minería de Textos. Esta técnica se detallará en el capítulo 3.

Aunque DM está orientado hacia la toma de decisiones en las empresas, no se trata de una herramienta para realizar consultas al estilo de las herramientas OLAP, que también muestran los datos de tal forma que el directivo de la organización puede responder preguntas, puede que decisivas en el futuro del negocio; la Minería de Datos pretende descubrir información que no había sido pensada por el analista.

De este modo, mientras que con herramientas tradicionales se responde a preguntas del tipo:

¿Quiénes son los mejores clientes de mi organización?

¿Los clientes que compran en la sección de electrónica compran también en la sección de deportes?

Sin embargo, si necesitamos responder a preguntas para las que la respuesta no parece tan evidente, como puede ser la siguiente:

¿Existen clientes más predispuestos a las rebajas?

Tendremos que echar mano de herramientas de Minería de Datos para encontrar respuestas, aunque no podemos pasar por alto que también es posible aplicar dichas técnicas sobre las preguntas del primer tipo.

2.1.2.1 Aplicaciones de la Minería de Datos

Las metas y aplicaciones de DM son variadas, vemos algún ejemplo de ellas en la siguiente enumeración:

- Mejorar las capacidades de marketing: orientación de campañas hacia determinados clientes, fragmentación de la bolsa de clientes, web marketing, análisis de mercado (CRM, Venta cruzada).
- Detectar patrones anormales: la prevención de fraudes.
- En la industria: control de producción y logística.
- En el ámbito bancario: análisis de riesgos, créditos, prevención de morosos.

- En Gestión de Bases de Datos: para Ingeniería Inversa, mejorar la calidad de los datos, mejorar las consultas descubriendo dependencias funcionales.
- En general, para predecir el futuro, basándose en experiencias del pasado y en tendencias actuales.
- Descubrimiento de Conocimiento en Texto: es uno de los nuevos campos de aplicación de las herramientas de Minería de Datos, se trata de la Minería de Textos que desarrollaremos más ampliamente en la sección 2.2 de este trabajo.

2.1.2.2 Relación entre KDD y DM

El proceso de extracción de conocimiento no es una ciencia exacta, aunque las técnicas utilizadas para extraerlo, como pueden ser la estadística, sí lo sean. De este modo no todos los autores se muestran de acuerdo a la hora de definir DM y KDD. Hay quienes destacan el DM como la fase principal de KDD y quienes encuentran equivalentes ambos conceptos, como si se tratara del mismo proceso. A continuación desarrollaremos estos enfoques:

- **DM como fase de KDD**

  La **Minería de Datos** puede destacarse como la fase más importante del KDD, aquella que integra los métodos de aprendizaje y métodos estadísticos para obtener hipótesis de patrones y modelos. El trabajo en este área incluye la aplicación de *machine learning* y técnicas de análisis estadístico a través de descubrimiento automático de patrones en bases de datos estructuradas, así como proporcionar entornos guiados por el usuario en la exploración de datos ([Feldman et al., 1998b] , [Iiritano and Ruffolo, 2001], [Paralic and Bednar, 2002]).

  Su meta fundamental es descubrir o derivar nueva información de los datos, o encontrando patrones a través de los conjuntos de datos dados o bien separando las señales del ruido (o ambos procesos a la vez).

  Para Paralic ([Paralic and Bednar, 2002]), DM es una parte del proceso KDD: La confusión, dice, entre los términos KDD y DM viene de razones históricas y del hecho de que muchos de los trabajos están focalizados en refinamiento y aplicación de experimentos de algoritmos de *machine learning*. El preprocesamiento, a menudo, se incluye en el DM como una parte del algoritmo de minería.

- **DM como sinónimo de KDD**

  Hay otros autores, como Agrawal y Hearst, para los que DM es equiparable con KDD ([Hearst, 1999b], [Agrawal and Shafer, 1996]).

  Una posible explicación a esta equivalencia puede encontrarse en ([Orallo, 2002]) donde achacan la causa de esta paridad entre los procesos a que, como DM es la fase de generación de hipótesis, la más vistosa y la que produce resultados sobre los que trabajar y tomar decisiones (aspecto primordial en las organizaciones del momento) no es extraño que pase a identificar todo el proceso global de Descubrimiento de Conocimiento. De todas formas, hay muchos casos en los que no se puede distinguir claramente la fase de minería dentro del proceso de Descubrimiento ([Delgado et al., 2003]) porque puede que no sea necesario realizar todas y cada una de las fases del mismo, como preprocesamiento, limpieza de datos ...

2.1.2.3 Modelos y Técnicas en Minería de Datos

Los algoritmos de Minería de Datos que se emplean en la actualidad se pueden agrupar en dos modelos, el descriptivo y el predictivo. Esta distinción atiende a que DM permite construir un modelo que representa relaciones entre descripciones y resultados (variables descriptivas y predictivas, respectivamente), veamos los posibles modelos ([Thuraisingham, 1999])

- **Modelo descriptivo**: informa acerca de la relación entre los datos y sus características, por ejemplo:

  Los clientes de gasolineras que compran pañales
  a las 5 de la tarde suelen comprar cerveza.

En este tipo de modelos, se proporciona un conjunto de ejemplos sin clasificar y se pretende encontrar regularidades dentro de estos ejemplos y se intenta caracterizar tanto como sea conveniente dichos ejemplos.

Las técnicas usadas serán de aprendizaje no supervisado, tales como: Clustering (SOM, k-means clustering, k-means, ...), correlación y asociaciones, análisis estadístico (estudio de la distribución de los datos, análisis de la dispersión, detección de datos anómalos).

- **Modelo predictivo**: responden a preguntas formuladas acerca de los datos,

por ejemplo:

¿Se ha realizado fraude en la última transacción?

En estos otros, se da un conjunto de ejemplos u observaciones que se clasifican en un número finito de clases. Se pretende introducir una hipótesis que clasifica correctamente todos los ejemplos dados y no vistos. El propósito del DM predictivo es generar hipótesis que puedan usarse para clasificación.

Técnicas de aprendizaje supervisado de métodos predictivos, como por ejemplo: Clasificación (k-nn, clasificadores de bayes, reglas CN2), interpolación (regresión lineal, regresión no lineal, ...), árboles de decisión (ID3, C4.5, CART).

Se suele tener la visión de que se puede determinar la técnica que se utilizará según sea el tipo de conocimiento que se desea extraer, por lo tanto, si se sabe lo que se quiere descubrir se habla de Minería de Datos Dirigida (*Directed Data Mining*) y si no se sabe qué se busca, se habla de Minería de Datos No Dirigida (*Undirected Data Mining*).

Una vez que nos hemos introducido en el mundo de la extracción del conocimiento en datos, tendremos que abordar la extracción de trozos de información en texto, desconocida a priori. Éste es el objetivo de este trabajo. Para ello y del mismo modo que hemos hecho con los datos, comenzaremos con la definición del proceso de Extracción de Conocimiento en Textos.

En la siguiente sección abordaremos el Descubrimiento de Conocimiento en Textos o KDT y la Minería de Textos. Analizaremos los distintos nombres que ha recibido y la fase de Preprocesamiento para poder obtener una Representación Intermedia adecuada a nuestras necesidades y revisaremos las técnicas más utilizadas por los autores.

2.2 Descubrimiento de Conocimiento en Textos (KDT, *Knowledge Discovery in Text*)

Tradicionalmente, el descubrimiento de conocimiento se ha venido realizando sobre los datos almacenados en bases de datos estructuradas. Sin embargo, la mayoría de la información de la que disponen las organizaciones está en formato textual y, como se ha mencionado anteriormente, puede estar en intranets, páginas web, informes

de trabajo, publicaciones, correos electrónicos, etc.

El procesamiento de esta información suele ser complejo debido a la gran cantidad de documentos ([Tan, 1999]), su heterogeneidad y su falta de estructura ([Feldman and Dagan, 1995]). Durante los últimos años, se han realizado operaciones sobre documentos tales como catalogación, se han generado referencias, se han creado índices, se han extraído términos relevantes y se han extraído resúmenes con el fin de agilizar las búsquedas de información sobre ellos sin tener que volver a leer y estudiar el documento.

De forma casi paralela, en el entorno de las bases de datos, se han ido construyendo herramientas que facilitan el interrogatorio sobre un conjunto de documentos, pudiendo formular preguntas con el fin de localizar la respuesta en dichos documentos. Ahora bien, ¿y si queremos encontrar información que no estamos buscando expresamente?

¿Se puede realizar un análisis del contenido de estos documentos y conseguir que, tratados convenientemente nos aporten información nueva, que desconozcamos a priori y que resulte útil para las empresas u organizaciones?

Frawley, en su artículo ([Frawley et al., 1992]) define el descubrimiento de conocimiento como la extracción de información no trivial, implícita, previamente desconocida y potencialmente útil a partir de los datos. En nuestro trabajo, adaptaremos esta definición considerando que los datos son textuales.

Es precisamente esta consideración de datos textuales y no de datos estructurados de bases de datos la que hace emerger una serie de problemas que necesitan un tratamiento adicional, pudiendo requerir algún método o proceso que los trate convenientemente. Dichos problemas son:

- La falta de estructura del texto. Incluso en lenguajes semiestructurados como el HTML, el texto sigue siendo carente de una estructura homogénea procesable de forma automática sin que se produzca pérdida de información.
- La naturaleza heterogénea y distribuida de los documentos. Como ya hemos mencionado, las fuentes externas que nutren los almacenes de texto pueden ser tan diversas como podamos imaginar: intranets, bases de datos documentales, redes sociales, censos, páginas web, informes empresariales, publicaciones, correo electrónicos, etc.
- El multilingüismo presente no sólo en diferentes conjuntos de documentos, sino también dentro de una misma colección de textos.

- El análisis de texto depende del contexto y del dominio de la aplicación, lo cuál implica el uso de diccionarios, tesaurus u ontologías específicas de dicho contexto para poder llevar a cabo el procesamiento de los textos.

El proceso de Descubrimiento de Conocimiento en Textos implicará, por lo tanto, a diferentes ámbitos de conocimiento, principalmente: Recuperación de Información (para filtrar y reunir documentos adecuados), Extracción de Información (que selecciona hechos específicos sobre tipos de entidades y relaciones de interés), Procesamiento del Lenguaje Natural (para realizar el preprocesamiento y etiquetado de los textos) y Minería de Datos (para descubrir asociaciones desconocidas entre hechos desconocidos).

2.2.1 KDT versus KDD

En general, las fases principales del proceso de Descubrimiento de Conocimiento en Textos son las mismas que las del Descubrimiento de Conocimiento en Datos, a saber ([Goebel and Gruenwald, 1999]):

- **Fase 1:** Desarrollar y comprender el dominio de la aplicación
- **Fase 2:** Adquirir el conjunto de datos inicial
- **Fase 3:** Integrar y comprobar el conjunto de datos
- **Fase 4:** Limpiar, procesar y transformar los datos
- **Fase 5:** Desarrollar el modelo y construir la hipótesis
- **Fase 6:** Elegir los algoritmos de minería adecuados
- **Fase 7:** Interpretar y visualizar los resultados
- **Fase 8:** Verificar y probar los resultados
- **Fase 9:** Uso y mantenimiento del conocimiento descubierto

Sin embargo, considerando los problemas comentados al comienzo de la sección 2.1.2, tales como la heterogeneidad de las fuentes externas, tiene relación con problemas que también nos encontramos durante el proceso de Descubrimiento de Conocimiento en Datos (KDD), fácilmente podemos pensar que ambos procesos (KDD y

	KDD	**KDT**
1	Comprender el dominio de la aplicación.	El usuario define qué conceptos le interesan.
2	Seleccionar un conjunto de datos objetivo.	Los textos se obtienen con herramientas de Recuperación de Información o de forma manual.
3,4	Limpieza, Preprocesamiento y Transformación de datos.	Se describen los conceptos y los textos serán analizados y representados mediante una Forma Intermedia (se eliminan *stop words*, se desestiman palabras frecuentes poco relevantes ...)
5	Desarrollo de modelos y construcción de hipótesis.	Identificación los conceptos en la colección de textos.
6	Elección y ejecución de algoritmos adecuados de Minería de Datos.	Tareas de Minería de Textos.
7	Interpretación de resultados y visualización.	Interpretación de los resultados por un humano.

Tabla 1: Comparativa entre los procesos KDD y KDT ([Paralic, 2001])

KDT) pueden ser similares en su desarrollo; sin embargo, existe una diferencia fundamental entre texto y datos que hace necesaria una revisión del proceso de descubrimiento de conocimiento: la falta de una estructura procesable automáticamente hace que las etapas de selección, preprocesamiento y transformación de los datos textuales sea imprescindible en el KDT, conformando todas ellas las fase de Preprocesamiento.

Ambos procesos aparecen comparados en la Tabla 1 ([Paralic, 2001]). En ella vemos las fases normalmente consideradas en la literatura. Hemos agrupado alguna de las fases de KDD a la hora de compararlas con KDT.

De forma general, podemos resumir las fases principales del KDT en estas tres: *Preprocesamiento, Minería de Textos y Visualización*. Hay que tener en cuenta, no obstante, que en la literatura está mucho más extendido el término Minería de Textos que el término Descubrimiento de Conocimiento en Textos. Esto es debido a que la mayoría de las veces se utiliza la fase Minería de Textos para identificar al proceso de descubrimiento completo. Este fenómeno nace de una metonimia semántica: la fase principal del proceso identifica al proceso genérico.

En la siguiente sección presentamos la fases principales del proceso del Descubrimiento de Conocimiento en Textos: la fase de Preprocesamiento, la Minería de Textos y la fase de Visualización.

2.2.2 Fases Principales en el KDT: Preprocesamiento, Minería de Textos y Visualización

En esta sección abordamos las principales fases del Descubrimiento de Conocimiento en Textos: el Preprocesamiento para darle al texto una Forma Intermedia que permita su computación, la Minería de Textos y las técnicas que la hacen posible y, por último, la Visualización de resultados.

2.2.2.1 Preprocesamiento de los Documentos: Cómo Obtener una Forma Intermedia Adecuada

Ya sabemos que el texto no presenta una estructura fácil para aplicarle las técnicas de Minería de Textos directamente, así que habrá que realizar una serie de operaciones sobre él hasta conseguir una representación adecuada a nuestras necesidades.

Para conseguir realizar la ansiada minería, antes hay que preparar el texto de forma que admita ser procesado por técnicas de extracción de patrones útiles. No basta sólo con homogeneizar dichos documentos (procedentes de distintas fuentes de información y que pueden estar en distintos formatos e idiomas), ya que si nos quedamos aquí, se trataría de un procesamiento de textos avanzado, que puede llevarse a cabo mediante lingüística computacional. Sin embargo, y como indica Hearst ([Hearst, 1999b]), la lingüística computacional trata sobre la comprensión del lenguaje, computa estadísticas sobre grandes colecciones de texto para descubrir patrones útiles que se usan para informar subproblemas de procesamiento de lenguaje natural: etiquetas de *part of speech*, desambigüedad del sentido de las palabras y creación de diccionarios bilingües; no obstante, los patrones que se obtienen en la lingüística computacional carecen del valor desde el punto de vista empresarial. Lo interesante son los patronse que aportan conocimiento no explícito en el texto, como ocurre con la Minería de Textos.

La fase en la que realizamos operaciones sobre el conjunto de documentos objeto de estudio, es el **Preprocesamiento** (algunos autores, como por ejemplo Tan ([Tan, 1999]) la llaman *Text Refining*). Este paso es primordial ya que, en función de cómo representemos los datos, el resultado de la minería puede variar. Este problema queda reflejado en la Tabla 2 ([Montes-y Gómez et al., 2002]), donde podemos observar que si en la fase de Preprocesamiento se elige una técnica determinada, ésta establecerá el tipo de información obtenida en el descubrimiento.

Teniendo en cuenta esta *determinación*, habrá que elegir con cuidado qué técnicas

Preprocesamiento	Representación	Descubrimiento
Categorización	Vector de tópicos	Relaciones entre tópicos
Análisis de texto completo	Sequenciamiento de palabras	patrones de lenguaje
Extracción de Información	Tabla de base de datos	Relaciones entre entidades

Tabla 2: Relación entre Preprocesamiento, Representación Interna y Descubrimiento

Figura 3: Fases por las que pasa un documento ([S. Iiritano and Rullo, 2003])

utilizaremos para darle una representación interna al texto. Se pueden definir estructuras menos simples que pueden ser extraídas del texto con un coste razonable y de forma automática. Esta estructura debe conservar la riqueza del documento y debe permitir que se realicen operaciones sobre ella. Aunque hay diversos autores que la mencionan, ([Paralic and Bednar, 2002], [Iiritano and Ruffolo, 2001], [Feldman et al., 1998b]), no todos los autores tratan la fase de Preprocesamiento. De entre los que hemos seleccionado, sólo Feiju Xu, Iiritano, Paralic, Rajman, Feldman y Montes y Gómez hablan de ella explícitamente. Estos autores adoptan distintos enfoques:

- No se trabaja con todo el texto de los documentos, sino que se utilizan *documentos categorizados* para realizar Preprocesamiento, es decir, se toman documentos etiquetados con términos que identifican su contenido. La única opción posible para el descubrimiento será encontrar relaciones entre tópicos, con lo que se está perdiendo gran parte de la riqueza semántica del texto. Los autores que optan por esta opción son Feiju Xu y Feldman.
- Se trabaja con el conjunto de documentos en su totalidad, el corpus se somete a técnicas de NLP para realizar un Preprocesamiento lingüístico, como el etiquetado o la extracción de términos. Esta opción tiene en cuenta los conceptos y puede dar lugar a descubrir desde patrones en el lenguaje hasta representaciones más complejas, como grafos conceptuales.

Algunas de las técnicas utilizadas para la transformación de documentos en una Forma Intermedia, se resumen en la Tabla 11. Vemos que estas técnicas se pueden englobar en distintas áreas, como son el Preprocesamiento de Lenguaje Natural como la Recuperación de Información.

Una vez que se han aplicado las técnicas adecuadas para el Preprocesamiento, habrá que tener en cuenta qué tipo de Representación queremos obtener, abordaremos el tema de las posibles Formas Intermedias que hemos encontrado en la literatura en el Capítulo 2.3.

2.2.2.2 Minería de Textos (TM, *Text Mining*): Un Poco de Historia

En esta fase, también llamada de *descubrimiento*, se localizan los patrones, hechos, tendencias, eventos ... normalmente no explícitos en el texto.

En la figura 4 se puede observar la evolución de las referencias a la Minería de Textos desde sus primeras aplicaciones, en las que se consideraba análoga a la Minería de Datos aplicada a repositorios de información textual. Con el paso del tiempo, esta información documental almacenada en formato digital ha ido creciendo exponencialmente y la importancia de la Minería de Textos ha ido aumentando, apareciendo en la literatura con diferentes denominaciones.

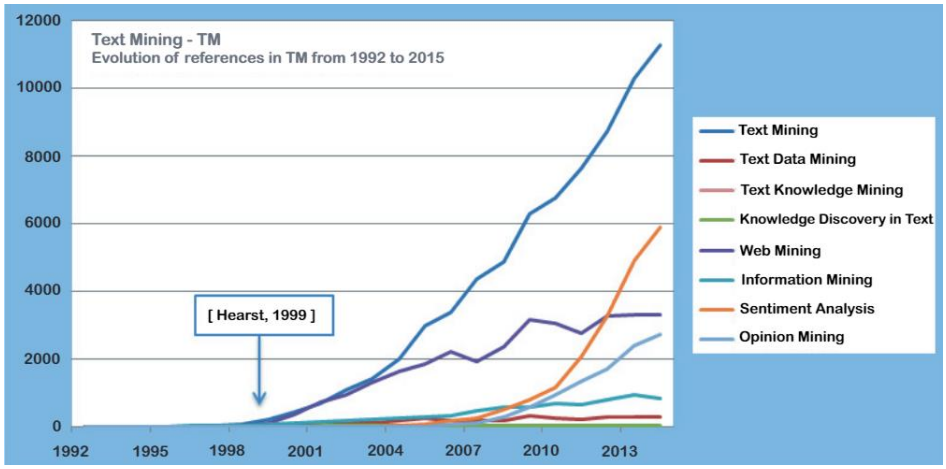

Figura 4: Evolución de referencias a Minería de Textos (Microsoft Academic Research)

Para conseguir un marco de trabajo adecuado para esta investigación, hemos realizado un análisis de la literatura científica localizando, sobre motores de búsqueda habituales como *IEEE Explore, dblp, scholar.google.es* y *Microsoft Academic Research*, referencias a artículos relacionados con la Minería de Textos y con otros términos que se suelen confundir con este tipo de minería.

Así, podemos observar que no sólo se utiliza indistintamente el concepto de Minería de Textos con el de Minería de Datos (conceptos que hemos diferenciado

anteriormente basándonos en la naturaleza estructurada y textual o no del conjunto de datos sobre el que se aplican) sino también con *Information Mining, Information Archeology, Knowledge Management, Data Mining, Knowledge Distillation, Document Mining, Text Data Mining, Opinion Mining, Web Document Mining* o *Sentiment Analysis*. Para los autores que hemos seleccionado para realizar este trabajo, no existe unanimidad: hay quienes identifican el proceso de Descubrimiento de Conocimiento en Texto con la Minería de Textos, mientras que otros, sin embargo, distinguen plenamente la Minería como fase principal del proceso global KDT. Hay autores que ven la Minería de Textos como un equivalente de la Minería de Datos pero aplicado a información textual y los hay que opinan que la Minería de Textos debería realizar algo más.

El concepto de Minería de Textos es relativamente reciente. Si observamos la imagen 4, no es hasta finales de los 90 cuando se produce el despegue de las investigaciones en este área. El trabajo de [Hearst, 1999b] marca un hito en este campo y sienta las bases tanto para diferenciar los conceptos de Descubrimiento de Conocimiento en Datos y en Textos como para definir qué es Minería de Textos, pero ¿qué hay de los otros conceptos mencionados?

Information Mining, que [Kruse et al., 1999] define como el proceso no trivial de identificar patrones válidos, nuevos, potencialmente útiles y comprensibles a partir de fuentes de información heterogéneas, se utiliza tanto para realizar minería sobre texto [Hsu and Yih, 1997], como sobre otros conjuntos de datos como, por ejemplo, imágenes [Datcu et al., 2003].

Web Mining sí es una expresión que claramente se confunde con *Text Mining* en la literatura. Son muchos los autores que lo utilizan para extraer información de texto pero reducen el conjunto de datos a la web [Kosala and Blockeel, 2000], [Svátek et al., 2004], [Wu et al., 2004], [Berendt et al., 2002]. *Web mining* trata sobre el uso de las técnicas de minería de datos para descubrir y extraer información automáticamente a partir de documentos y servicios Web [Etzioni, 1996], y tiene una estrecha relación con agentes de software o agentes inteligentes [Kosala and Blockeel, 2000]. Un interesante análisis de este término consistiría en encontrar las diferentes variantes en las que se presenta en la literatura: *Web data mining* [Madria et al., 1999], *semantic Web mining* [Berendt et al., 2002], *Web content mining* (si lo que interesa es la estructura interna de los documentos web) [Kosala and Blockeel, 2000], *Web structure mining* (si el objetivo son la estructura de los hiperenlaces dentro de la web) [Kosala and Blockeel, 2000] o *Web usage mining* [Kosala and Blockeel, 2000] que se focaliza en las técnicas que pueden predecir el comportamiento de los usuarios mientras interactúan con la web. Por lo tanto, podríamos encontrar una equivalencia entre el término *Text Mining* para documentos web y el término Web content mining o *Web data mining*.

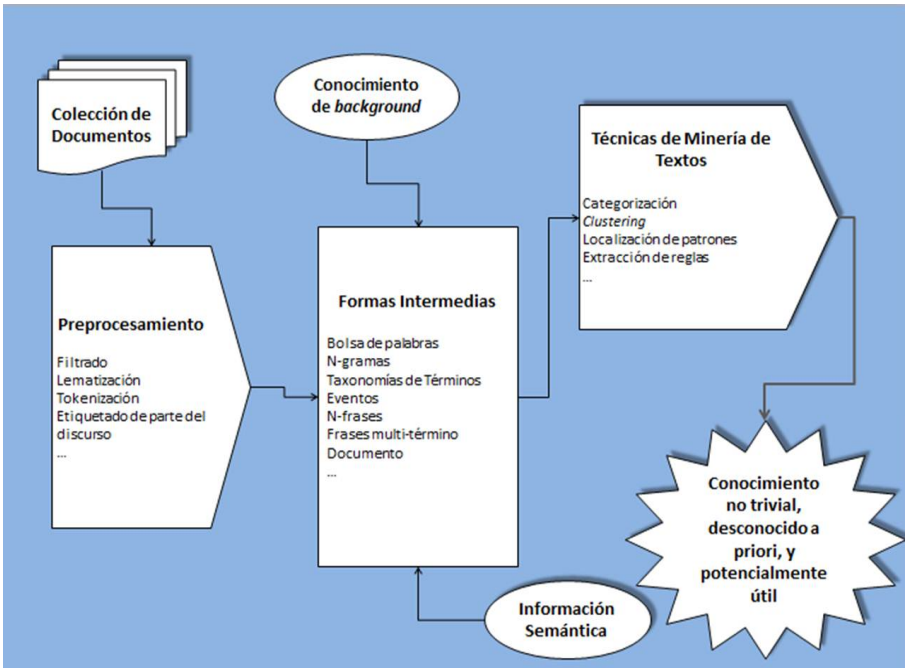

Figura 5: Proceso de Descubrimiento de Conocimiento en Texto

A principios del 2000, aparece el término *Sentiment Analysis* ([Thelwall, 2000], [Brewster and Miller, 2000]) unido con frecuencia a *Opinion Mining* [Dave et al., 2003] y algunos autores llegan a intercambiarlos [Pang and Lee, 2008], [Pak and Paroubek, 2010]. En concreto, *sentiment*, se puede encontrar relacionado con el análisis automático de texto de evaluación y seguimiento de los juicios predictivos que aparecen en ella. A partir de aquí, se ha focalizado en redes sociales, como Twitter, obteniendo diversidad de aplicaciones.

El término *Opinion Mining* se hizo popular entre las comunidades fuertemente asociadas con la búsqueda de información web o recuperación de información. [Dave et al., 2003]. Para Dave, el proceso de Opinion Mining sería aquel proceso de búsqueda de resultados dado un ítem, generando la lista de atributos (calidad, características) y aplicando técnicas de agregación sobre cada uno de ellos, de este modo dicho ítem se englobaría finalmente en categorías del tipo *(pobre, mixto, bueno)*.

En nuestro review nos centraremos en *Text Mining* ya que entendemos que la información textual, englobando no sólo a datos de la web o de redes sociales, sino a cualquier conjunto de datos de tipo textual.

Hemos seleccionado una serie de artículos escritos por autores que se aproximan

al área de la Minería de Textos y, gracias a sus estudios, intentaremos arrojar algo de luz sobre este problema. Primero, presentaremos qué opinan sobre la **Minería de Textos** y con qué la comparan; una vez que conozcamos los distintos enfoques y que nos hayamos posicionado respecto a este problema, hablaremos sobre las técnicas necesarias.

El proceso que trabaja sobre dichos documentos se ha identificado con *Information Mining, Information Archeology, Knowledge Management, Data Mining, Knowledge Distillation, Document Mining, Text Data Mining* o *Web Document Mining*. Para los autores que hemos seleccionado para realizar este trabajo, no existe unanimidad: hay quienes identifican el proceso de Descubrimiento de Conocimiento en Texto con la Minería de Textos, mientras que otros, sin embargo, distinguen plenamente la Minería como fase principal del proceso global KDT. Hay autores que ven la Minería de Textos como un equivalente de la Minería de Datos pero aplicado a información textual y los hay que opinan que la Minería de Textos debería realizar algo más.

Debido, probablemente a su corta vida, estos desacuerdos, lagunas y, sobre todo, distintos puntos de vista acerca de qué es lo que debe realizar la Minería de Textos en realidad son frecuentes en la literatura científica. Así, cada autor define, realiza experimentos y opina sobre su visión particular de Minería de Textos.

En la Tabla 3, se observa cómo llama cada autor al proceso de Minería de Textos, algunos de ellos le dan más de un nombre.

Observando la Tabla 3, podemos concretar que las posibles tendencias en Minería de Textos se pueden resumir con los siguientes enfoques principales: Atendiendo al ámbito dentro del proceso de Descubrimiento y Dependiendo de los resultados esperables de la Minería de Textos:

- Atendiendo al ámbito dentro del proceso de Descubrimiento:
  - **TM** como fase del proceso **KDT**: TM es la fase principal del proceso global KDT porque es en ella donde se descubren los patrones interesantes y desconocidos. En este punto, existen autores, como Feldman, que avanzan en su visión de Minería de Textos: mientras que en su artículo de 1996, hablaba exclusivamente de KDT, en el de 1998 ya realiza una distinción clara de TM como fase del KDT.
  - **TM** como **KDT**: quizá debido a que el TM es la fase principal del KDT y, en muchos casos, no es necesario realizar más trabajo sobre la información, hay autores como Delgado que indican que ambos conceptos (KDT y TM) pueden identificarse en algunas ocasiones.

- Dependiendo de los resultados esperables de la Minería de Textos:
  - **TM** como extrapolación de DM a texto: estos autores, como Iiritano, consideran una buena opción realizar la Minería de Textos del mismo que modo que se ha venido haciendo hasta ahora con los datos. Como veremos a lo largo del capítulo, no se pueden aplicar directamente las técnicas tradicionales sobre el texto ya que éste está sin estructurar y esas técnicas, basadas en la sintaxis, necesitan que los datos sobre los que actúan estén estructurados. Por ello, será necesario representar el texto de tal forma que pueda realizarse minería sobre él.
  - **TM** como un avance sobre DM: Hearst es de los primeros investigadores que hace la anotación de que el texto tiene más riqueza que los datos y, que si se le aplican técnicas que sólo tienen en cuenta la sintaxis, se estará perdiendo la semántica del mismo. Podemos utilizar una clasificación de Hearst (Tabla 4) donde detalla cómo se pueden obtener 'trozos de información nueva', es decir, cómo conseguir realizar Minería de Textos

Respecto a quienes lo denominan *Knowledge Extraction* o Minería de Información, vamos a puntualizar que sendos conceptos son demasiado amplios y pueden abarcar desde datos almacenados en bases de datos hasta texto, por ello, concretaríamos algo más el nombre y lo rescribiríamos como Minería de Información Textual o Extracción de Conocimiento a partir de Textos.

Ya puntualizamos en las sección 2.4.3 que Rajman ([Rajman and Besançon, 1998]) también habla de Extracción de Información cuando se refiere a Minería de Textos, pues bien, debido a que el término de Extracción de Información ya se utiliza para identificar un proceso ampliamente tratado en este trabajo y teniendo en cuenta que en los resultados que obtiene no aparece información desconocida sino que se extraen fragmentos del documento, debemos insistir en que nosotros usaremos IE como una fase previa a la Minería de Textos y no como la Minería de Textos en sí misma.

Los autores aportan una serie de definiciones para este proceso dentro de sus artículos, las más comunes son las siguientes:

	Text Mining (TM)	Text Data Mining (TDM)	Knowledge Discovery in Texts (KDT)	Information Mining (IM)	Knowledge Extraction (KE)	Text Refining	Full Text Mining
DELGADO [Delgado et al., 2002a]	X		X			X	
DUBOIS [Dubois and Quafafou, 2002]			X				
FELDMAN [Feldman and Hirsh, 1996] **FELDMAN** [Feldman et al., 1998b]	X		X				
GELFAND [Gelfand and Wulfekuhler, 1998]	X						
HEARST [Hearst, 1999b]		X					
IIRITANO [Iiritano and Ruffolo, 2001]			X		X		
MONTES Y GOMEZ [Montes-y Gómez et al., 2002]			X				
NAHM [Nahm and Mooney, 2002]	X						
PACK CHUNG [Wong et al., 2000]	X						
PARALIC [Paralic and Bednar, 2002]		X					
RAJMAN [Rajman and Besançon, 1998]	X						X
TAKEDA [Takeda et al., 2000]		X					
AH-HWEE TAN [Tan, 1999]			X	X			
FEIJU XU [Xu et al., 2002]	X						

Tabla 3: Sinónimos de Minería de Textos

	Patrones	Trozos de información no nueva	Trozos de información nueva
Datos no textuales	DM estándar	Consultas de bases de datos	?
Datos textuales	Lingüística Computacional	Recuperación de Información	TDM real

Tabla 4: Búsqueda de información nueva.

Proceso de extraer patrones interesantes a partir de grandes colecciones de textos para descubrir conocimiento [Tan, 1999] [Delgado et al., 2002a].

Descubrimiento de conocimiento en documentos sin estructurar [Nahm and Mooney, 2002].

Descubrimiento de Reglas de Asociación importantes dentro del corpus de documentos [Wong et al., 2000].

Instancia del descubrimiento óptimo de patrones [Takeda et al., 2000].

Descubrimiento de información útil y previamente desconocida a partir de texto sin estructurar [Xu et al., 2002].

Algunas de estas definiciones, como la de Pack [Wong et al., 2000], se centran demasiado en una técnica concreta de descubrimiento, otras, como la de Nahm, no especifican el tipo de conocimiento que debe descubrirse, es decir, la meta de la minería debería ser encontrar información desconocida, no escrita explícitamente. Los demás, se basan en el descubrimiento de patrones pero, como veremos en las siguientes secciones, el conocimiento descubierto puede representarse de diversas formas, desde patrones hasta grafos conceptuales, por lo que no podemos restringirlo en la definición.

Para nosotros, y basándonos en las definiciones presentes en la literatura científica, la Minería de Textos es:

El proceso que descubre información útil que no está presente explícitamente en ninguno de los documentos objeto de análisis y que surge cuando se estudian adecuadamente y se relacionan dichos documentos.

El hecho de *descubrir información útil* tiene que ver con la idea de descubrimiento de conocimiento, del que habla Nahm ([Nahm and Mooney, 2002]), es decir, se trata de obtener información que pueda ser interpretada y utilizada de forma válida, por ejemplo, para la toma de decisiones en una empresa. Es aquí cuando nos damos cuenta de que la Lingüística Computacional (ver Sección 2.2.2.1), que podría analizar el texto, no nos sirve como ejemplo de minería porque los resultados que obtiene no son adecuados para su explotación en entornos empresariales. En esta definición, no indicamos qué tipo de información se va a descubrir (patrones, tendencias, hechos ...) porque ésta dependerá no tanto de nuestras expectativas sino de la Forma Intermedia que elijamos para representar el documento.

La información descubierta no debe *estar presente explícitamente en ninguno de los documentos objeto de análisis*, porque de otro modo no se trataría de descubrimiento, objetivo principal de la minería. Para obtener fragmentos explícitos del documento, ya disponemos de herramientas de Extracción de Información y de Recuperación de Información.

Cuando se estudian adecuadamente y se relacionan dichos documentos. Estos documentos, con una estructura demasiado compleja, son los que vamos a tener que procesar y preparar para que puedan aplicarse herramientas de Minería de Textos sobre ellos, así que, volvamos a revisar la fase de Preprocesamiento del Documento, dentro del proceso global KDT, para conocer qué Formas Intermedias de Representación del Documento podemos obtener.

Existen, a nuestro parecer, diferentes *enfoques en Minería de Textos* que expondremos en el capítulo 3, basados principalmente en las técnicas que se utilicen para obtener la Forma Intermedia y en el tipo de conocimiento que se pretenda extraer a partir del texto.

Si lo que se pretende es extraer verdadero conocimiento desconocido a primera vista dentro de los documentos, no podemos dejar de lado la semántica del texto porque es ésta la que puede aportarnos más información interesante acerca de lo que el autor quiso expresar en el texto, sin embargo, las corrientes de estudio en TM, debido entre otras cosas a la extrapolación de las técnicas de DM a texto, pasan por alto en muchos casos el significado del texto, también hay que tener en cuenta la carga computacional que requiere trabajar con conceptos y con significado, no siempre puede resultar rentable utilizar dicha información.

2.2.2.3 Visualización

Una vez obtenido el conocimiento deseado (tras las fases de Preprocesamiento, Representación Intermedia y Minería de Textos) existe una fase de representación necesaria para finalizar el proceso de Descubrimiento de Conocimiento en Textos que puede resultar importante para el usuario final a la hora de interpretar los resultados de la minería: la **visualización**.

En esta fase se proporciona un entorno de exploración de datos guiado para el usuario que sea lo más amigable posible. Las últimas tendencias presentan los resultados mediante gráficas tridimensionales, páginas web o *tag clouds* ([Kim et al., 2011]).

El proceso que nos lleva hasta la fase de visualización, ([Wong et al., 1999]), podemos resumirlo como: a un conjunto de documentos se le aplica una serie de transformaciones (como Preprocesamiento, Representación Interna y Minería de Textos) y se obtienen, como resultado, Reglas de Asociación que se representan gráficamente en 3D, en ese caso.

Una vez obtenidos los conceptos, los términos, las tendencias o cualquiera que sea el resultado, se pueden utilizar métodos automáticos de visualización o bien pueden interpretarse los resultados directamente. Si se ha elegido realizar el proceso de visualización, pueden ser útiles las técnicas utilizadas por alguno de los autores que se han adentrado en este ámbito: [Dubois and Quafafou, 2002], [Paralic and Bednar, 2002], [Wong et al., 2000], entre otros.

Si se ha elegido realizar el proceso de visualización, pueden ser útiles las técnicas utilizadas por: ([Wong et al., 1999], [Wong et al., 2000]), que presentan sistemas para visualizar reglas de asociación aplicadas a Minería de Textos, permitiendo así una mejor detección de tópicos, ([Dubois and Quafafou, 2002]) que sugiere algoritmos de visualización basados en gráficos SOM para mostrar estructuras dinámicas obtenidas por la minería de textos, ([Paralic and Bednar, 2003]) que muestra, vía html, los resultados obtenidos en el clustering, ([Mei and Zhai, 2005]) que visualiza los temas mediante un *plot*, ([Lopes et al., 2007]) que afirma que las técnicas de visualización de la información deberían jugar un papel primordial en el descubrimiento y sugieren el uso de técnicas VTM (*Visual Text Mining*) que van desde mapas cognitivos hasta SOM, ([Federico et al., 2011]) que sugiere una solución basada en capas para la visualización de los resultados de minería aplicados a redes sociales utilizando algoritmos de Teoría de Grafos, ([Van Eck and Waltman, 2011]) que utiliza la aplicación VOSviewer, utilizada para explorar mapas de bibliométricos científicos, ([Kontopoulos et al., 2013]) que visualiza las ontologías utilizando la herramienta

OntoGen.

([Chen, 2005]) menciona algunos problemas existentes a la hora de representar la semántica de la información, puesto que debe trabajar con datos no numéricos, no espaciales y altamente dimensionados, entre ellos es necesario mencionar: la visualización del conocimiento del dominio, inferencia visual y predicción, medidas de cualidad intrínsecas o la propia estética de la representación. Aún así, sugiere que la navegación a través de la información visual requiere.

Hasta aquí, hemos presentado el proceso de **Minería de Textos**. En los siguientes apartados, presentaremos los distintos enfoques de Minería de Textos que podemos encontrar atendiendo al contenido semántico de los documentos y veremos qué estudios futuros podemos esperar sobre este tema.

2.2.3 Aplicaciones de la Minería de Textos

Las aplicaciones prácticas de la Minería de Textos son muy diversas, aunque el diseño de dichas aplicaciones no es sencillo debido fundamentalmente a la complejidad que presenta el problema de obtener una estructura adecuada a partir del texto, estructura que depende del problema concreto que se esté tratando. Además, tal y como ocurre en el ámbito de la extracción de conocimiento a partir de datos en general, se requiere asimismo la participación de expertos humanos.

Existen diversas técnicas de Minería de Textos que se pueden aplicar a problemas del mundo real. Weiss ([Weiss et al., 2010]) proporciona una detallada lista de aplicaciones. Nosotros hemos seleccionado una serie de aplicaciones adicionales que también relatamos a continuación y que se pueden ver de forma esquemática en la tabla 5. Nuestra recopilación no pretende ser exhaustiva, sino proporcionar un catálogo de casos que demuestren la utilidad real de las herramientas desarrolladas en este área:

1. **Web Semántica** ([Berners-Lee et al., 2001]). Se trata de una extensión de las capacidades de la Web actual, a la cual se la dota de nuevos elementos semánticos como metadatos, lenguaje estructurado XML y ontologías para conseguir que se pueda razonar con la Web. Algunas aplicaciones generales de Minería de Textos en el contexto de la Web Semántica se muestran en ([Knublauch et al., 2004], [Berendt et al., 2002], [Liu and Wong, 2009], [Maedche and Staab, 2001]). Una aplicación más concreta es la proporcionada por Medelyan ([Medelyan et al., 2009]): los autores plantean cómo interactúa Wikipedia con el público y si supone una amenaza para la Web Semántica. La

Formas Intermedias sugeridas son las ontologías y realizarán reconocimiento de entidades para extraer hechos y relaciones semánticas del texto y a partir de ellos, construir ontologías. Utilizarán varias medidas basadas en relaciones semánticas: medida del path normalizado o medidas de desambiguación basadas en enlaces.

2. **Redes sociales**: Herramienta imprescindible en nuestra vida diaria, donde ya es inconcebible no estar concectado con el mundo mediante *Facebook* o *Twitter*. Son múltiples las aplicaciones en este campo:

   Filtrado de emails: gracias a la Minería de Textos se puede desarrollar una herramienta útil y configurable para organizar en carpetas la ingente cantidad de correos electrónicos que un usuario puede recibir diariamente. Cada mensaje se tokeniza con las 15 palabras más interesantes (entendiendo por interesante aquella palabra que tiene poca probabilidad de ser *spam*). El clasificador realizará esta operación y decidirá si el mensaje es o no *spam* ([Weiss et al., 2010]).

   Personalización de perfiles web, publicaciones digitales: las noticias de última hora aparecen en tiempo real en la infinidad de páginas web fuentes de noticia. Las editoriales deben dedicar efectivos para determinar si esa información es útil o no. La aplicación propuesta, utilizará un recuperador de documentos interesantes de los sitios web de noticias deseadas y se clasificarán los artículos en función de varios temas que se conocerán a priori. Mediante un cluster, los documentos entrarán en una u otra categoría basándose en la similaridad entre ellos. Estos artículos podrían mezclarse para obtener un nuevo artículo o bien se podrían obtener cabeceras de noticias interesantes ([Weiss et al., 2010]).

   Minería de redes sociales para aplicar teorías sociales ([Tang et al., 2014]), **Detección de comunidades Web** ([Tang and Liu, 2010]), existen millones de usuarios que juegan, se etiquetan, trabajan, interactúan y socializan online demostrando nuevas formas de colaboración, comunicación e inteligencia, lo cual hace reformular los modelos de negocios y de emociones.

   También son múltiples las aplicaciones dirigidas a **Twitter**, ([Bifet and Frank, 2010], [Pak and Paroubek, 2010]) es un reflejo de lo que está ocurriendo justo ahora. En estas aplicaciones, cada *tweet* se puede recuperar o bien en formato XML o bien en formato JSON. Ambos formatos

ya aportan estructura al texto y sobre ellos las posibilidades de aplicaciones son extensas, realizar *Opinion Mining*, *Sentiment Analysis*, *Survey Mining*...

Otro autor que realiza **Análisis de redes sociales** ([Kosala and Blockeel, 2000]), descubre diferentes tipos de páginas mediante el análisis de los hiperenlaces, tanto entrantes como salientes.

3. ***Opinion Mining* y *Sentiment Analysis***: son muchas las aplicaciones que ofrece la Minería de Textos sobre redes sociales como *Twitter* si además aplicamos análisis de sentimientos o de opinión ([Pang and Lee, 2008], [Pak and Paroubek, 2010]).

   Sentiment Analysis ([Bifet and Frank, 2010], [Pak and Paroubek, 2010], [Pang and Lee, 2008], [Nasukawa and Yi, 2003], [Gamon et al., 2005]): el *Sentiment Analysis* se puede resumir someramente como un problema de clasificación donde la tarea principal es ubicar mensajes en dos categorías: sentimientos positivos o negativos, incluso sentimientos neutros.

   Partiendo de un conjunto de términos con etiquetas referentes al sentimiento expresado dentro de los *tweets* y aplicando clasificación de términos y *part of speech tagging*, los autores ([Bifet and Frank, 2010]) sugieren aplicaciones diversas como sistemas de recomendación, detección de *flames* en correos electrónicos, detección de contenido inapropiado en páginas web, resúmenes desde diversos puntos de vista, responder a preguntas dirigidas a la opinión general... Mientras que Pak ([Pak and Paroubek, 2010]) aplica n-gramas y clasificadores de Bayes para obtener el clasificador de sentimientos positivos, negativos y neutros en documentos.

   La aplicación descrita ([Bifet and Frank, 2010]), muestra *tweets* relacionados con una pregunta y los englobará en el sentimiento principal predominante otorgándoles un color dependiendo si el contenido del *tweet* es positivo, negativo o neutro. Por ejemplo, si buscamos el *hastag #SuperBowl*, encontraremos tanto *tweets* positivos *"liifex : RT Pharrell: You were phenomenal at the #SuperBowl MissyElliot. So great seeing you up there, sis. http://t.co/NRtK8ngMC7"* como neutros *"liifex : RT Pharrell: You were phenomenal at the #SuperBowl @MissyElliot. So great seeing you up there, sis. http://t.co/NRtK8ngMC7"*. Sin embargo, si buscamos *#GlobalCrisis*, encontraremos sólo resultados negativos *"Djalvi80Alvi : RT Hmansoor: Plague kills dozens of people in Madagascar via CNN. Ebola, polio, measles and now ...the #plague? #publichealth #globalcris..."*, entre otros.

Opinion Mining: otro ejemplo concreto de aplicación sobre *Twitter* son las **encuestas de opinion** ([O'Connor et al., 2010]) en las que se conectan medidas de opinión pública con medidas de *Sentiment Analysis*. Analizando varias encuestas de opinión de consumidores y opiniones políticas para encontrar frecuencias de palabras con cargas de sentimientos en *Twitter*.

También se puede utilizar **para detección de puntos calientes en foros y realización de pronósticos,** ([Li and Wu, 2010]). Partiendo de los foros deportivos Sina, se pretende realizar búsquedas de información con objetivos de marketing, persiguiendo orientar y aconsejar en las inversiones en los mercados financieros. Tras analizar los enlaces de los foros, se completan una serie de plantillas XML con los temas principales encontrados en esos enlaces. Se le añade un valor adicional procedente del *Sentiment Analysis* para todas las palabras clave del foro. Sobre el conjunto de plantillas XML se aplican las siguientes técnica de Minería no supervisada: *Machine Learning, K-Means, SVM forecasting*. Los autores crean un algoritmo para analizar automáticamente la polaridad emocional de un texto y obtener, así, el valor de cada parte del mismo. Gracias a la Minería de Textos, agrupan los foros en varios clústers, el centro de los mismos será el foro "*hotspot*" dentro de la correspondiente ventana de tiempo.

Otros trabajos de Análisis de Sentimientos los encontramos en ([Munezero et al., 2014]), **Detección Automática de Comportamiento Antisocial en Textos**: mediante la ayuda de una ontología de emociones, construyen una aplicación que identifica las emociones que existen en los siguientes corpus textuales especializados (ASB: corpus de documentos hostiles, agresivos y violentos, ISEAR: corpus de informes de estudiantes en los que expresan emociones de miedos, disgusto, vergüenza. Reseñas de películas. Extractos de textos de Wikipedia relacionados con emociones fuertes) y ayuda a prevenir u observar comportamiento perjudicial presente en el texto. Extrae una bolsa de palabras y aplica clasificación supervisada: *Multinomial Naïve Bayes, SMO - Support Vector Machines*, Árboles de Decisión (J48).

4. **Métodos de síntesis y de organización**: en esta categoría englobaremos aquellos autores que sintetizan documentos o colecciones de documentos bien sea con revisiones sistemáticas de documentos, mediante la realización de resúmenes, la obtención de titulares ([Thomas et al., 2011]) o la extracción de ideas útiles.

El **resumen de documentos** y **síntesis de información** es un proceso mediante el cual se extraen las fases más destacadas basándose en una serie de supuestos como son que un lector humano descompone cada frase en un conjunto de fragmentos de información a la que la frase se refería, que los fragmentos de información son independientes entre sí y que un fragmento de información tiene una determinada importancia. Existirán varias técnicas que excluyan la información redundante y dispersa por todo el documento y que ordenarán la frases extraídas de acuerdo con la relación cronológica de los temas ([Weiss et al., 2010], [Barzilay and Elhadad, 2002], [Thomas et al., 2011], [Neto et al., 2000], [Mei and Zhai, 2005], [Thomas et al., 2011], [Turney, 2000], [Tseng et al., 2007], [Mei and Zhai, 2005], [Hu and Liu, 2004]).

Extracción de ideas útiles ([Thorleuchter et al., 2010]): los proyectos de investigación del NIST (*National Institute of Standards and Technology*) sirven, en este artículo, de base para la obtención de ideas útiles del texto sin estructurar para aplicarlo a sistemas de defensa. Se extraen patrones del texto en forma de vectores de términos y sobre ellos aplican técnicas de clasificación de textos como (*tokenization, term filtering methods, Euclidean distance measure etc.*).

Otras herramientas útiles: la extracción de *keyphrases* que se capturarán y conformarán los **principales tópicos de discusión de artículos científicos** como los de
([Nguyen and Kan, 2007]), y los trabajos sobre *Knowledge Mining* de Swanson ([Swanson and Smalheiser, 1999]) y nuestra búsqueda de contradicciones en textos (según se expone en este trabajo en los capítulos 3 y 4).

La **Organización bibliográfica y de autores** es otra aplicación de la Minería de Textos mediante la cual se puede utilizar minería para **organizar documentos**: cartas oficiales, ficheros personales, artículos científicos, colecciones de documentos, artículos de periódicos o bases de datos corporativas de texto completo ([Kaski et al., 1998]). También se puede **organizar la literatura médica** online en un tiempo razonable ([Chen et al., 2005]) o bien se puede utilizar para la **atribución de autoría de artículos** ([Kešelj et al., 2003], [Peng et al., 2003]): identificación del autor de un texto anónimo o aquellos cuya autoría presenta dudas. La aplicación consistiría en construir un perfil por autor que consistiría en un conjunto de los n-gramas más frecuentes de ese autor. Para comprobar la auditoría de otros autores, se comparará la similaridad entre n-gramas incluso para **detección de plagios** ([Kešelj et al., 2003]).

El **análisis de patentes** es otro campo de organización: los documentos de patentes contienen una gran cantidad de resultados de investigación importantes, pero por tener una longitud considerable y ser tan ricos en terminología técnica, suponen un gran esfuerzo para el ente humano que se enfrenta a ellos ([Tseng et al., 2007]).

5. **Minería sobre Thesaurus, Fuentes Abiertas...**: una **herramienta de código abierto para realizar minería sobre Wikipedia** la encontramos en ([Milne and Witten, 2013]): proponen una herramienta Wikipedia Miner Toolkit que puede identificar temas dentro de los documentos y generar medidas de relación semántica entre ellos. Además de proponer el cálculo de medidas de relación semántica y detección de enlaces en búsquedas, presenta trabajos basados en esta herramienta para construir modelos de los temas para realizar resúmenes multi documento, e identificar las mejores sentencias para ello ([Nastase et al., 2009]). El proceso seguido comienza con el preprocesamiento del conjunto de artículos de la Wikipedia, que devuelve documentos iguales a los originales pero etiquetados con marcas y etiquetas html que ayudarán si hay que realizar una desambiguación de contenido. Mientras se realiza dicho etiquetado, se obtiene una lista de temas del documento utilizando n-gramas. Para la desambiguación de términos, utiliza medidas semánticas y para construir un buscador de thesaurus que permita a los usuarios explorar los conceptos de la Wikipedia, utiliza técnicas de etiquetado.

6. **Marketing y Comercio Electrónico**: una de las aplicaciones de este área es el **Sistema Inteligente de Mercado vía web** cuya la misión fundamental es la toma de decisiones y suele requerir la interacción de un agente humano que establecerá las condiciones iniciales y direccionará los resultados. Para ello, se utilizarán las diversas noticias existentes en la web que tengan que ver con el producto deseado (fundamentalmente los textos escritos por la propia compañía y los que han sido escritos por la competencia), se analizará la información de dichos artículos y se obtendrá inteligencia de mercado en tiempo real. El conjunto de textos utilizado es las noticias de Reuters o *Associated Press*. Los documentos se limpiarán de información irrelevante y se convertirán a XML. Este etiquetado proporcionará la estructura necesaria para el texto antes de aplicar las técnicas de minería. Sobre los documentos etiquetados se utiliza un categorización de textos para encontrar un diccionario de las palabras más frecuentes que se encuentren dentro de los documentos. Posteriormente se vectoriza cada documento con las características binarias y se aplicarán reglas de inducción para clasificarlos. El agente humano recuperará

aquellos documentos que cumplen las reglas de su interés ([Weiss et al., 2010]).

Entre los trabajos de Minería de Textos aplicados a **marketing** encontramos ([Fan et al., 2006], [Gupta and Lehal, 2009]): La tecnología de Minería de Textos puede ayudar al marketing profesional encontrando *information nuggets*. El análisis de mercado implica fuentes variopintas de información: transacciones de tarjetas de crédito, tarjetas de fidelización, cupones de descuento, quejas de clientes... Es útil encontrar un grupo modelo de usuarios con intereses, hábitos de consumo e ingresos comunes, determinar patrones de adquisición a través del tiempo, por ejemplo, al pasar de soltero a casado...

Otros autores hablan de **e-commerce** en este ámbito, entre ellos: ([Zhai et al., 2004], [Song and Shepperd, 2006], [Thorleuchter and Van Den Poel, 2012]).

7. **E-Learning y Aplicaciones Help Desk**:

   Una aplicación práctica de **e-aprendizaje**, como puede ser la construcción de clasificadores automáticos de habilidades de e-tutores para aprendizaje colaborativo la podemos estudiar en ([Santana Mansilla et al., 2013]) y ([Mansilla et al., 2014]).

   Sobre la **Generación de Casos Modelo en Aplicaciones Help Desk**, Weiss, en su artículo, ([Weiss et al., 2010]) concreta el objetivo de una aplicación Help Desk, que será obtener un conjunto conciso de informes resumen que sirvan para solucionar problemas sin la intervención de expertos en un servicio de información. Cada una de las consultas online o de los problemas presentados por los usuarios en un *help desk* será el conjunto de documentos inicial sobre el que se trabajará. La información contenida en ellos es, con frecuencia, redundante puesto que varios usuarios pueden realizar las mismas consultas. La redundancia se eliminará con técnicas de clustering de alta dimensión. Para limpiar los documentos restantes se utilizarán técnicas de extracción y de resumen de documentos. Para proporcionar estructura a los documentos, se utiliza un vector con las palabras clave existentes en el título y las palabras más frecuentes del mismo.

   Las aplicaciones *Help Desk* las estudian diversos autores: ([Chang et al., 1996], [Sakurai and Suyama, 2005], [Weiss et al., 2000], [Nasukawa and Nagano, 2001]).

([Nasukawa and Nagano, 2001]) en concreto, presenta un análisis de informes de Help Desk. La aplicación de minería de textos permite inspeccionar todos los datos. Permite detectar fallos del producto, problemas que hayan tenido los clientes, las razones de dichos problemas, analizar la productividad de los trabajadores de help desk y analizar los cambios en los comportamientos de los clientes sobre algún producto en particular en un período de tiempo.

En esta sección hemos recorrido los trabajos más relevantes sobre Minería de Textos en los que se aportan soluciones a problemas reales de la vida diaria. En la siguiente sección reflexionaremos sobre otras vertinentes de la minería y propondremos una nueva visión.

2.2.4 Discusión

El Descubrimiento de Conocimiento en Textos presenta grandes similitudes y múltiples diferencias respecto al Descubrimiento de Conocimiento en Datos. La especifidad del texto, el idioma, su procedencia, su almacenamiento, cualquier característica lo convierte en único y diferente. Y nos aparta del camino del KDD.

Este estudio nos ha llevado por otros términos, que se pueden confundir en la literatura científica en ocasiones con Minería de Textos pero que, a nuestro parecer, se corresponden con esta fase del KDT sino que se trata de otras disciplinas de conocimiento, obsérvense las definiciones de *Opinion Mining, Sentimental Analysis*, entre otros.

Hemos podido comprobar la enorme cantidad de representaciones intermedias diferentes que se le puede proporcionar al texto, recorriendo un abanico que varía tanto en complejidad como en eficiencia computacional, englobando desde la bolsa de palabras hasta el grafo semántico.

Elegida la Forma Intermedia, hemos presentado diferentes aplicaciones sobre las que se aplica actualmente la Minería de Textos y que están muy cercanas al usuario convencional: aplicaciones para web semántica, redes sociales o herramientas de *help-desk* se benefician en la actualidad de esta disciplina.

En la siguiente sección propondremos una técnica de Minería de Textos, presentaremos el concepto de *Text Knowledge Mining* como el proceso de extracción de verdadero conocimiento a partir de textos sin estructurar y expondremos las razones que nos llevan hasta esta definición.

Área de Aplicación	**Uso**
Web Semántica	Servicios Web de Descubrimiento
	Aprendizaje de ontologías
Redes Sociales	Filtrado de Email
	Personalización de perfiles web
	Detección de Comunidades Web
	Teorías Sociales
Opinion Mining, *Sentimental Analysis*	Clasificación de Opiniones
	Hotspots en Foros
	Predicciones
	Detección de Comportamiento Antisocial
	Encuestas de Opinión
Síntesis, Organización	*Summarization*
	Revisiones sistemáticas
	Obtención de Titulares
	Extracción de Ideas Útiles y Nuevas
	Discusión de Temas Principales
	Organización de Documentos
	Detección de Plagios
	Búsqueda de Contradicciones
Minería de Fuentes Abiertas, Tesauros	Identificación de Temas
	Exploración de Tesauros
e-Commerce	Toma de Decisiones
Marketing	Localización de "trozos" de información
	Análisis de Mercado
e-Learning	Herramientas Colaborativas
Help Desk	Generación de Casos Modelo
	Detección de Fallos

Tabla 5: Aplicaciones de la Minería de Textos

2.3 Estudio de las Formas Intermedias

Uno de los requisitos principales del proceso de Descubrimiento de Conocimiento en Textos es proporcionar una representación intermedia al texto que sea manejable computacionalmente. Este paso, crucial y complicado, debe lidiar con datos procedentes de fuentes multilingues, debe identificar relaciones semánticas entre conceptos nuevos y antiguos o relacionar identificadores, todo ello procurando que el texto original no pierda su integridad. Esta representación intermedia del texto es conocida también como Forma Intermedia y en este trabajo se utilizarán ambos términos indistintamente ([Justicia de la Torre et al., 2008]).

En este capítulo nos adentraremos en las unidades mínimas en las que se puede dividir una colección de documentos y veremos cómo se eligen las Formas Intermedias según el tipo de descubrimiento que se desea obtener y las aplicaciones posibles de los resultados obtenidos.

2.3.1 Unidades Mínimas dentro de la Colección

Cuando nos planteamos el hecho de analizar los corpus textuales, la primera prueba con la que nos encontramos es la *aparente* uniformidad de los elementos que componen la colección. Decimos que es sólo aparente ya que estos elementos, pueden ser no sólo diferentes en su temática (documentos con información financiera o registros de ventas), sino en su estructura (correos electrónicos simples o informes técnicos) y en el idioma en el que han sido redactados. De este modo nos encontramos ante un conjunto heterogéneo de información a partir del cual pretendemos obtener conocimiento.

Supongamos que tenemos una colección de documentos Ω sobre la que pretendemos extraer conocimiento, nuestro principal objetivo será proporcionar cierta estructura a dicho texto intentando representar, mediante formas intermedias existentes, partes fundamentales de la colección. Para ello, dividiremos el texto en unidades mínimas como pueden ser las elementales en un documento, como la palabra, un concepto, una frase o párrafo o el documento completo. Sin embargo, podríamos pretender realizar este análisis sintáctico dividiendo el texto en otros fragmentos no tan habituales.

Una de las posibilidades a la hora de analizar la colección, podría ser la de concatenar sus documentos de tal forma que dejaremos de ver nuestra colección de documentos como un repositorio de artículos, informes o documentos web y la

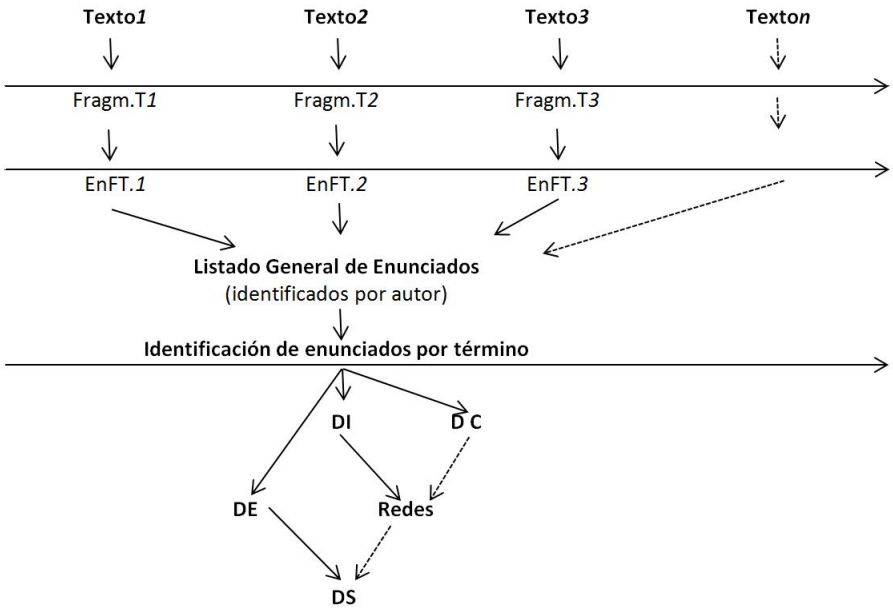

Figura 6: Fragmentar textos ([Winchkler, 1997])

veremos como una unidad superior formada por segmentos textuales de longitudes distintas con o sin marcas de división entre ellos.

A partir de esta forma de representar la colección o de cualquier otra, extraeremos trozos de texto que representaremos mediante una estructura que permita mantener la mayor información sintáctica y semántica posible.

En la figura 6, vemos un ejemplo de extracción de fragmentos de texto donde Giovana Winchkler ([Winchkler, 1997]) indica cómo pasar de fragmentos textuales numerados a una definición de un diccionario, todo ésto a partir de un conjunto de documentos arqueológicos, y seleccionando enunciados donde aparece un término concreto. En dicho gráfico, cada *Fragm.Ti* es un Fragmento de texto, los *EnFT.j* son enunciados correspondientes a un fragmento de texto, *DI* son definiciones implícitas (cuando el término no está en la cabecera del enunciado), *DC* son definiciones contextuales (como las implícitas), *DE* definiciones explícitas (aquellas en las que el término se define expresamente a través de un verbo copulativo, por ejemplo, *"La metafísica es ..."*) y *DS* son definiciones sintéticas (compuestas por varias de las anteriores).

Las distintas técnicas para unir los documentos no las abordaremos en esta sección, proporcionaremos una visión genérica de la mezcla de ontologías en el

capítulo , puesto que una de las finalidades de este trabajo será trabajar con una forma intermedia que proporcione estructura al texto.

Los métodos para obtener los segmentos textuales, también los estudiaremos en este trabajo:

1. Podríamos extraer los elementos básicos que componen un texto, por ejemplo, quedarnos desde la unidad mínima de significado para nosotros como es la palabra, una frase, un párrafo, ..., así hasta el documento completo (que podría estar identificado dentro de la colección con unas marcas de comienzo y de fin).
2. O bien, seleccionar aquellas partes del texto que contengan algún término en concreto que nos sirva para el estudio perseguido, por ejemplo, troceamos los párrafos en busca de algún concepto en concreto, con lo que no sólo conseguimos el concepto sino la información adyacente al mismo.

Los métodos para extraer información sintáctica del texto pasan por un análisis sintáctico adecuado. Disponemos de una definición del mismo en el capítulo 2 pero puede que presente una serie de inconvenientes que preferimos no abordar puesto que nuestra misión no es tanto realizar un procesamiento lingüístico como extraer conocimiento.

Para solventar algunos de los problemas que puede presentar el análisis sintáctico, como pueden ser la ambigüedad estructural o la complejidad algorítmica para realizar dicho análisis, existen algunas alternativas entre las que encontramos el *análisis parcial*, el *chunking* (troceado) ([Abney and Abney, 1991], [Bisbal et al., 2004]) o el *análisis superficial* (*shallow parsing*) ([Sha and Pereira, 2003], [Molina and Pla, 2002]).

El *análisis parcial de oraciones* consiste en obtener información sintáctica de interés a partir del texto, evitando los conflictos del análisis global, consiguiendo un análisis robusto para textos donde los algoritmos son más eficientes ([Abney and Abney, 1991]).

El *chunking* (o troceado) implica dividir, mediante un análisis superficial, las frases en segmentos que no se superpongan, cuyos elementos sólo pertenezcan a uno de los segmentos cada vez, por ejemplo: *[$_N$ Some bankers $_N$] [$_V$ are reporting $_V$] [$_N$ more inquiries than usual $_N$] [$_N$ about CDs $_N$] [$_N$ since Friday $_N$]* siendo N y V las etiquetas proporcionadas a las secciones de la frase según contengan o no partículas verbales ([Ramshaw and Marcus, 1995], [Tjong Kim Sang and Buchholz, 2000]).

El *análisis superficial* (*shallow parsing*) ([Sha and Pereira, 2003],

[Molina and Pla, 2002]) divide el texto en ciertas unidades sintácticas y, aunque puede perder cierto detalle respecto del parsing total, se ha convertido en una buena alternativa al mismo.

Con el fin último de no perder información en estas divisiones de la colección original, presentamos una serie de elementos que se pueden extraer *literalmente* de los documentos, estos elementos son: la **palabra**, el **concepto**, la **frase**, el **párrafo** o un **documento completo** de la colección.

2.3.1.1 Palabra

Las palabras que forman un texto, aunque aparentemente en sí mismas no aportan demasiada información, sí que pueden proporcionar un primer análisis del corpus de textos de tal forma que se determinen ciertos factores no semánticos del mismo, por ejemplo, gracias a su conteo y análisis conseguiremos determinar el tamaño de la colección, las principales funciones sintácticas de dichas palabras y la riqueza léxica del texto (la proporción existente entre el número de palabras distintas y el total de palabras del corpus). Lo que sí hay que tener en cuenta es que las palabras son independientes del contexto, por lo que servirán para una representación más general de los documentos a costa de perder información semántica.

Como es el elemento mínimo de estudio sobre el texto, la identificamos con el símbolo w_k, donde $k = 1, .., m$ siendo m el total de palabras de un documento. Cada w_k será una secuencia de caracteres que se distingue dentro de la frase por ir delimitada por caracteres en blanco.

2.3.1.1.1 Bolsa de Palabras (*Bag-of-words ó BOW*) La bolsa de palabras es un tipo de representación de documentos que procede de la Recuperación de Información clásica, basada en los modelos probabilístico, booleano y espacio-vectorial, como vimos en la introducción. Este tipo de forma intermedia se puede conseguir aplicando técnicas de procesamiento de lenguaje natural al texto o técnicas de indexación en recuperación de información.

Es una de las formas intermedias más frecuentes en la literatura: ([Ahonen et al., 1998], [Delgado et al., 2002a], [Feldman et al., 1998b], [Kosala and Blockeel, 2000], [Iiritano and Ruffolo, 2001], [Martín-Bautista et al., 2003], [Nahm and Mooney, 2001], [Rajman and Besançon, 1998], [Takeda et al., 2000], [Weiss et al., 2010], [Wong et al., 2000], [Yang et al., 1999]).

Una representación basada en bolsa de palabras es aquella en la que se toman todas las palabras del documento, se eliminan sus relaciones tanto sintácticas como semánticas y se almacenan de tal modo que quede accesible para futuros tratamientos.

Dentro de esta bolsa elemental, encontraremos elementos repetidos (las palabras que aparezcan más de una vez en el documento), otros que no aporten información nueva para el análisis o palabras con gran carga semántica. Veamos cuáles son estos tipos de palabras:

1. **Palabra vacía (*Stop word*)** : Palabras que ocurren frecuentemente en el texto pero que no aportan significado relevante, por ejemplo, los artículos, preposiciones y conjunciones.
2. **Término Índice** (***Keyword* o Palabra Clave**) : Palabra cuya semántica ayuda a recordar los temas principales del documento, es decir, identifica la temática del texto o del corpus de documentos. Distintos términos índices tienen distinta relevancia en una colección de documentos, dicha relevancia se consigue asociando un peso numérico a cada término índice del documento. Normalmente, suelen ser nombres o grupos de nombres. Existe un tipo de bolsa de palabras basada exclusivamente en este tipo de palabras, con lo que se consigue reducir, considerablemente, el dominio de los elementos, entre los autores que lo usan, encontramos a Rajman ([Rajman and Besançon, 1998]) y Pack Chung ([Wong et al., 2000]) que habla de **conceptos**.

   Ejemplo: *En una colección de documentos relativa a tipos de uva, un concepto podría ser 'uva cabernet sauvignon'.*

   Los métodos para extraer estas palabras clave ([Hernández, 2002]) residen en la comparación de patrones de frecuencias de palabras del corpus objeto de estudio con un corpus de referencia y la utilización de métodos de relevancia estadística como el test χ^2 y *Log likelihood*. De este modo, una palabra clave elegida será la que tenga una frecuencia significativa al comparar ambos corpus.

   Gracias a estas palabras clave, también se pueden detectar grupos semánticos o categorías conceptuales dentro de los documentos. Estas super categorías nos pueden servir para determinar estructuras más complejas dentro del corpus, pero este punto lo dejaremos para futuras investigaciones.

   Ejemplo: *del estudio de dos corpus textuales, uno de oncología y otro de referencia, se obtienen una serie de palabras clave con sus frecuencias de*

aparición, por ejemplo: pacientes, revista, artículo, tratamiento, receptor, humano..., podemos englobar algunas de las palabras en ciertos grupos semánticos, como puede ser ***Enfermo****, que englobaría a humano, pacientes, y receptor.*

Como inconveniente a la utilización de esta estructura, indicar que las áreas conceptuales encontradas no tienen relación aparente entre sí.

Hasta aquí la definición de bolsa de palabras y los tipos de palabras que puede contener, pero veamos cómo se puede representar una bolsa de palabras:

Aunque una bolsa de palabras puede ser, simplemente, todas las palabras de un documento, intentaremos incluir algún tipo de información que permita discriminar las palabras, esta información será un peso asociado a cada uno de los términos.

Sea $\Omega = \{D_1, ..., Dn\}$ una colección de documentos donde cada D_i es un documento y sea $V_D = \{w_1, ..., w_j\}$ el conjunto de palabras extraídas del documento, la versión estructurada del mismo mediante bolsa de palabras la notaremos como B_i y estará formada por pares $(w_j, D_i(w_j))$ donde $w_j \epsilon V_D$ en el documento D y $D_i(w_j)$ es el peso o frecuencia de la palabra j en el documento i. Dependiendo de cómo sea este número, la bolsa de palabras podrá ser binaria, real o difusa. Veámoslo a continuación ([Martín-Bautista, 2000]):

- **Representación binaria**: Dícese de aquella bolsa de palabras en la que la ausencia o presencia de un término en el documento se identifica mediante el valor 0 ó 1, es decir, para el par $(w_j, D_i(w_j))$. La presencia del término $D_i(w_j)$ equivale a 0 si la palabra w_j no está en el documento D_i, mientras que valdrá 1 si la palabra w_j sí se encuentra en dicho documento D_i.

- **Representación real**: una bolsa de palabras tendrá una representación mediante números reales cuando la presencia o ausencia de un término en el documento venga dada por un valor real.

  - Frecuencia Relativa: el valor derecho del par $(w_j, D_i(w_j))$ será la frecuencia relativa, por lo que el par podemos representarlo como $f_D(w_j)$ donde f_D es la frecuencia relativa de la palabra w_j en el documento D_i.
    Esta frecuencia relativa no es más que el número de veces que aparece un determinado término en un documento, calculado sobre el número de términos de dicho documento, se calculará como:

$$D_i(t_j) = \frac{f_{ij}}{\sum_{i=1}^{m} f_{ij}} \tag{2.1}$$

Este esquema asigna valores altos a los términos que ocurren de forma frecuente en el documento.

- Frecuencia Invertida del Documento (*Inverse Document Frequency* ó IDF): el valor de la frecuencia será el producto entre la frecuencia del término y la frecuencia invertida del documento (IDF):

$$D_i(t_j) = w_{ij}^* = f_{ij} * IDF_j^* \tag{2.2}$$

donde IDF se puede calcular como:

$$IDF_j = log_2(|\Omega| - |D_j|) - log_2(|D_j|) \tag{2.3}$$

La principal diferencia, y ventaja, sobre el esquema de frecuencia relativa es que, aunque este esquema también asigna valores altos a los términos que ocurren de forma frecuente en el documento, si esos términos también aparecen frecuentemente en la colección, el peso disminuirá.

- **Representación Difusa**: en este tipo de representación se tiene en cuenta la imprecisión y la subjetividad de la información, características fundamentales del texto con las que no se trabaja en los otros modelos de bolsas de palabras.

Sea $\Omega = \{D_1, ..., Dn\}$ una colección de documentos donde cada D_i es un documento y sea $V_D = \{w_1, ..., w_j\}$ el conjunto de palabras extraídas del documento, la versión estructurada del mismo mediante bolsa de palabras difusa la notaremos como D_i y estará formada por pares $(w_j, D_i(\mu w_j))$ donde $w_j \epsilon V_D$ en el documento D y $D_i(\mu w_j)$ es el peso difuso de la palabra en el documento, que indica el grado de pertenencia del término al documento.

Formalmente, la función de pertenencia difusa que define esta relación entre los términos y los documentos es ([Kraft and Buell, 1993]):

$$F_D : DxT \rightarrow [0, 1] \tag{2.4}$$

Entre los inconvenientes de esta representación nos encontramos que es un conjunto demasiado amplio y con demasiadas palabras innecesarias, así mismo, como se pierden las relaciones entre las palabras del documento, se limita bastante

Descripción del Libro
Título: Harry Potter y el Cáliz de Fuego (Libro 4)
Autor: Joanna K. Rowling
Comments: Este libro fue el mejor libro que nunca he leído
Tema: Ficción, Misterio, Magia, Niños, Escuela, Ficción Juvenil, Fantasía, Hechiceros
Representación
Autor = { " joanna "," rowling " }
Título = { " harry potter "," cáliz "," fuego "," libro " }
Comentarios = { " book "," book "," leer " }
Tema= { " ficción "," misterio "," magia "," niños "," escuela "," ficción "," juvenil ", " fantasía "," hechiceros" }

Tabla 6: Bolsa de Palabras ([Nahm and Mooney, 2001])

la calidad el conocimiento descubierto; pero por otro lado, se pueden otorgar pesos a las palabras para identificar la relevancia de la misma dentro del documento. En la Tabla 6 se muestra una representación mediante bolsa de palabras ([Nahm and Mooney, 2001]).

2.3.1.2 Concepto

Un concepto es un *elemento del pensamiento, expresado en general por un término, por un símbolo o por otros medios* ([Hernández, 2002]).

Desde este punto de vista, un concepto será una representación mental de un objeto basándose en la selección de aquellas propiedades relevantes que definan una clase de objetos que serán las que diferencien una clase de objetos de otra ([Hernández, 2002]).

En la figura 7, vemos cuáles son las dimensiones que forman la unidad conceptual, donde según la norma ISO 704 de 1987 ([ISO704, 2000]), los conceptos son *representaciones mentales de objetos dentro de un contexto específico*, así que un concepto en sí mismo tendrá una forma de representación, estará compuesto por una serie de objetos de la realidad y, además, poseen un aspecto discursivo que les proporciona a dichos conceptos la cualidad de representar dichos objetos de la realidad ([Hernández, 2002]).

Uno de los inconvenientes de los conceptos a la hora de tratar de representarlos, es su dependencia del contexto, al contrario de lo que les ocurre a las palabras, un concepto debe ser modelado en un ámbito y cualquier representación que se le

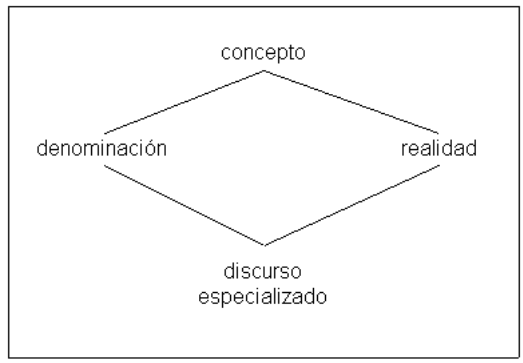

Figura 7: Representación de un concepto ([Hernández, 2002])

proporcione en dicho contexto no presenta garantías de servir para otro contexto diferente.

Presentamos varios tipos de estructuras basadas en las relaciones de jerarquía que se presentan entre los conceptos de un documento ([Hernández, 2002]).

2.3.1.2.1 Jerarquía de Conceptos o Taxonomía de Términos *Una taxonomía es una clasificación ordenada respecto a una serie de relaciones naturales* ([Wason, 2006]).

Entendemos por taxonomía, aquella clasificación compuesta por uno o más niveles, siendo la forma más simple aquella que consta sólo de un nivel, como ejemplo tendríamos un vocabulario en el que cada nivel presentaría un término o concepto. Cada una de estas fases suele identificarse por un término y por un identificador alfanumérico ([Wason, 2006]).

La forma más habitual de representación de las taxonomías es mediante una jerarquía de términos en la que se aprecian en los niveles superiores aquellos conceptos más generales y en los niveles secundarios, especificaciones de los mismos formando así una gradación de conceptos.

Este tipo de construcción también puede representarse mediante un árbol, en el que los términos sólo tienen un padre o mediante un **Grafo Dirigido Acíclico** donde un nodo tiene varios ancestros y permite determinar asociaciones a diferentes niveles de granularidad. Este último tipo de representación permite representar jerarquías solapadas u ortogonales ([Klösgen, 1996]).

Mediante este tipo de representaciones y dado un concepto de la colección, con-

seguimos limitar el número de opciones o de conceptos hijo y construir diversos 'caminos taxonómicos' a partir de él. De hecho, la taxonomía más simple es un vocabulario controlado ([Wason, 2006])

Entre los autores que las emplean: [Wason, 2006], Feldman en varios de sus artículos [Feldman et al., 1998b], [Feldman et al., 1998a], [Feldman et al., 2002], y otros como [Tseng et al., 2007], [Wu et al., 2004], [Kostoff et al., 2001].

Ejemplo de Jerarquía de Términos:

1. *Difuso*
   (a) *Sistema Difuso*
      i. *Sistema Neuro Difuso*
      ii. *Sistema Difuso Neuro Evolutivo*
      iii. *Implementación de Sistema Difuso*
   (b) *Conjunto Difuso*
   (c) *Control Difuso*
      i. *Circuitos Integrados para el Control Difuso*

Dicha taxonomía, la podemos representar gráficamente mediante el árbol que vemos en la figura 8.

Una **taxonomía de atributos** se puede ver como una agrupación de valores de atributos que deberían generarse o deberían explotarse mediante tareas KDT. Los items pueden ser los valores de un único atributo o bien ser los atributos en sí mismos. Estos atributos capturados en una base de datos se colocan en una taxonomía, se pueden colocar en la misma clase aquellos atributos altamente correlacionados. La forma de agrupar los atributos puede ser mediante una técnica estadística como la correlación o mediante aspectos de dominio.

Sea D el dominio de un atributo A, donde A pertenece a un esquema de base de datos o bien puede ser un atributo general que define un conjunto de atributos binarios. Una taxonomía T es un conjunto de subconjuntos del dominio D, normalmente disjuntos y usualmente ordenados por inclusión [Klösgen, 1998]. Formalmente, definiremos taxonomía T, como el conjunto de categorías (Ct), conceptos (C) e identificadores de conceptos (I), del siguiente modo: T= (C,Ct,I).

Dentro de la taxonomía, puede aparecer un conjunto de conceptos 'especiales' llamados 'tipos', que consisten en una abstracción o generalización de un concepto

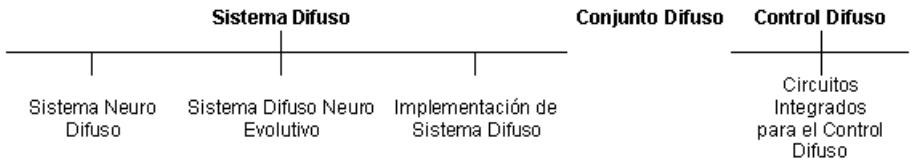

Figura 8: Taxonomía de Términos

de forma que representen una idealización del mismo, pero no siempre tiene que aparecer este 'tipo' dentro de la taxonomía, sobre todo si se está construyendo de forma automática sobre la colección.

Hay características que sí deben presentar los elementos de una taxonomía como son ([Rodríguez, 1999]):

- Que se correspondan con elementos o conceptos clave de la colección, de tal forma que, los elementos seleccionados se correspondan con los términos más importantes del texto, porque, por ejemplo hayan sido seleccionados por el usuario como tales.

  De hecho, el inconveniente principal de este tipo de representación es que no se trata de un esquema portable. Habrá que construir una taxonomía diferente por cada documento de la colección. Pero se intentará que las clasificaciones sí sean constantes para el período de tiempo y los documentos implicados en cada estudio que se realice.

- Que permitan clasificar de forma genérica para que la limitación de portabilidad de la que hemos hablado en el punto anterior se limite al máximo.

- Que no se incluyan 'tipos' ideales de forma abusiva ya que ésto añade complejidad a la taxonomía.

- No abusar de categorías diferentes indiscriminadamente, todo lo que se pueda agrupar y colgar del mismo concepto padre debe asociarse para evitar el crecimiento exponencial de elementos generales en la taxonomía.

- Debe existir jerarquía entre los conceptos, es decir, es fundamental que exista, al menos un nivel por cada concepto general, ésto implicará que exista agrupación de conceptos y, por tanto, posibilidad de intercambio de conceptos en futuros análisis mediante esa taxonomía.

2.3.1.2.2 Grafo semántico , Grafo de Relaciones Semánticas o Red Semántica

Un grafo o red semántica (*Semantic Relationship Graph* ó SRG) es un esquema de representación de conocimiento que se basa principalmente en el uso de nodos y de relaciones al mismo tiempo que utilizan la inferencia como procedimiento para operar sobre la información representada en los nodos. Mediante estos nodos se pueden representar conceptos, estructuras o marcos.

Con esta definición tan somera, nos encontramos que existen varios tipos de redes que se pueden agrupar dentro de las redes semánticas, como son las **redes IS-A**, los **Grafos Conceptuales** de los que nos encargaremos en la sección 2.3.1.2.4 y **Redes de Marcos** ([Moreno Ortiz, 2000]).

La principal diferencia entre los distintos tipos de redes semánticas radica en las características representadas por los enlaces entre los nodos, así, mientras que en las redes IS-A, los enlaces no están etiquetados, en las redes de marcos, los enlaces forman parte de la etiqueta que se le proporciona al nodo. Otra diferencia es los tipos de nodos que existen, ya que en los grafos conceptuales las relaciones también son un tipo de nodo, no así en las otras dos categorías de redes semánticas.

Una característica representativa de las redes semánticas es el uso de la herencia, donde las características de los nodos padre son heredadas por los nodos hijos.

La representación mediante red semántica permite representar relaciones entre conceptos. Intenta proporcionar una descripción para el significado de las palabras y las condiciones bajo las que los significados interactúan para ser compatibles con otros aspectos del lenguaje.

Este tipo de representación es muy compleja pero contiene gran carga semántica (ver Figura 9). Lo utilizan Dubois ([Dubois and Quafafou, 2002]) y Gelfand ([Gelfand et al., 1998]).

2.3.1.2.3 Red IS-A

Una red IS-A es un modelo de representación en forma de taxonomía de conceptos con relaciones de herencia entre sí. Se trata de la red semántica por excelencia y en algunas ocasiones no se distingue como una categoría o especialización de red semántica sino como una red semántica en sí misma.

Como vimos en su definición, la red semántica consta de dos elementos: los nodos y los enlaces entre nodos los cuales se nombran con las etiquetas 'IS-A', 'SUPERC', 'AKO' o 'SUBSET' ([Moreno Ortiz, 2000]).

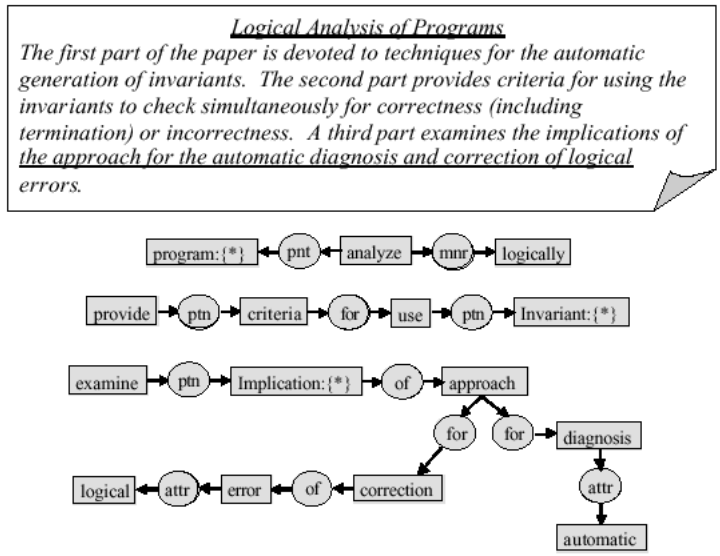

Figura 9: Grafo Conceptual [Montes-y Gómez et al., 2002]

Ejemplo: *un perro* ***es un*** *cánido, un cánido* ***es un*** *mamífero, un mamífero* ***es un*** *animal* ([Brachman et al., 1993]).

Mediante las redes semánticas IS-A, se pueden generalizar los conceptos, es decir, partiendo del concepto 'perro' hemos llegado a la generalización 'animal'. Asimismo, hemos llegado a la generalización de varios individuos partiendo de la especialización de uno sólo.

Por lo tanto, y en función de lo que representen, encontramos dos tipos de nodos en las redes IS-A:

- Token: es el nodo que se encuentra en los niveles inferiores de la red y representa individuos concretos. Heredan las características de los nodos de capas superiores.
- Tipo (*type*): es el nodo de los niveles superiores y representa clases de individuos.

Los enlaces entre los nodos se ven afectados según los tipos de nodos que estén

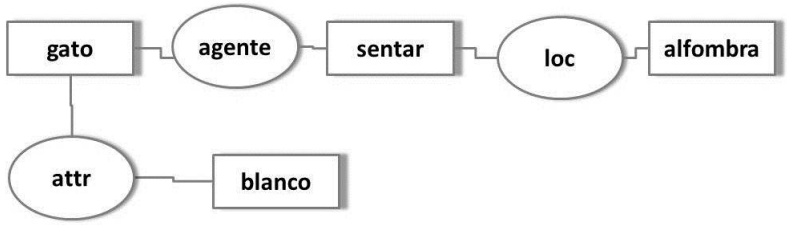

Figura 10: Representación básica de un grafo conceptual

relacionando, así, el enlace que conecte tipos de nodos no será igual al que conecte un nodo con su tipo, tampoco lo será la etiqueta del mismo y podemos encontrar todas estas: Subconjunto/superconjunto, Generalización/especificación, AKO (*a kind of*), Contenido conceptual, Restricción de valores, Tipo característico del conjunto, Pertenencia al conjunto, Predicación, Contenido conceptual y Abstracción ([Moreno Ortiz, 2000]).

Las cuatro últimas se utilizan para unir un nodo con su tipo, mientras que las primeras unen categorías entre sí, sean o no genéricas.

2.3.1.2.4 Grafo Conceptual Un grafo conceptual es un grafo finito, dirigido y bipartito, donde cada nodo o es un nodo concepto o un nodo relación ([Mishne, 2003]), donde cada relación se une a un número de conceptos y viceversa y que se utiliza para la representación de conocimiento.

Según Sowa ([Sowa and Way, 1986]) un grafo conceptual sirve para *expresar significado de forma lógicamente precisa, legible por un humano y extensible computacionalmente.*

Veamos la representación de una sentencia mediante un grafo conceptual ([G.A.Ringland, 1988]) en la Figura 10:

Un gato blanco está sentado en una alfombra

Los elementos básicos de un grafo conceptual son el tipo de concepto, el concepto en sí, y las relaciones conceptuales:

- Un **tipo de concepto** representa clases de entidades, atributos, estados y

eventos. Por ejemplo, *'gato'*.

- El **concepto**. Llamamos así a cada uno de los nodos que representan entidades, atributos, estados y eventos. Un concepto es una instancia de un tipo de concepto. Se trata de la forma más simple de grafo conceptual y su significado será: *'existe un gato'*.

  Si se quiere referenciar a un individuo específico, habrá que particularizar el concepto.

  Los valores asociados a cada nodo, independientemente de la clase a la que pertenezcan son: un tipo y un referente, éste último puede ser individual o genérico.

  Existirá un soporte (también llamado *canon*) relacionado con el grafo conceptual, que será una base de conocimiento que proporcionará la información de dominio para el contexto en el que se ha construido el grafo.

  Este soporte contendrá ([Mishne, 2003]):

  - Un conjunto de tipos de conceptos, una red jerárquica simple, la relación entre dos tipos de conceptos vecinos A y B permitirá la herencia múltiple y se leerá como *B es una clase de A*.
  - Un conjunto de tipos de relación.
  - Una indicación de qué clase de tipos de conceptos se pueden conectar. Esta información se proporciona mediante un conjunto de grafos estrella para cada tipo de relación.
  - Una colección de conjuntos referencia para cada tipo de concepto. Cada conjunto referente debe incluir al menos, el referente genérico etiquetado como " $*$ ". Es decir, el soporte se representa mediante una 4-tupla $S = \langle T_c, T_r, B, R\rangle$ donde:
    1. T_c es el conjunto de tipos de concepto formado por una red finita de orden $\leq$. Siendo 1 el máximo y 0 el mínimo y con $\bigwedge$ y $\bigvee$ representando los límites.
    2. T_r es el conjunto finito de tipos de relación, siendo T_r y T_c disjuntos.
    3. B es el conjunto de grafos estrella, B_{r_i},$r_i \in T_r$. Cada elemento r_i de T_r etiqueta a un vértice de B_{r_i}, el cual tendrá un conjunto no vacío y ordenado de vecinos, dichos vecinos están etiquetados elementos de T_c en pares no ordenados.
    4. R es un conjunto de agrupaciones contables de referentes individuales, donde cada conjunto R_t está asociado con un tipo de concepto $t \in T_c$. A cada conjunto $R_t \bigcup *, 0$ se le asocia una red con un orden representado como $<_t$ de tal forma que los elementos de R_t son incomparables y $\forall r_t \epsilon R_t : 0 < r_t < *$, siendo 0 el absurdo y " $*$ " el genérico.

Un concepto representará a una entidad, pero una misma entidad puede ser representada por varios conceptos o nodos del grafo ([Chein and Mugnier, 2004]) llamado 'co-referentes'. Para facilitar la construcción del grafo conceptual, y su entendimiento, se puede conseguir mezclar estos nodos que representan la misma entidad en uno sólo sin que se pierda el significado original del grafo, es decir, una entidad representada por un nodo tipo t_1 y por otro nodo tipo t_2, necesitamos un tipo más general que sea la conjunción de los tipos t_1 y t_2.

- Las **Relaciones conceptuales**, muestran los roles que los conceptos toman cuando se relacionan unos con otros, es decir, enlazan dos conceptos. Estas relaciones conceptuales se indican en el lenguaje con el orden de las palabras o las preposiciones, sin embargo, en grafos conceptuales habrá que definir una serie de ellas:

  Ejemplo:

  ATTR: BLANCO es un atributo de GATO

  AGNT: GATO es un agente de SENTAR

  LOC: Un EVENTO ocurre en un lugar

Por lo tanto, definimos grafo conceptual como *un grafo finito bipartito representado por una 5-tupla* $g=\langle N_C, N_R, E, ord, label\rangle$, *tal que ([Chein and Mugnier, 2004])*:

1. N_C *y* N_R *son nodos concepto y relación, respectivamente, con* $N_C \bigcap N_R = \emptyset$ *y* $N_C \neq \emptyset$.
2. *E es el conjunto de bordes del grafo. Los bordes adyacentes al nodo relación r estarán totalmente ordenados por 'ord'.*
3. *'label' representa el mapeo de cada nodo concepto del grafo con una etiqueta,* $label(c) = (t, r), t\epsilon T_C, r\epsilon R$.

Entre los autores que trabajan con este tipo de Forma Intermedia, destacamos a: ([Montes-y Gómez et al., 2002], [Tan, 1999], [Mishne, 2003] y [Jin and Srihari, 2007]).

El grafo conceptual proporciona una de las notaciones más flexibles y sólidas para el razonamiento deductivo y para la resolución de problemas del mundo real aunque, como inconveniente, encontramos que no es fácil de representar debido a su alto coste computacional tanto en la comparación de grafos como a la hora de encontrar elementos comunes dentro de ellos ([Montes-y Gómez et al., 2002],

[G.A.Ringland, 1988]).

Ejemplos de campos de aplicación de los Grafos Conceptuales :

- Representación del contenido del texto. Para ello, y una vez construido el/los grafo/s, se pueden realizar comparaciones de grafos ([Mishne, 2003]), realizar clustering conceptual sobre los mismos, descubrir asociaciones o detectar desviaciones de conceptos, realizar análisis de tendencias o clasificación de texto ([Montes-y Gómez et al., 2002]).
- Para conseguir modelos de representación genéricos de documentos, como pueden ser documentos escritos en lenguaje de programación ([Mishne, 2003]).

Similaridad entre un Diagrama Entidad Relación y un Grafo Conceptual:

Si analizamos el grafo conceptual visualmente, podemos encontrar varios puntos de similaridad con los diagramas entidad relación de Chen ([Chen, 1981]), sobre todo en que los elementos principales de ambos son los conceptos o entidades y las relaciones entre ellos. Sin embargo, el grafo conceptual es una representación más avanzada que otros tipos de diagramas que existen en la literatura porque permiten ocultar la dificultad de la lógica tradicional ([Polovina, 2008]).

2.3.1.2.5 Grafo Conceptual Difuso Un grafo conceptual difuso G_D = *(C_D,R_D,A_D)* es un grafo dirigido formado por tres clases de componentes: conceptos difusos (C_D), relaciones difusas (R_D) y atributos de relación difusos (A_D) ([Mulhem et al., 2001]).

- Un *concepto difuso* c_D es una 3-tupla *[T,e,f]* donde T es el conjunto de tipos de conceptos t_i, e es un referente y $f(t_i)$ es una función miembro difusa, tal que $0 \leq f(t_i) \leq 1$ para cada t_i, según esta definición, un *concepto crisp* c puede verse como una clase especial de concepto difuso cuyo conjunto T consistirá sólo en un tipo de concepto t, con *f(t) = 1* ([Mulhem et al., 2001]).
- Una *relación difusa* r_D es una 2-tupla *(t,v)* donde t es un tipo de relación y el valor v es un valor difuso que representa la probabilidad de ocurrencia de la relación r en el tipo de objeto que estemos estudiando. Una relación crisp o no difusa, será aquella en la que *v=1*.

- Un *atributo de relación difusa* es una 3-tupla *[t,e,f]* donde t es un atributo de relación crisp, e un referente y *f(e)* una función miembro difusa $0 \leq f(e) \leq$. El equivalente crisp a esta relación es aquella en la que *f(e) = 1*.
- Llamamos *D(t)*, *denotación* del tipo t, al conjunto de todos los posibles referentes individuales de t.
- *Especialización* o *subtipo* de t es *t'*, siempre que $t' \leq t$ y $D(t') \subseteq D(t)$.
- Un grafo modelo representa un modelo para una escena conocida donde se sabe cuáles son los tipos de conceptos.

  Ejemplo: *si tenemos una escena del tipo 'montaña-y-lago', tendremos identificados los conceptos 'montaña', 'cielo', 'agua', 'árbol', ... conocidos a priori.*

2.3.1.2.6 Dependencia Conceptual La dependencia conceptual asume que, si dos sentencias tienen el mismo significado, se deberían representar del mismo modo independientemente de las palabras utilizadas para ello y que toda la información implícita de la sentencia se debería expresar de forma explícita ([Becker, 2005]).

Veamos dos ejemplos de sentencias que, independientemente de su sintaxis, tienen idéntico significado:

Ejemplo:

Luis paseó desde el parque hasta la casa

Luis caminó desde el parque hasta la casa

Ambos verbos se pueden representar de forma canónica gracias a una serie de primitivas semánticas, en concreto, el ejemplo anterior se puede representar mediante la primitiva PTRANS, de transferencia física en el espacio ([Becker, 2005]):

1. ACTOR: un HUMANO inicia la primitiva PTRANS
2. OBJECT: un OBJETO FÍSICO es movido (PTRANSed)
3. FROM: una LOCALIZACION, en la que comienza PTRANS
4. TO: una LOCALIZACION, en la que termina PTRANS

Las primitivas intentan encontrar una forma canónica para los verbos, así, como 'caminar' y 'pasear' pueden implicar diferentes connotaciones como más o menos velocidad, el concepto de movimiento sí se verá reflejado.

Las primitivas enlazan información similar para que las reglas de inferencia no necesiten ser escritas para cada verbo individual. Las principales primitivas son ([Becker, 2005]):

1. ATRANS - Transferencia de una relación abstracta como posesión, propiedad o control
2. PTRANS - Transferencia de localización física de un objeto
3. PROPEL – La aplicación de la fuerza física a un objeto
4. MOVE – El movimiento de una parte de un cuerpo de un animal
5. GRASP – Agarrar un objeto
6. EXPEL – La expulsión del cuerpo de un animal al mundo
7. MTRANS – La transferencia de información mental entre animales o dentro de un animal
8. CONC – La conceptualización o pensamiento de una idea por parte de un animal
9. MBUILD – La construcción por parte de un animal de nueva información a partir de información antigua
10. ATTEND - La acción de dirigir uno de los sentidos a un objeto
11. SPEAK – La acción de producir sonidos con la boca

Aunque en sus comienzos, las dependencias conceptuales eran uno de los inconvenientes de las redes semánticas, se han convertido en un método óptimo de representar conceptos.

Quinllian ([Quillian, 1968]) pretendía que su red semántica expresara cualquier concepto que se pudiera expresar en lenguaje natural, sin embargo, la intuición humana no se puede representar, así, la dependencia conceptual intenta solucionar este problema, procurando que la representación del lenguaje natural saque conclusiones y que dicha representación sea independiente del lenguaje utilizado por el autor del texto.

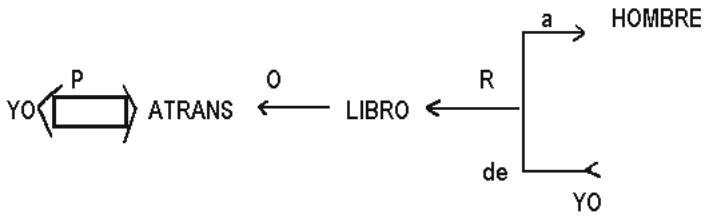

Figura 11: Dependencia conceptual ([Bonillo and Betanzos, 1998])

Veamos un ejemplo de dependencia conceptual en la Figura 11. Aquí, las flechas unidireccionales indican una dependencia y las bidireccionales un enlace. Existen varias relaciones: 'P' que indica pasado, 'O' que expresa una relación causal con un objeto y 'R' que indica una relación causal relativa a *libro*. También aparece una primitiva ATRANS, que refleja la acción e indica transferencia o posesión.

La representación no utiliza primitivas que correspondan a palabras concretas del texto, sino que utiliza primitivas conceptuales que pueden utilizarse en cualquier lenguaje.

La teoría de Schank ([Schank, 1972]) se basa en que existe una base conceptual sobre todas las estructuras lingüísticas, así, la dependencia conceptual extrae los conceptos en los que se basan las palabras y los relaciona.

Para una misma sentencia se podrán realizar numerosas conceptualizaciones a la vez.

Clases de conceptos o categorías conceptuales según Schank ([Becker, 2005]):

1. Nominal: los nombres. Los denomina PP o *picture producer* (productores de dibujos).
2. Acción: los verbos. Los denomina ACT o *base actions* (acciones básicas).
3. Modificador: son los adjetivos, PA o *picture aiders* (ayudantes de dibujo) y los adverbios, AA o *action aiders* (ayudantes de la acción).

Para analizar una sentencia mediante dependencias conceptuales, se necesitará:

1. Un procesador sintáctico, que encuentre los verbos y los nombres en el texto y que establezca las relaciones sintácticas entre ellos.

2. Un procesador conceptual, que enumere las relaciones subyacentes de una sentencia del texto basándose en el conocimiento de background.
3. Interfaz de conceptualización de memoria.
4. Memoria intermedia.
5. Memoria de largos términos.

Encontramos ejemplos de sistemas basados en Dependencias Conceptuales como MEMORY ([Becker, 2005]), se trata de un programa que realiza inferencia sobre el texto, cada vez que se lee una sentencia.

2.3.1.2.7 Ontologías Una ontología representa un marco conceptual que incluye la descripción de los conceptos y las relaciones entre ellos ([Mizoguchi et al., 2006]).

Entre los autores que hablan del uso de **ontologías** como herramienta de estructuración intermedia a la hora de realizar Minería de Textos, encontramos a [Maedche and Staab, 2000a], [Sánchez et al., 2008], [Paralic and Bednar, 2003], [Smrž et al., 2007], [Spasic et al., 2005], [Milne and Witten, 2013], [Medelyan et al., 2009], [Buitelaar et al., 2005], [Kim et al., 2008], [Paralic and Bednar, 2003], [Kontopoulos et al., 2013]. Una ontología se puede utilizar para definir un vocabulario básico de términos así como para especificar hasta un cierto grado, su significado. Esta estructura incluye o puede incluir definiciones de conceptos y cómo están interrelacionados. Estos conceptos imponen colectivamente una estructura en un dominio de discurso y limitan las posibles interpretaciones de los términos [Smrž et al., 2007].

Esta Forma Intermedia será ampliamente estudiada en el capítulo 4.

2.3.1.2.8 AP-Sets El problema de la representación y manejo de los campos textuales dentro de las bases de datos como si fueran cualquier otro campo lleva a los autores ([Martín-Bautista et al., 2008]) a proporcionar una estructura intermedia llama AP-Set, que se obtiene através del algoritmo Apriori ([Agrawal et al., 1994]).

Los atributos textuales de una base de datos se procesan para obtener una representación semántica estructurada del siguiente modo ([Martín-Bautista et al., 2008]):

Sea R una relación con los atributos $\{ A_1, A_2, A_T, ..., A_n \}$, donde A_T es un atributo de texto sin una estructura predecible. Para obtener su ADT (*Abstract*

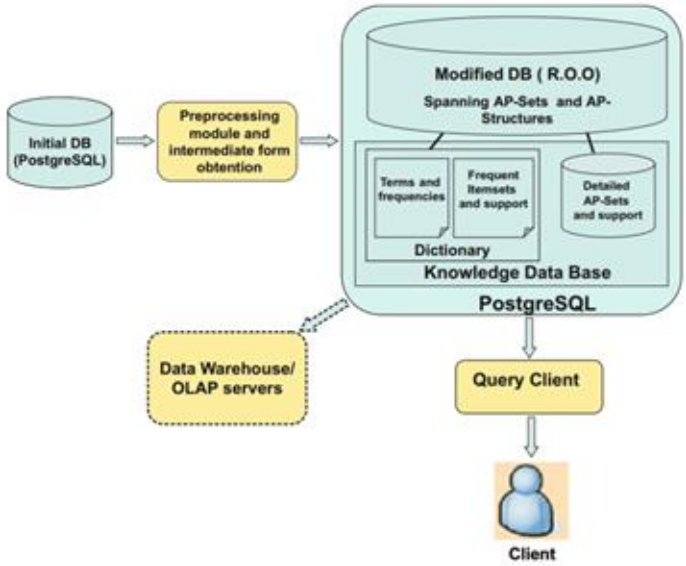

Figura 12: Sistema de Extracción de AP-Sets ([Martín-Bautista et al., 2008])

Data Type) habrá que:

1. Obtener el diccionario de datos que consiste en una lista de términos sin los *stop-words*.
2. Transformar los datos una base de datos transaccional en la que los atributos serán los diferentes términos del diccionario de datos y en el que cada tupla se corresponderá con un registro.
3. Obtener los itemsets frecuentes de la base de datos transaccional según el algoritmo Apriori.
4. Los itemset maximal formarán la estructura reticular, AP-Sets, siguiendo la propiedad Apriori.

En la figura 12 vemos un ejemplo de la arquitectura de un sistema que convierte los campos textuales a AP-Sets.

2.3.1.3 Frase

Llamaremos frase a la secuencia de palabras, con cierto nexo sintáctico, que aparecen en un texto. Lo representamos como $F_i =< (w_1)(w_2)...(w_n) >$.

Desde el punto de vista computacional, existen distintos tipos de frases que se pueden obtener del documento.

2.3.1.3.1 1-Frase Se denomina *1-frase* a aquella sucesión de elementos donde cada uno de ellos es una frase propiamente dicha.

2.3.1.3.2 n-Frases Generalizando la definición anterior, una n-frase es aquella sucesión de elementos donde cada uno de ellos es una frase de hasta n niveles.

Veamos el ejemplo propuesto por Lent ([Lent et al., 1997]) para diferenciar estas definiciones:

Ejemplo: *Una 1-frase puede ser aquella en la que aparezca 'IBM' y 'data-mining' en un párrafo, la notaremos como:* $<< (IBM) >, < (data)(mining) >>$

Una 2-frase será del tipo:

$<< (IBM) >, < (Data)(Mining) >><< (Anderson)(Consulting) >, < (soporte)(decisiones) >>$

donde 'Anderson Consulting' y 'soporte de decisiones' están en un párrafo distinto a 'IBM' y 'data mining'.

También mencionan el uso de esta forma intermedia en uno de sus artículos: [Lent et al., 1997], [Scott and Matwin, 1999], [Delgado et al., 2002a], [Yang et al., 1999], [Nguyen and Kan, 2007], [Turney, 2000],[Lewis, 1992], [Tseng et al., 2007].

2.3.1.3.3 Frases de Texto Multi-términos En este tipo de representación, los elementos serán frases de texto con múltiples términos y los documentos serán las transacciones. El tipo de minería que se puede obtener con esta representación tiene que ver con la coocurrencia de las frases en el documento ([Delgado et al., 2002b]).

[Sahami and Heilman, 2006] introduce una medida de similaridad para determinar la similitud entre dos breves extractos textuales y nos invitan a utilizar **frases multitérminos**, ([Delgado et al., 2002a], [Feldman and Sanger, 2007], [Ahonen-Myka et al., 1999] y [Chen et al., 2005]).

2.3.1.4 Párrafo

Un párrafo, de manera informal, es un conjunto de frases que concluyen en un punto y aparte. Conceptualmente, un párrafo contiene el desarrollo de una idea central o un conjunto de ellas enlazadas entre sí, expresadas mediante una o más frases que presenten coherencia en el discurso.

Representaremos un párrafo como una colección de frases que aparecen de forma secuencial en el texto, es decir, como $P = \{F_1, ..., F_p\}$, siendo p el número de frases que lo componen.

Como formas intermedias asociadas a este tipo de construcción, utilizaremos las *n-frases* definidas en la sección 2.3.1.3.

2.3.1.5 Documento

Un Documento D_i, es cada uno de los textos que forman la colección Ω. Estará formado por palabras, conceptos, frases y párrafos persiguiendo, todos ellos, representar una idea fundamental.

Si se pretende realizar la minería de textos sobre el cuerpo completo de los documentos, nos encontramos con la dificultad de la ausencia de estructura del lenguaje natural en el que está escrito el documento.

Las formas intermedias basadas en documentos, nos permitirán deducir patrones, tendencias y relaciones de interés en un dominio específico. La forma intermedia basada en documentos se puede transformar en una basada en conceptos, extrayendo información relevante de acuerdo a los objetos de interés de un dominio específico.

2.3.1.5.1 Documento Prototípico Un documento prototípico es aquel que se corresponde con la información que ocurre de modo repetitivo en una colección de documentos ([Rajman and Besançon, 1998]). Es una estructura típica de la Extracción de Información donde el objetivo es rellenar una especie de plantilla con unos datos básicos extraídos del texto.

En la Figura 13 tenemos un ejemplo. Con esta representación se extraen las secuencias de palabras que aparecen con frecuencia en el conjunto de documentos.

Para extraer los documentos prototípicos se pueden utilizar técnicas de proce-

<DOC2088>
Nissan_Motor_Co_Ltd "NSAN.T" is **issuing** a 35_billion_yen eurobond **due** March_25 1992 **paying** 5-1/8_percent and **priced** at 103-3/8, Nikko_Securities_Co (Europe) Ltd said.
The non-callable_issue is **available** in **denominations** of one_million Yen and will be **listed** in **Luxembourg**.
The **payment_date** is March_25.
The selling_concession is 1-1/4_percent while **management** and **underwriting combined** pays 5/8_percent.
Nikko said it was still completing the syndicate.

Figura 13: Documento Prototípico ([Rajman and Besançon, 1998])

samiento de Lenguaje Natural basadas en extracción de información como: etiquetado de parte del discurso y la extracción de términos, que servirán como un procedimiento automático que extrae estructuras lingüísticas significativas del contenido completo del texto, según el artículo de Rajman ([Rajman and Besançon, 1998]).

Aunque, básicamente, la extracción de documentos prototípicos consiste en la identificación de secuencias frecuentes de términos en los documentos, es decir, se generan los conjuntos de palabras clave cuyo soporte sea superior a un umbral determinado.

- Etiquetado de parte del discurso (*part of speech tagging*): con este proceso se asignan etiquetas a las palabras del texto. Estas etiquetas pueden ser categorías morfosintácticas como /N para nombre, /ADJ para adjetivo o /VERB para verbo. Hay que tener cuidado con la ambigüedad léxica del lenguaje natural.

  Una de las ventajas del etiquetado es que permite filtrar automáticamente las palabras sin significado en base a las categorías morfosintácticas.

- Extracción de términos.

Como inconveniente hayamos la pérdida de contenido semántico y parte de información que puede ser importante aunque no se repita de forma continuada en la colección.

Entre los autores que lo mencionan, encontramos a Rajman ([Rajman and Besançon, 1998], [Rajman and Besançon, 1997]) y Feiju Xu ([Xu et al., 2002]).

2.3.1.6 Otros Elementos Susceptibles de ser Representados

2.3.1.6.1 N-gramas *Un n-grama es una subcadena de n caracteres obtenida a partir de una palabra dada de mayor tamaño.*

Un n-grama proporciona una representación distribuida de las palabras del documento porque cada una de ellas estará representada por un conjunto de n-gramas ([Cavnar, 1994]).

Dependiendo de *n*, los n-gramas se denominarán bigramas o digramas si n=2, trigramas si n=3, quadgramas si n=4 y así sucesivamente.

Ejemplo: *la palabra TEXTO podemos representarla del siguiente modo: como un un bigrama, _T, TE, EX, XT, T_, como un trigrama, _TE, TEX, EXT, XT_ o como un quadgrama, _TEX, TEXT, EXT_*

Las palabras similares, entendiendo por similar, por ejemplo, aquellas que comparten una misma raíz o tienen un sufijo diferente o diferente deletreo, compartirán un amplio número de n-gramas.

Por ejemplo, veamos los bigramas que comparten las palabras *RETRIEVE, RETRIEVAL y RETRIEVING*, que tienen diferente sufijo:

RETRIEVE: _***R***,***RE***,***ET***,***TR***,***RI***,***IE***,***EV***,*E_*

RETRIEVAL: _***R***,***RE***,***ET***,***TR***,***RI***,***IE***,***EV***,*VA,AL,L_*

RETRIEVING: _***R***,***RE***,***ET***,***TR***,***RI***,***IE***,***EV***,*VI,IN,NG,G_*

Ahora, veamos que también comparten varios n-gramas *RETRIEVE* y *RETREIVE* (con un deletreo equivocado):

RETRIEVE: _***R***,***RE***,***ET***,***TR***,*RI,IE,EV,E_*

RETREIVE: _***R***,***RE***,***ET***,***TR***,*RE,EI,IV,VE,E_*

Estas secuencias de caracteres se pueden representar mediante un grafo dirigido acíclico y probabilístico en el que cada uno de los caminos del grafo consta de *n*

nodos, de los cuales, los nodos del 1 al *n-1* se corresponden con eventos pasados. El nodo *n-1* tiene adyacencias con los nodos que representan todos los eventos que pueden posibilitar la secuencia $1..n-1$ y el enlace del nodo *n-1* al nodo n en cada uno de los caminos se etiqueta como la probabilidad de que n siga $1..n-1$ ([Khomra, 2005]).

Los trigramas se suelen utilizar en reconocimiento del habla. Los n-gramas, en general, son utilizados para encontrar las frecuencias de las palabras en ciertos idiomas, como el inglés ([Khomra, 2005]).

La ventaja de la representación mediante n-gramas es que no se requiere preprocesamiento lingüístico de los documentos como es el *stemming* o la eliminación de las *stop words*, además de que es independiente del lenguaje utilizado ([Cavnar, 1994]).

Algunos ejemplos de *n-gramas* encontrados en la literatura son:
[Cavnar, 1994], [Ari Pirkola and Järvelin, 2003], [Yang et al., 1999], [Kaski et al., 1998], [Kargupta et al., 1997], [Kešelj et al., 2003], [Peng et al., 2003], [Ananiadou and Mcnaught, 2005], [Pak and Paroubek, 2010], [Furnkranz, 1998], [Khomra, 2005].

Las búsquedas dentro de los n-gramas pueden ser exactas o mediante emparejamiento difuso:

- **Emparejamiento exacto**: en este tipo de n-gramas, el texto se descompone en cadenas de longitud n.
- **Emparejamiento difuso (*fuzzy matching*)**: mediante una mínima desviación del término en el documento, encontramos otro término más adecuado, por ejemplo, aunque no se conozca exactamente cómo se deletrea un nombre propio, se puede encontrar dicho nombre aplicando técnicas de emparejamiento difuso. Esta técnica tiene que ver con la idea de que palabras con ortografía similar tengan una alta probabilidad de estar relacionadas con el mismo concepto semántico.

  El grado de similaridad entre dos palabras cualesquiera w_i y w_j se mide en función del número de n-gramas que tienen ambas en común y el número de n-gramas únicos para cada una de ellas. ([Ari Pirkola and Järvelin, 2003])

$$Sim(w_i, w_j) = \frac{|N_i \bigcap N_j|}{|N_i \bigcup N_j|} \tag{2.5}$$

  donde N_i y N_j se refieren al conjunto de n-gramas que se derivan de la palabra

w_i y w_j, respectivamente.

Ejemplos de representaciones basadas en n-gramas:

- Sistema para emparejamiento de ciudades, estados y códigos ZIP *U.S. Postal Service* ([Cavnar, 1994]), que permite pasar por alto los errores en las direcciones postales y los fallos de los sistemas OCR ya que los sistemas basados en n-gramas permiten eliminar la información redundante.
- Herramienta de recuperación de texto *zview* ([Cavnar, 1994]) que proporciona un acceso sencillo a un conjunto de documentos ASCII sin necesidad de utilizar expresiones de búsqueda booleanas, gracias al deletreo y a las variaciones de las palabras en las consultas.

2.3.1.6.2 Consultas y Url Una estructura ampliamente utilizada tanto por las consultas en buscadores como en sistemas de bases de datos es la consulta. Existen una serie de relaciones semánticas que son capturadas implícitamente cada vez que un usuario realiza una consulta. Dichas consultas se pueden representar mediante gráficos semánticos ([Baeza-Yates and Tiberi, 2007], [Silvestri, 2010]) a partir de un grafo bipartito obtenido de hacer click para obtener los resultados e inferir relaciones semánticas interesantes entre dichas consultas.

Entre los trabajos a destacar, ([Lu et al., 2004]) extrae traducciones de consultas Web a través de la minería de anclajes de texto y enlaces.

2.3.1.6.3 Tendencia (*trend*) El análisis de tendencias consiste en la comparación de distribuciones de conceptos de datos nuevos con distribuciones de datos antiguos. Una tendencia es una n-frase seleccionada por una consulta modelo con información añadida. Dicha información consiste en los períodos de tiempo en las que aparece la tendencia, es decir, asociada a cada frase del documento existe una historia de la frecuencia de ocurrencia de la frase, esta frecuencia se obtiene particionando los documentos en función de un tiempo asociado que indica el período de tiempo en el que se incluyó en la colección. Por lo tanto, la frecuencia de ocurrencia en un momento determinado será el número de documentos que contienen dicha frase ([Lent et al., 1997]).

Volvemos a redefinir el concepto de frases para adaptarlo a las tendencias.

- Sea P un conjunto de frases: $P = \{F_1, F_2...F_n\}$ donde $F_i = (w_1)(w_2)...(w_k) >$

y $w_i \epsilon W$ siendo (w_i) una palabra y una frase es $< (w_1)(w_2)...(w_k) >$, es decir, una frase es una secuencia de palabras

- Sea F un campo de texto, $F = \{f_1, f_2, ...f_n\}$, donde $f_i \subseteq W \bigcup P$
- Sea $\Omega = \{D_1, D_2, ...D_n\}$ un conjunto de documentos, y sea $D_i = \{DS_i, f_1, f_2, ...f_n\}$ la representación de un documento, donde DS_i es el espacio de tiempo de D_i y $f_i \epsilon F$. Este espacio de tiempo del documento es el momento en el que el documento es registrado en la base de datos

Existe un lenguaje llamado SDL (*Shape Definition Languaje*) que sirve para definir tendencias específicas con o sin soporte. Así, las tendencias son aquellas n-frases recuperadas de una *shape query* con restricciones como el período de tiempo durante el cual se soporta la tendencia ([George Chang and Wang, 2001]).

Encontramos referencias de este tipo de representación en: [Delgado et al., 2002a], [Lent et al., 1997].

2.3.1.7 Otras Representaciones *Adecuadas* para Estructurar un Texto

En esta sección destacamos representaciones que, aunque no se suelen encontrar directamente en el texto objeto de análisis como puede ser el caso de la palabra, el concepto, la frase, el párrafo o el propio documento, como vimos en la sección 2.3.1, y si bien es cierto que existe cierta controversia sobre si son realmente representaciones intermedias propiamente dichas, en algunos casos pueden encontrarse en la literatura realizando dicha función.

2.3.1.7.1 Eventos Los millones de eventos que ocurren constantemente en el mundo, se registran en ciertas agencias como CNN y Reuters. Estas historias se archivan en agencias de noticias en orden cronológico como texto plano o en formato estructurado como SGML. Como incrementa el volumen de información disponible, habrá que poner atención en estos eventos ([George Chang and Wang, 2001]).

Un *evento Ev* es un conjunto significativo de términos que ocurre en un lugar y en un tiempo determinado. Ambas características, de tiempo y lugar, son las que convierten la definición de evento en algo especial porque un evento puede ocurrir sólo en un momento determinado mientras que otro puede ser cierto en muchos sitios y en muchos momentos.

Es una estructura propia de la extracción de información, obtenida tras la aplicación de técnicas de preprocesamiento de lenguaje natural y con la que se consigue, por ejemplo en el artículo de Karanikas ([H. Karanikas and Theodoulidis, 2000]), etiquetar documentos de distintas compañías gracias a eventos seleccionados de los mismos. Aunque también puede utilizarse en la recuperación de información si tenemos en cuenta que una consulta es un objeto de interés para la caracterización basada en contenidos pero no sirve para responder cuestiones del tipo: "*¿Qué es nuevo?*", a ésto se le llama Detección de Tópicos y Persecución de un objetivo (TDT, *Topic Detection y Tracking*) con métodos como la clasificación basada en el vecino más cercano, el filtrado basado en relevancias y las evaluaciones de punto de referencia (*benchmark*) ([George Chang and Wang, 2001]).

Ejemplo: *'La firma de la constitución europea tuvo lugar el 29 de octubre de 2004 en Roma' es un evento que tuvo lugar en un momento y en un lugar determinados, mientras que 'La familia de Picasso' es un evento que se repite a lo largo de la historia y en diversas ciudades.*

La detección de eventos es un proceso con el que se intenta identificar historias en continuamente que se corresponden a nuevos eventos o a eventos no identificados previamente.

Delgado menciona la detección de eventos en uno de sus artículos ([Delgado et al., 2002a]), mientras que ([Allan et al., 1998]) y ([Yang et al., 1998]) presentan aplicaciones con este tipo de representación.

Una forma de describir eventos puede ser mediante un conjunto de atributos que se almacenarán en una tabla, en el artículo de Karanikas ([H. Karanikas and Theodoulidis, 2000]), un evento tipo como *Take-over* se definirá como se recoge en la tabla 7:

Tipo de Evento	***Take-over***
Fecha	March 15
Compañía Adquisidora	Neurosoft SA
Compañía Destino	IBM
Tipo de *take-over*	Amistoso
Valor	240,000,000 libras
ID del Documento	Doc1

Tabla 7: Tabla que representa un evento ([H. Karanikas and Theodoulidis, 2000])

Sobre cómo se realiza la detección de eventos, encontramos dos tareas fundamentales: la detección retrospectiva y la detección *on-line* ([George Chang and Wang, 2001]).

1. Detección retrospectiva: es el proceso de descubrir eventos, sin identificar, en una colección de noticias.

   Los algoritmos usados para llevar a cabo esta tarea se basan, principalmente, en dos técnicas: detección de cambios rápidos en el tiempo y en el uso de similaridad léxica y de la proximidad en el tiempo de las nuevas historias.

2. Detección *on-line*: es el proceso de identificar nuevos eventos en tiempo real según llegan las nuevas noticias.

   Los algoritmos usados para la detección *on-line* se basan en un modelo de umbrales en función de varios parámetros:

   - Un umbral de detección t_d, que especifica la mínima medida de similaridad requerida por el sistema como confianza para que la historia actual pertenezca a un nuevo evento.
   - Un umbral de clustering t_c, que especifica la mínima medida de similaridad requerida por el sistema para que la historia actual se añada como nuevo miembro de un cluster existente.
   - Un tamaño de ventana t_w, que especifica el máximo número de clusters disponible para comparar con la actual historia.

En ninguno de los procesos se conoce la naturaleza de los eventos estudiados, sólo la fecha en la que se introdujeron en la base de datos.

Para identificar los eventos, se tiene en cuenta un hecho no trivial que tiene lugar en un sitio o en un período de tiempo determinado, por lo que se seleccionarán los eventos, manualmente, y se etiquetan, posteriormente, los documentos con alguno de los eventos seleccionados. Si se analizan nuevos hechos basándose en los eventos existentes, y estos eventos tienden a ser próximos en el tiempo, se sugiere una similaridad léxica y una proximidad temporal que servirá para el algoritmo de clustering, sin embargo, un lapso de tiempo entre hechos similares suele indicar eventos diferentes, como pueden ser crisis políticas, lo que nos sugerirá que se monitorice el clustering para conocer su evolución en el tiempo ([Yang et al., 1998]).

La detección de eventos se aplica a *microblogging* y redes sociales como *Twitter* ([Weng and Lee, 2011]), debido a su características de *text stream*, es decir, secuencias de documentos ordenados cronológicamente en las que puede resultar

interesante obtener *hot bursty events*, es decir el mínimo conjunto de ráfagas de características que ocurren juntas en ventanas de tiempo con un un fuerte apoyo de los documentos de la secuencia de texto ([Fung et al., 2005]).

2.3.1.7.2 Episodios Un episodio de texto es un par que se puede obtener a partir de datos secuenciales, en el caso que nos ocupa, los obtendremos del texto. Se tratan de una modificación del concepto de regla de asociación asociado a datos secuenciales. Por lo tanto, dada una secuencia de texto S, un episodio de texto constará de dos componentes $\alpha = (V, \leq)$ ([Ahonen et al., 1997]), donde

- V es el **vector de características**, es decir, es un conjunto ordenado de características. Entendiendo por características una palabra, su género o número, un signo de puntuación.
- $\leq$ es el **orden** parcial dentro del vector, o la posición de la palabra dentro de la secuencia de texto.

El episodio de texto α ocurrirá en S si se satisface el vector de características respetando el orden.

Entendemos por característica una propiedad de la palabra (la forma base de la palabra, la raíz), una cualidad gramatical (parte del discurso, el número o el género), un signo de puntuación o una estructura etiquetada.

Ejemplo: *supongamos un texto que trate sobre text 'knowledge discovery in databases', y en un vector de características que contenga, la forma base de la palabra, la parte del discurso, y el número de la palabra. Obtendríamos la secuencia de episodios siguiente: (knowledge_N_SG, 1) (discovery_N_SG, 2) (in_PP, 3) (database_N_PL, 4)*

El texto, en este tipo de aproximaciones, se convierte en datos secuenciales ($\alpha = (V, \leq)$) del mismo modo que ocurre cuando intentamos representar datos procedentes de algún sistema, como puede ser un sensor.

Es posible obtener unas reglas 'especiales' sobre episodios, estas reglas se denominan reglas de episodios y son una modificación de las reglas de asociación habituales.

Mientras Delgado menciona estas nuevas reglas de episodio ([Delgado et al., 2002a]), Ahonen utiliza ejemplos prácticos de los mismos ([Ahonen et al., 1997]).

2.3.1.7.3 Marcos (*Frames*) Se trata de la extracción de cierta estructura sustancial del texto. En realidad es la extracción de un prototipo para que represente una situación determinada. Entre los distintos tipos de información que deben incluir, nos encontramos con: información acerca de lo que puede ocurrir después de esa situación, qué opciones se pueden tomar si no se cumplen ciertas condiciones, qué acciones hay que realizar ([Minsky, 1975]).

Hasta ahora se suelen utilizar para describir sucesos de percepción visual, donde los marcos actúan como perceptores de objetos ya conocidos. Por ejemplo, cuando se entra en un restaurante hay objetos que esperamos encontrar como: 'platos', 'mesas', 'camareros' ... y se venían utilizando en ingeniería o para la representación de conocimiento en inteligencia artificial.

Es una estructura muy útil para representar la *semejanza*, para acumular experiencias pasadas sin tener que volver a representarlas ante un nuevo hecho. Son un tipo especial de red semántica donde la construcción de la misma está directamente asociada con el concepto de semejanza y que utiliza el concepto de *herencia*.

Uno de los inconvenientes de esta estructura, junto con los ya mencionados de las redes semánticas, es encontrar el hecho inicial ya que, si vamos a reutilizar los marcos, ante una nueva situación necesitamos partir de algún punto y lo habitual será tomar alguno de los estados iniciales de los que dispongamos, el cual no será exactamente el que necesitamos. Además, se trata de una construcción demasiado ajustada a un hecho en particular por lo que sería necesario encontrar una forma intermedia más general para que nos sirviera para todo tipo de situaciones y ámbitos.

MARCOS	
Papeles:	cliente, cajero, camarero, dueño
Elementos:	mesa, platos, menú, dinero, cubertería, comida, propina, servilletas, cuenta.
Condiciones de entrada:	el cliente tiene dinero, el cliente tiene hambre, el restaurante tiene comida, el restaurante está abierto.
Resultados:	el restaurante tiene más dinero, el cliente tiene menos dinero, el cliente ya no tiene hambre.

Tabla 8: Representación del Marco restaurante ([Leahey and Jackson, 1998])

Cuando la información del marco se organiza secuencialmente de cierto modo temporal más o menos predeterminado, hablamos de **guiones** o *scripts* (ver Tabla 9).

GUIONES		
Escena 1:	**Entrada:**	el cliente entra en el restaurante,
		el cliente decide la mesa,
		el cliente va a la mesa,
		el cliente se sienta,
Escena 2:	**Encargo:**	el cliente recibe el menú
		el cliente mira el menú
		el cliente ordena la comida
Escena 3:	**Comida:**	el camarero lleva la comida
		el cliente se come la comida
Escena 4:	**Salida:**	el camarero lleva la cuenta al cliente
		el cliente paga al cajero
		el cliente deja propina
		el cliente sale del restaurante

Tabla 9: Guión de un Restaurante ([Leahey and Jackson, 1998])

2.3.2 Clasificación de las Formas Intermedias según su Aplicabilidad

Un factor a tener en cuenta a la hora de decantarse por una u otra representación intermedia es que la Forma Intermedia y los procedimientos de inferencia empleados están fuertemente unidos. Por ejemplo, si se desea utilizar técnicas de minería basadas en lógica, será necesaria una representación intermedia del texto sobre la que se pueda realizar dicha inferencia. Si, por el contrario, lo que se pretentende es jugar con la mayor cantidad de información contenida en el documento, habrá que decidir primero qué representación intermedia nos conviene. La transformación del documento en una versión estructurada de sí mismo implica, generalmente, la pérdida de una gran cantidad de información. Si la técnica elegida para proporcionarle esa estructura no tiene en cuenta el significado y simplemente se queda con los términos relevantes y alguna relación entre ellos, estaremos perdiendo información semántica irrecuperable ([Zhong et al., 2012]).

Hay que valorar, no obstante la aplicación de estas técnicas en función del coste computacional y la complejidad del problema. Las formas intermedias más simples están normalmente más cercanas a lo sintáctico y más alejadas de lo semántico, pero también suelen ser más fáciles de obtener y son computacionalmente, menos costosas. Las formas intermedias enriquecidas conservan mejor la semántica del documento pero son más complejas de ser generadas y manipuladas.

Como vimos en la subsección 2.3.1, no es lo mismo tener como objeto básico de minería una simple palabra que un conjunto de ellas con cierto significado, por ello, será importante aplicar técnicas de Preprocesamiento adecuadas, de tal forma que las estructuras obtenidas contengan la máxima información semántica. Algunos trabajos como los de Feiju Xu ([Xu et al., 2002]) y nosotros mismos, consideran la posibilidad de utilizar otras fuentes de datos y bases de conocimiento para obtener tanto relaciones léxicas (sinónimos, antónimos o relaciones 'pertenece a') como relaciones conceptuales (hipónimos y merónimos, causas o suposiciones entre los términos extraídos de los documentos). Estos autores obtienen el dominio de la aplicación a partir de bases de conocimiento, tales como *WordNet, GermanNet* y *EuroWordNet* ([Vossen, 1998]) (ver Anexo D).

La decisión sobre la representación, por lo tanto, conllevará evaluar la necesidad de un determinado coste computacional, determinar si la carga semántica es importante o si se tiene claro el tipo de descubrimiento que se desea obtener.

Las formas intermedias se pueden obtener a través de diversos procesos como puede ser el de enriquecimiento y desambiguación con información semántica y contextual proveniente de fuentes de datos y bases de conocimiento tales

como tesaurus ([Berry and Castellanos, 2004], [Spasic et al., 2005]), taxonomías ([Feldman et al., 2002], [Tang and Liu, 2010], [Zhong et al., 2012]) u ontologías ([Kim et al., 2008], [Maedche and Staab, 2001], [Paralic and Bednar, 2003]), pero, ¿qué hacemos con ellas una vez obtenidas?.

En la tabla 11 podemos comprobar las aplicaciones que se les han dado a dichas FI en la literatura.

2.4 Estudio de las Técnicas de Minería de Textos existentes en la literatura

A lo largo de este trabajo se ha mencionado ampliamente la existencia de técnicas para realizar Minería de Textos, ayudadas por ámbitos tales como la Recuperación de Información, la Extracción de Información, el *Machine Learning* o la Inteligencia Artificial. En esta sección realizaremos una revisión de la literatura científica para resumir las técnicas más utilizadas en el área.

2.4.1 Técnicas de Minería de Textos

En esta sección se ofrece una visión general de las técnicas más utilizadas en la Minería de Textos, con referencias a la literatura científica. Se pretende ofrecer una visión de los enfoques más comunes y más empleados en las aplicaciones prácticas que se discutirán en secciones posteriores.

2.4.1.1 Reglas de Asociación

Las Reglas de Asociación se utilizan para descubrir hechos comunes dentro de un determinado conjunto de textos. Detectan implicaciones de la forma $X \rightarrow Y$ donde X es un conjunto de antecedentes e Y es el consecuente. Son una técnica poderosa de análisis de datos que aparece con frecuencia en la Minería de Datos.

Son muchos los autores que las utilizan para realizar minería como una herramienta muy útil en la toma de decisiones en las empresas: [Delgado et al., 2001], [Hu and Chen, 2006], [Sánchez et al., 2009], [Delgado et al., 2005].

En lo que respecta a los corpus textuales, también son varios los au-

Forma Intermedia	**Autores que la utilizan**
AP-Sets	[Martín-Bautista et al., 2008]
Bolsa de Palabras	[Ahonen et al., 1998], [Delgado et al., 2002a], [Feldman et al., 1998b], [Kosala and Blockeel, 2000], [Iiritano and Ruffolo, 2001], [Martín-Bautista et al., 2003], [Nahm and Mooney, 2001], [Rajman and Besançon, 1998], [Takeda et al., 2000], [Weiss et al., 2010], [Wong et al., 2000], [Yang et al., 1999]
Consultas	[Baeza-Yates and Tiberi, 2007], [Silvestri, 2010], [Lu et al., 2004]
Documentos Prototípicos	[Rajman and Besançon, 1998], [Rajman and Besançon, 1997], [Xu et al., 2002]
Episodios	[Delgado et al., 2002a], [Ahonen et al., 1997]
Eventos	[Delgado et al., 2002a], [Allan et al., 1998], [Yang et al., 1998], [Ananiadou and Mcnaught, 2005] , [Weng and Lee, 2011], [Fung et al., 2005], [Ananiadou et al., 2010]
Frases	[Lent et al., 1997], [Scott and Matwin, 1999], [Delgado et al., 2002a], [Yang et al., 1999], [Nguyen and Kan, 2007], [Turney, 2000], [Lewis, 1992], [Tseng et al., 2007]
Frases Multi - Términos	[Delgado et al., 2002a], [Feldman and Sanger, 2007], [Ahonen-Myka et al., 1999], [Chen et al., 2005]
Grafo Conceptual	[Montes-y Gómez et al., 2002], [Tan, 1999], [Mishne, 2003], [G.A.Ringland, 1988], [Jin and Srihari, 2007]
Grafo Semántico	[Dubois and Quafafou, 2002], [Gelfand et al., 1998]
Ontologías	[Maedche and Staab, 2000a], [Sánchez et al., 2008], [Paralic and Bednar, 2003], [Smrž et al., 2007], [Spasic et al., 2005], [Milne and Witten, 2013], [Medelyan et al., 2009], [Buitelaar et al., 2005], [Kim et al., 2008], [Paralic and Bednar, 2003], [Kontopoulos et al., 2013]
Taxonomía de Términos	[Wason, 2006], [Feldman et al., 1998b], [Feldman et al., 1998a], [Feldman et al., 2002], [Tseng et al., 2007], [Wu et al., 2004], [Kostoff et al., 2001]
N-gramas	[Cavnar, 1994], [Ari Pirkola and Järvelin, 2003], [Yang et al., 1999], [Kaski et al., 1998], [Kargupta et al., 1997], [Kešelj et al., 2003], [Peng et al., 2003], [Ananiadou and Mcnaught, 2005], [Pak and Paroubek, 2010], [Furnkranz, 1998], [Khomra, 2005]
Tendencias	[Delgado et al., 2002a], [Mei and Zhai, 2005]

Tabla 10: Formas Intermedias en la literatura

Forma Intermedia	Coste - Beneficio	Técnicas de Minería
AP-Sets	Mínimo Preprocesamiento [Martín-Bautista et al., 2008]	Categorización Naive-Bayes Clustering Textual Redes Neuronales Reglas de Asociación ...
Bolsa de Palabras	Mínimo Preprocesamiento Existencia de Sinónimos Herramientas ampliamente desarrolladas como WordNet [Stavrianou et al., 2007]	Categorización Naive-Bayes Clustering Textual Redes Neuronales Reglas de Asociación ...
Consultas	Gestión de la semántica entre consultas [Baeza-Yates and Tiberi, 2007] Evaluación de interdepencias entre términos de consultas [Silvestri, 2010]	Análisis de Grafos Semánticos Clustering Textual Análisis de Tendencias
Documentos Prototípicos	Trabaja con *full text* [Rajman and Besançon, 1998] Obtención de *itemsets* frecuentes [Rajman and Besançon, 1998]	Summarization de *Text Collections* Clustering Textual Reglas de Asociación
Eventos	Requiere corpus anotados o *full parsing* [Ananiadou et al., 2010]	Visualización
Tendencias	Manejo de *text stream* [Delgado et al., 2002a] [Mei and Zhai, 2005]	Grafos evolutivos

Tabla 11: Aplicaciones de las Formas Intermedias en la literatura

tores que emplean esta técnica utilizando diversos algoritmos que realizan búsquedas sobre las reglas de asociación en entornos textuales: *Apriori* [Mahgoub et al., 2008], [Lopes et al., 2007], [Holt and Chung, 2001], *FP-growth* [Lin et al., 2011], [Zhao and Bhowmick, 2003], *Rapid Association Rule Mining* [Zhao and Bhowmick, 2003], *Pruning* [Holt and Chung, 2001].

2.4.1.2 Reglas de Asociación Difusas

Las Reglas de Asociación Difusas [Martín-Bautista et al., 2003], [Kaya and Alhajj, 2005], que se pueden utilizar para *query refinement* y para visualización de *Text Mining* [Lopes et al., 2007], [Wong et al., 1999].

Una regla de asociación difusa se puede definir según [Delgado et al., 2008] como una expresión de la forma "Si X es A entonces Y es B", donde X, Y $\subseteq$ ER.

X = { x_1, ..., x_p } e Y = { y_1,...,y_q } son conjuntos de atributos disjuntos y A = { a_{x1}, ..., a_{xp} } y B = { b_{y1}, ..., b_{yq} } contienen los conjuntos difusos asociados a los correspondientes atributos de X e Y.

Las reglas de asociación difusas se diferencian de las reglas de asociación tradicionales en que intentan representar la imprecisión de la información, lo cual las convierte en una herramienta muy útil cuando se trabaja sobre texto. Algunos de

los usos, aplicados a Minería de Textos son: *query reformulation* o *query expansion* [Delgado et al., 2005].

En el trabajo de [Kaya and Alhajj, 2005], los autores señalan que conocer a priori el conjunto difuso más apropiado para cubrir el dominio de los atributos cuantitativos para minería de reglas de asociación difusas es bastante difícil, porque las características de los datos cuantitativos son desconocidos, en general, el uso de reglas difusas y de algoritmos genéticos puede solucionar en parte el hecho de que sea siempre un usuario el que deba refinar los conjuntos para la minería.

En [Martín-Bautista et al., 2003], las reglas de asociación se emplean para *query refinement*, técnica empleada para mejorar y guiar la búsqueda en Recuperación de Información con el fin de descartar documentos no interesantes, o incluso, recuperar documentos que pueden ser de interés pero que no se obtendrían en la búsqueda inicial.

2.4.1.3 Clustering

Esta técnica es una de las más utilizadas en la minería de texto [Smrž et al., 2007], [Larsen and Aone, 1999], [Berry and Castellanos, 2004], [Tan, 1999], [Neto et al., 2000], [Hotho et al., 2005], [Kostoff et al., 2001]. El **Clustering de documentos** es un método no supervisado que realiza descubrimiento en agrupaciones de documentos, basándose en su contenido o en un tema determinado que caracteriza a cada uno de los grupos en los que se puede dividir una colección de documentos. Para preprocesar los documentos se puede combinar con formas intermedias como *bags-of-words*, o incluso se pueden enriquecer los documentos mediante ontologías [Zhao and Karypis, 2005].

El clustering requerirá una correcta selección de las características y del algoritmo. En [Aggarwal and Zhai, 2012] diferencian entre el tipo de documentos sobre los que se realizará el clustering, aplicado a la minería de textos, catalogando dichas aplicaciones como dinámicas (si la información varía en el tiempo, habrá que tratar *streaming text*) o heterogéneas (donde el texto se compagina con otro tipo de información no textual, como es el caso de la red social *Flickr*).

2.4.1.4 Clustering Difuso

Los algoritmos de Clustering Difuso se han utilizado de forma satisfactoria en Minería de Textos debido a la gestión que realizan, adecuada, de los datos in-

completos o del "ruido"; por un lado, se han empleado para mejorar la compresión de los patrones [Bezdek et al., 1984], [Jipkate and Gohokar, 2012], y por otro, para tratar el manejo de los clusters que se superponen, sustituyendo el uso de métricas *crisp* por conjuntos difusos que reflejan el grado de pertenencia a un cluster [Arotaritei and Mitra, 2004], [Saxena et al., 2016].

Tanto los algoritmos ***c-means* difusos** como los **algoritmos jerárquicos de clustering difuso** se han utilizado en diferentes aproximaciones en el área de Minería de Textos [Goswami and Shishodia, 2013]. Se han realizado varias aproximaciones relevantes, que podemos consultar en la literatura, en el caso de los *c-means* [Deng et al., 2010], [Patil and Dongre, 2015], [Prabha et al., 2013] y en el caso del clustering jerárquico difuso, [Kumar et al., 2012], [Rodrigues and Sacks, 2004].

2.4.1.5 Árboles de Decisión

Un Árbol de Decisión [Ghosh et al., 2012], [Weiss et al., 2010], [Aggarwal and Zhai, 2012], [De Weerdt et al., 2012], [He et al., 2013] es una técnica de clasificación supervisada donde existen regiones no superpuestas y hojas. Esta técnica se utiliza sobre todos para hacer predicción en Minería de Textos [Weiss et al., 2010]. Un ejemplo de árbol de decisión es el ***Hoeffding Tree*** [Bifet and Frank, 2010] es un árbol de decisión que emplea una estrategia de poda previa basada en el margen de *Hoeffding*. Los nodos se expanden en cuanto hay evidencias estadísticas.

2.4.1.6 Árboles de Decisión Difusos

Tal y como se define en [Do Prado, 2007], en general un árbol de decisión difuso es una estructura de árbol donde los valores básicos de los atributos de una rama se caracterizan por sus funciones de pertenencia.

Algunos de los enfoques que utilizan esta técnica en el área de Minería de Texto incluyen la clasificación de documentos de texto, donde la minimización de la entropía difusa y la ambigüedad de clasificación se utilizan para seleccionar el atributo discriminante en el nodo a la partición [Wahiba and Ahmed, 2016]. En [Wang and Wang, 2005], los autores presentan una extracción de la regla de categorización de texto basada en árboles de decisión difusa.

Este enfoque reduce las características del texto en términos de la estadística

mejorada del *chi-cuadrado*, y por lo tanto reduce en gran medida las dimensiones del espacio vectorial, aumentando la precisión de la categorización y la comprensibilidad de las reglas de categorización. En [Abdessalem et al., 2016], se presenta un enfoque combinado de árbol de decisiones difuso y similaridad de documentos para el clustering de documentos.

2.4.1.7 Algoritmos Genéticos

Los algoritmos genéticos son optimizaciones heurísticas cuyos mecanismos son análogos a la evolución biológica. Después de una población inicial de elementos, denominados cromosomas, generada aleatoriamente, se ejecutan las funciones de variación y selección hasta que se alcance un criterio establecido. Entre los autores que los mencionan como técnica de Minería de Textos encontramos a [Ghosh et al., 2012], [Atkinson-Abutridy et al., 2004], [Krallinger et al., 2008], [Kaya and Alhajj, 2005].

2.4.1.8 Redes Neuronales

Las Redes Neuronales Artificiales (ANNs) sirven para tratar datos complejos y de los que tenemos escaso conocimiento. Entre las ventajas que presentan encontramos: las ANNs no realizan suposiciones sobre la naturaleza de la distribución de los datos; las redes neuronales trabajan bien con datos incompletos u ocultos lo que las hace útiles para *time-series data*, además de ser una herramienta no lineal idónea para dichas series dinámicas [Ghosh et al., 2012].

Castillo [Rojas and Villegas, 2012] habla de las ventajas de las redes neuronales con, al menos, una capa intermedia. Dichas redes usan los datos para desarrollar una representación interna de la relación entre las variables. Las series de datos son dinámicas, así que es necesario utilizar herramientas no lineales para discernir relaciones entre los datos de series de tiempo, asimismo ayudarán a descubrir relaciones no lineales.

Mientras que el análisis de regresión tradicional no es adaptativo, las redes neuronales permiten adaptar y trabajar tanto con datos antiguos como con datos nuevos.

2.4.1.9 Redes Neuronales Difusas

Esta técnica sirve como potente enfoque de auto aprendizaje no lineal para clasificación y minería de patrones [Tian et al., 2009]. En [Liu et al., 2010], se aplica un enfoque de red neural difusa a la minería de textos, encontrando el número óptimo de clusters por adelantado, utilizando los vectores centrales difusos como los pesos de la red neuronal. En [Tian et al., 2009] se construye una estructura de red neural difusa para la clasificación de texto web, donde se utiliza una adaptación del algoritmo de optimización de *Levenberg-Marquat* para entrenar la red neural difusa, mejorando la tasa de convergencia y la precisión de clasificación.

Un tipo de red neuronal bastante utilizado es el ***Self-Organizing map*** [Kaski et al., 1998], [Paralic and Bednar, 2003], mapa de categoría de palabras. La finalidad de este tipo de mapa es facilitar la búsqueda interactiva. Se utiliza para codificar los documentos de tal manera que expresen explícitamente la similaridad de los significados de las palabras. Dado que la información contextual relacionada con cada palabra del vocabulario se puede codificar, para conseguir dicho mapa, [Kaski et al., 1998] crea un vector aleatorio r para cada palabra del vocabulario. Encuentra todas las instancias de cada palabra que se desean tener en cuenta, y serán todas, palabras clave. A partir del vector aleatorio, cada palabra clave se asocia con una marca contextual y el documento quedará codificado.

2.4.1.10 Naïve Bayes y Multinomial Naïve Bayes

El clasificador Naive Bayes es un clasificador generativo que modela la distribución de documentos en cada clase, utilizando un modelo probabilístico con supuestos independientes sobre las distribuciones de diferentes términos. Según Aggarwal, [Aggarwal and Zhai, 2012], los clasificadores probabilísticos fueron diseñados para usar un modelo implícito de mezcla para generar documentos subyacentes. Cada componente de la mezcla es un modelo generativo que proporciona la probabilidad de un término dado con ejemplos para dicho componente.

Existe una amplia variedad de algoritmos de los que pondremos algunos ejemplos: la *Multinomial Naïve Bayes* [Bifet and Frank, 2010] es un clasificador para clasificación de documentos que considera un documento como un *bag-of-words*. Para cada clase c, la probabilidad de observar una palabra dada dicha clase, se estima a partir del conjunto de datos de entrenamiento computando la frecuencia relativa de cada palabra de la colección de documentos de entrenamiento de dicha clase.

Los algoritmos Naive Bayes se han utilizado en la literatura en combinación con técnicas difusas para resolver problemas de minería. En [Fernandez et al., 2006] se utiliza un criterio de combinación difusa para evaluar la relevancia de las características en el texto de una página web. El sistema difuso asigna las funciones de peso y su combinación y genera reglas difusas. En [Roy and Toshniwal, 2010] se propone un algoritmo para el agrupamiento difuso de documentos de texto utilizando el concepto *naive-bayes* en lo que se denomina independencia condicional de clúster. Se basa en el supuesto de que un término probabilidad de pertenecer a un grupo particular es independiente de sus probabilidades de pertenecer a los otros grupos. Aplicando este concepto a los términos, esto se traduce en el hecho de que la probabilidad de co-ocurrencia es independiente de los términos.

2.4.1.11 Otras Técnicas

Existen muchas otras técnicas sean o no de clasificación empleadas en la literatura de Minería de Textos. A modo de ejemplo, ***Support Vector Machines* (SVM)** se emplean en [Ghose and Ipeirotis, 2011], donde los autores comparan con la técnica de clasificación de SVM para la creación de un modelo predictivo y obtienen un mejor comportamiento utilizando SVM.

***Stochastic Gradient Descendet* (SGD)** [Bifet and Frank, 2010] es un algoritmo utilizado para realizar entrenamiento en una red neuronal. Se basa en buscar la dirección en la que una pequeña variación del vector de pesos hace que el error decrezca más rápidamente.

Todas las técnicas tradicionales de minería vistas en esta sección se pueden adaptar para trabajar sobre datos *stream* que requieren de técnicas incrementales que se adapten rápidamente a los cambios sobre dichos datos, así, entre los autores que trabajan con ***Streaming data*** nos encontramos a [Bifet and Frank, 2010], [Hotho et al., 2005], [Liu, 2007], [O'Connor et al., 2010], [Mei and Zhai, 2005], [Aggarwal and Zhai, 2012].

Para este tipo de aplicaciones, los autores suelen recomendar el uso de un proceso de extracción de estructuras de conocimiento a partir de registros de datos continuos o rápidos. Un ejemplo claro de aplicación de estas estructuras y técnicas adaptadas son los posts de los blogs o los *tweets* cuyo flujo varía continuamente.

2.4.2 La Recuperación de Información como el paso previo al Descubrimiento de Conocimiento en Textos

Cuando pretendemos realizar Minería de Textos, la primera idea que se nos viene a la mente es que hay que determinar el conjunto de documentos que van a ser analizados y habrá que estudiar el método de conseguirlos del modo más óptimo. Necesitaremos herramientas que nos devuelvan una serie de documentos para poder trabajar con ellos.

La Recuperación de Información (IR ó *Information Retrieval*) trata con la representación, almacenamiento, organización, y acceso a las unidades de información y los textos a partir de una colección de documentos escritos que deben satisfacer una información que necesita el usuario y cuya solicitud ha sido expresada en Lenguaje Natural.

Un Sistema de Recuperación de Información en Lenguaje Natural es aquel que realiza las funciones de: recoger la información de un usuario (hablada o escrita en LN), procesarla y generar una respuesta adecuada (utilizando un modelo de recuperación y con métodos de inferencia inductivos). Es capaz de acceder a la información almacenada previamente en las bases de datos y puede ejecutar funciones sobre los datos recuperados, como parte del proceso de la solicitud que ha sido realizada por el usuario.

Un Modelo de Recuperación de Información es un conjunto de premisas y un algoritmo para ubicar los documentos de acuerdo a una consulta de usuario. Formalmente, es una cuádrupla *[D, Q, F, R(q_i,d_j)]* donde D es un conjunto de vistas lógicas de documentos, Q es un conjunto de consultas de usuario, F es un marco de trabajo para modelar documentos y consultas y *R(q_i,d_j)* es una función ranking que asocia un ranking numérico a la consultas q_i y al documento d_j.

Los modelos de Recuperación de Información han sido utilizados con éxito en la representación de expresiones de lenguaje natural, es decir, en las consultas del usuario, representaciones simples de sus necesidades de información, y las definiciones de los términos del glosario. Algunos de los modelos clásicos de Recuperación de Información son los siguientes: *Booleano* (modelo que se fundamenta en la ***Teoría de Conjuntos*** y en el ***Álgebra de Boole***. Se basa en la presencia o ausencia de un término en un documento, de este modo, diremos que un peso para un término índice será de la forma $w_ij\epsilon\{0,1\}$. Las consultas de este tipo de modelos, enlazan los ítems con conectivas lógicas, como son **and**, **or** y **not**), *Espacio Vectorial ó MEV* (donde la frecuencia de la ocurrencia de términos en un documento es un esquema de pesos. La frecuencia del documento se combinará con el factor de frecuencias

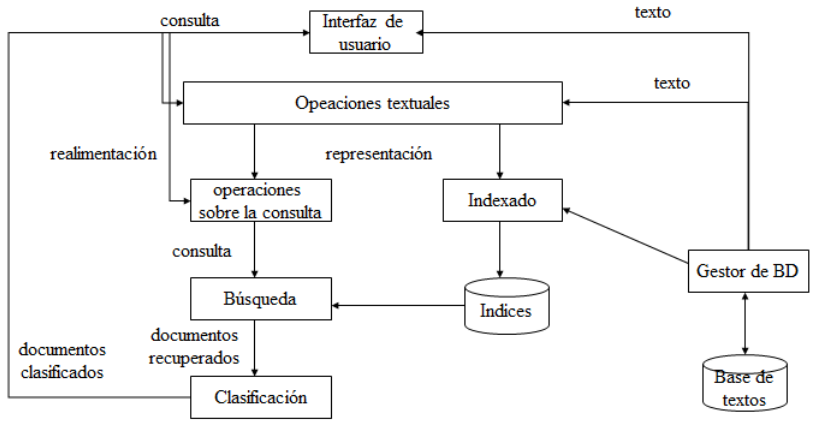

Figura 14: Proceso de Recuperación de Información ([Rodríguez, 1999])

de la colección, que se usa para discriminar los documentos), *Probabilístico* o variantes (el peso de los términos índices será todos los binarios, una consulta será un subconjunto de términos índices, es decir, para una consulta de usuarios, existirá un conjunto de documentos que formarán el conjunto respuesta ideal), *Bayesiano*, *Redes de Inferencia*, *Redes de Creencia*, *Modelo vectorial generalizado* y *Modelo booleano extendido*.

Tres de estos modelos son ampliamente aceptados y han probado su efectividad: el modelo de espacio vectorial (MEV), el modelo probabilístico y el modelo booleano. En la Figura 14 vemos representado el proceso ([Rodríguez, 1999]).

En los comienzos, los Sistemas de Recuperación de Información se diseñaron para recuperar documentos con respecto a una petición formulada por un usuario experto en un lenguaje regular. Después, comenzó a realizar nuevas tareas: filtrado y enrutamiento de la información entrante o análisis de grandes corpus de documentos, medir y usar la relevancia de las características del texto.

En el Descubrimiento de Conocimiento en Textos se han utilizado diferentes técnicas relacionadas con la IR. La Recuperación de Información es necesaria a la hora de recuperar los documentos objeto del análisis pero también se pueden utilizar técnicas de IR para la fase de Preprocesamiento de documentos, bien sean propias de la Recuperación de Información (como el indexado o el *ranking*) o ajenas como *soft computing* ([Delgado et al., 2002b]) o el indexado semántico.

Los autores que hemos seleccionado en este trabajo, hacen referencia a las he-

Autores	Técnicas aplicadas a IR
Hearst [Hearst, 1999b]	Clustering Generación automática de asociaciones de términos Cocitación
Gelfand [Gelfand and Wulfekuhler, 1998]	Cocitación
MAck [Mack and Hehenberger, 2002]	Clustering Indexado Ranking Operadores Difusos

Tabla 12: Fase de Visualización en KDT

rramientas de Recuperación de Información como métodos que sirven para extraer documentos en la fase de Minería de Textos que satisfagan la información que necesitan los usuarios ([Iiritano and Ruffolo, 2001], [Hearst, 1999b], [Tan, 1999], [Gelfand and Wulfekuhler, 1998], [Mack and Hehenberger, 2002]).

En la Tabla 12, destacamos las técnicas utilizadas por los autores para acometer la Recuperación de Información, bien sea para recuperar documentos como para la fase de Preprocesamiento.

Aunque las fases iniciales de Descubrimiento de Conocimiento en Textos se pueden basar en la descomposición del texto en palabras, igual que la Recuperación de Información, el fin último de ambos procesos dista mucho. El objetivo de la IR es conseguir optimizar las consultas del usuario, agilizar las búsquedas de documentos en la base de datos o incluso analizar grandes cantidades de documentos, en ningún momento descubre información que no haya sido escrita por los autores de los documentos. Por el contrario, la meta del KDT es extraer conocimiento que no está explícitamente en los datos.

2.4.3 La Extracción de Información en el Descubrimiento de Conocimiento en Textos

Cuando nos enfrentamos al texto con la idea de descubrir conocimiento, nos encontramos con el problema de la falta de estructura del mismo. Esta falta de estructura es sólo aparente, porque, realmente, el texto presenta una estructura demasiado compleja y difícil de tratar computacionalmente. Para vencer este pro-

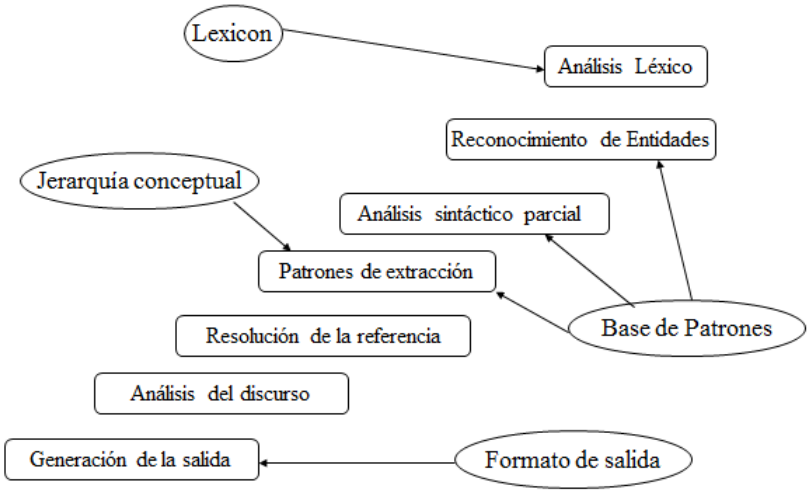

Figura 15: Arquitectura de un Sistema de Extracción de Información ([Rodríguez, 1999])

blema, podemos aplicar técnicas de Extracción de Información (IE o *Information Extraction*) que se define como el proceso de localizar las porciones de un texto dado, que contengan información relevante para las necesidades de un usuario y proporcionar dicha información de forma adecuada al proceso.

Típicamente, un Sistema de Extracción de Información (SEI) (Figura 15) extrae informaciones sobre entidades, relaciones y eventos a partir de una serie de documentos que habitan en un dominio restringido. Este sistema puede utilizar recursos de Lenguaje Natural como son Ontologías, Lexicones y Corpus de documentos para realizar sus tareas. Su aplicación se caracteriza por la presencia de técnicas tanto de Lenguaje Natural, como de Análisis de Textos (Análisis Léxico, Reconocedor de Nombres Propios ...). Algunas de estas técnicas son las siguientes:

- Técnicas empíricas como el Aprendizaje Automático (ML) ([Cardie and Mooney, 1999]).
- Análisis Léxico: Identificación de la lengua.
- División del texto en unidades (tokens).
- Consulta a diccionarios: por ejemplo, PROTEUS (NYU), Comlex, Nombres propios (personas, geográficos, empresas), ...
- Procesadores específicos de fechas, cantidades, siglas, locuciones, términos multipalabras, ...

- Reconocedores de nombres propios (*Named Entities*): Lexicones especializados, patrones (expresiones regulares).
- *Pos-tagging*: desambiguación morfosintáctica.

La aplicación de técnicas de Extracción de Información al documento o conjunto de documentos, constituye una fase dentro del descubrimiento. Algunos autores la llaman **Refinamiento de Textos** (*Text Refining*), para otros es la fase de Preprocesamiento mientras que hay autores que, simplemente, no la mencionan. En todo caso, con la aplicación de técnicas de Extracción de Información, se consigue un mapeo del texto escrito en Lenguaje Natural en una representación estructurada y predefinida o bien en una plantilla que se rellena con una extracción de claves del documento. Esta información, que ya está estructurada, puede almacenarse en una base de datos y a partir de ahí se podrá consultar, realizar minería sobre ella, etc ...

No todos los autores estudiados utilizan Extracción de Información en el proceso de descubrimiento de conocimiento, pero entre aquellos que la emplean, existe una serie de motivaciones similares. Veamos para qué utilizan la IE:

- Para proporcionar una estructura al texto antes de realizar Minería de Textos ([Nahm and Mooney, 2002], [Montes-y Gómez et al., 2002]). En este acaso, la información que se desea extraer del texto la determina una plantilla con una serie de datos que se deben completar con partes del texto que se está analizando (Gráfico 16). Con los datos obtenidos se crea una base de datos que servirá como entrada a la fase de Descubrimiento de Conocimiento en Datos (para Nahm, la Minería de Textos se compone de IE y técnicas de KDD. Ver gráfico 17).

  Como veremos a lo largo de este trabajo, la realización de Minería de Términos no tiene por qué implicar la utilización de técnicas de Extracción de Información y, para nosotros, será un fase independiente dentro del proceso de Descubrimiento de Conocimiento en Texto.

- Para extraer características del texto ([Feldman et al., 1998b], [Delgado et al., 2002a]). Las características que se pueden obtener del documento son: palabras, términos, claves, eventos, episodios o etiquetas de XML, entre otras; con todas ellas o con algunas, se podrá construir la Forma Intermedia.

- Para aprender patrones léxico sintácticos, que indican relaciones sintagmáticas relevantes (relaciones paradigmáticas y de dominio) ([Xu et al., 2002]).

- Para extraer términos de los documentos o incluso hechos (trozos específicos de texto) ([Nahm and Mooney, 2002], [Mack and Hehenberger, 2002]).

Filled Template

- title: "Web Development Engineer"
- location: "Beaverton, Oregon"
- languages: "C/C++", "Java", "Javascript", "VBScript", "Perl","PHP","ASP"
- platforms: "Solaris", "Linux", "IBM AIX", "Windows/NT"
- applications: "Oracle", "ODBC", "JDBC", "Flash/Shockwave", "FrontPage", "Cold Fusion"
- areas: "Database", "CGI", "scripting"
- degree required: "BSCS"
- years of experience: "2+ years"

Figura 16: Plantilla de Extracción de Información ([Nahm and Mooney, 2002])

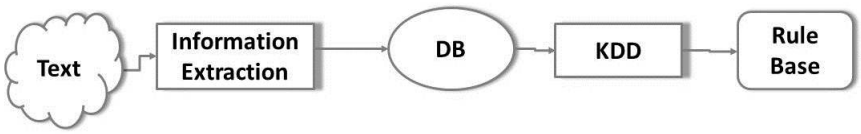

Figura 17: Minería de Textos y Extracción de Información ([Nahm and Mooney, 2000])

- Para realizar comprensión de textos, muy relacionado con análisis léxico y sintáctico de los documentos ([Nahm and Mooney, 2002]).

Los pasos básicos dentro del proceso de Extracción de Información son: la extracción de hechos, la integración de los mismos y la representación del conocimiento. A continuación, presentamos brevemente cada uno de ellos:

- Extracción de hechos: búsqueda de hechos individuales contenidos dentro del documento, para ello, es muy importante el conocimiento específico de dominio. De este modo estaremos consiguiendo la información realmente relevante del documento. Las técnicas utilizadas en esta fase son:
  - *Pattern matching*
  - Análisis léxico
  - Extracción de la estructura sintáctica y semántica
- Integración de hechos: vinculada a la co-referencia. Uno de los primeros pasos son las referencias anafóricas (si se menciona a 'ella' debe saberse a qué persona se está refiriendo el texto). Otro de los pasos, éste debe estar entre los últimos, será el concepto de unir eventos:

  Laura protagonizó la película. Alejandro fue su compañero.

  Con estos dos hechos podemos razonar que Alejandro también participó en la película.
- Representación de conocimiento: aunque es una fase trivial del proceso, puede ser importante para el usuario final. Abarca desde rellenar una plantilla que sirva como entrada a bases de datos, hasta la visualización gráfica de resultados.

Como resumen de las fases, podemos ver un ejemplo: Si se dispone de un corpus de documentos financieros y se someten a un proceso de Extracción de Información, se obtendrá una representación como la disponible en la Tabla 13, donde se muestran los términos y los eventos financieros más importantes extraídos durante el proceso. Esta información servirá como entrada para los algoritmos de Minería de Textos propiamente dichos.

Uno de los inconvenientes de la Extracción de Información (también del KDT) es su dependencia total del dominio en el que se encuentren los documentos, así,

Documentos	Términos y eventos extraídos
Doc1	{Microsoft, Neurosoft, Take-over, Departamento de Computación, ... Evento1 ...}
Doc2	{Microsoft, IBM, Share capital increasee, ...Event2 ...}
...	

Tabla 13: Ejemplo de la extracción de términos en IE

Autores	Herramientas EI
Feldman [Feldman et al., 1998b]	Categorización de tópicos
Feiju Xu [Xu et al., 2002]	Adquisición de patrones léxico sintácticos
Gelfand [Gelfand and Wulfekuhler, 1998]	Extracción automática de términos
Mack [Mack and Hehenberger, 2002]	Localización de trozos específicos de texto

Tabla 14: Herramientas de Extracción de Información

las plantillas generadas para documentos médicos no servirán para documentos jurídicos. Feiju Xu aporta una posible solución a este problema mediante la extracción de sinónimos próximos durante el proceso de Extracción de Información ([Xu et al., 2002]).

En la tabla 14 podemos ver algunas de las técnicas de Extracción de Información usadas por los autores en el Preprocesamiento. Hay autores, ([Rajman and Besançon, 1998]), que apuestan por la utilización de herramientas que no son propias de la Extracción de Información, como son la extracción de claves automáticas y la minería de documentos prototípicos (información que ocurre de modo repetitivo sobre un conjunto de documentos, ver Gráfico 13), para llevar a cabo esta fase. Aunque, basándonos en los resultados obtenidos en dicho artículo, el autor se refiere al Descubrimiento de Conocimiento dentro del Texto cuando habla de Extracción de Información, es decir, sobre documentos indexados con técnicas de NLP o sobre documentos prototípicos aplica técnicas de NLP (*part of speech tagging* y extracción de términos) para encontrar secuencias frecuentes de términos en los documentos.

Para finalizar esta sección, es importante notar que el coste computacional de las técnicas de Extracción de Información, como pueden ser las técnicas de Preprocesamiento lingüístico (*part of speech tagging, lemmatization, tokenization*), puede ser demasiado alto para la organización: habrá que evaluar si la relación coste / resultados es positiva a la hora de aplicar estas técnicas.

2.5 Conclusión

El Descubrimiento de Conocimiento en Datos es un proceso que ha tenido una gran acogida en el mundo empresarial y científico por su capacidad para encontrar información útil en la toma de decisiones. Su baza principal es la Minería de Datos que se encarga de encontrar esas relaciones o patrones interesantes que después se utilizarán convenientemente por expertos para distintos fines.

Si nos paramos un poco en este punto, podemos razonar que, si este tipo de tratamiento se le puede aplicar a los datos, también podrá realizarse sobre información textual, que, al fin y al cabo, está presente en mayor medida dentro de las empresas.

El Descubrimiento de Conocimiento en Textos presenta grandes similitudes y múltiples diferencias respecto al Descubrimiento de Conocimiento en Datos. La especifidad del texto, el idioma, su procedencia, su almacenamiento, cualquier característica lo convierte en único y diferente. Y nos aparta del camino del KDD.

Este estudio nos ha llevado por otros términos, que se pueden confundir en la literatura científica en ocasiones con Minería de Textos pero que, a nuestro parecer, se corresponden con esta fase del KDT sino que se trata de otras disciplinas de conocimiento, obsérvense las definiciones de *Opinion Mining, Sentimental Analysis*, entre otros.

Hemos podido comprobar la enorme cantidad de representaciones intermedias diferentes que se le puede proporcionar al texto, recorriendo un abanico que varía tanto en complejidad como en eficiencia computacional, englobando desde la bolsa de palabras hasta el grafo semántico.

Elegida la Forma Intermedia, hemos presentado diferentes aplicaciones sobre las que se aplica actualmente la Minería de Textos y que están muy cercanas al usuario convencional: aplicaciones para web semántica, redes sociales o herramientas de *help-desk* se benefician en la actualidad de esta disciplina.

En la siguiente sección presentaremos el concepto de *Text Knowledge Mining* como el proceso de extracción de verdadero conocimiento a partir de textos sin estructurar y expondremos las razones que nos llevan hasta esta definición.

Capítulo 3 Una Nueva Visión de la Minería de Textos

In formal logic, a contradiction is the signal of defeat,
but in the evolution of real knowledge it marks the first
step in progress towards a victory
(Alfred North Whitehead [Whitehead, 2011])

La generación de nuevo conocimiento, bien por parte del ser humano o por medios automáticos, se realiza a través de procesos de razonamiento, que toman como punto de partida conocimiento existente. En el caso de las computadoras, como es bien conocido, los procesos de generación de nuevo conocimiento requieren del uso de lenguajes formales de representación del conocimiento, dotados de mecanismos formales de razonamiento.

Esta afirmación general tiene también como caso particular a la minería de datos (en su acepción como sinónimo de descubrimiento de conocimiento en bases de datos). Los modelos de datos utilizados en las bases de datos sobre los que habitualmente se realizan procesos de minería son un formalismo de representación de conocimiento, si bien semánticamente pobre. Por ejemplo, cada tupla de una tabla que contenga datos sobre un empleado de una empresa puede verse como la representación de conjunciones de *hechos* relativos a un empleado concreto: *su nombre es Luis* y *su edad es 27* y *su sueldo es 1500* y ...[1] De la misma forma, en una base de datos transaccional, cada una de las transacciones es una conjunción de *items* (la cesta de compra contiene pan y leche y galletas y ...). Los hechos, en la terminología de la ingeniería del conocimiento, constituyen el tipo más simple de conocimiento.

[1] Hay otras formas de interpretar tuplas como hechos, por ejemplo la utilizada en el modelo lógico de bases de datos, en el cual una tabla con n atributos puede modelarse como un predicado con n argumentos, entre otras posibilidades.

El tipo de estructura de una base de datos relacional o transaccional puede verse así como una lista no ordenada de conjunciones de hechos, que describen objetos, ejemplos, transacciones, etc.[2]

En la minería de datos, el conocimiento se obtiene principalmente a través de razonamiento de tipo *inductivo* a partir de los datos, tanto supervisado como no supervisado. Este tipo de razonamiento consiste en obtener *patrones* o *modelos* generales que expliquen los datos, en un proceso que va desde lo particular a lo general. El razonamiento inductivo a partir de datos, que se basa en la *semejanza* de objetos/ejemplos, y en la frecuencia de repetición de los mismos, es el que se utiliza en las técnicas de clustering, reglas de asociación, árboles de decisión, patrones secuenciales, etc. típicas de la minería de datos.

La minería de textos, como hemos visto en el capítulo anterior, surge inmediatamente después de la minería de datos, y después de que esta última ha probado su utilidad con éxito. Probablemente por ello, la visión convencional de la minería de textos ha consistido en verla como un caso particular de minería de datos, con la particularidad de que se necesita un proceso previo de transformación del texto en datos, utilizando para ello diversas estructuras de datos (formas intermedias). Así, realizar minería de textos se interpreta directamente como realizar minería de datos sobre las correspondientes formas intermedias extraídas de los textos. En la literatura se han utilizando prácticamente todas las técnicas existentes de minería de datos para la minería de textos, utilizando para ello una gran diversidad de formas intermedias, como hemos visto en el capítulo anterior.

Este enfoque convencional, que en definitiva concibe la minería de textos como la aplicación de razonamiento inductivo sobre datos extraídos del texto, ha resultado de mucha utilidad en gran cantidad de aplicaciones. Sin embargo, esta visión ignora un hecho fundamental: *el texto es un formalismo de representación con una capacidad expresiva mucho mayor que las estructuras de datos*, por lo que, al reducir el contenido del texto a una forma intermedia, se pierde una enorme cantidad de información valiosa para la obtención de nuevo conocimiento. El texto contiene conocimiento, expresado mediante lenguaje natural, mucho más rico que una estructura de datos. Más allá de una simple lista de conjunciones de hechos, los textos incluyen de manera habitual expresiones condicionales, disyunciones y negaciones, etc. A partir de este punto, en esta Tesis utilizaremos el término *conocimiento* y *base de conocimiento* para referirnos a representaciones con semántica más rica que los datos y bases de datos.

En el ámbito de la Inteligencia Artificial, transformar una base de conocimiento

[2]No consideramos aquí bases de datos deductivas, con capacidades de representación y razonamiento más avanzadas, pero que raramente son utilizadas en procesos de minería de datos.

semánticamente rico en una base de datos, con la consiguiente pérdida de semántica, puede resultar una idea bastante extraña. ¿Por qué hacerlo entonces? En nuestra opinión, los motivos que han llevado a ello son los siguientes:

- Existe una gran cantidad de información disponible en bases de datos convencionales, documentales, y en la propia web, en forma de texto, lo que motiva la necesidad de su análisis para la obtención de nuevo conocimiento.
- El paradigma actual de obtención de nuevo conocimiento a partir de la información digital es la minería de datos, por lo que un primer enfoque natural es tratar de convertir el problema de la minería de textos en un problema de minería de datos.
- Las técnicas de minería de datos se basan fundamentalmente en razonamiento inductivo sobre datos, que requiere como punto de partida un conjunto de datos con una cierta estructura, donde se buscan distintos tipos de patrones y modelos sobre la base de la semejanza de características y/o la repetición. Para poder aplicar estas técnicas en el ámbito de la minería de textos, es necesario *transformar* el texto en datos.

Una de las principales aportaciones de esta Tesis es la propuesta de una nueva visión de la minería de textos más natural, a la vez que complementaria a la que acabamos de exponer, y que abre un enorme abanico de posibilidades de obtención de conocimiento que las técnicas basadas en minería de datos sobre formas intermedias no pueden llevar a cabo. En este nuevo paradigma, la obtención de conocimiento novedoso, no trivial, previamente desconocido y potencialmente útil a partir de colecciones de textos puede realizarse mediante la aplicación de técnicas de razonamiento sobre el conocimiento, mucho más rico, disponible en los textos. En el ámbito de los sistemas inteligentes, existen formalismos y mecanismos para llevar a cabo razonamiento deductivo y abductivo, entre otros. Este capítulo está dedicado a describir este nuevo paradigma de minería de textos, sus fundamentos, contribuciones, y los problemas abiertos que plantea.

3.1 Un Nuevo Paradigma: Minería de Textos basada en *Conocimiento*

La idea básica que subyace nuestra propuesta es que, dado que el texto es una representación de conocimiento mucho más sofisticada que una base de datos, estando más cerca del concepto de base de conocimiento propio de los sistemas inteligentes,

resulta natural utilizar razonamiento deductivo, abductivo y de otros tipos para el descubrimiento de conocimiento a partir de textos.

Estos tipos de razonamiento han sido utilizados de manera exhaustiva desde hace muchos años en el desarrollo de sistemas inteligentes (sistemas expertos, agentes inteligentes, etc.). Sin embargo, el desarrollo de sistemas de minería de textos afronta problemas muy distintos y tiene diferencias significativas con el desarrollo de sistemas inteligentes en muchos aspectos.

En primer lugar, existen diferencias importantes entre un texto y la base de conocimiento de un sistema inteligente:

- Los sistemas inteligentes suelen utilizar formalismos de representación de conocimiento que permitan llevar a cabo razonamiento automático con el mismo, de tipo principalmente deductivo, aunque también de otros tipos. Por el contrario, el texto viene descrito en lenguaje natural (más concretamente, en uno de los muchos lenguajes naturales utilizados por el ser humano en el mundo), que no es una representación adecuada para llevar a cabo razonamiento. En la actualidad, las representaciones más utilizadas son lógicas formales de distintos tipos y representaciones relacionadas como los sistemas basados en reglas, entre otros. En lo sucesivo, en esta Tesis nos centraremos en el uso de lógicas formales.
- Las bases de conocimiento de sistemas inteligentes se diseñan cuidadosamente para cumplir una función específica, concretamente son un elemento fundamental de un sistema computacional que utiliza el conocimiento para resolver un problema real concreto. Por el contrario, los textos tienen una función muy diferente, que puede variar extraordinariamente según el ámbito en el que se generen (informes de gestión, noticias en prensa, correos electrónicos que formen parte de una conversación o que difunden información a un grupo, mensajes cortos en redes sociales y editoriales que contienen opiniones, textos legales o reglas internas de empresa de tipo normativo, textos científicos o académicos, etc.).
- Finalmente, las bases de conocimiento tienen como destinatario un sistema inteligente, y por tanto se diseñan para contener todo el conocimiento relevante para el sistema, asumiendo en muchas ocasiones la hipótesis de mundo cerrado (si no puedo demostrar que algo es cierto con el conocimiento disponible, entonces es falso), así como evitar ambiguedades y contradicciones. Por el contrario, el texto tiene como destinatario seres humanos, por lo que la relevancia, interpretación y valoración del mismo estará sujeta al contexto del conocimiento previo, las creencias y opiniones de dicha persona. Además, se

asume que puede haber desconocimiento con respecto a la verdad de determinadas expresiones. Por todo ello (al igual que ocurre en la valoración de los resultados en la minería de datos) el conocimiento previo del destinatario final (en inglés, *background knowledge*) juega un papel fundamental.

De la misma forma, hay grandes diferencias entre los objetivos de un sistema inteligente y de un sistema de minería de textos. El primero genera conocimiento en respuesta a cuestiones específicas que van surgiendo durante la operación del sistema, que suelen estar *provocadas* por la interacción del sistema con su entorno, con el objetivo de tomar la decisión más adecuada a la situación actual. Por el contrario, en un sistema de minería de textos, el objetivo es tratar de obtener nuevo conocimiento a través de un conjunto de textos proporcionado, en respuesta a un planteamiento o cuestión mucho menos específico. Esta diferencia es, de alguna manera, similar a la que existe entre un test estadístico y la minería de datos: mientras que el primero plantea cuestiones muy específicas (por ejemplo, ¿son independientes las variables X_1 y X_4?), la segunda plantea cuestiones más generales (por ejemplo, ¿es posible encontrar asociaciones entre valores de las variables del problema?).

El nuevo paradigma de minería de textos que proponemos toma como punto de partida las siguientes hipótesis:

1. Es posible obtener formas intermedias a partir del texto, basadas en lógicas formales, que representen aspectos relevantes del conocimiento descrito en lenguaje natural. Como ocurre siempre en la minería de textos, se asume que este paso supone una pérdida de información, aunque mucho menor que la que se produce en las formas intermedias que representan un texto mediante una estructura de datos. Esto no supone un problema dado que, como es habitual en la minería de textos, y de forma contraria a lo que ocurre con los sistemas inteligentes, se asume que no se requiere exhaustividad en la representación de conocimiento de los textos (la forma intermedia puede no recoger completamente la semántica del texto).

2. Mediante el uso de razonamiento sobre estas formas intermedias, es posible obtener conocimiento novedoso, no trivial, previamente desconocido y potencialmente útil a partir de textos, que no podría obtenerse mediante razonamiento inductivo a partir de datos obtenidos de los mismos textos.

3. Es posible desarrollar técnicas de minería de textos basadas en los elementos anteriores. De la misma forma que hay muchas técnicas distintas de minería de textos basadas en razonamiento inductivo sobre datos, también es posible desarrollar muchas técnicas distintas basadas en razonamiento sobre conocimiento

> más rico, abriéndose un campo enorme de posibilidades. Contrariamente a lo que ocurre en los sistemas inteligentes, y como es habitual en procesos de minería de datos y de textos, obtener algunas piezas de conocimiento interesante es ya un éxito, no siendo imprescindible obtener todas las posibles.

Es importante destacar que, aunque las técnicas habituales de descubrimiento de conocimiento a partir de textos se basan principalmente en aprendizaje inductivo a partir de datos, las definiciones que existen en la literatura no exigen el uso de este tipo de técnicas. Como hemos visto en el capítulo anterior, las definiciones existentes hacen referencia a *la obtención de nuevo conocimiento novedoso, no trivial, previamente desconocido y potencialmente útil a partir de textos*. Por lo tanto, no existe restricción alguna en cuanto al tipo de técnicas que pueden utilizarse en minería de textos, y por ello nuestra propuesta encaja plenamente en el marco del descubrimiento de conocimiento en textos.

En los siguientes apartados hablaremos de la extracción de formas intermedias basadas en lógicas y de razonamiento sobre conocimiento. En la sección 3.2 hablaremos sobre las particularidades del proceso de minería de textos basada en este tipo de razonamiento.

3.1.1 Extracción de Formas Intermedias basadas en Lógicas Formales

La primera hipótesis en la que se basa nuestra propuesta de un nuevo paradigma de minería de textos es que es posible obtener formas intermedias a partir del texto, basadas en lógicas formales, que representen aspectos relevantes del conocimiento descrito en lenguaje natural. Esta afirmación es completamente cierta. En la literatura, existe una inmensa cantidad de técnicas y algoritmos para la representación del conocimiento contenido en textos mediante formalismos lógicos, destacando por su abundancia la extracción de ontologías a partir de textos. Se trata de un área de investigación que sigue siendo muy activa en la actualidad. Por lo general, estas propuestas se centran en un tipo de formalismo concreto, e incluso en la búsqueda de determinadas fórmulas específicas.

La extracción de ontologías ([Maedche and Staab, 2000a]), ([Maedche and Staab, 2001]), ([Buitelaar et al., 2005]), ([Lima et al., 2015]), ([Conde et al., 2015]), ([Rios-Alvarado et al., 2015]), ([Wong et al., 2012]), ([Cimiano, 2006]), ([Biemann, 2005]), ([Lehmann and Voelker, 2014]), ([Hajic et al., 2010]), bases de conocimiento ([Kiddon and Domingos, 2012, Bobrow et al., 2009]), representaciones basadas en *lógica natural* ([MacCartney and Manning, 2009]), ([Andreasen and Nilsson, 2014]), relaciones

causales ([Puente et al., 2012]), ([Puente et al., 2010]), y otros formalismos ([de Paiva et al., 2007]), ([Bobrow et al., 2009]), suele realizarse con distintos objetivos, no solo el uso de textos como fuentes para el modelado del conocimiento básico de un problema, sino como indexación en técnicas de recuperación de información, para facilitar la realización de consultas sobre el contenido del texto ([Sobrino et al., 2014]), para generar resúmenes del texto ([Puente et al., 2015]), etc. También se han utilizado en el ámbito de la minería de textos, pero aplicando técnicas inductivas de minería de datos para determinar patrones de aparición de determinadas piezas de conocimiento (hechos, reglas, etc.) en una colección de textos, considerando la aparición de dichas piezas en el texto como datos ([Mooney and Bunescu, 2005]).

Los avances y la gran cantidad de técnicas y resultados disponibles en este ámbito nos permiten afirmar que es factible contar con formas intermedias de representación del texto que permitan razonamiento sobre conocimiento, proporcionándonos uno de los elementos fundamentales del nuevo paradigma que proponemos.

También podemos afirmar que, mediante el uso de razonamiento sobre estas formas intermedias, es posible obtener nuevo conocimiento que no podría obtenerse mediante técnicas convencionales de minería, es decir, que también se cumple la segunda de las hipótesis en las que se basa nuestro paradigma de minería de textos basada en el conocimiento. Para demostrar este punto, vamos a considerar como ejemplo el caso de un conjunto de seis textos, de cada uno de los cuales se extrae una frase (también podemos considerar el caso de que varias frases estuviesen en el mismo documento y tuviésemos menos documentos):

- Documento 1: Cada persona es mortal
- Documento 2: Nick es una persona
- Documento 3: Aixu es mortal
- Documento 4: No puedo encontrar a una persona que no sea mortal
- Documento 5: Un dios no es persona y no es mortal
- Documento 6: Thor no es mortal

Siguiendo el paradigma basado en conocimiento, podemos utilizar bases de conocimiento expresadas mediante lógica de predicados como forma intermedia para representar el conocimiento contenido en cada una de las frases. En este ejemplo tan sencillo, la base de conocimiento para cada documento consiste en una única sentencia lógica:

- Documento 1: $\forall x\ (persona(x) \rightarrow mortal(x))$
- Documento 2: $persona(Nick)$
- Documento 3: $mortal(Aixu)$
- Documento 4: $\neg\exists x\ (persona(x) \wedge \neg mortal(x))$
- Documento 5: $dios(x) \rightarrow (\neg persona(x) \wedge \neg mortal(x))$
- Documento 6: $\neg mortal(Thor)$

Mediante un proceso de deducción que puede llevarse a cabo de forma automática por alguno de los múltiples algoritmos existentes, es fácil obtener, a partir del conocimiento contenido en estos textos, nuevo conocimiento como por ejemplo:

- $mortal(Nick)$ (Nick es mortal)
- $\neg dios(Nick)$ (Nick no es un dios)
- $\neg dios(Aixu)$ (Aixu no es un dios)

Cada una de estas piezas de conocimiento se han obtenido de forma no trivial (no se trata de una consulta o resumen simple obtenido del texto) y novedosa (en el sentido de que no formaban parte del texto original). Con respecto a su utilidad, dependerá del problema concreto y de las necesidades y conocimientos de la persona destinataria del nuevo conocimiento obtenido, como ocurre siempre en los procesos de descubrimiento de conocimiento.

Pero además, el conocimiento obtenido *no podría haberse obtenido mediante técnicas de razonamiento inductivo a partir de datos* basadas en semejanza y repetición. Por ejemplo, la búsqueda de palabras frecuentes y reglas de asociación entre las mismas habría partido de formas intermedias basadas en bolsas de palabras o secuencias, que en este caso podrían ser:

- Documento 1: $\{persona, mortal\}$
- Documento 2: $\{Nick, persona\}$
- Documento 3: $\{Aixu, mortal\}$
- Documento 4: $\{persona, mortal\}$
- Documento 5: $\{dios, persona, mortal\}$

- Documento 6: $\{Thor, mortal\}$

A partir de esta representación pueden obtenerse *patrones* de frecuencia y asociación de palabras, como por ejemplo que la palabra *persona* aparece en 2/3 de los documentos de la colección, *mortal* en 5/6, y el par $\{persona, mortal\}$ en 1/2, así como que las reglas de asociación $persona \Rightarrow mortal$ y $mortal \Rightarrow persona$ tienen confianza 3/4 y 6/10, respectivamente, teniendo ambas un soporte de 1/2. Estos patrones pueden eventualmente convertirse en conocimiento tras un proceso de interpretación y valoración por parte del ser humano, pero conocimiento de un tipo muy distinto al obtenido mediante razonamiento deductivo. Sin embargo, la pérdida de semántica del texto al utilizar bolsas de palabras, así como el tipo de técnicas de obtención de patrones utilizadas, no permiten en ningún caso obtener que *Aixu no es un dios* como nuevo conocimiento, algo que sí es posible obtener mediante deducción como acabamos de ver. De hecho, no existe ningún documento en el que aparezcan conjuntamente las palabras *Aixu* y *dios*, así como ninguna información que pueda determinar ninguna relación de semejanza bajo la cual se produzca esa aparición conjunta. Lo mismo puede decirse de técnicas que realizan un agrupamiento (clustering) de documentos (y que determinarían que el documento 1 y el 4 deben estar en un mismo agrupamiento, quizá también con el 5, etc.), y otras técnicas basadas en inducción. Por ello, las técnicas basadas en semejanza y frecuencia no pueden obtener el tipo de conocimiento que nuestro paradigma puede proporcionar.

Podemos concluir en este apartado que las dos primeras hipótesis en las que se basa nuestro nuevo paradigma se cumplen, y que la enorme variedad de representaciones y técnicas de razonamiento existentes pueden ser potencialmente utilizadas en el desarrollo de una gran cantidad de técnicas de minería de textos basada en conocimiento.

3.1.2 Modelos CHC

La mayoría de los sistemas inteligentes utilizan razonamiento de tipo deductivo, es decir, obtienen nuevas piezas de conocimiento que cumplen la condición de ser *consecuencia lógica* del conocimiento existente. Este tipo de razonamiento es el que hemos empleado para ilustrar las posibilidades de la minería de datos basada en conocimiento en los ejemplos del apartado anterior.

Sin embargo, es posible llevar a cabo otros tipos de razonamiento sobre representaciones lógicas de conocimiento, lo que amplía aún más las posibilidades de nuestro nuevo paradigma. Estos otros tipos de razonamiento constituyen de hecho la mayor parte del razonamiento que aplica el ser humano de manera cotidiana, junto

al razonamiento por analogía (como caso particular, el razonamiento basado en casos) ([García-Honrado and Trillas, 2012]), y por tanto resultan muy interesantes de cara a obtener nuevo conocimiento a partir de textos de la misma forma en que lo haría un ser humano.

En este apartado vamos a describir la propuesta de formalización de estos tipos de razonamiento, descrita por Trillas y otros en ([Trillas et al., 2000, García-Honrado and Trillas, 2012]), llamada *modelos CHC* (de *Conjeturas, Hipótesis y Consecuencias*) por Qiu ([Qiu, 2007]). En esta propuesta, las sentencias lógicas y proposiciones propias del razonamiento humano se representan mediante elementos de un retículo ortomodular, lo que permite aplicar sus propuestas en distintos tipos de lógicas. Dado que la presente Tesis se centra exclusivamente en lógica de predicados y ontologías formulables como subconjuntos de la misma a través de lógicas descriptivas, y por mayor sencillez en la descripción de estos modelos, vamos a simplificar la formalización restringiéndonos al caso de álgebras de Boole, que son las estructuras adecuadas para formalizar la estructura que forman las sentencias lógicas con respecto a los operadores de conjunción ($\wedge$), disyunción ($\vee$), y negación ($\neg$)[3].

En los modelos CHC, el resultado de un proceso de razonamiento a partir de un cuerpo de conocimiento (base de conocimiento que no contenga una contradicción) se denomina *conjetura* ([Trillas et al., 2000]). Una conjetura puede definirse como una expresión que no es incompatible con el cuerpo de conocimiento existente, y por tanto es *posible* con respecto al mismo. Los modelos CHC distinguen tres tipos de conjeturas:

- *Consecuencias*, expresiones que se derivan como consecuencias lógicas del cuerpo de conocimiento de manera necesaria y segura. El proceso de obtener consecuencias se denomina *razonamiento deductivo*.
- *Hipótesis*, expresiones contingentes que son *explicaciones* tentativas del cuerpo de conocimiento, en el sentido de que todas las expresiones de éste último son consecuencias lógicas de la hipótesis. El proceso de obtener consecuencias se denomina *razonamiento abductivo*.
- *Especulaciones*, aquellas conjeturas que no son consecuencias ni hipótesis. No tienen relación en términos de consecuencia lógica con el cuerpo de conocimiento, siendo su única relación que no son incompatibles con el mismo. Son elucubraciones contingentes con respecto al cuerpo de conocimiento.

[3] Las estructuras algebráicas necesarias para la representación completa de la expresividad de la lógica de predicados de primer orden son las *álgebras cilíndricas*, que son álgebras de Boole respecto a $(\wedge, \vee, \neg)$ a las que se añaden ciertos operadores y axiomas para modelar la cuantificación existencial y la igualdad.

Es importante destacar que las expresiones lógicas no son conjeturas per se, así como cuando son conjeturas no lo son de uno u otro tipo, sino con respecto a un cuerpo de conocimiento dado.

Aquellas expresiones que no son conjeturas se denominan *refutaciones* ([García-Honrado and Trillas, 2012]), y se caracterizan por ser incompatibles con el cuerpo de conocimiento, es decir, la conjunción de las expresiones que forman el cuerpo de conocimiento y una refutación respecto al mismo dan lugar a una *contradicción*, una expresión que es falsa bajo toda interpretación lógica. En los trabajos donde se hace referencia a los modelos CHC, las refutaciones juegan un papel secundario o carecen de interés, asumiéndose aparentemente que tanto en el razonamiento humano como en el automático propio de los sistemas inteligentes, el objetivo es obtener conjeturas a partir de un cuerpo de conocimiento. Sin embargo, como veremos en el capítulo 4, la búsqueda de refutaciones con respecto a un cuerpo de conocimiento (o equivalentemente, contradicciones) es muy interesante como base para el desarrollo de técnicas de minería de textos basada en el conocimiento.

3.1.2.1 Formalización

Sea L un conjunto de expresiones lógicas. Sea $(L, 0, 1, ', +, \cdot)$ un álgebra de Boole, con $0, 1 \in L$ representando *Falso* y *Verdadero* respectivamente, y $'$, $+$ y $\cdot$ las operaciones del álgebra que implementan los operadores lógicos de negación, disyunción y conjunción, respectivamente.

Sea $\leq$ una relación en L tal que $p \leq q$ si y solo si $p \cdot q = p$ o $p + q = q$. La relación $p \leq q$ representa *q se deduce (es consecuencia) de p*. Utilizamos $p < q$ cuando $p \leq q$ y $p \neq q$. Para todo $q \in L$ se verifica $0 \leq q \leq 1$. Aquellos $q \in L$ tales que $0 < q < 1$ se denominan *contingentes*.

Sea $P = \{p_1, \ldots, p_n\} \subset L$ y sea $r(P) = p_1 \cdot p_2 \cdot \ldots \cdot p_n$ una expresión llamada el *resumen* de P. Se dice que $P \subset L$ es un cuerpo de conocimiento, es decir, que no contiene inconsistencias, si $r(P) \neq 0$.

Se dice que $C : \mathcal{P}(L) \to \mathcal{P}(L)$ es un *operador de consecuencias de Tarski consistente* si verifica las siguientes propiedades:

- $P \subset C(P)$ $\forall P \in \mathcal{P}(L)$ (C es *extensivo*).
- Si $P \subset Q$ entonces $C(P) \subset C(Q)$ (C es *monótono*).
- $C(C(P)) = C(P)$ (C es una *clausura*).

- Si $r(P) \neq 0$ y $q \in C(P)$ entonces $q' \notin C(P)$ (C es *consistente*).

Los operadores de consecuencias consistentes son *abstracciones* de procesos deductivos ([García-Honrado and Trillas, 2012]). Es decir, $C(P)$ contiene todas las expresiones que son consecuencia lógica de P (o equivalentemente, de $r(P)$, es decir, $C(P) = C(\{r(P)\})$). Dado que $P \subset C(P)$, si P es inconsistente entonces $C(P)$ también lo es ya que en ese caso $r(P) = 0$ y, como $0 \leq q \; \forall q \in L$, entonces $C(P) = L$. Por tanto, otra forma de establecer que P es consistente (y por tanto un cuerpo de conocimiento) es $C(P) \neq L$. Nótese que L no es un cuerpo de conocimiento porque para todo $q \in L$ se cumple $q' \in L$ y por tanto $r(L) = 0$.

Los conceptos de conjetura y sus distintos tipos, así como el concepto de refutación, pueden formalizarse como sigue[4] ([Trillas et al., 2000, García-Honrado and Trillas, 2012]). Sea P un cuerpo de conocimiento. Podemos definir los siguientes subconjuntos de L relativos a P:

- Conjeturas: $Conj(P) = \{q \in L \mid q' \notin C(P)\}$.
- Consecuencias: $C(P) = \{q \in Conj(P) \mid r(P) \leq q\}$.
- Hipótesis: $Hip(P) = \{q \in Conj(P) \mid q < r(P)\}$.
- Especulaciones: $Esp(P) = \{q \in Conj(P) \mid r(P) \not\leq q \wedge q \not< r(P)\}$.
- Refutaciones: $Ref(P) = \{q \in L \mid q' \in C(P)\}$.

Es evidente que

$$Conj(P) = C(P) \cup Hip(P) \cup Esp(P) \tag{3.1}$$

y $C(P) \cap Hip(P) = C(P) \cap Esp(P) = Hip(P) \cap Esp(P) = \emptyset$, por lo cual consecuencias, hipótesis y especulaciones de P forman una partición de las conjeturas de P. Por otro lado,

$$L = Conj(P) \cup Ref(P) \tag{3.2}$$

con $Conj(P) \cap Ref(P) = \emptyset$, y por tanto conjeturas y refutaciones forman una partición de L. En definitiva, dado un cuerpo de conocimiento P, toda expresión lógica $q \in L$ es necesariamente consecuencia, hipótesis, especulación o refutación con respecto a P, pero solo puede pertenecer a una de estas categorías.

[4] La deducción puede representarse mediante distintos operadores de consecuencias que, en el caso de álgebras de Boole, son equivalentes en el sentido de que proporcionan un mismo resultado. Sin pérdida de generalidad, utilizamos aquí uno de los propuestos en ([Trillas et al., 2000, García-Honrado and Trillas, 2012]).

Un resultado muy interesante que se muestra en ([García-Honrado and Trillas, 2012]) es el siguiente: si $q \in Esp(P)$ entonces $q \cdot r(P) \in Hip(P)$, es decir, es posible obtener una hipótesis mediante la reunión de la evidencia existente y una especulación.

En el contexto de la lógica de predicados existen algoritmos para la determinación automática del conjunto de consecuencias de un cuerpo de conocimiento en un ordenador. Para algunos subconjuntos concretos de la lógica de predicados, como por ejemplo ciertos tipos de lógicas descriptivas utilizadas en la formalización de ontologías y el razonamiento con las mismas, existen algoritmos que siempre emplean un tiempo acotado en ese proceso, aportando la respuesta correcta. Estos procedimientos pueden utilizarse para obtener no solo consecuencias, sino tanto otras conjeturas como refutaciones de un cuerpo de conocimiento P, puesto que éstas pueden reformularse en términos de consecuencias como sigue:

- $Hip(P) = \{q \in L \mid q' \notin C(P) \wedge r(P) \in C(\{q\}) \backslash \{q\}\}$.
- $Esp(P) = \{q \in L \mid q' \notin C(P) \wedge r(P) \notin C(\{q\}) \wedge q \notin C(\{r(P)\})\}$.
- $Ref(P) = \{q \in L \mid q' \in C(P)\}$.

Es decir, es posible computacionalmente diseñar algoritmos de minería de textos que, utilizando formas intermedias basadas en subconjuntos apropiados de lógica de predicados, y algoritmos de cálculo de consecuencias, obtengan nuevo conocimiento en forma de los distintos tipos de conjeturas y refutaciones. En el siguiente apartado mostramos algunos ejemplos para motivar la utilidad potencial de tales técnicas.

3.1.2.2 Ejemplos

En el apartado 3.1.1 hemos visto ejemplos de nuevas piezas de conocimiento obtenidas por razonamiento deductivo a partir del conocimiento contenido en un conjunto de textos. Recordemos el conocimiento de partida expresado mediante lógica de predicados:

- Documento 1: $\forall x \ (persona(x) \rightarrow mortal(x))$
- Documento 2: $persona(Nick)$
- Documento 3: $mortal(Aixu)$
- Documento 4: $\neg\exists x \ (persona(x) \wedge \neg mortal(x))$

- Documento 5: $dios(x) \rightarrow (\neg persona(x) \wedge \neg mortal(x))$
- Documento 6: $\neg mortal(Thor)$

El conocimiento obtenido en el ejemplo pertenece al conjunto de consecuencias de la base de conocimiento formada por la unión de los textos. Si representamos por B_i la base de conocimiento del documento i del ejemplo del apartado 3.1.1, y definimos

$$B = \bigcup_{i=1}^{6} B_i \tag{3.3}$$

entonces, como vimos en el apartado 3.1.1, podemos decir $mortal(Nick) \in C(B)$, $\neg dios(Nick) \in C(B)$, y $\neg dios(Aixu) \in C(B)$. Incluso, más concretamente:

- $mortal(Nick) \in C(B_1 \cup B_2)$
- $\neg dios(Nick) \in C(B_2 \cup B_5)$
- $\neg dios(Aixu) \in C(B_3 \cup B_5)$

Un ejemplo de especulación es $\neg hombre(Aixu) \in Esp(B)$, ya que no está en contradicción con los textos (y por tanto, con B), pero tampoco se deduce de los mismos, ni los textos se deducen de esta pieza de conocimiento.

Podemos obtener un ejemplo de hipótesis aplicando el resultado descrito en ([García-Honrado and Trillas, 2012]) y que hemos apuntado en el apartado anterior: la conjunción de una especulación y el resumen de un cuerpo de conocimiento proporcionan una hipótesis. En este ejemplo, tenemos que $r(B_i)$ está compuesto por la única sentencia presente en B_i, y

$$r(B) = r(B_1) \cdot r(B_2) \cdots r(B_6) \tag{3.4}$$

y es fácil demostrar $((\neg hombre(Aixu)) \cdot r(B)) \in Hip(B)$. También es fácil demostrar

$$q = ((\neg hombre(Aixu)) \cdot r(B_1) \cdot r(B_2) \cdot r(B_5) \cdot r(B_6)) \in Hip(B) \tag{3.5}$$

ya que $r(B_3) \in C(q)$ y $r(B_4) \in C(q)$. Otro ejemplo de hipótesis es $(dios(Thor) \cdot r(B_5)) \in Hip(B_6)$.

Como ya hemos indicado anteriormente, una expresión no es una consecuencia, hipótesis o especulación per se, sino con respecto a un cuerpo de conocimiento. Y con respecto a un cuerpo de conocimiento, como también hemos visto, solo puede ser consecuencia, hipótesis, especulación o refutación, y solo una de ellas. Por ejemplo, $mortal(Nick) \in C(B_1 \cup B_2)$ pero $mortal(Nick) \in Esp(B_6)$.

Un ejemplo de refutación es $persona(Thor) \in Ref(B)$. También tenemos que $persona(Thor) \in Ref(B_1 \cup B_6)$. Sin embargo, $persona(Thor) \in Esp(B_2)$.

Finalmente, queremos destacar una vez más que la extracción de estos diversos tipos de conjeturas y refutaciones no se realiza por procedimientos basados en similitud y repetición, sino mediante la aplicación de procesos de razonamiento basados en deducciones lógicas.

3.1.2.3 Algunas Aplicaciones Potenciales

Existe una enorme cantidad de aplicaciones potenciales de la minería de textos basada en el uso de conjeturas y refutaciones:

- La búsqueda de consecuencias tiene muchas aplicaciones potenciales, pero a modo de ejemplo podemos citar la búsqueda de textos redundantes en una colección de textos. En ([Everett et al., 2002]) se cita el ejemplo concreto de la colección de textos contenida en el sistema *Eureka* de Xerox. Se trata de alrededor de 40,000 textos libres creados por técnicos que contienen pistas y claves que cubren aspectos que no aparecen en los manuales oficiales de la empresa. El objetivo, tal y como se indica en ([Everett et al., 2002]), es identificar textos o partes de los mismos que contienen la misma información, o que expanden la información dada por otros. Es decir, en el caso de documentos completos, $C(B_i) \subset C(B_j)$ ó $C(B_i) = C(B_j)$. Esto ocurre por ejemplo en el caso de los documentos 1 y 4 del ejemplo del apartado 3.1.1, en el que ambas sentencias son lógicamente equivalentes, y por tanto ambos documentos (aunque mediante expresiones textuales distintas) son redundantes.

  Esta información resulta de mucha utilidad a la hora de organizar los textos para su consulta, para determinar distintas opiniones de distintos técnicos de la empresa o la variación de dichas opiniones con el tiempo, obtener resúmenes de los textos relativos a una misma cuestión con la misma información (ahorrando así al usuario de Eureka la necesidad de leer muchos textos redundantes), etc.

Este ejemplo puede generalizarse al caso de los foros en Internet, que contienen textos en los que se plantean problemas en el uso de aplicaciones, dudas de tipo médico, legal, consejos en distintos ámbitos, y un largo etc.

- En el mismo contexto, la búsqueda de refutaciones resulta de mucha utilidad para detectar inconsistencias y contradicciones entre textos, que también pueden facilitar enormemente la gestión de la colección de textos y su explotación ([Everett et al., 2002]). La búsqueda de contradicciones es también particularmente interesante en el ámbito del *análisis de sentimiento* (sentiment analysis), en el cual la detección de opiniones discrepantes sobre un determinado tema en foros, noticias, textos científicos o divulgativos, etc. se considera una forma de *diversidad de sentimiento*, que puede deberse a que un mismo autor presenta opiniones divergentes sobre un tema, o a que diversos autores expresan distintas opiniones en varios textos ([Tsytsarau and Palpanas, 2011, Kim and Zhai, 2009]). Al igual que en el caso de las consecuencias, también puede utilizarse para analizar la evolución temporal de las opiniones de un autor sobre un tema, por ejemplo si en un momento dado la opinión del autor coincide con la de otro autor, para después ser contradictorias, etc.

  Entre las muchas aplicaciones de la búsqueda de contradicciones en textos destaca desde hace muchos años el caso de textos legales. Cuando se dispone de un corpus legal compuesto por una enorme cantidad de leyes, que además están en continuo cambio, no es infrecuente encontrar contradicciones que, lógicamente, deben ser resueltas ([Caracciolo, 1979]). No solamente contradicciones entre lo dispuesto por una ley y otra, sino entre lo dispuesto por el corpus legal y los objetivos del mismo (es decir, tratar de evitar el viejo dicho de *hecha la ley, hecha la trampa*). Otra aplicación potencial es la verificación formal de procedimientos de reparación y mantenimiento de sistemas (por ejemplo, aviones) y configuraciones (por ejemplo, en función de los sistemas disponibles en la red de ferrocarril ante fallos, determinar la seguridad de circular, etc.). Todos ellos pueden basarse en los manuales y protocolos existentes en formato textual.

  Otras aplicaciones posibles incluyen el destacar temas en los que participantes en un debate mantienen posiciones conflictivas (por ejemplo, en debates políticos), detectar información inconsistente en informes de inteligencia y otros informes internos en organizaciones, detectar conflictos en los efectos de la aplicación de medicamentos o en la interacción entre proteínas en bioinformática, etc. ([de Marneffe et al., 2008]).

- El descubrimiento de hipótesis que *expliquen* la información disponible tiene también muchas aplicaciones potenciales. Uno de los ejemplos más claros es la generación de hipótesis en el ámbito científico. En el ámbito de la minería de datos existe desde hace pocos años un subárea de investigación denomi-

nada *Discovery Science* (DS) que se define como "la disciplina que estudia el desarrollo, análisis y aplicación de métodos y herramientas computacionales que den soporte al descubrimiento automático o semi automático de conocimiento en campos de la ciencia como medicina, ciencias naturales y sociales ([Fürnkranz and Hüllermeier, 2016]).

DS sobre datos utiliza, como es natural, técnicas de ámbitos como el aprendizaje automático y minería de datos, análisis inteligente de datos, estadística, optimización, algoritmos y complejidad, bases de datos y sistemas de información. Además de centrarse en el valor científico de los resultados obtenidos, y no solo en el valor económico habitual en el ámbito de la minería, una particularidad fundamental del DS es que al contrario de lo que ocurre en el análisis estadístico, en el cual se utilizan datos y diseño de experimentos para verificar la validez de hipótesis planteadas por los investigadores, *el DS tiene como objetivo el descubrimiento de las hipótesis en si*, y por tanto se centra en el estudio de los procesos de generación de hipótesis y su aplicación práctica ([Fürnkranz and Hüllermeier, 2016]).

Aunque muy anteriores a la aparición del concepto de DS, existen precedentes del uso de conocimiento presente en textos para generar nuevas hipótesis. Un ejemplo de esta visión la podemos encontrar en los trabajos de Swanson y Smalheiser ([Swanson and Smalheiser, 1999]), plasmados en el proyecto Arrowsmith, que trata de extraer nuevas hipótesis a partir de relaciones causales presentes en la literatura médica (aunque lo realiza mediante la frecuencia de cadenas que puedan enlazar hipótesis y conclusión). Su sistema es capaz de generar hipótesis no planteadas previamente mediante el análisis de títulos de artículos biomédicos. Por ejemplo, Swanson extrajo los siguientes textos de títulos de artículos de la literatura biomédica sobre migrañas y elementos asociados:

- El estrés está asociado con las migrañas
- El estrés puede provocar pérdida de magnesio
- Los bloqueadores de los canales de calcio previenen algunas migrañas
- El magnesio es un bloqueador natural de los canales de calcio
- La difusión de la depresión cortical (SCD) está implicada en algunas migrañas
- Niveles altos de magnesio inhiben el SCD
- Pacientes con migraña tiene una alta agregación plaquetaria
- El magnesio puede suprimir la agregación plaquetaria

Estas pistas sugieren que un déficit de magnesio puede jugar un papel importante en algunos tipos de migrañas, una hipótesis que no existía hasta ese

momento y que los autores pudieron derivar de forma parcialmente automática, demostrándose posteriormente de manera experimental.

Es destacable en este punto que, aunque permiten modelar de manera general procesos de razonamiento ordinario, buena parte de la motivación del desarrollo de los modelos CHC ha tenido como objetivo la formalización del proceso de descubrimiento científico. Como se apunta en ([García-Honrado and Trillas, 2012]),

> *Hence, the real process to built up science's models consists on working with conjectures. That is, building up possible explanations (conjectures called hypotheses) from observations, that can change with new observations. Then, after deducing some necessary consequences of the hypothesis, they must be checked by repeated experiments to test its suitability.*

La generación de hipótesis, es decir, el razonamiento abductivo, también es típico de las tareas de diagnóstico, como el diagnóstico médico, la búsqueda de causas para fallos en sistemas computacionales o redes, la investigación criminal o de accidentes, etc. Por ejemplo, si tenemos un texto que indica que *Juan tiene fiebre*, y nuestro conocimiento del mundo nos dice que *La gripe causa fiebre*, podemos mediante razonamiento abductivo generar la hipótesis *Juan tiene gripe* (en realidad, la hipótesis es la conjunción de *La gripe causa fiebre* y *Juan tiene gripe*, pero como la primera parte forma parte del conocimiento previo contrastado, se suele llamar hipótesis a aquella parte de la conjunción de cuya veracidad no estamos seguros). Como ya se ha indicado anteriormente, las hipótesis son contingentes, y deben ser verificadas posteriormente, contrariamente a las consecuencias, cuya veracidad deriva de la veracidad del conocimiento de partida.

Además, es posible en general obtener muchas hipótesis diversas que expliquen la misma información. En el ejemplo anterior, si nuestro conocimiento previo nos dice también que *Las infecciones causan fiebre*, tendremos que *Juan tiene una infección* es asimismo una hipótesis. Un ejemplo de ello es la gran cantidad de hipótesis que existen en la literatura en relación a las causas de la muerte de Alejandro Magno en 323 a.C. Los últimos días de su vida, así como la evolución de su estado de salud y los síntomas que manifestaba, están ampliamente descritos en los textos del historiador griego Diodoro. En base a distintos resultados publicados en la literatura médica y de otros ámbitos científicos, distintos autores han atribuido la muerte de Alejandro a la malaria, la fiebre tifoidea, el paludismo, el virus del Nilo, neumonía, leucemia, envenenamiento por eléboro blanco, pancreatitis aguda, y un largo etc. Estas hipótesis, todas ellas publicadas, podrían haber sido potencialmente generadas con ayuda de técnicas de minería de textos basada en conocimiento a partir de los textos

anteriormente históricos y científicos mencionados, con un interés puramente histórico ya que nunca podrán ser comprobadas.

- Finalmente, en lo que concierne a las especulaciones, de nuevo es posible encontrar múltiples aplicaciones. Por ejemplo, en procesos de argumentación como debates políticos, etc. es muy habitual encontrar razonamientos que son realmente falacias, es decir, argumentos que parecen válidos pero realmente no lo son. Un ejemplo es la falacia conocida como *post hoc ergo propter hoc* ("después de esto, luego a consecuencia de esto"), que asume que si un acontecimiento sucede después de otro, el segundo es consecuencia del primero, cuando probablemente el segundo con respecto al primero es simplemente una especulación. Otro caso de falacia habitual es el *argumento ad consequentiam*, que concluye que una premisa es verdadera o falsa en función de si la consecuencia es más o menos beneficiosa. Por ejemplo, cuando se afirma que *el líder del partido no ha llevado a cabo conductas ilícitas porque, en ese caso, no le habrían votado.* O el *argumento ad baculum*, donde se usa una amenaza para justificar una conclusión, como por ejemplo *si nuestro partido no gana las elecciones, el país volverá a hundirse económicamente, por tanto hay que votar a nuestro partido*, o *Iraq tiene armas de destrucción masiva con las que pueden atacarnos, por lo que es necesario entrar en guerra con ellos.*

  Adicionalmente, tal y como se indicó anteriormente y se detalla en ([García-Honrado and Trillas, 2012]), las especulaciones son una herramienta muy útil para la generación de hipótesis, mediante la conjunción de la especulación con todo o parte del conocimiento existente. También permiten determinar que dos textos, aunque puedan versar sobre la misma temática, se refieren a aspectos que no entran en conflicto entre sí, y entre los cuales no existe ninguna relación de consecuencia lógica.

En este apartado hemos visto una serie de ejemplos (no una descripción exhaustiva) para demostrar que la minería de textos basada en conocimiento, concretamente en lógicas y modelos CHC, tiene muchas aplicaciones potenciales interesantes. En el siguiente apartado vamos a centrarnos en los aspectos técnicos del desarrollo de mecanismos de minería de textos basados en estos modelos, y las particularidades que presentan en relación al desarrollo de técnicas de minería de datos o minería de textos convencional.

3.2 El Proceso de Minería de Textos basada en *Conocimiento*

El proceso de minería de textos basada en conocimiento consta en general de las siguientes fases:

1. Formular un objetivo del proceso en términos del dominio de la aplicación y determinar la técnica de minería más apropiada para obtener conocimiento que permita responder a la cuestión planteada.
2. Recopilar el conjunto de textos adecuado.
3. Extraer las formas intermedias apropiadas para la técnica y estructurarlas.
4. Aplicar el algoritmo de minería elegido.
5. Visualizar, interpretar y validar los resultados.
6. Uso y mantenimiento del conocimiento descubierto.

Como puede verse, la estructura del proceso de minería de textos basada en conocimiento no se diferencia significativamente de la que hemos descrito para la minería de textos convencional en el capítulo 2. La principal diferencia se encuentra en el tipo de tareas que se llevan a cabo en cada fase y las herramientas a utilizar en cada una de ellas. El proceso de minería de textos bajo el nuevo paradigma que proponemos plantea nuevos desafíos, que vamos a estudiar en los siguientes apartados.

3.2.1 Formulación del Objetivo y Selección de Técnicas

El proceso de minería de textos debe ir siempre guiado por un objetivo, habitualmente planteado por una cuestión general, y la elección de técnicas concretas que permitan obtener conocimiento que sirva para responder a la cuestión y por tanto cumplir el objetivo. Ejemplos de objetivos posibles serían:

- ¿Qué grupos de autores podemos determinar en un blog en función de su opinión acerca de un tema concreto?
- ¿Qué posibles hipótesis a investigar podemos plantear en cuanto a las causas o factores que propician la enfermedad X, en base a la literatura?

- ¿Garantizan los procedimientos descritos en los manuales para la reparación de averías la seguridad?
- La nueva normativa que hemos planteado, ¿entra en conflicto con el resto de normas existentes?
- ¿Cómo pueden agruparse los textos en la literatura en relación a su opinión sobre un conjunto concreto de temas?
- ¿Qué nuevo conocimiento puede derivarse de la información contenida en los textos más recientes sobre un tema concreto?

Para resolver estas y otras cuestiones, podemos utilizar distintos tipos de técnicas:

- Técnicas que deriven nuevas piezas de conocimiento a partir del existente. Se trata de obtener piezas de conocimiento que no estén contenidas en ninguno de los textos, y que guarden una relación determinada con el conocimiento contenido en los textos (conjetura, refutación, etc.)
- Técnicas que determinen la relación entre el conocimiento contenido en un texto con respecto a otros, sin derivar nuevas piezas de conocimiento. Por ejemplo, determinar que una pieza de conocimiento de un documento es consecuencia del conocimiento contenido en otros dos, o hipótesis de parte del conocimiento contenido en otro documento, etc.
- Técnicas que realicen agrupamiento de textos en función de la relación entre los mismos (agrupar todos los textos que son coherentes en su contenido, etc.).
- Técnicas que añaden a las anteriores información adicional de los textos (autor, fecha, etc.) para encontrar relaciones entre autores, determinar *escuelas* de opinión, determinar la evolución temporal de todo lo anterior, etc.

Esta relación de posibles tipos de técnicas no es exhaustiva, pero cubre muchos de los posibles procesos de minería de textos basada en conocimiento.

3.2.2 Recopilación de Textos

La recopilación de los textos es una tarea clave, porque de la representatividad y relevancia de los mismos con respecto al objetivo planteado dependerá la calidad de los resultados obtenidos. Como ocurre siempre en un proceso de minería, los

resultados obtenidos son veraces en relación a los textos obtenidos, y su veracidad con respecto a la parcela de la realidad de nuestro interés dependerá de lo representativos que sean los textos con respecto a dicha realidad. Asimismo, la realidad de nuestro interés presenta múltiples aspectos y facetas, por lo que para que el conocimiento obtenido sea útil será necesario que los documentos sean relevantes a aquella parte de la realidad relacionada con el objetivo planteado.

A pesar de su importancia, la recopilación de textos no es en general la fase del proceso de minería de textos que recibe más atención en la literatura o en la práctica. Por lo general, el conjunto de textos se asume como dado a priori, o bien se utiliza solamente para demostrar experimentalmente cuestiones de eficiencia de nuevas técnicas o algoritmos, o novedades en el tipo de patrones de conocimiento generados. En lo que respecta a la representatividad y relevancia, la cuestión se resume en indicar que se utilizarán técnicas de recuperación de información, que es el área que se centra en el estudio de la selección de documentos apropiados a una cierta consulta. Sin embargo, hay muy poca información en la literatura acerca de, dado un objetivo concreto del proceso de minería de textos, cómo plantear las consultas adecuadas a un sistema de recuperación de información para obtener el conjunto de documentos más apropiado para ese objetivo.

En el contexto del nuevo paradigma que proponemos, la minería de datos basada en conocimiento, existe un problema adicional: los sistemas de recuperación de información suelen basar la indexación de los documentos en representaciones cercanas a las formas intermedias utilizadas en la minería de textos basada en datos (bolsas de términos, etc.), y por tanto la información de la que disponen acerca del conocimiento contenido en textos es limitada. Usando recuperación de información podemos asumir razonablemente la temática de un documento hasta cierto punto, pero no podemos garantizar que contenga conocimiento interesante.

Este último aspecto entronca con otro tema importante: al contrario que ocurre con los sistemas de minería de textos basados en datos, donde la frecuencia de repetición es un aspecto clave, el número de documentos de los que se dispone es menos importante que la cantidad y calidad del conocimiento contenido en los mismos. Por ejemplo, para deducir una nueva pieza de conocimiento no es necesario que ésta se repita en una gran cantidad de documentos, simplemente necesitamos que el cuerpo de conocimiento del que podemos derivar esa nueva pieza (como conjetura o refutación) esté contenido en el conjunto de textos, siendo posible obtener nuevo conocimiento incluso de un solo texto breve, como ya vimos mediante ejemplos sencillos en este mismo capítulo. Por supuesto, algunas aplicaciones (como la mencionada anteriormente en relación al sistema Eureka) puede tomar como punto de partida un conjunto de documentos prefijado, e incluso de un tamaño considerable, pero en líneas generales y como ocurre en cualquier proceso de minería, es

interesante procurar partir de la cantidad más pequeña posible de textos que permitan alcanzar los objetivos del proceso. Por supuesto, determinar el conjunto más pequeño de textos que cumpla esta condición no es nada trivial, y constituye uno de los desafíos más importantes a nuestro juicio para la investigación en este área.

El hecho de que la minería de textos basada en conocimiento no requiera un gran número de textos, dado que no se basa en repetición de la presencia de patrones en una gran cantidad de casos, puede llevar a pensar que solo es adecuada para problemas de menor *tamaño* o complejidad. Sin embargo, esto no es cierto, dado que lo que define realmente la complejidad del proceso de minería de textos basada en conocimiento no es el número de documentos, sino el número de piezas de conocimiento, de la misma forma que en el caso de la búsqueda de reglas de asociación en una base de datos la complejidad no reside en el número de transacciones, sino en el número de items. De esta forma, un proceso de minería de textos basada en conocimiento sobre 100 documentos que contengan 1000 piezas de conocimiento tiene una complejidad mucho mayor que un proceso de búsqueda de reglas de asociación sobre una base de datos de un millón de transacciones definidas sobre un conjunto de 100 items. Trataremos este tema con mayor detalle en el apartado 3.2.4.

Otro aspecto particularmente importante en el nuevo paradigma que proponemos es que el conocimiento sobre el contexto no es solo conveniente, sino necesario. Como ya hemos comentado, la interpretación de la semántica de los textos se realiza en términos de un conocimiento previo de contexto, relativo al ámbito de la realidad donde se enmarcan los textos, conocimiento previo que puede ir desde conocimiento muy general sobre el mundo (por ejemplo, leyes físicas), a conocimiento más específico (reglas internas de negocio de la empresa). Podemos ilustrar este punto con un ejemplo muy sencillo: si al ejemplo del apartado 3.1.1 añadimos *Jack es el padre de Nick*, no podemos deducir nuevo conocimiento. Sin embargo, si añadimos también el conocimiento contextual *El padre de una persona es una persona*, entonces podremos deducir *Jack es una persona*, *Jack es mortal*, *Jack no es un dios*, etc.

Los procesos de minería de datos y textos pueden beneficiarse mucho del uso del conocimiento contextual. Una ventaja importante del paradigma que proponemos es que el conocimiento de contexto puede integrarse con el conocimiento contenido en los textos y ser utilizado de la misma forma, como acabamos de ver, algo que resulta más complejo de realizar en procesos de minería basados en datos. Además, el conocimiento textual puede tener un mayor impacto ya que, como acabamos asimismo de ver, el añadir una sola pieza de conocimiento contextual puede permitir la generación de muchas nuevas piezas de conocimiento. Por ello, es interesante explotar estas posibilidades y dedicar esfuerzo a la recopilación de conocimiento contextual en esta primera fase.

Afortunadamente, el conocimiento propio del ámbito particular donde se plantea el objetivo de la minería suele poder encontrarse en forma de textos (normas de empresa, principios de actuación, líneas editoriales, etc.) que pueden añadirse al conjunto de textos sobre los que realizar el proceso de minería, e incluso en ocasiones es posible encontrar el conocimiento contextual representado mediante alguna forma intermedia apropiada para el razonamiento, particularmente ontologías. Esto mismo ocurre también con el conocimiento más general, donde existen proyectos que han realizado un gran esfuerzo en la recopilación de razonamiento de sentido común (*commonsense reasoning*) utilizando ontologías, tales como el proyecto CYC ([Lenat, 1995]), comenzado en 1984 y cuya última versión, lanzada en 2012, reúne alrededor de 239,000 términos, 2,093,000 axiomas, y 69,000 enlaces *owl:sameAs* a otros recursos sobre conceptos como DBpedia, UMBEL, WordNet, Wikicompany, CIA World Factbook, RDFAbout SEC, RDFAbout y FOAF, como se indica en la web del proyecto[5]. Este conocimiento (o al menos aquel subconjunto que pueda ser relevante para los objetivos del problema) puede asimismo incorporarse y utilizarse directamente en el proceso de minería de textos basada en conocimiento.

3.2.3 Extracción de Formas Intermedias

Una vez determinado el conjunto de textos para responder al objetivo planteado, el siguiente paso es determinar la forma intermedia adecuada para obtener el tipo de conocimiento deseado. Un requisito fundamental en este paso es disponer de medios automáticos o semiautomáticos para la extracción del tipo de forma intermedia elegida a partir de los textos. Como ya vimos en el apartado 3.1.1, en la literatura existen muchas herramientas y resultados que permiten extraer representaciones del contenido del texto utilizando lógicas de diversos tipos, que son los formalismos en los que nos centramos en la presente Tesis.

De entre todas las propuestas existentes, el uso de ontologías (basadas en lógicas descriptivas, básicamente subconjuntos de la lógica de predicados de primer orden) es particularmente interesante por varios motivos:

- Existe una gran cantidad de técnicas y herramientas para la extracción de conocimiento expresado mediante ontologías a partir de textos.
- Es posible encontrar bases de conocimiento representadas mediante ontologías que contienen conocimiento de contexto, tanto general (por ejemplo el sistema CYC mencionado en el apartado anterior), como de dominio específico, estas últimas accesibles a través de herramientas de búsqueda como por ejemplo

[5] `http://www.opencyc.org/`

Swoogle[6], o bien diseñadas expresamente. De esta forma, es posible integrar de manera más sencilla el conocimiento contenido en los textos y el conocimiento de contexto, a la vez que cada vez más conocimiento de contexto está disponible.

- Existen diversos estándares y herramientas ampliamente utilizadas para el diseño y manejo de ontologías, incluyendo herramientas muy importantes para nuestros propósitos como herramientas de integración de ontologías, razonadores eficientes, etc.

Es necesario reiterar que en el proceso de generación de formas intermedias siempre se pierde información con respecto al texto original. Sin embargo, los procesos de minería de textos no necesitan ser exhaustivos, ni en la extracción de conocimiento de los textos ni en la generación de nuevo conocimiento, para tener éxito. Como ya hemos apuntado anteriormente, la obtención de una sola pieza de conocimiento novedoso y potencialmente útil se considera un éxito. Lógicamente, cuanto menos información se pierda, más probabilidades de obtener resultados importantes.

La extracción de formas intermedias no se limita a tomar cada texto individual proporcionado y extraer a partir del mismo una ontología utilizando las técnicas existentes. Hay pasos adicionales fundamentales:

1. En primer lugar, es necesario poder integrar adecuadamente las distintas ontologías obtenidas, proceso conocido como *ontology alignment* o también *ontology matching*. Por ejemplo, debido a las características del texto en sí, es posible que haya distintas representaciones en distintas ontologías para un mismo concepto del dominio del problema, lo que puede impedir realizar razonamiento con piezas de conocimiento procedentes de dos textos distintos. También es necesario integrar las ontologías procedentes de los textos con las ontologías que contienen el conocimiento previo de contexto. Como hemos indicado anteriormente, existen herramientas y una gran cantidad de resultados de investigación en este ámbito, que sigue siendo activo hoy en día.
2. En segundo lugar, es necesario poder estructurar adecuadamente todo el conocimiento extraido de los textos y el conocimiento previo. Habitualmente, el texto se considera estructurado en un corpus o colección de documentos, y el conocimiento obtenido puede heredar esa estructura considerando que de cada texto se obtiene una base de conocimiento. Tendríamos pues una colección de bases de conocimiento, incluyendo el conocimiento previo, que es la entrada a los algoritmos de minería de textos basada en conocimiento. Por ejemplo,

[6] `http://swoogle.umbc.edu/`

para el caso de los textos del sistema Eureka descrito en apartados anteriores, las entradas del blog constituyen la colección de textos, obteniéndose una base de conocimiento para cada entrada.

Sin embargo, esta estructuración *natural* puede no ser la más adecuada según el problema en muchos casos:

- En ocasiones, un documento puede contener diversas bases de conocimiento. Por ejemplo, un documento en el que se transcribe un debate entre varias personas puede separarse en varias bases de conocimiento, cada una de ellas correspondiente a las opiniones de una de las personas, cuando el objetivo es la minería de sentimiento ó determinar la coherencia o discrepancia entre las opiniones de distintos autores.
- La búsqueda de conjeturas y refutaciones requiere cuerpos de conocimiento, es decir, bases de conocimiento consistentes (que no contengan contradicciones). Cuando el conocimiento contenido en un documento no es consistente, tenemos pues dos opciones: o bien descartarlo, o bien dividirlo en varias bases de conocimiento que sí sean consistentes, utilizando algún criterio apropiado al objetivo del proceso de minería.
- De la misma forma, en ocasiones puede ser conveniente integrar las bases de conocimiento obtenidas de distintos textos en una sola base de conocimiento, siempre que ésta sea consistente y que contenga un conocimiento *homogéneo* con respecto a algún criterio relevante para el objetivo del problema. En relación con esto, también es posible considerar agrupaciones jerárquicas de bases de conocimiento. Por ejemplo, un conjunto de bases de conocimiento etiquetadas por (autor, tema, época) puede agruparse jerárquicamente para dar una sola base de conocimiento por par (autor, tema), o (autor, época), exclusivamente por autor, etc.

3. En tercer lugar, es conveniente tratar de eliminar en la medida de lo posible conocimiento que no contribuya al objetivo final, dado que la complejidad del proceso de minería, como veremos, depende fundamentalmente del número de piezas de conocimiento existentes. Por ejemplo, si tenemos una base de conocimiento de contexto, eliminar de dicha base todo el conocimiento que no tenga relación concreta con el conocimiento contenido en los textos seleccionados. También es posible eliminar redundancias dentro de un mismo texto (es decir, varias piezas de conocimiento que expresan la misma información), eliminar errores en los textos o en su traducción a forma intermedia que pueden provocar inconsistencias, etc. Esta idea puede llevarse más lejos en algunos casos, considerando redundancias e inconsistencias en más de un texto. Aunque esto último es ya propiamente una tarea de minería de textos basada en conocimiento (por ejemplo, la determinación de que dos bases de conocimiento son

equivalentes, es decir, que expresan la misma información), también puede utilizarse como paso previo para otras tareas, de la misma forma que en el caso de minería basada en datos, el agrupamiento (clustering) de datos es una técnica de minería, pero también puede emplearse como paso previo a la aplicación de otras técnicas.

Desde el punto de vista de la estructuración el caso más simple, particularmente cuando los distintos textos no están caracterizados de manera relevante para el problema (por autor, temática, fecha, etc.) y el objetivo del proceso de minería es muy inespecífico (encontrar todo lo que se pueda), es considerar el conjunto completo de piezas de conocimiento dado por todos los textos, e ir construyendo bases de conocimiento como los posibles subconjuntos de piezas de conocimiento del problema. Para cada par de subconjuntos A y B que sean cuerpos de conocimiento, y por tanto consistentes, el proceso de minería trataría de determinar si la conjunción de las piezas de conocimiento contenidas en A es consecuencia, hipótesis, especulación o refutación con respecto a la base de conocimiento B. En este caso, la estructuración no viene dada a priori, sino que las estructuras se construyen durante el proceso de minería.

Como vemos, existe una amplia casuística (que no hemos descrito de forma exhaustiva), que demuestra que no solamente la elección de la forma intermedia y la extracción de la misma, sino también la estructuración del conocimiento de los textos en bases de conocimiento, es un aspecto clave de la minería de textos basada en conocimiento.

3.2.4 Algoritmos

Como hemos comentado anteriormente, es posible concebir una gran cantidad de técnicas de minería de textos basada en conocimiento. Cada una de esas técnicas requerirá un algoritmo para la extracción de nuevo conocimiento. De manera general, dada una colección de cuerpos de conocimiento, el objetivo de los algoritmos de extracción de nuevo conocimiento es generar expresiones del tipo

A es X de B

donde A y B son cuerpos de conocimiento con $A \not\subseteq B$ y $B \not\subseteq A$, y X representa *consecuencia, hipótesis, especulación o refutación.* El significado de estas expresiones puede formalizarse como sigue:

- *A es consecuencia de B* si y solo si $r(A) \in C(B)$.
- *A es hipótesis de B* si y solo si $r(A) \in Hip(B)$.
- *A es especulación de B* si y solo si $r(A) \in Esp(B)$.
- *A es refutación de B* si y solo si $r(A) \in Ref(B)$.

donde $r(A)$ es la conjunción de todas las piezas de conocimiento de A, como vimos en el apartado 3.1.2.1. Por lo visto en el mismo apartado, una y solo una de las cuatro expresiones anteriores se cumple para cada par (A, B) de cuerpos de conocimiento. Como caso particular tenemos que A y/o B sean singletons, es decir, que contengan una sola pieza de conocimiento. De esta forma, el hecho de que una sentencia lógica q tenga relación X con respecto a un cuerpo de conocimiento B puede expresarse como

$\{q\}$ *es X respecto a B*

o por simplicidad *q es X respecto a B*. También como caso particular podemos extraer expresiones del tipo

A y B son contradictorios

con el significado *r(A) es una refutación de B* (equivalentemente, *r(B) es una refutación de A*), o expresiones del tipo

A y B son equivalentes

con el significado *r(A) es consecuencia de B y r(B) es consecuencia de A*.

Como ya hemos indicado, este tipo de expresiones pueden generarse considerando cuerpos de conocimiento para formar A y B de distintas formas:

- Considerando exclusivamente un conjunto de cuerpos de conocimiento prefijados, proporcionados en la fase de estructuración de las formas intermedias del texto
- Tomando subconjuntos de dichas bases de conocimiento elegidos por algún criterio relevante al objetivo del problema (en particular, piezas individuales) que no formen parte de la estructura inicial.

- Uniendo cuerpos de conocimiento existentes que no sean inconsistentes entre sí.
- Siendo $A = \{q\}$ una nueva pieza de conocimiento generada mediante razonamiento a partir de B.

Es posible determinar de manera automática si expresiones del tipo *q es X respecto a B* (las más generales) se cumplen mediante algoritmos de generación de consecuencias, ya que tanto los distintos tipos de conjeturas como las refutaciones pueden expresarse en función de la pertenencia de determinadas piezas de conocimiento al conjunto de consecuencias de un cuerpo de conocimiento, como ya comentamos en el apartado 3.1.2.1. Los sistemas de representación y manejo de conocimiento basados en ontologías disponen, para ciertas expresividades restringidas, de razonadores decidibles, que permiten calcular el conjunto de consecuencias de un cuerpo de conocimiento, proporcionando siempre una respuesta correcta en un tiempo finito. De hecho, se ha llevado a cabo un gran esfuerzo de investigación y desarrollo por parte de los investigadores en el ámbito de las ontologías para obtener razonadores cada vez más eficientes en tiempo y espacio. Por ello, es posible construir algoritmos, basados en la tecnología existente, para la extracción de nuevo conocimiento en forma de conjeturas y refutaciones a partir de las ontologías extraidas de los textos.

Un aspecto clave en el desarrollo de algoritmos de minería, y que no lo es menos en el nuevo paradigma de minería de textos basada en conocimiento, es sin duda la complejidad computacional del proceso. Aunque en algunas técnicas el resultado final no sean conjeturas y refutaciones, sino que éstas se utilicen para obtener otro tipo de información (por ejemplo, agrupamiento de textos), el proceso computacionalmente más costoso es la generación de todas las conjeturas y/o refutaciones. En el peor caso, y empleando un algoritmo de exploración exhaustivo, la complejidad de la extracción de conjeturas y refutaciones depende del número de piezas de conocimiento y de la complejidad del algoritmo de razonamiento. Supongamos que tenemos n piezas de conocimiento en total. El número de bases de conocimiento posible es del orden de 2^n (incluyendo singletons correspondientes a piezas de conocimiento individuales). Por tanto, el número de posibles pares de bases de conocimiento es

$$N = \left(\begin{array}{c} 2^n \\ 2 \end{array} \right) = \frac{(2^n)!}{(2^n-2)!2!} = \frac{1}{2}(2^n \times (2^n-1)) = \frac{1}{2}(2^{2n} - 2^n) \qquad (3.6)$$

Es decir, el número potencial de expresiones del tipo *A es X de B* es del orden

$O(2^{2n})$. Este resultado indica claramente que, incluso con un conjunto de textos aparentemente pequeño en comparación con las colecciones empleadas en minería de textos basada en datos, es posible encontrarnos con un problema de alta complejidad computacional, típica de los procesos de minería.

En cuanto a la complejidad del algoritmo de razonamiento, depende del tipo de lógica descriptiva en la que se basa la representación de la ontología. Es conocido que la complejidad del razonamiento va ligada inversamente a la expresividad de la lógica. En muchos casos, como por ejemplo en OWL DL, uno de los estándares más utilizados y basado en la lógica descriptiva $\mathcal{SHOIN}(\mathcal{D})$, la complejidad de los algoritmos de razonamiento pertenece a la clase EXPTIME, es decir, $O(2^{p(m)})$ siendo $p(m)$ una función polinomial sobre m y siendo m el tamaño de la base de conocimiento. En el caso que acabamos de ver el tamaño de la base de conocimiento con la que hay que razonar para valorar una expresión tipo *A es X de B* es $|A \cup B|$ y dado que no consideramos los casos triviales al imponer $A \not\subseteq B$ y $B \not\subseteq A$, tenemos que en el peor caso $m = n - 2$. Dependiendo del tipo de conjetura, hay que realizar entre uno y tres procesos de razonamiento por cada expresión, lo cual no afecta al orden de complejidad. Por tanto, un algoritmo de fuerza bruta (exploración exhaustiva) tendría como complejidad de peor caso

$$O(N \times 2^{p(n)}) = O(2^{2n} \times 2^{p(n)}) = O(2^{2n+p(n)}) \tag{3.7}$$

es decir, EXPTIME en el número de piezas de conocimiento. Sin embargo, en la práctica, es posible diseñar algoritmos cuya complejidad va a estar muy lejos de la complejidad teórica que acabamos de ver, debido a tres factores fundamentales que pueden darse de manera separada o simultánea[7]:

- En muchos casos la complejidad se reduce drásticamente por los objetivos concretos de la técnica y la estructuración previa del conocimiento. Si el conjunto de n piezas de conocimiento que aparecen en los textos está previamente estructurado en un conjunto $\mathcal{C}$ de cuerpos de conocimiento con $|\mathcal{C}| << 2^n$, y se restringe la búsqueda de posibles expresiones *A es X de B* a aquellas en las que $A, B \in \mathcal{C}$, tenemos que el número de posibles expresiones es como mucho de orden $|\mathcal{C}|^2 << N$. También existe la posibilidad de que, debido a los objetivos del problema, el número de posibles expresiones a evaluar se reduzca drásticamente. Por ejemplo, si tenemos un texto que contiene los síntomas de un enfermo (el texto de Diodoro sobre Alejandro Magno por ejemplo) y tenemos una colección de textos que describen los síntomas de distintas enfermedades (un texto por enfermedad), el problema consistirá en comprobar

[7] No descartamos que existan otros aspectos que puedan contribuir a reducir la complejidad del problema.

para cada enfermedad si el texto correspondiente es una hipótesis válida para el texto de Diodoro, y por tanto el número de pares a evaluar se reduce al número de enfermedades consideradas. Las propias exigencias de que A y B sean cuerpos de conocimiento con $A \not\subseteq B$ y $B \not\subseteq A$ también contribuyen a reducir el número de pares a considerar en la práctica.

- El uso de heurísticas para acotar la búsqueda de conjeturas y refutaciones. Existe una serie de propiedades de las conjeturas y refutaciones que pueden (y deben) utilizarse para acotar la búsqueda. Se trata de propiedades de monotonía y antimonotonía que permiten llevar a cabo un proceso similar al de generación de candidatos en la búsqueda de reglas de asociación. En el caso de las reglas de asociación, y más concretamente en la búsqueda de itemsets frecuentes (que es el paso de mayor complejidad) se sabe que si A es un conjunto de items no frecuente, entonces B no es frecuente para todo B tal que $A \subseteq B$, y por tanto puede evitarse el análisis para todo subconjunto que contenga a A. En el caso de conjeturas y refutaciones, tenemos las siguientes propiedades ([García-Honrado and Trillas, 2012]):

  - Si *A es consecuencia de B* y *B es consecuencia de C* entonces *A es consecuencia de C*. En particular, si *A es consecuencia de B* y $B \subseteq C$ entonces *A es consecuencia de C*. Esto implica a su vez que si *A y B son equivalentes* y *B y C son equivalentes* entonces *A y C son equivalentes* por transitividad.
  - Si *A es consecuencia de B* y $B \neq A$ entonces *B es hipótesis de A*.
  - Si *A es hipótesis de B* entonces *B es consecuencia de A*.
  - Si *A es hipótesis de B* y *B es hipótesis de C* entonces *A es hipótesis de C*. En particular, si *A es hipótesis de B* y $C \subset B$ entonces *A es hipótesis de C*.
  - Si *A es especulación de B* entonces $A \cup \{r(B)\}$ *es hipótesis de B*.
  - Si *A y B son contradictorios*, y se da $A \subseteq A'$ y $B \subseteq B'$, entonces *A' y B' son contradictorios*.
  - Si *A y B son equivalentes* y *B y C son contradictorios* entonces *A y C son contradictorios*.

  Estas propiedades (algunas de las cuales pueden demostrarse en función de las otras) permiten limitar el número de razonamientos que es necesario llevar a cabo para determinar el conjunto de expresiones de tipo *A es X de B*. En el caso de la búsqueda exhaustiva que hemos utilizado como peor caso, también nos permiten adicionalmente acotar la búsqueda comenzando el análisis por aquellos pares (A, B) de cuerpos de conocimiento de menor tamaño (si buscamos

consecuencias, por monotonía) o mayor (si buscamos hipótesis, por antimonotonía), de forma que si se obtiene *A es consecuencia de B* o *A es hipótesis de B*, por ejemplo, podamos también afirmar el mismo tipo de expresión para una enorme cantidad de otros pares de cuerpos de conocimiento A', B' para los que se cumple $A \subseteq A'$ y/o $B \subseteq B'$. Por supuesto, el alcance de la reducción en tiempo en la práctica dependerá del conjunto de conocimiento disponible y sus características.

- El uso de razonadores incrementales. La investigación para el desarrollo de nuevos razonadores ha dedicado mucho esfuerzo a la incrementalidad del razonamiento ([Grau et al., 2007, Kazakov and Klinov, 2013]), debido a su capacidad para reducir enormemente el tiempo empleado. La incrementalidad hace referencia a la posibilidad de mantener las estructuras utilizadas en un proceso de razonamiento de manera que, a la hora de llevar a cabo de nuevo el razonamiento tras una pequeña actualización de la base de conocimiento, no sea necesario comenzar el razonamiento desde cero, sino que puedan reutilizarse las estructuras de razonamiento previas, aplicando pequeños cambios coherentes con la modificación, y obtener resultados válidos. Existen hoy en dia razonadores ampliamente utilizados en los estándares de ontologías, como *Pellet* ([Sirin et al., 2007]) para OWL DL, que son incrementales.

  Este tipo de razonadores permite ganar rapidez en la ejecución de algoritmos de búsqueda de conjeturas y refutaciones. Por ejemplo, supongamos que mediante un proceso de razonamiento se ha podido determinar que *A es consecuencia de B*. Un razonador incremental permitiría que podamos evaluar $A \cup \{q\}$ *es consecuencia de B* sin tener que hacer el razonamiento desde cero, sino reutilizando mediante pequeños cambios las estructuras de datos empleadas para determinar que *A es consecuencia de B*, y por lo tanto ganando tiempo. Este proceso puede seguirse a continuación añadiendo una a una nuevas piezas de conocimiento, con lo que podremos minimizar enormemente el tiempo necesario para evaluar expresiones tipo *C es consecuencia de B* con $A \subset C$.

El desarrollo de algoritmos que se beneficien de estas propiedades para realizar minería de textos basada en conocimiento de manera eficiente es uno de los desafíos más interesantes del nuevo paradigma que proponemos en esta Tesis. Algunos de estos aspectos se utilizarán en la propuesta concreta de minería de textos basada en conocimiento que presentamos en el capítulo 4.

3.2.5 Resultados

Debemos recordar que en todo caso la validez de los resultados dependerá de la calidad del conocimiento de partida, su representatividad y relevancia. En la discusión subsiguiente, no tomaremos estos importantes aspectos en cuenta, por cuanto asumimos que deben haber sido resueltos en fases anteriores del proceso de minería.

Dependiendo de la técnica de minería de textos basada en conocimiento utilizada, los resultados pueden ser muy diversos. Por ejemplo, algunas técnicas van a proporcionar agrupamiento de textos en función de la relación entre los mismos (más concretamente, de la relación existente entre sus bases de conocimiento, sea esta de equivalencia, contradicción, etc.). En este caso, en principio, no es necesario establecer ningún tipo de orden de los resultados, siendo simplemente necesario, como se indicó en el capítulo anterior, el uso de técnicas de visualización para presentar estos resultados de forma que resulten fáciles de comprender por el destinatario final de la información.

Sin embargo, otras técnicas proporcionan listas de expresiones del tipo *A es X de B*, como ya hemos indicado. La cuestión que se plantea es, ¿cómo presentar esta información? Por supuesto, puede presentarse una simple lista si ningún orden particular, pero esto puede dificultar también el análisis y la valoración de los resultados. ¿Qué expresiones son más interesantes y, por tanto, deberían ocupar las primeras posiciones en la lista?

En el caso de minería de textos basada en datos, los patrones extraídos suelen venir acompañados de *medidas objetivas de calidad* que permiten ordenarlos a la hora de su presentación. Por ejemplo, en el caso de las reglas de asociación, las medidas habituales de soporte y confianza (entre otras muchas propuestas de medidas) pueden utilizarse para ordenar las reglas. También se utilizan en técnicas de visualización para destacar mediante algún código visual (grosor de líneas, colores, tamaño de círculos o esferas que representan items, etc.) las relaciones que pueden ser más relevantes desde el punto de vista de las medidas. Por supuesto, puede (y suele) ocurrir que las reglas más importantes en términos de los valores de sus medidas no son luego las más relevantes desde el punto de vista de los criterios de la minería (novedad, potencial utilidad, etc.).

En la extracción de expresiones del tipo *A es X de B* no se generan medidas, ya que éstas últimas se basan fundamentalmente en la frecuencia de repetición de patrones y sus componentes que, como ya hemos visto, no se utilizan en las técnicas de minería de textos basada en conocimiento. Sin embargo, es posible determinar criterios para ordenar las expresiones obtenidas, aunque la conveniencia de utilizar

uno y otro de estos criterios, o alguna combinación de los mismos, dependerá de la aplicación concreta. Podemos considerar los siguientes criterios *objetivos*:

- Tamaño de los cuerpos de conocimiento. Si tenemos dos expresiones *A es X de B* y *A' es X de B'*, un criterio para ordenarlas es el número de piezas de conocimiento que componen A, A', B y B'. Particularmente, si $|A| < |A'|$ y $|B| \leq |B'|$ o bien $|A| \leq |A'|$ y $|B| < |B'|$ entonces *A es X de B* se presenta antes que *A' es X de B'*. La idea que subyace a este criterio es que el conocimiento más simple es más fácil de entender y también más valioso en el sentido de que es más difícil en general que se verifiquen relaciones lógicas con bases de conocimiento pequeñas. Como criterio más general, podemos utilizar el tamaño de $|A \cup B|$ y $|A' \cup B'|$ para ordenar las expresiones, anteponiendo las de menor tamaño conjunto.
- En el caso de que $B \subseteq B'$ y se verifique *A es consecuencia de B*, podemos anteponer esta expresión a *A es consecuencia de B'*, que incluso podría excluirse de la lista, salvo que B' corresponda a una estructuración previa del conocimiento que requiera proporcionar información sobre la misma.
- Si se desea proporcionar como resultados piezas de conocimiento generadas y no presentes previamente en los textos, sin especificar a partir de qué cuerpos de conocimiento se han obtenido, un criterio posible es determinar el número de cuerpos de conocimiento con respecto a los cuales mantienen una determinada relación. Por ejemplo, si *A es consecuencia de* B_i para $i \in \{1, \ldots, k\}$ y *A' es consecuencia de* B_j para $j \in \{1, \ldots, k'\}$ y se verifica $k > k'$ entonces presentaremos A antes que A' en la lista, ya que está *soportada* lógicamente por más cuerpos de conocimiento.

Estos criterios no pretenden ser una lista exhaustiva, sino simplemente mostrar que es posible utilizar medidas *objetivas* para realizar la ordenación de los resultados.

En el ámbito de la minería de datos, existen también estudios sobre criterios *subjetivos* para valorar los resultados, es decir, criterios que permitan tener en cuenta la novedad y utilidad de los patrones obtenidos. Este tipo de criterios requiere de la representación computacional de información de la visión del problema que tiene el usuario (entendiendo por tal el destinatario final del conocimiento), concretamente sus intereses y su conocimiento previo particular del ámbito de la realidad al que se refiere el conocimiento. Por ejemplo, en la búsqueda de reglas de asociación, el usuario puede especificar que está especialmente interesado en reglas que involucren a ciertos items, o ventas correspondientes a ciertas épocas del año, o que quiere enfocar su atención en cestas de compra que tengan un valor mínimo dado. Para determinar la novedad, es necesario saber qué conoce el usuario sobre el problema.

El uso de modelos CHC proporciona una herramienta muy interesante para valorar la novedad y utilidad. Para ello se requiere que el usuario represente el conocimiento de que dispone acerca de la realidad en un cuerpo de conocimiento D, que puede incluir también conocimiento previo de tipo general. Cada nueva pieza de conocimiento o conjunto A puede estar en distinta situación respecto a D:

- Si A *es consecuencia de* D entonces A es coherente con lo que se sabía y consiste en un desarrollo de D. No puede descartarse su utilidad.
- Si A *es hipótesis de D'* con $D' \subseteq D$ entonces A permite explicar algún aspecto de lo que sabemos. Si podemos comprobar empíricamente que A se cumple siempre en el ámbito de la realidad objeto del problema, A es útil.
- Si A *es especulación de* D entonces A es muy novedosa a la vez que inesperada. Su utilidad puede venir dada por el hecho de que $A \cup \{r(D)\}$ es una hipótesis de D.
- Si A *es refutación de* D entonces A es muy novedosa (ya que contradice nuestras creencias) e inesperada. Puede ser particularmente útil si se demuestra que dentro de la creencia inicial D existe $D' \subseteq D$ que es falso.

Los resultados obtenidos pueden inicialmente ordenarse en función de su relación a D y dependiendo del interés del usuario. Es importante destacar que, una vez ordenados los resultados inicialmente, la inspección de los mismos puede cambiar el orden. Por ejemplo, si una vez ordenados los resultados tenemos en primer lugar A *es refutación de* D y determinamos que efectivamente existe $D' \subseteq D$ que es falso, podemos sustituir el conjunto D actual por $D_1 = (D \cup A) \backslash D'$, de forma que la relación entre las distintas piezas de conocimiento obtenidas y D_1 sea distinta a la relación existente con respecto a D, lo que implica cambiar la ordenación. En general, tras la inspección de cada pieza de conocimiento, es posible que el conjunto D cambie. Este proceso es coherente con lo que ocurre en el ser humano cuando valora los resultados de un proceso de minería de datos, ya que se actualiza su conocimiento sobre el problema, lo cual afecta al conjunto de sus creencias y a la novedad y utilidad de las mismas. Es típico también de manera particular del ámbito científico.

Es importante destacar que no es imprescindible (aunque sea conveniente) que el conjunto D inicial contenga todo el conocimiento disponible del usuario acerca del problema, sino que éste puede irse actualizando incluso durante el proceso de valoración de los resultados. Imaginemos por ejemplo que incluso $D = \emptyset$ inicialmente. En un proceso de inspección del conocimiento obtenido por parte del usuario, éste puede determinar ir añadiendo a D las piezas de conocimiento que se le muestran

que para él/ella son parte de sus creencias, construyendo de esta forma el conjunto D a la vez que se analizan los resultados. Por supuesto, el conjunto final D contendrá sus creencias iniciales siempre que éstas puedan obtenerse del conjunto de textos utilizados inicialmente. En caso contrario, puede ocurrir que, o bien el usuario vaya añadiendo aparte nuevas piezas de conocimiento que vienen a su mente por relación de analogía y otras con el conocimiento obtenido, o bien que haya parte de ese conocimiento que quede sin incluir en D.

De nuevo, las opciones que hemos descrito con respecto a la valoración subjetiva de resultados no pretenden ser exhaustivas, y su validez y utilidad a la hora de ordenar e inspeccionar los resultados debería ser probada empíricamente. Pero demuestra que los procesos de minería de datos basada en conocimiento, particularmente en modelos CHC, permiten integrar la valoración de novedad y utilidad de manera automática o semiautomática utilizando de forma natural los mismos mecanismos de razonamiento empleados en la generación del nuevo conocimiento. De nuevo, los aspectos discutidos en este apartado abren un amplio horizonte de posibles líneas de investigación futuras a explorar.

3.2.6 Uso y Mantenimiento del *Conocimiento* Descubierto

El conocimiento obtenido mediante procesos de minería de textos basada en conocimiento puede utilizarse de distintas formas. Por ejemplo, el agrupamiento de textos puede usarse para determinar relaciones de semejanza basada en semántica entre documentos científicos, para mejorar la recuperación de información, y un largo etc.

En el apartado 3.1.2.3 hemos indicado una larga lista, no exhaustiva, de posibles usos del conocimiento en aplicaciones. También hemos visto en el apartado anterior cómo el conocimiento obtenido puede ser utilizado para modificar y mejorar el conocimiento previo de una persona acerca del ámbito de la realidad al que se refieren los textos. Como ocurre también en la minería de datos, cuando el ámbito real consiste en alguna organización, empresa o ámbito científico, estas personas suelen ser las responsables de la toma de decisiones que permita avanzar en los objetivos de mejora de la organización. El conocimiento puede utilizarse asimismo para generar resúmenes de los textos capaces de condensar la información contenida en los mismos, mostrando la información más pertinente, e incluso incluyendo en el resumen la relación entre el contenido de distintos textos, como ocurre por ejemplo en ([Puente et al., 2015]).

Otra posibilidad interesante es incluir el conocimiento obtenido en bases de conocimiento utilizables computacionalmente, por ejemplo en el desarrollo de la web

semántica o en sistemas de soporte a la decisión, sistemas inteligentes, etc. Dentro de este ámbito, y en relación al mantenimiento del conocimiento, resultan destacables los sistemas de mantenimiento de conocimiento y creencias, que permiten registrar la evolución del conocimiento a través del tiempo y proporcionan una versión actualizada en cada momento.

Este último punto está relacionado también con un aspecto importante en las técnicas de minería de datos actuales, como es el mantenimiento de los patrones obtenidos, particularmente las medidas de los mismos. Las bases de datos suelen ser dinámicas, registrando continuamente modificaciones que afectan a los patrones que pueden obtenerse de las mismas. El objetivo de los sistemas mencionados es poder mantener de forma incremental los patrones y sus medidas sin necesidad de lanzar el proceso completo de minería cada vez que se hagan modificaciones a la base de datos. En este aspecto, la minería de textos basada en conocimiento podría beneficiarse de los razonadores incrementales que hemos descrito en apartados anteriores a través del mantenimiento de las estructuras necesarias, abriéndose una línea interesante en el ámbito del mantenimiento de conocimiento obtenido mediante el nuevo paradigma de minería.

Por último, queremos destacar que, como es habitual en procesos de minería, el proceso de minería de textos basada en conocimiento no es siempre lineal, es decir, no siempre se llevan a cabo las fases que hemos descrito de manera sucesiva, terminando en la fase que nos ocupa en este apartado. Es frecuente que los resultados obtenidos en cualquiera de las fases puedan llevarnos a volver hacia atrás para modificar lo realizado en alguna de las fases anteriores. Particularmente, el propio uso y mantenimiento del conocimiento puede motivar la aparición de nuevos objetivos que inicien un nuevo proceso de minería de textos. Se trata pues de un proceso dinámico que puede ser casi cíclico en algunas aplicaciones.

3.3 Deducción, Inteligencia y Minería en la Literatura

En la literatura es posible encontrar propuestas de minería de datos y de textos *deductiva*, *inteligente* o que utilizan la palabra *conocimiento*. Sin embargo, ninguna de ellas se corresponde con el nuevo paradigma que proponemos en esta tesis.

Por ejemplo, en ([Russell, 1998]) se propone una técnica llamada *deductive data mining*, donde la parte *deductiva* consiste realmente en averiguar qué datos de la base de datos soportan a una regla, como parte del proceso de minería, pero las reglas se siguen obteniendo mediante inducción utilizando técnicas convencionales. Esta

idea de proporcionar los datos relevantes, bien en el proceso de preparación de datos o en la validación de reglas, está también detrás del uso del término "deducción" en otros trabajos ([Simoudis et al., 1994, Giannotti et al., 1999, Greco et al., 2001, Manco et al., 2008]). En todos ellos, las reglas se obtienen de manera inductiva sobre datos.

El concepto de *deductive text mining* fue propuesto en ([Kodratoff, 2001]) para referirse simplemente a la extracción de información en textos, específicamente la búsqueda de instancias de ciertos patrones en textos. Este concepto, muy lejano a nuestro paradigma de minería de textos basada en conocimiento, ha sido extendido con el mismo significado al caso de la web en lo que se denomina *deductive web mining* ([Svátek et al., 2004]).

En ([Michalski, 2003]) el término *knowledge mining* aparece en el ámbito de la minería de datos haciendo referencia a la propuesta del autor de representar el conocimiento previo y objetivos del destinatario del conocimiento para guiar el proceso de descubrimiento, manteniendo el conocimiento obtenido en una base de conocimiento. Esta idea general coincide con nuestra propuesta de ordenación de resultados, aunque solo considera la actualización del conocimiento añadiendo el nuevo obtenido, y la resolución de conflictos.

Con respecto al uso de formas intermedias de tipo lógico, en la literatura pueden encontrarse técnicas que utilizan las mismas pero que realizan minería sobre las bases de conocimiento buscando patrones de estructuras de conocimiento en base a repetición y semejanza, es decir, mediante técnicas inductivas ([Shen, 1992]), obteniéndose resultados del tipo $p \rightarrow q$ *aparece en el 56% de las bases de conocimiento.*

Hasta donde nuestro conocimiento alcanza, la técnica existente que se corresponde mejor con nuestro nuevo paradigma es la propuesta en ([Kontos et al., 2002]). En este trabajo se extraen relaciones causales de los textos, de las que se obtienen nuevas relaciones causales mediante encadenamiento como mecanismo deductivo. Este trabajo aparece citado en ([Kroeze et al., 2003]) como ejemplo de lo que denominan *Intelligent text mining* (minería de textos inteligente):

> *The essence of text mining is the discovery or creation of new knowledge from a collection of documents. The new knowledge may be the statistical discovery of new patterns in known data (standard text mining) or it may incorporate artificial intelligence abilities to interpret the patterns and provide more advanced abilities such as hypotheses suggestion (intelligent text mining).*

En ([Kroeze et al., 2003]), los autores realizan un análisis sobre el trabajo de Hearst ([Hearst, 1999b]) y muestran su desacuerdo con la propuesta de minería de textos real. Los autores consideran que la distinción de Hearst es entre técnicas tradicionales de gestión de textos (recuperación de información, extracción de información, etc., llamadas técnicas *no nóveles*) y la minería de datos aplicada a textos (que es a lo que Hearst llamaría minería de textos real según estos autores, y denominada *seminóveles*). Aunque no se elabora lo suficiente sobre el concepto de minería de textos inteligente (también llamadas técnicas *nóveles* de minería de textos), ésta se presenta como un proceso semiautomático (interacción entre personas y sistemas inteligentes) que sea capaz de determinar el significado de los patrones obtenidos, lo que implican, qué decisiones pueden ocasionar en el contexto de la organización, impacto social, etc. Es decir, el concepto de minería de textos inteligente se centra más en que el sistema sea capaz de realizar acciones inteligentes con el conocimiento obtenido, de la forma en que lo haría un ser humano, que en el proceso de obtención del conocimiento. Por otro lado, el hecho de que consideren la propuesta de ([Kontos et al., 2002]) como cercana a su paradigma hace pensar que los autores en realidad asocian las capacidades inteligentes con la obtención de conocimiento mediante razonamiento, aunque en ningún momento discuten sobre el proceso de minería ni las fases que lo componen, así como las herramientas de representación y manejo del conocimiento necesarias.

3.4 Conclusiones

En este capítulo hemos presentado un nuevo paradigma que llamamos *minería de textos basada en conocimiento*, cuya característica distintiva con respecto a las técnicas de minería de textos convencionales es el uso de representaciones formales del conocimiento contenido en los textos, y el uso de técnicas formales de razonamiento para generar nuevo conocimiento a partir del contenido en los textos. Este nuevo enfoque permite limitar la perdida de información que se produce al representar el contenido del texto mediante formas intermedias basadas en datos, y se basa en el hecho obvio de que el contenido de los textos es en sí conocimiento. Al mismo tiempo, el uso conjunto de formas intermedias y mecanismos de razonamiento formales permite obtener nuevo conocimiento que no puede obtenerse mediante técnicas basadas en inducción a partir de datos, basadas en semejanza de patrones y en repetición de aparición de los mismos, propios de la minería de textos basada en datos.

La nueva propuesta abre un enorme abanico de posibilidades, tanto en las potenciales aplicaciones prácticas, como en los retos de investigación apasionantes que

plantea. En este capítulo hemos enumerado un número importante de tales posibilidades que, estamos convencidos, no deja de ser una pequeña muestra del potencial de la propuesta. Asimismo, hemos constatado que las hipótesis de partida necesarias para la viabilidad práctica de técnicas de minería de textos basada en conocimiento se corresponden con el estado del arte en temas como la extracción de representaciones formales a partir de textos (particularmente ontologías), la disponibilidad de razonadores eficientes, etc.

Al tratarse de un paradigma novedoso y cuya descripción exhaustiva resulta inabarcable en el marco de una Tesis doctoral, y a pesar de que hemos centrado progresivamente nuestro interés en el uso de ontologías y modelos CHC para concretar tanto como ha sido posible la visión de los aspectos novedosos del problema, el contenido de este capítulo ha sido necesariamente descrito a un cierto nivel de abstracción, utilizando ejemplos muy simples y describiendo posibilidades en las distintas fases del proceso de minería que ilustrasen nuestra visión del nuevo paradigma. Es importante destacar que el paradigma en sí no se restringe al uso de ontologías y modelos CHC, sino que comprende cualquier formalismo de representación y razonamiento basado en el conocimiento.

Consideramos que se trata de un aporte importante porque abre un nuevo campo de investigación interesante y no explorado hasta ahora, en el que pueden esperarse muchas contribuciones importantes de las evoluciones en áreas de investigación bien establecidas y activas, como la representación del conocimiento y desarrollo de razonadores eficientes, especialmente ontologías, lingüística computacional y extracción de conocimiento, heurísticas, etc. En los siguientes capítulos vamos a proponer un mecanismo concreto que se dedica a la búsqueda de contradicciones, utilizando ontologías y los mecanismos de detección de inconsistencias existentes, como técnica concreta que nos permitirá demostrar la factibilidad del nuevo paradigma que hemos propuesto.

Capítulo 4 Una Aplicación en el marco de la Minería de Textos: Búsqueda de Contradicciones en Texto

La contradicción existe en el proceso de cada cosa
y lo recorre desde el comienzo hasta el fin
(Mao Tse Tung [Tse-Tung, 1937])

Las contradicciones están presentes en el mundo que nos rodea, forman parte de lo cotidiano tanto en el pensamiento como en la utilización del lenguaje natural. Históricamente estudiadas, desde Aristóteles ([Aristotle, 1994]) hasta Trillas ([Trillas et al., 2002]), en disciplinas diversas: filosofía, lógica, dialéctica... pero ¿qué es una contradicción?

> *[...] it is impossible for anything at the same time to be and not to be, and by this means have shown that this is the most indisputable of all principles (Aristóteles [Aristotle, 1994]).*

> *Es claro que la misma cosa no estará dispuesta al mismo tiempo a hacer o sufrir cosas contrarias con respecto a lo mismo y en relación al mismo objeto (Sócrates [Bloom, 1991]).*

La localización de contradicciones puede resultar muy útil en diversos campos: para evaluar la consistencia de una colección de textos o para comprobar la validez

de cualquier nuevo documento que se pueda incorporar a la misma, para localizar diferentes grupos de ideas en repositorios que contengan artículos de opinión, microblogs, redes sociales... o para detectar la consistencia de las ideas de los autores o los posibles errores en su discurso. Otras posibles aplicaciones para la búsqueda de contradicciones pueden ser la generación de debates científicos respecto a un tema en concreto o alcanzar consensos, la revisión de textos legales con el fin de evitar contradicciones y redundancias que podrían conllevar problemas posteriores a la hora de aplicar dichos textos, la localización de problemas en guiones de películas (en inglés *plot/ script holes*) de películas o la obtención de paradojas o ironías, el análisis de sentimientos contradictorios o encontrados, la detección de mentiras o falsas afirmaciones, o la posibilidad de ubicar incongruencias en artículos.

A lo largo de este capítulo ofreceremos una definición de contradicción en el ámbito de la lógica y se proporcionará una metodología para nuestra aplicación práctica, Búsqueda de Contradicciones sobre Textos.

4.1 Definición de Contradicción

El concepto de *contradicción lógica* se define por medio de una operación de implicación $\rightarrow$ y una negación $\neg$, donde

Definición: *p es contradictorio con q si y sólo si $p \rightarrow \neg\, q$ es verdad. ([Trillas et al., 1999])*

Nótese que la relación de contradicción es simétrica por cuanto $p \rightarrow \neg\, q$ es lógicamente equivalente a $q \rightarrow \neg\, p$. Utilizando la terminología de los modelos CHC, tal y como vimos en el capítulo anterior, la definición anterior es equivalente a decir que $p \in Ref(q)$ (y también por lo tanto $q \in Ref(p)$).

Podemos encontrar en tratados filosóficos (vemos uno de ellos es [Priest et al., 2004]) una serie de afirmaciones que nos ayudarán a lo largo de este trabajo:

- *Contradicción Simple*: ninguna contradicción es cierta.
- *Contradicción Ontológica*: ningún ser puede instanciar propiedades contradictorias.
- *Contradicción Racional*: es irracional aceptar una contradicción.

Además de aportar estas afirmaciones, distinguiremos las ideas de contrarios y contradictorios ([Priest et al., 2004]):

- p y q son *contrarios* si no pueden ser, ambos, verdaderos.
- p y q son *subcontrarios* si no pueden ser, ambos, falsos.
- p y q son *contradictorios* si ambos son contrarios y subcontrarios.

Y, continuemos con dos conceptos también implicados en la contradicción: *falsedad* e *imposibilidad*. Según asegura Trillas en su artículo ([Trillas et al., 2002]), en la lógica, la idea de la *falsedad* apareció después de la idea de *imposibilidad*, entendida esta última como "auto contradicción". El concepto de falsedad era normal en lógicas antiguas mientras que se volvió más frecuente encontrar el concepto de imposibilidad en lógicas modernas.

Pero veamos algunas de las ideas que nos llevan hasta el concepto de contradicción en lógica.

4.1.1 Contradicciones en Textos

Desde un punto de vista lógico, las contradicciones se pueden detectar de una forma más evidente que sobre el lenguaje natural en el que se escriben los textos, donde una contradicción puede mostrarse bajo diferentes facetas como pueden ser: antónimos, negaciones, falta de contexto, incoherencias semánticas...

Tomemos los siguientes fragmentos de texto que se utilizan en el trabajo de Marneffe en el grupo de trabajo de Procesamiento de Lenguaje Natural de la Universidad de Stanford ([de Marneffe et al., 2008]), como parte de su trabajo localizando contradicciones en textos y nos servirán como ejemplos ilustrativos sobre la diferente apariencia que puede presentar una contradicción en un texto.

Fragmento 1a: *A Russian Soyuz spacecraft docked with the International Space Station, after a two-day flight from the Baikonur Cosmodrome in Kazakhstan.*

Fragmento 2a: *Soyuz spacecraft were never built by Russia.*

Ambas sentencias son contradictorias: mientras que en el fragmento **2a** se asegura que la nave espacial "Soyuz" no fue construida por Rusia, encontramos una aserción explícita en el fragmento **1a** en la que se habla de la nave espacial rusa "Soyuz". Podría ocurrir que ambas naves espaciales sólo compartieran el nombre y que, en realidad, ambos fragmentos hablen de astronaves diferentes, ambigüedad que podría eliminar un humano o la propia restricción del contexto en el que se encuentren incluidos los artículos de los que se han extraído los fragmentos.

Fragmento 1b: *Mr. Fitzgerald revealed he was one of several top officials who told Mr. Libby in June 2003 that Valerie Plame, wife of the former ambassador Joseph Wilson, worked for the CIA.*

Fragmento 2b: *Valerie Plame did not work for the CIA.*

La contradicción presente en ambos fragmentos se puede deber a una inculpación, cierta o no, por parte de uno de los declarantes: "Mr. Fitzgerald", en el fragmento **1b** afirma que "Valerie Plame" trabajó para la *CIA* mientras que en el fragmento **2b** "alguien" asegura que "Valerie Plame" sí que trabajó para esta compañía. Se desconoce, con sólo estas líneas quién está en lo cierto pero sí se detecta una contradicción.

Sin embargo, no todas las contradicciones están explícitamente representadas ni se observan de forma evidente al leer la información. Comprobemos un ejemplo en los siguientes extractos de información de los que no se puede determinar si existe o no contradicción porque falta información:

Fragmento 1c: *Brigham Young University began its Jerusalem study program in 1968, first at the City Hotel in East Jerusalem.*

Fragmento 2c: *Brigham Young University was founded in 1968.*

Sin más datos que los escritos en ambos fragmentos, gracias al primero de ellos, **1c**, sabemos que la universidad *Brigham Young* comenzó su ”programa de estudios” en ”1968”, es el segundo fragmento **2c** el que nos habla de que en ese año fue fundada. ¿Existe contradicción?. En principio puede parecer que la segunda frase complementa a la primera, no hay más información que nos lleve a pensar en que exista un error o contradicción en los hechos descritos.

Observemos otro caso interesante en el que, a no ser que introduzcamos algún sinónimo, y por muy evidente que pueda parecer para un lector humano, un sistema automático no tendría por qué detectar una contradicción:

Fragmento 1d: *His wife Strida won a seat in parliament after forging an alliance with the main anti-Syrian coalition in the recent election.*

Fragmento 2d: *Strida elected to parliament.*

Será necesario conocimiento externo que identifique si *won a seat in parliament* y *elected to parliament* son sentencias sinónimas y que indican que Strida, de hecho, ”es parlamentario”, de otro modo sólo quedará como información redundante en la representación.

Cada uno de estos fragmentos, para ser tratados computacionalmente de forma eficiente, deben ser representados mediante una Forma Intermedia. A lo largo de este capítulo proporcionaremos representaciones mediante ontologías para estos segmentos textuales e intentaremos descubrir cómo se comportan al integrarse y a la hora de localizar contradicciones sobre ellas.

En el anexo B se amplía el concepto de ontología, Forma Intermedia que hemos elegido gracias a sus particulares características de similitud con el pensamiento humano y con el mundo real, aportamos una definición para la misma, damos ejemplos de cómo se construye y de cómo se maneja.

4.2 Aplicación de Minería de Textos - Algoritmo de Búsqueda de Contradicciones en Texto

Hemos considerado, en el capítulo 2, sección 2.3, que las ontologías pueden ser útiles a la hora de desarrollar técnicas de Minería de Textos, puesto que permiten representar el mundo real y la semántica de los conceptos y sus relaciones.

Los posibles resultados del proceso deductivo estarán influidos en gran medida por la cantidad y calidad del conocimiento representado en nuestra Forma Intermedia, pero éste no es tan crítico como, por ejemplo, en el diseño de un sistema experto. Como hemos indicado previamente, un proceso de minería no fracasa si no descubre todo el conocimiento novedoso y potencialmente útil posible (de hecho, ninguna técnica existente puede garantizar esto). Descubrir una sola pieza de conocimiento constituye ya un éxito y un aporte valioso ([Justicia de la Torre et al., 2006]).

En este capítulo, utilizaremos las ontologías como forma intermedia para representar la colección de documentos, aprovechando su rica capacidad semántica de representación ya que las ontologías están compuestas por una serie de elementos clave que consiguen representar tanto la sintaxis como la semántica del texto escrito en lenguaje natural.

El **Algoritmo de Búsqueda de Contradicciones**, localiza, dentro de la colección de documentos, aquellos textos y concatenaciones de textos que contienen contradicciones. Consideraremos *contradicciones*, aquellas ontologías o concatenaciones de ontologías en las cuales existen clases insatisfacibles, inconsistencias semánticas o incoherencias en el modelado (véase la sección 4.4.5 donde se detallan las posibles contradicciones existentes en una ontología).

Supongamos que disponemos de un conjunto de textos $T = \{t_1, \ldots, t_n\}$ y que para cada uno de ellos disponemos de una representación lógica tan rica como haya sido posible. Supongamos un procedimiento *res(t_i)* que lleva a cabo un proceso de razonamiento con el conocimiento obtenido de t_i, devolviendo *falso* en caso de encontrar una contradicción en el mismo. Notaremos $t_i t_j$ la concatenación de los textos t_i y t_j (un texto a su vez). Sea PT el conjunto de concatenaciones de subconjuntos de textos de T. V representa el conjunto de concatenaciones de textos que no son contradictorios. Nuestro algoritmo para la búsqueda de contradicciones es el siguiente:

Algoritmo 1 Búsqueda de Contradicciones en Textos

1. $V_1 \leftarrow \{\ t_i \in T \mid res(t_i) \neq falso\ \}$
2. $V \leftarrow V_1$
3. Para k=2 hasta n

(a) $V_k \leftarrow \{\ t't'' \mid t' \in V_{k-1},\ t'' \in V_1,\ t^* t'' \in V$
$\forall\ t^* \subset\ t' \mid y\ res(t't'')\ \neq falso\ \}$

(b) $V \leftarrow V \cup V_k$

4. *Contradicciones* $\leftarrow$ PT\V

Las iteraciones de nuestro algoritmo de **Búsqueda de Contradicciones en Textos** (**Algoritmo 1**), como podemos ver en la figura 19, recibe como entrada un conjunto de ontologías candidato y como salida devuelve los miembros del conjunto que no son consistentes (que son contradictorios).

El número máximo de iteraciones del algoritmo dependerá, tanto de la consistencia del conjunto de candidatos, como del tamaño del corpus inicial. Así, para un corpus de n ontologías, el número máximo de uniones de ontologías candidatos será la suma de las combinaciones de los elementos $\binom{n}{k}$ siendo $m = 1..n$ y el número máximo de iteraciones del mismo será *n-1*:

- Primera vuelta: se comparan todos los elementos del corpus inicial, de dos en dos, para un tamaño de corpus de 5 ontologías, los candidatos explorables en busca de contradicciones serán $\binom{5}{2} = 10$, siempre que no aparezcan ontologías insatisfacibles durante la mezcla.
- Segunda vuelta: se comparan todos los elementos del corpus inicial, la mezcla ahora será de tres en tres, para un tamaño de corpus de 5 ontologías, los candidatos explorables en busca de contradicciones serán $\binom{5}{3} = 10$, de nuevo si no encontramos inconsistencias.
- Tercera vuelta: añadidos los obtenidos anteriormente, los comparamos de cuatro en cuatro, para un tamaño de corpus de 5 ontologías, los candidatos explorables no contradictorios en busca de contradicciones serán $\binom{5}{4} = 5$.
- Cuarta vuelta: se incrementa el conjunto de explorables con la unión de las ontologías de cuatro en cuatro, para un tamaño de corpus de 5 ontologías, los candidatos explorables en busca de contradicciones serán $\binom{5}{4} = 1$.

	ontologias	candidatos (antes)	explorables	candidatos (después)
vueltas = 0 i=0..4 j=0..4 Candidatos posibles: (5 2) = 10	{a} {b} {c} {d} {e}	{a} {b} {c} {d} {e}	{ab} {ac} {ad} {ae} {bc} {bd} {be} {cd} {ce} {de}	{ab} {ac} {ad} {ae} {bc} {bd} {be} {cd} {ce} {de}
vueltas = 1 i=0..4 j=0..4 Candidatos posibles: (5 3) = 10	{a} {b} {c} {d} {e}	{ab} {ac} {ad} {ae} {bc} {bd} {be} {cd} {ce} {de}	{abc} {abd} {abe} {acd} {ace} {ade} {bcd} {bce} {bde} {cde}	{abc} {abd} {abe} {acd} {ace} {ade} {bcd} {bce} {bde} {cde}
vueltas = 2 i=0..4 j=0..4 Candidatos posibles: (5 4) = 5	{a} {b} {c} {d} {e}	{abc} {abd} {abe} {acd} {ace} {ade} {bcd} {bce} {bde} {cde}	{abcd} {abce} {abde} {acde} {bcde}	{abcd} {abce} {abde} {acde} {bcde}
vueltas = 3 i=0..4 j=0..4 Candidatos posibles: (5 5) = 1	{a} {b} {c} {d} {e}	{abcd} {abce} {abde} {acde} {bcde}	{abcde}	{abcde}

Figura 19: Búsqueda de Contradicciones - Candidatos

4.3 Metodología General en la Búsqueda de Contradicciones

Esta sección describe el marco general de nuestra aplicación, Búsqueda de Contradicciones en Textos, y describen casos - estudio reales que realizaremos sobre un textos reales. Estos ejemplos nos servirán para guiar este trabajo hasta el siguiente capítulo donde se experimentará de forma práctica nuestra aplicación.

La puesta en marcha de nuestro sistema requerirá los elementos y fases que se ven reflejados de forma gráfica en la figura 20. En ella observamos cómo es necesario un corpus inicial de ontologías (que construiremos de forma automática y presentaremos en el siguiente capítulo, aunque podría obtenerse de un repositorio), un razonador que será necesario para detectar las contradicciones en las ontologías y permitirá realizar razonamiento incremental y no incremental (las contradicciones que localizará el razonador se manifiestan en forma de consistencia de ontologías o como clases insatisfacibles en la misma, el razonador elegido para esta labor es *pellet* ([Clark and LLC, 2011])).

Los módulos empleados en nuestro proceso de Búsqueda de Contradicciones son los siguientes (se pueden explorar gráficamente en la figura 20):

1. **Obtención de un Corpus Inicial**, bien por **Generación Automática**, bien a partir de un **repositorio**: partiendo de un conjunto de ontologías de libre elección (la generación del conjunto inicial se detallará más adelante en el capítulo 5), $O = \{o_1, \ldots, o_n\}$, cada una de estas ontologías se volcarán en una estructura de datos que facilite su posterior combinación.
2. **Comprobación de la Consistencia de las ontologías**, cada uno de los elementos iniciales o_i será evaluado por un razonador, un clasificador que permitirá comprobar si esa ontología es consistente (sección 4.3.2), desde el punto de vista semántico o semántico. En el caso de encontrar una *inconsistencia* en el modelo, se determinará que en dicha ontología existe una contradicción y se eliminará del conjunto inicial de candidatos.
3. **Fusión de los candidatos**: si nuestra ontología o_i no presenta contradicciones, se fusionará con el siguiente elemento de nuestro corpus inicial o_j, una vez determinada sus consistencia y devolverá la ontología fusionada $o_i o_j$. La fusión de dos ontologías es una tarea costosa, con diferentes modalidades (mapeo, alineamiento, emparejamiento) y, en ocasiones puede requerir la acción de un elemento humano. Nos adentraremos en las diferentes opciones de fusión en secciones posteriores (ver 4.3.3). El candidato fusionado,

si ha resultado consistente, en nuestro trabajo equivaldrá a que no presenta contradicciones, se incluye al conjunto de candidatos, que se fusionará con el resto de elementos de dicho conjunto. Si resulta contradictorio, se añadirá a la bolsa de contradicciones e inconsistencias resultante de nuestro algoritmo.

4. El algoritmo **finalizará** cuando no queden más candidatos explorables, bien porque se hayan agotado todas las vueltas posibles (como vimos en la sección anterior, el número de vueltas vendrá determinado por el número de elementos del conjunto inicial) o por la imposibilidad de continuar debido a la presencia de contradicciones en las ontologías restantes.

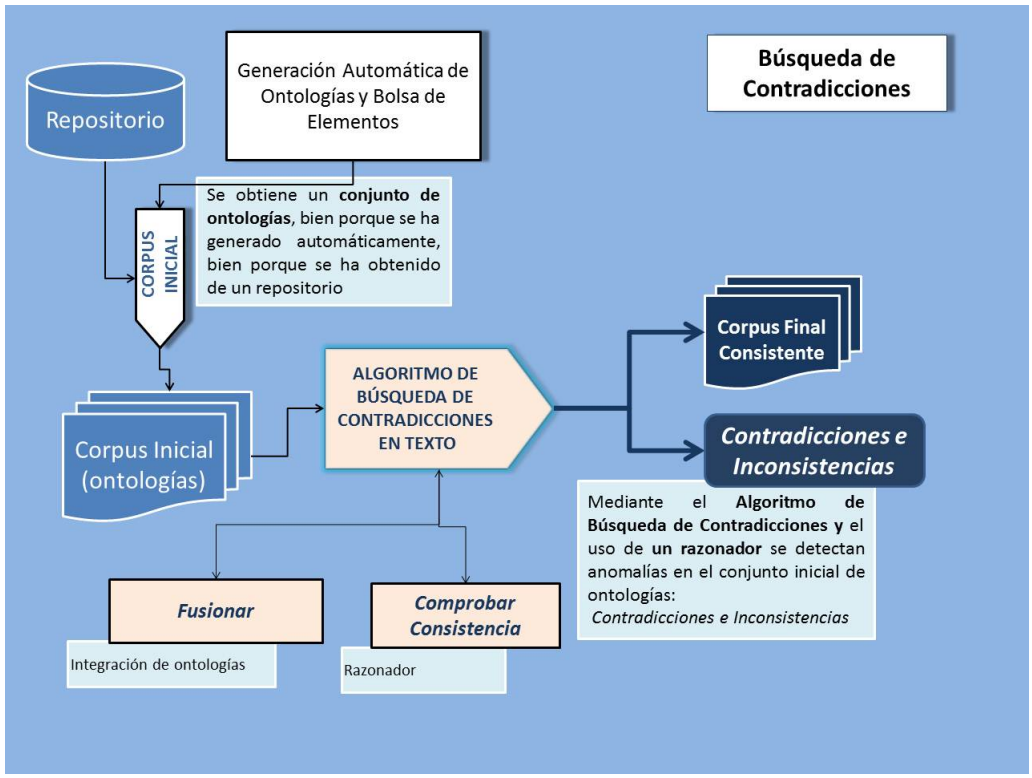

Figura 20: Búsqueda de Contradicciones

Tras presentar nuestro algoritmo de Búsqueda de Contradicciones y para demostrar la utilidad del mismo sobre casos reales, pasemos a los experimentos controlados con los que comprobaremos la fortaleza de nuestra aplicación como los fragmentos textuales que presentábamos en la sección 4.1.1.

4.3.1 Paso 1: Obtención del Corpus Inicial

Siguiendo la metodología anterior, detallemos el **primer paso**, **Obtención de un Corpus Inicial**: partiendo del conjunto de fragmentos textuales de la sección 4.1.1, construiremos un conjunto de ontologías que sirva de corpus inicial.

Se pueden utilizar herramientas de etiquetado semántico, como las que presentan en su artículo Witte ([Witte et al., 2010]) o la herramienta de análisis de texto que ofrece el grupo S4 ([Zendesk, 2014]) para obtener los conceptos necesarios para construir ontologías que representen ambos fragmentos.

Entre las herramientas que nos permiten, dado un texto, escrito en lenguaje natural, obtener una ontología de forma automática encontramos: ASIUM ([Faure et al., 1998]), TextToOnto ([Maedche and Staab, 2000b]) y Ontolearn ([Navigli et al., 2003]). Muchos de esos sistemas dependen de un análisis superficial y de algoritmos de *machine learning* para encontrar conceptos potencialmente interesantes entre ellos.

- ASIUM ([Faure et al., 1998]), basado en clustering conceptual gracias al que se forman clústeres básicos a partir de palabras cabecera que aparecen con el mismo verbo después de la misma preposición.
- TextToOnto ([Maedche and Staab, 2000b]), en este trabajo se realiza una extracción semi automática de taxonomías utilizando un algoritmo de reglas de asociación, este algoritmo detecta el nivel de asociación entre conceptos y el nivel de abstracción de cada relación.
- Ontolearn ([Navigli et al., 2003]), utilizan procesamiento de lenguaje natural y técnicas de *machine learning*.
- OntoLT ([Buitelaar et al., 2003]), proporciona un *plug-in* para Protégé con el que se pueden extraer conceptos (*Protégé clases*) y relaciones (*Protégé slots*) a partir de colecciones de texto con anotaciones. Se basa en el conocimiento lingüístico - semántico a través del uso de patrones incorporados que mapean una lingüística compleja (análisis morfológico, funciones gramaticales)

y la semántica (clases semántico léxicos, predicados). Este mapeo se realiza directamente con los conceptos y las relaciones.

- En el trabajo de Ahmad ([Ahmad and Gillam, 2005]), se muestra una aplicación que identifica y extrae términos candidatos y ontologías candidatas a partir de textos.
- PoolParty: herramienta que incluye extracción de conceptos y términos y ofrece soluciones para la organización del conocimiento ([Company, 2014]).
- S4: sistema web que proporciona servicios de análisis de texto para noticias y redes sociales ([Zendesk, 2014]).
- Fox: proporciona una demo con la que se puede obtener un grafo owl a partir de un fragmento textual ([Fox, 2014]).

Uno de los problemas con los que nos enfrentamos al utilizar este tipo de herramientas es que la expresividad del lenguaje puede no ser la que estamos buscando, por lo que habrá que realizar operaciones posteriores sobre la ontología obtenida.

La primera tarea a acometer consistirá en representar, mediante lógica descriptiva (LD), aquellas secciones de los fragmentos textuales en las que, claramente, apreciamos una contradicción:

Fragmento 1a: *A Russian Soyuz spacecraft [..]*

LD 1a: Spacecraft $\sqcap \exists$ builtIn.Russia, Spacecraft(Soyuz)

Fragmento 2a: *Soyuz spacecraft were never built by Russia*

LD 2a: Spacecraft $\sqcap \nexists$ builtIn.Russia, Spacecraft(Soyuz)

Con la información del etiquetado semántico, la procedente de la lógica descriptiva e información adicional obtenida a partir de *dbpedia*, construimos manualmente dos ontologías que sirvan como Forma Intermedia para sendos fragmentos, de este modo y según la figura 21, nuestra primera ontología *Ontología 1* será la

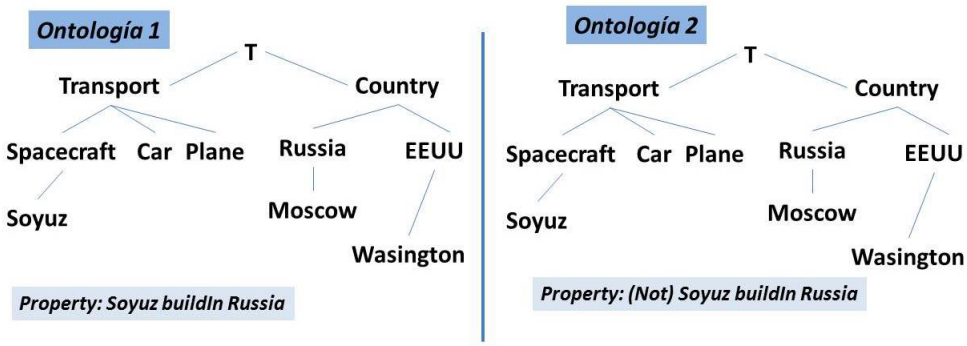

Figura 21: Representación Intermedia de los fragmentos textuales 1a y 2a

que represente al **fragmento 1a** y la segunda, *Ontología 2*, será la que represente al **fragmento 2a**.

La construcción de las ontologías se completará con conceptos adicionales no presentes explícitamente en la frase pero necesarios para construir los modelos (por ejemplo, las clases *Transporte, Country, Car, Plane...*).

Continuando con nuestra construcción de conjuntos iniciales a partir de los fragmentos de la sección 4.1.1, comprobemos la transformación siguiente, representada en la figura 22:

Fragmento 1b: *[...] that Valerie Plame,[...], worked for the CIA.*

LD 1b: Human $\sqcap \exists$ workAt.CIA, Human(Valerie_Plame)

Fragmento 2b: *Valerie Plame did not work for the CIA.*

LD 2b: Human $\sqcap \nexists$ workAt.CIA, Human(Valerie_Plame)

Las ontologías *Ontología 1* y *Ontología 2* obtenidas a partir de los fragmentos

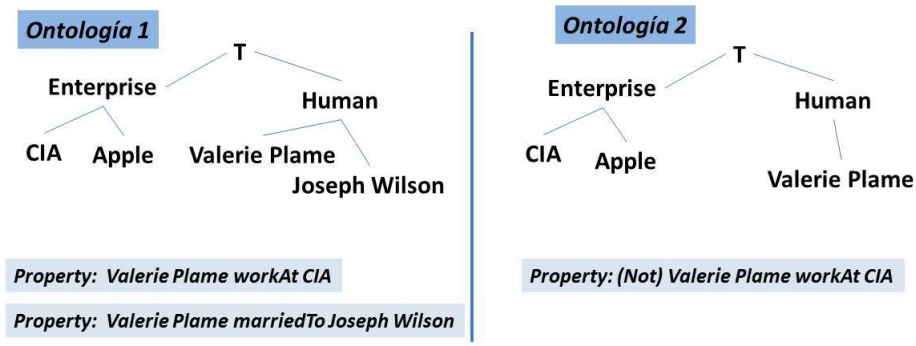

Figura 22: Representación Intermedia de los fragmentos textuales 1b y 2b

1b y **2b** también requieren conceptos adicionales para proporcionarle coherencia al modelo (por ejemplo, las clases *Enterprise, Human...*).

Y evaluemos los fragmentos en los que no se conoce, a priori, si existe contradicción con la información existente, nos referimos a los fragmentos **1c** y **2c** y **1d** y **2d**, representados respectivamente en las figuras 23 y 24:

Fragmento 1c: *Brigham Young University began its Jerusalem study program in 1968[...]*

LD 1c: University(Brigham_Young_University)
University ⊓ ∃ Study_Program ⊓ startedIn.1968, Study_Program(Jerusalem)

Fragmento 2c: *Brigham Young University was founded in 1968.*

LD 2c: University ⊓ ∃ foundedIn.1968, University(Brigham_Young_University)

Fragmento 1d: *His wife Strida won a seat in parliament [...]*

LD 1d: Politician ⊓ ∃ wonASeatIn.Parliament, Politician(Strida)

Fragmento 2d: *Strida elected to parliament.*

LD 2d: Politician ⊓ ∃ electedTo.Parliament, Politician(Strida)

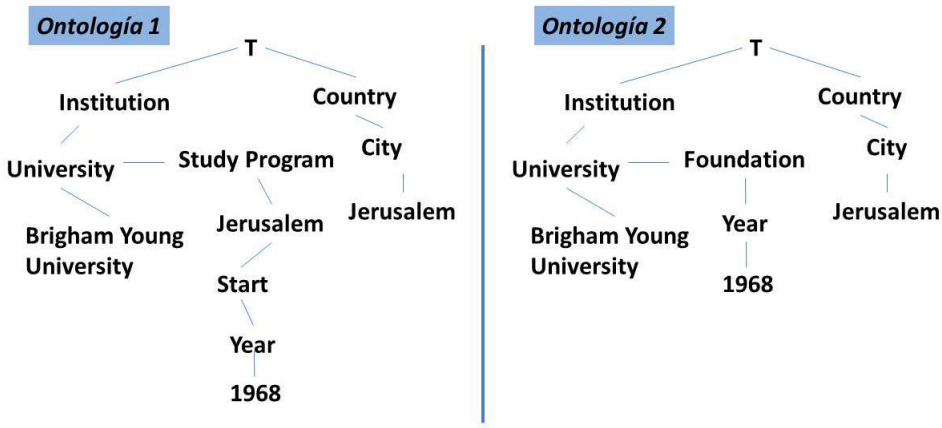

Figura 23: Representación Intermedia de los fragmentos textuales 1c y 2c

La construcción de las ontologías *Ontología 1* y *Ontología 2* obtenidas a partir de los fragmentos **1c** y **2c** continúa con la dinámica anterior, añadimos clases al modelo (por ejemplo, las clases *Institution, Country, City ...*), igual ocurre con la figura 24 donde se han incluido clases como *Politician*.

Tras la propuesta de representación intermedia de los fragmentos textuales, y conociendo que no existe un único modelo para construir ontologías sino que podrían haberse dibujado diferentes versiones para reproducir los fragmentos, se nos plantean diversas preguntas derivadas del análisis textual: ¿es consistente nuestro corpus inicial? ¿está bien construido? ¿existen problemas de modelado en la representación intermedia de nuestro corpus?

Las siguientes secciones describen los siguientes pasos de nuestra metodología

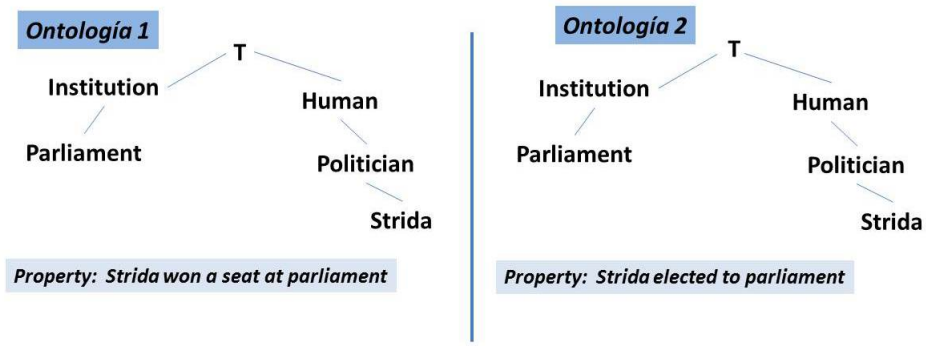

Figura 24: Representación Intermedia de los fragmentos textuales 1d y 2d

para implementar el algoritmo de Búsqueda de Contradicciones y pretende responder a dichas preguntas.

4.3.2 Paso 2: Comprobación de la Consistencia de las Ontologías - Razonamiento

Presentada la aplicación que se va a estudiar en este trabajo y elegidas tanto la estructura como la expresividad de las ontologías, se utilizará un razonador para comprobar si la semántica y la sintaxis de la misma son correctas.

Existen diversos razonadores para ontologías pero no todos permiten obtener la consistencia de una ontología y realizar razonamiento incremental. Se entiende por razonamiento incremental la capacidad del razonador para procesar actualizaciones, véase: añadir o eliminar información, añadir individuos, sin tener que volver a realizar todos los pasos de razonamiento. Habrá que tener en cuenta este hecho si se pretende incluir el razonamiento incremental como parte de la experimentación. En el capítulo 5 veremos que éste es nuestro caso.

El razonador podrá realizar varias tareas ([Horridge et al., 2004]):

1. Detectar inconsistencias: Una clase será inconsistente si no tiene instancias (también se puede llamar clase insatisfacible).

   El test de satisfacibilidad de conceptos definidos se puede definir como

([Horrocks and Sattler, 2002]):

*SAT(*C, T*) si y sólo si hay un modelo* I *de* T *con* $[C^I]$ *[*$\neq \emptyset$*]*

Si existen conceptos definidos e insatisfacibles, habrá fallos en el modelado

2. Detectar clases satisfacibles de forma trivial.
3. Encontrar consecuencias de las declaraciones o subclases ([Horrocks and Sattler, 2002]).

   *SUBS(*C,D,T*) si y sólo si [*$C^I \sqsubseteq D^I$*] para todo modelo* I *de* T

   Si existen subclases no deseadas o perdidas, existirá imprecisión o errores en el modelado
4. Redundancias: se localizarán equivalencias que ayudarán a evitar malentendidos ([Horrocks and Sattler, 2002]).

   *EQUIV (*C,D,T*) si y sólo si [*C^I*] = [*D^I*] para todo modelo* I *de* T

Las ontologías construidas y detalladas en las figuras 21, 22, 23 y 24 no presentan inconsistencias, a priori, puesto que: las clases tienen instancias (*CIA* es instancia de *Enterprise*, *Strida* es instancia de *Politician* ...), no existen subclases perdidas en el modelo, todas ellas tienen una clase padre (*University* es subclase de *Institution*, *City* es subclase de *Country*...) y, a simple vista, no existen redundancias evidentes.

4.3.3 Paso 3: Fusión de los Candidatos

La fusión de los candidatos será un proceso, nada trivial, de generación de una ontología sencilla a partir de dos de los candidatos que se estén evaluando ([Choi et al., 2006]).

También llamado proceso de integración de ontologías (fusión, mapeo, integración y alineamiento), la fusión de candidatos se puede considerar como un proceso de reutilización de ontologías ([Choi et al., 2006]) y se puede ampliar en el anexo C donde se detallan los problemas de la integración y las diferentes opciones de mezcla de ontologías.

En el paso 3 de nuestro proceso de Búsqueda de Contradicciones, realizaremos una fusión simple de ontologías en la que se fusionarán los axiomas de las dos ontologías candidato.

Veámoslo de forma gráfica utilizando nuestros casos de estudio. En la figura 25 comprobamos cómo sería la fusión de las ontologías 1 y 2 de la figura 21, obtendríamos la propiedad *Soyuz buildIn Russia* y su negada *(NOT) Soyuz buildIn Russia*. Al construir la ontología, volveríamos a llamar al razonador, el cual nos encontraría una contradicción y esta ontología fusionada no formaría parte del conjunto de candidatos sino de las contradicciones detectadas.

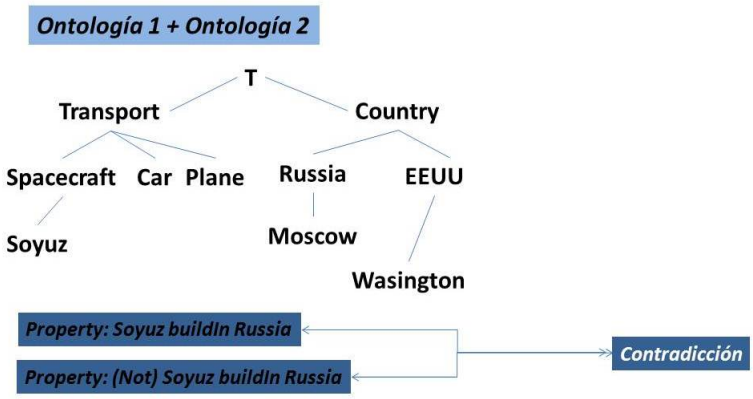

Figura 25: Fusión de los candidatos representando los fragmentos textuales 1a y 2a

El siguiente caso que nos ocupa, figura 26 representa la fusión de las ontologías 1 y 2 de la figura 22. En esta ocasión, también obtendríamos dos propiedades contradictorias *Valerie Plame workAt CIA* y su negada *(NOT) Valerie Plame workAt CIA*. Contradicción que también detectaría nuestro razonador y añadiría al conjunto de contradicciones detectadas.

Respecto a la fusión de las ontologías 1 y 2 (fragmentos **1c** y **2c**) de la figura 23 que mostramos en la figura 27, la ontología resultado de la mezcla, se completaría con las clases de la primera y de la segunda ontología, no existiendo propiedades, axiomas o instancias que se contradigan, por lo que nuestra ontología resultado de la fusión formaría parte, tras invocar al razonador, del conjunto de candidatos.

Finalicemos este estudio con la fusión de las ontologías 1 y 2 de los fragmentos **1d** y **2d** que vimos en la figura 24. La fusión de estas representaciones intermedias, véase la figura 28, sería, tras identificar los conceptos uno a uno como idénticos, sería una nueva ontología sin contradicciones evidentes y a la que se le añaden los

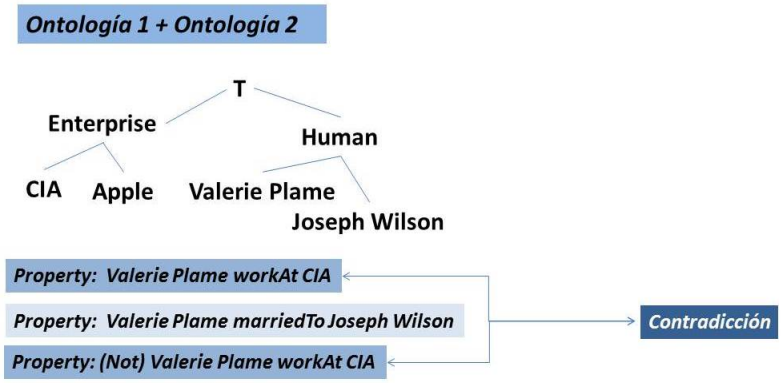

Figura 26: Fusión de los candidatos representando los fragmentos textuales 1b y 2b

axiomas *Strida won a seat at parliament* y *Strida elected to parliament*. Esta nueva ontología también formaría parte del conjunto de candidatos sin contradicciones.

4.4 Construcción del Conjunto Ejemplo de Ontologías OWL DL

Hemos mencionado en diversas ocasiones a lo largo de este trabajo que las ontologías serán una Forma Intermedia adecuada para la representación del texto pero habrá que decidir qué tipo de expresividad se desea para construirlas.

Una de las decisiones que nos han llevado a elegir ontologías descritas en lenguaje OWL DL es que pueden ser procesadas por un razonador, elemento que podemos integrar en nuestro algoritmo de forma que nos devuelva la inconsistencia o no de las ontologías que se están tratando, otro de los hechos que nos han llevado a decidirnos por este lenguaje es su capacidad para representar contradicciones.

OWL es un ”fragmento” de las lógicas de primer orden, y está basado en lógicas descriptivas. Existen varios tipos de OWL, a saber, OWL Lite $\subseteq$ OWL DL $\subseteq$ OWL Full. Cada una de las variantes de OWL permite expresar un número determinado de propiedades, nosotros elegiremos OWL DL, basado en SHOIN(D), cuyas

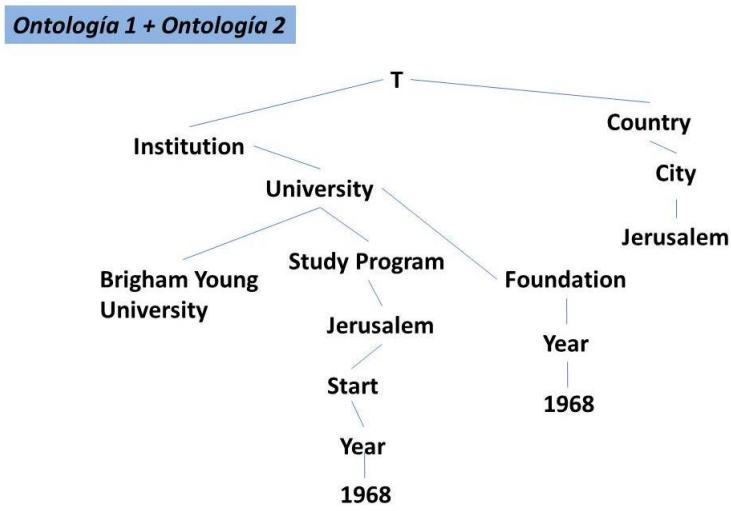

Figura 27: Fusión de los candidatos representando los fragmentos textuales 1c y 2c

características las podemos leer en la sección 4.4.3.

En esta sección nos acercaremos a la representación de contradicciones en OWL y mostraremos cómo se construye un conjunto de ontologías ejemplo utilizando el lenguaje OWL DL.

4.4.1 Localizando Contradicciones en OWL

El lenguaje elegido (OWL, acrónimo de *"Web Ontology Language"*, es el estándar del W3C y está basado en lógicas descriptivas utilizadas para representar conocimiento) utiliza la hipótesis de "mundo abierto" (en inglés, *Open World Assumption* (OWA)) con dos ítems principales ([Horridge et al., 2004]):

- Negación: explícita. Si no lo encuentro en mi mundo, asumiré que es posible salvo que sea imposible en cualquier mundo, en cuyo caso será insatisfacible, por lo tanto, una contradicción. Se utiliza en los razonadores de Owl.
- Asumimos las contradicciones planteadas en el lenguaje:

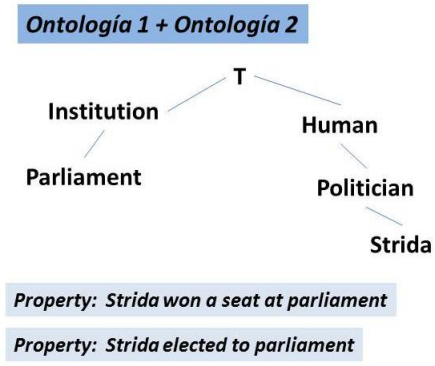

Figura 28: Fusión de los candidatos representando los fragmentos textuales 1d y 2d

- Cualquier restricción existencial (*someValuesFrom*) que se rellene con una contradicción es en sí misma una contradicción.
- Las contradicciones se propagan a través de los enlaces *SomeValuesFrom*.
- Una restricción universal (*allValuesFrom*) (*only*) que se rellene con una contradicción puede ser cumplida de forma trivial.
- No hay ninguna contradicción en decir que algo sólo puede ser cumplido por "*nothing*", aunque seguramente signifique que haya un error.
- Una posible forma de detectar contradicciones es localizar las tautologías F, en cuyo caso $\neg\ F$ será una contradicción.
- A una fórmula falsa para toda interpretación se le llamará contradicción

La redacción de una ontología se puede complicar tanto como se quiera pero el fin último de este apartado es mostrar la construcción de un conjunto adecuado y útil de ontologías que permita representar el corpus textual y mostrar cómo utilizar la sintaxis OWL DL, por lo tanto, cada una de las ontologías ejemplo construidas será una versión simplificada de una ontología.

4.4.2 Modelado de las Ontologías OWL DL

Ya sabemos que una ontología es un conjunto de clases, objetos, propiedades y relaciones entre ellas, así como una serie de individuos que la completa pero ¿cuántas partes debe tener una ontología para que sea considerada como tal? En la sección B.1.1, se revisaron diferentes definiciones de ontología encontradas en la literatura, en esta sección nos interesa construir una, por lo que será imprescindible otorgar un enfoque práctico a dichas definiciones.

Lo que sí necesitaremos es una metodología formal para desarrollar ontologías consistentes, eficientes y, por qué no, que ese desarrollo sea distribuido ([Shack, 2013]).

Según Noy ([Noy and McGuinness, 2001]), no existe una metodología correcta y única para construir ontologías pero estos autores sí que proporcionan una *Metodología de Construcción* que consta de ocho pasos que pueden guiar a quien pretenda afrontar el reto de escribir una ontología, en esta sección se aplicará esta metodología a la resolución del problema propuesto:

- **Paso 1: Determinar el dominio y alcance de la ontología.** Este paso no es relevante en este trabajo, puesto que se trata de obtener un conjunto de ejemplo, las ontologías no estarán aplicadas a ningún ámbito en concreto. Si se estuviera construyendo una ontología en otros términos sí que habría que plantearse para qué se va a utilizar la ontología (*especificación*), sobre qué dominio actuará para desarrollar un modelo conceptual (*conceptualización*), formalizar dicho modelo conceptual en uno manejable computacionalmente (*formalización*), implementar dicho modelo o definir quién lo mantendrá, entre otras cuestiones.
- **Paso 2: Considerar reutilizar ontologías existentes.** Este paso tampoco nos afectará demasiado en este trabajo, la construcción del conjunto ejemplo se hará a partir de cero.
- **Paso 3: Enumerar importantes términos en la ontología.** Llegados a este punto, puede resultar útil enumerar los términos de la ontología para el futuro usuario que las utilice, igualmente, sería importante conocer las propiedades asociadas a esos términos. Si la ontología va a tratar sobre *exploradores*, los términos relevantes podrían ser: *ocupación, profesión, nacionalidad, época ...*, si el dominio elegido es *vinos*, los términos podrían ser: *vino, uvas, grano, ...* Una vez identificados los términos, habría que establecer la jerarquía de clases y las propiedades de los conceptos.

- **Paso 4: Definir las clases y la jerarquía de clases.** El desarrollo elegido sigue la orientación *top-down*, es decir, partiendo de clases generales se van añadiendo especializaciones de conceptos a posteriori. Si la ontología trata sobre *exploradores*, los conceptos generales pueden ser *exploradores*, y las especializaciones serían *Exploradores Franceses*, *Exploradores Británicos*, *Exploradores del siglo XVI* y así tan detallado como se quiera. Si es sobre *vinos*, *vino blanco, vino tinto, Ca- bernet Sauvignon, ...* Para organizar las clases dentro de una jerarquía de clases, habrá que preguntarse si por ser una instancia de una clase, un objeto será necesariamente instancia de otra clase.

  > Si cada clase A es una superclase de B, entonces cualquier instancia de B también lo será de A [Noy and McGuinness, 2001]

  Sea $\mathcal{C} = (c_1, c_2, \ldots, c_n), n = random(1, pm)$,
  siendo pm el valor del parámetro definido por el usuario , el conjunto total de clases disponibles. Cada una de las ontologías tendrán un número n de clases, pudiendo estar repetidas en alguna de las ontologías. El parámetro pj sirve para definir la jerarquía de clases, indica el porcentaje de subclases que contendrá la ontología.

- **Paso 5: Definir las propiedades de clases - *solts*.** Una vez definidas las clases, hay que describir la estructura interna de los conceptos, lo cual aportará más información sobre la clase. Una propiedad es describe los atributos de las instancias. En el ejemplo del *vino*, *cada vino tiene un color, textura, olor*, así que las propiedades serían: *sabor*, *color* o *cuerpo* y habría que indicar a qué clase pertenecen dichas propiedades. Las propiedades pueden ser intrínsecas a la clase (como el *sabor* lo es al *vino*), extrínsecas (como el *nombre* del *vino*), partes de un objeto (como el *primer plato* de una *comida*) y relaciones entre individuos (como el *fabricante*, que relaciona el *vino* con la *bodega* y con la *uva*).

  Entre las propiedades, entendidas como relaciones binarias, ([w3c, 2004]) detallamos los siguientes tipos:

  - ***Datatype Properties***: relaciones entre instancias de clases y literales RDF y tipos de datos del esquema XML.
  - ***Object Properties***: relaciones entre instancias de dos clases.

  Al definir la propiedad, hay una serie de maneras de limitar la relación: se pueden especificar el rango y el dominio, la propiedad se puede definir para ser una especialización de una propiedad existente (subpropiedad), se pueden establecer propiedades transitivas, simétricas, funcionales, inversas o inversas funcionales.

Las propiedades tendrán, a su vez, una serie de restricciones, entre las que se encuentran:

- ***allValuesFrom, someValuesFrom***: restricciones locales a su definición de clase.
- ***Cardinality***: permite la especificación del número exacto de elementos en una relación

  < owl:Class rdf:ID="Vintage" >
  < rdfs:subClassOf>
  < owl:Restriction>
  < owl:onProperty rdf:resource="#hasVintageYear" >
  < owl:cardinality rdf:datatype="&xsd;nonNegativeInteger" >1
  </owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
  </owl:Class>

  Vintage será una clase exactamente con un *VintageYear*. En OWL DL, están permitidos valores enteros positivos distintos de 0 y 1. Se podrá utilizar la cardinalidad máxima y la mínima para limitar el valor entre un intervalo numérico.
- ***hasValue*** : en OWL DL, se permite especificar clases basadas en la existencia de valores una propiedad particular. Un individuo pertenecerá a una clase si al menos uno de sus valores de propiedad es igual al recurso *hasValue* .

  <owl:Class rdf:ID="Burgundy">
  ...
  <rdfs:subClassOf>
  <owl:Restriction>
  <owl:onProperty rdf:resource="#hasSugar" />
  <owl:hasValue rdf:resource="#Dry" />
  </owl:Restriction>
  </rdfs:subClassOf>
  </owl:Class>

  Los vinos Burgundy tienen el valor *dry*. Por lo tanto la propiedad *hasSugar* debe tener al menos un valor igual a *dry*.

- **Paso 6: Definir las facetas de los slots.** Cada una de las características de los *solts*. Los tipos permitidos de valores son: (*String, Number, Boolean, Enumerated, Instance-type* que permite definición de valores entre individuos),

xsd:string	xsd:normalizedString	xsd:boolean	
xsd:decimal	xsd:float	xsd:double	
xsd:integer	xsd:nonNegativeInteger	xsd:positiveInteger	
xsd:nonPositiveInteger	xsd:negativeInteger		
xsd:long	xsd:int	xsd:short	xsd:byte
xsd:unsignedLong	xsd:unsignedInt	xsd:unsignedShort	xsd:unsignedByte
xsd:hexBinary	xsd:base64Binary		
xsd:dateTime	xsd:time	xsd:date	xsd:gYearMonth
xsd:gYear	xsd:gMonthDay	xsd:gDay	xsd:gMonth
xsd:anyURI	xsd:token	xsd:language	
xsd:NMTOKEN	xsd:Name	xsd:NCName	

Tabla 16: Datatypes recomendados ([w3c, 2012b])

valores permitidos, cardinalidad (número de valores), dominio (clases definidas por un *solt* o clases a las que se asocia el mismo) o rango (clases permitidas para los *solts* de un tipo de instancia). Los tipos de datos recomendados se pueden ver gráficamente en la tabla 16.

- **Paso 8: Crear Instancias.** Las instancias también se llaman individuos de la ontología. Para el conjunto de clases C que se va a construir, C, se elegirá una de ellas *[c_i]*, se creará una instancia individual de la misma *[i_i]* y se rellenarán los *solts*.

4.4.3 Obtención de la Expresividad SHOIN(D) - *How To*

La expresividad deseada para el conjunto de ontologías ejemplo es SHOIN(D), para llegar a ella habrá que ir introduciendo factores adicionales, a continuación enumeramos los axiomas y constructores del lenguaje OWL DL de la siguiente manera:

1. **AL**, Lenguaje Atributivo, permite negación atómica ($\neg$C), intersección de conceptos o conjunción (C$\sqcap$D), conjuntos disjuntos (C$\sqcup$D), restricciones universales ($\forall$R.C) y cuantificación existencial limitada ($\exists$R.C).
2. **ALC**, AL + negación compleja de conceptos.
3. **S**, ALC + roles transitivos (S: $\sqsubseteq^+$).
4. **SI**, S + roles inversos (I: $\sqsubseteq^+$).
5. **SH**, SI + roles jerárquicos. Relaciones de subpropiedades (H: R $\sqsubseteq$ S).

6. **SHIQ**, SHI + restricciones numéricas. Cardinalidad cualificada. Este lenguaje es una modalidad de OWL DL sin tipos de datos y nominales (*O:* {a}).
7. **SHOIN**, ALCQI + restricciones de cardinalidad con objetos de dominios (nominales, clases enumeradas, *oneof, hasvalue*) (*N*: $\geq$ n R y $\leq$ n R).
8. **SHOIN(D)**, SHOIN + dominios concretos de tipos de datos (*D*).

4.4.4 Escritura de la Ontología

Cuando empezamos a escribir la ontología desde cero, existen una serie de características que hay observar. Unas tienen que ver con el lenguaje elegido para la implementación de la misma, (OWL DL), otras con las especificaciones (granularidad y alcance), otras con la conceptualización (cómo se organizará el conocimiento adquirido) ([Corcho et al., 2003]).

En las siguientes subsecciones se analizará cómo se construye sintácticamente una ontología utilizando la gramática del lenguaje OWL DL, se revisará la cabecera, el cuerpo y el pie de la misma.

4.4.4.1 Cabecera de la Ontología OWL DL

Toda ontología tiene una cabecera de declaraciones que la ubican en un contexto sintáctico y semántico (esta cabecera es una "herencia" procedente de su origen firmemente ligado a los lenguajes de marcado), la información que se puede encontrar o se puede indicar en ella es la siguiente:

1. **Declaración XML:** línea principal del documento, puede incluir: versión, codificación e independencia de la Definición del Tipo de Documento (*Document Type Definition o DTD*).
   - Versión: especifica la versión del estándar XML que se ha seguido para construir el documento.
   - *Encoding*: estándar de codificación utilizado, puede ser alguno de los siguiente: UTF8, UTF16, ISO10646UCS2, ISO10646UCS4, ISO88591 to ISO88599, ISO2022JP, Shift_JIS, EUCJP. Es recomendable incluirlo por si el documento va a contener caracteres especiales distintos al ASCII, por ejemplo: Æ, å, ø, å...

- *Standalone*: Identifica si el documento presenta o no DTD interna. En el ejemplo que se está construyendo en esta sección, la DTD es interna al documento. La diferencia entre una DTD interna o una externa radica en que si en el primer caso, la estructura del documento se especificará en el propio documento mediante declaraciones a partir de la etiqueta *DOCTYPE* o en un documento externo, aunque también es posible mezclar ambas.

2. **Directivas:**

- **mmDOCTYPE**: Declaración de Tipo de Documento (DTD), debe ir seguida del primer elemento del documento (*root element*), en este caso *rdf:RDF*. Puede o bien ser interna o bien referenciar a una DTD externa. Un DTD define las restricciones en la estructura del documento XML.
- **ENTITY**: las entidades referencian a los datos, los cuales se pueden encontrar en una localización externa. Ayudan a reducir información repetitiva. En el ejemplo que se está construyendo, se corresponden con información gramatical necesaria para construir la ontología. Recomendaciones *owl, xsd, rdfs, ...*

4.4.4.2 Cuerpo

Sección más extensa de toda la ontología debido a que será aquí donde se construirán, definirán y rellenarán los conceptos: clases, objetos, relaciones e instancias.

- Clases: cada uno de los conceptos de la ontología, en el cuerpo de la misma aparecen entre *owl:Class*. Se podrá encontrar el tipo de clase dentro de la definición de la misma.
- Propiedades de Objetos: Delimitadas por las etiquetas de apertura y cierre *owl:ObjectProperty*, contendrán el nombre, el tipo y puede que el dominio al que pertenecen los objetos, así como la url de la web semántica donde está la definición de la misma.
- Propiedades de Datos: al igual que las Propiedades de Objetos, también se detallarán el nombre, el tipo y las restricciones de dominio y de valores convenientes. Se pueden reconocer en el cuerpo de la ontología buscando la etiqueta *owl:DatatypeProperty*.
- Instancias: cada uno de los individuos de la ABox, enmarcados entre las etiquetas *owl:Thing*.

Clase Java	**ProcesarParametros.java**
Restricción	**Motivo**
Mínimo número de **clases**: 2	Para que se pueda añadir una **subclase**
Mínimo número de **instancias**: 2	Para poder añadir **sameas**
Mínimo número de **data properties**: 2	Una llevará **maxcardinality**, otra **mincardinality**
Mínimo número de **object properties**: 3	Inversa,algunos valores (de las 2 propiedades mínimas introducidas), transitiva
Todas las instancias irán a la misma clase, la primera	
ObjectProperties: Inversa	

Tabla 17: Restricciones del programa de generación de ontologías

4.4.4.3 Pie

En este apartado se cierra la descripción de recursos del modelo RDF construido para modelar la ontología gracias a la etiqueta de cierre del lenguaje elegido, por ejemplo *</rdf:RDF>*.

4.4.5 Restricciones Definidas en la Ontología a la hora de definir el Programa de Escritura

En la tabla 17 se detallan algunas de las limitaciones que se han establecido a la hora de construir el conjunto ejemplo para delimitar el ámbito de la ontología.

En la siguiente sucesión de imágenes, 29 - 30 - 31, se muestra el código fuente de la ontología *ejemshoin*, ejemplo final de ontología escrita en lenguaje OWL DL2, con la expresividad SHOIN(D) deseada:

En el capítulo de experimentos (capítulo 5) ofrecemos, de forma pormenorizada, el proceso de generación de ontologías de expresividad SHOIN(D), además, ofrecemos una propuesta de *Text Knowledege Mining* en la que necesitaremos realizar operaciones con las ontologías generadas o bien con algunas obtenidas de un repositorio existente.

a) Razonamiento en OWL Existen varias opciones de razonadores, concretamente, para trabajar con librerías Java: *Pellet*, *Jena reasoner* que funciona mejor con RDF que con OWL, *Owlapi reasoner* que funciona mejor para OWL pero es menos detallado que Jena.

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
<!ENTITY owl "http://www.w3.org/2002/07/owl#" >
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
<!ENTITY ejemshoin "http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#" >
]>
<rdf:RDF xmlns="http://www.w3.org/2002/07/owl#"
xml:base="http://www.w3.org/2002/07/owl"
xmlns:ejemshoin="http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:owl="http://www.w3.org/2002/07/owl#">
<owl:Ontology
rdf:about="http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl">
<owl:versionInfo rdf:datatype="&xsd;string"> version 1.5</owl:versionInfo>
</owl:Ontology>
<!– ////////////////////////
// Object Properties
////////////////////////
–>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#I –>
<owl:ObjectProperty rdf:about="&ejemshoin;I">
<rdf:type rdf:resource="&owl;InverseFunctionalProperty"/>
</owl:ObjectProperty>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#n –>
<owl:ObjectProperty rdf:about="&ejemshoin;n">
<rdfs:domain>
<owl:Restriction>
<owl:onProperty rdf:resource="&ejemshoin;dn"/>
<owl:someValuesFrom rdf:resource="&rdfs;Literal"/>
</owl:Restriction>
</rdfs:domain>
<rdfs:domain>
<owl:Restriction>
<owl:onProperty rdf:resource="&ejemshoin;dnm"/>
<owl:someValuesFrom rdf:resource="&rdfs;Literal"/>
</owl:Restriction>
</rdfs:domain>
</owl:ObjectProperty>
```

Figura 29: Ontología Ejemplo 1/3

```
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#t –>
<owl:ObjectProperty rdf:about="&ejemshoin;t">
<rdf:type rdf:resource="&owl;TransitiveProperty"/>
</owl:ObjectProperty>
<!– ///////////////////////
// Data properties
///////////////////////
–>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#dn –>
<owl:DatatypeProperty rdf:about="&ejemshoin;dn">
<rdfs:domain>
<owl:Restriction>
<owl:onProperty rdf:resource="&ejemshoin;dn"/>
<owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">
21</owl:maxCardinality>
</owl:Restriction>
</rdfs:domain>
</owl:DatatypeProperty>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#dnm –>
<owl:DatatypeProperty rdf:about="&ejemshoin;dnm">
<rdfs:subPropertyOf rdf:resource="&owl;topDataProperty"/>
<rdfs:domain>
<owl:Restriction>
<owl:onProperty rdf:resource="&ejemshoin;dnm"/>
<owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">
1</owl:minCardinality>
</owl:Restriction>
</rdfs:domain>
</owl:DatatypeProperty>
```

Figura 30: Ontología Ejemplo 2/3

```
<!– ////////////////////////
// Classes
////////////////////////
–>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#CLASE1 –>
<owl:Class rdf:about="&ejemshoin;CLASE1">
<owl:disjointWith rdf:resource="&ejemshoin;CLASE2"/>
</owl:Class>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#CLASE2 –>
<owl:Class rdf:about="&ejemshoin;CLASE2">
<rdfs:subClassOf rdf:resource="&owl;Thing"/>
</owl:Class>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#SUBCLASE1 –>
<owl:Class rdf:about="&ejemshoin;SUBCLASE1">
<rdfs:subClassOf rdf:resource="&ejemshoin;CLASE1"/>
</owl:Class>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#SUBCLASE2 –>
<owl:Class rdf:about="&ejemshoin;SUBCLASE2">
<rdfs:subClassOf rdf:resource="&ejemshoin;CLASE1"/>
</owl:Class>
<!– ////////////////////////
// Individuals
////////////////////////
–>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#INDIVIDUO1 –>
<owl:Thing rdf:about="&ejemshoin;INDIVIDUO1">
<rdf:type rdf:resource="&ejemshoin;SUBCLASE2"/>
<sameAs rdf:resource="&ejemshoin;INDIVIDUO2"/>
</owl:Thing>
<!– http://www.semanticweb.org/ontologies/2011/9/ejemshoin.owl#INDIVIDUO2 –>
<owl:Thing rdf:about="&ejemshoin;INDIVIDUO2">
<rdf:type rdf:resource="&ejemshoin;SUBCLASE2"/>
</owl:Thing>
</rdf:RDF>
```

Figura 31: Ontología Ejemplo 3/3

Todos tienes sus pros y sus contras, uno de los principales problemas encontrados es la cantidad de versiones de librerías y de actualizaciones que pueden modificar las importaciones de una a otra (owlapi2 y owlapi3).

El razonador elegido será *Pellet*, un razonador de código abierto para OWL2 DL en java que soporta dos técnicas de razonamiento incremental: la comprobación de consistencia incremental y la clasificación incremental.

El razonador *pellet* se basa en los algoritmos de Tableau, algoritmo que se puede aplicar sobre lógicas descriptivas basadas en expresividad ALC, como son las ontologías elegidas (ver sección 4.4.3). Recordemos que este algoritmo se aplica para comprobar la satisfacibilidad de las fórmulas lógicas, infiriendo que su negación es una contradicción, los nodos finales del árbol representarán el valor de verdad de la fórmula que se está evaluando.

Aplicado sobre Lógicas Descriptivas, si el resultado del algoritmo de Tableau es cerrado, la base de conocimiento será insatisfacible ([Shack, 2013]).

b) Relaciones Semánticas - Ejemplos concretos Concretando la mezcla a nivel de conceptos, utilizaremos las siguientes relaciones semánticas a la hora de comparar conceptos a y b de las ontologías *o'* y *o"* (siempre que los nombres de a y b sean el mismo):

1. Igualdad de conceptos, Equivalencia $a = b$

   a (o'):
   $< owl : Classrdf : ID = "OnionTopping" / >$

   b (o"):
   $< owl : Classrdf : ID = "OnionTopping" / >$
   $\Downarrow$
   a = b (o' $\bigcup$ o"):
   $< owl : Classrdf : ID = "OnionTopping" / >$

2. Subconjunto o "Menos general que", $a \subseteq b$

   a (o'):
   $< owl : Classrdf : ID = "OnionTopping" / >$

   b (o"):
   $< owl : Classrdf : ID = "OnionTopping" >$

$< owl : onProperty >$
$< owl : ObjectPropertyrdf : about = "sostenido_hasTopping"/ >$
$< /owl : onProperty >$
$< /owl : Class >$
$\Downarrow$
a = b (o' $\bigcup$ o"):
$< owl : Classrdf : ID = "OnionTopping" >$
$< owl : onProperty >$
$< owl : ObjectPropertyrdf : about = "sostenido_hasTopping"/ >$
$< /owl : onProperty >$
$< /owl : Class >$

3. El contrario, $a = \neg\ b$, la propiedad *complementOf* vincula una clase con una descripción de clase. Describe una clase para la cual la extensión de clase contiene exactamente aquellos individuos que no pertenecen a la clase objeto de la declaración. Es análoga a la negación lógica: en el siguiente ejemplo se representa la expresión *no carne*.

   $< owl : Class >$
   $< owl : complementOf >$
   $< owl : Classrdf : about = "\#Meat"/ >$
   $< /owl : complementOf >$
   $< /owl : Class >$

c) Enfoques y técnicas de Emparejamiento de Ontologías En la literatura encontramos diversas herramientas que ofrecen formas más o menos automáticas de combinación de ontologías. Las herramientas planteadas suelen resolver alguno de los problemas que plantea dicha operación aunque suelen ser tantos que se hace difícil abarcarlos todos.

Una vez revisada la teoría en la sección 4.3.3, echemos un vistazo al estado de la ciencia y a las aplicaciones web que podríamos usar para realizar la deseada mezcla:

- *OWL Patch*: desarrollado por la Universidad de Manchester, transforma una ontología A en otra B mediante una herramienta online ([of Manchester, 2011]).
- *OWLOntologyMerger API*: aplicación que realiza el *merge* de dos ontologías. Es la clase utilizada por Protégé ([API, 2011]).
- *Mezcla.java*: es la opción en la que se basará nuestra solución final. El resultado es una mezcla de ontologías también en lenguaje OWL DL.

d) Consistencia del Modelo En un entorno de mundo cerrado (CWA *Closed World Assumption*) se presupone que la base de conocimiento con la que trabajamos contiene todos los individuos existentes; sostiene que cualquier cosa de la que no se pueda determinar que es cierta, será falsa y no será necesario declarar dicha falsedad. A la pregunta *¿los leones llevan gafas?* la respuesta será negativa si nuestra tabla de hechos sólo contiene las afirmaciones *los pingüinos no llevan gafas* y *los humanos llevan gafas*.

Sin embargo, en la web semántica, como en el mundo real, los modelos se pueden extender permitiendo que se reutilicen y se amplíen gracias a la hipótesis de mundo abierto (OWA *Open World Assumption*). En OWA se asume que existe información incompleta y no todo estará reflejado en nuestra tabla de hechos, por lo tanto, a la pregunta *¿los leones llevan gafas?*, la respuesta podría ser *no lo sé* si no hay una cláusula explícita que indique que efectivamente *los leones no llevan gafas*.

Partiendo de estas premisas, surgen una serie de **problemas** a la hora de realizar la inferencia como podemos ver en el trabajo de Shack ([Shack, 2013]), siendo C y D clases complejas, KB la base de conocimiento, y a,x individuos,

1. Inconsistencia global de la base de conocimiento: ¿tiene sentido la base de conocimiento? $KB \models \bot$.
2. Inconsistencia de clases: ¿debe una clase estar vacía? $C \equiv \bot$.
3. *Subsumption*, inclusión de clases: estructura de la base de conocimiento $C \sqsubseteq D$.
4. Equivalencia de clases: ¿son dos clases la misma? $C \equiv D$.
5. Clases Disjuntas: ¿son dos clases disjuntas? $C \sqcap D = \bot$.
6. Miembros de clases: ¿está un individuo incluido en una clase? C(a).
7. Generación de instancias: encuentra todos los individuos conocidos de la clase C (*find all x with C(x)*).

Para cada uno de estos problemas debe existir un algoritmo que finalice en un tiempo finito, a ésto se le conoce como **decidibilidad** ([Shack, 2013]), como podrían ser los de Tableau y Resolución (que, aunque se corre el riesgo de que no terminen, se podrían adaptar). Dichos algoritmos permiten mostrar la **insatisfacibilidad** de la base de conocimiento, así que una forma de adaptar dichos algoritmos sería detectando contradicciones posibles en la base de conocimiento (en este caso, en nuestra ontología), por ejemplo, probar la insatisfacibilidad de dicha base. Un modelo será satisfacible si ninguna de sus sentencias se contradicen.

Una forma práctica de detectar inconsistencias, como hemos visto al principio de esta sección es mediante algoritmos de Tableau ([Shack, 2013]):

1. Atómicas: un individuo pertenece a una clase y a su complemento.
2. Cardinalidad: un individuo presenta cardinalidad máxima (*owl:maxCardinality*) y se relaciona con más individuos distintos, el razonador debe asignar equivalencia entre esos individuos para satisfacer la restricción de cardinalidad. La cardinalidad puede ser explícita (*owl:cardinality*, *owl:minCardinality owl:maxCardinality*) por lo que se requiere que cada restricción se incluya en un bloque *owl:equivalentClass, owl:intersectionOf* para asegurar que el conjunto de de propiedades y restricciones es necesario y suficiente, o implícita (*owl:someValuesFrom*)
3. Tipo de datos: un literal viola el rango en la propiedad de tipo de datos.

Estos algoritmos pueden resultar útiles ya que existen numerosas razones por las que pueden aparecer **inconsistencias** en las ontologías, Kalyanpur ([Kalyanpur et al., 2005b]) nos muestra varias de ellas: dificultades para comprender el modelo que se pueden deber a errores que pueden proceder tanto de usuarios como de desarrolladores que pasan por alto ciertas características importantes que tienen que ver con el diseño de ontologías (características del lenguaje OWL, de la taxonomía de las ontologías, de las herramientas necesarias ...), ontologías OWL sin alinear que proceden de errores de modelado que se introducen en las ontologías mediante importaciones debido a diferentes puntos de vista de los creadores de las ontologías. Estos puntos de vista pueden ocasionar conflictos (que se manifestarán en forma de conceptos insatisfacibles); también se pueden producir por migraciones a OWL: si la ontología OWL procede de otros lenguajes como XML, DAML podrían ocurrir problemas o errores al traducir / migrar dichos esquemas.

Siguiendo esta línea, el autor mencionado engloba las inconsistencias en diferentes categorías ([Kalyanpur et al., 2005b]):

- **Inconsistencias Sintácticas:** producidas por diferentes motivos tan diferentes como la sintaxis barroca, el uso de URIs o RDF/XML. La mayoría de ellas se pueden corregir usando validadores de código RDF y XML pero para OWL DL existe otra capa de estructura sintáctica que se corresponde con el grafo RDF y el número de restricciones impuestas por el mismo. Estas restricciones no se suelen tener en cuenta y se violan con facilidad y el impacto final es importante, tanto que se puede importar un documento OWL Lite y el resultado puede ser un documento OWL Full (o viceversa).

Este tipo de inconsistencias están controladas por el propio lenguaje OWL DL, el más mínimo cambio en la estructura de la ontología provocará una variación en la expresividad, por ejemplo, al añadir un individuo sin padre, se provoca un error en el análisis sintáctico.

- **Inconsistencias Semánticas:** un razonador puede detectar errores semánticos (clases insatisfacibles) en una ontología sintácticamente correcta. Una clase insatisfacible es aquella que no puede tener individuos, por ejemplo, la clase A es insatisfacible si es una subclase de C y de $\neg C$ al mismo tiempo, en cuyo caso se produciría una contradicción. Determinar por qué un concepto es una ontología es insatisfacible puede ser una tarea difícil incluso para expertos en ontologías ya que el problema se puede agravar a medida que aumenta el número de axiomas en la ontología. Una ontología es inconsistente si presenta una contradicción en sus instancias. En razonadores basados en algoritmo de Tableau ([Kalyanpur et al., 2005a]), el test de insatisfacibilidad se reduce al test de consistencia: una de las clases se elige como clase de prueba y se realiza el control de consistencia sobre la base de conocimiento resultante.

  Si una clase S es subclase de C y de su disjunta D, se presentará una inconsistencia.

  Al introducir una clase como subclase, a su vez, de dos clases disjuntas, se puede comprobar que el razonador *pellet* la marcará como inconsistente.

- **Defectos de Modelado:** se trata de defectos que no tienen por qué ser inválidos sintáctica o semánticamente, sino que se mostrarán en forma de discrepancias en la base de conocimiento o como resultados imprevistos del modelado. En este tipo de inconsistencias habrá que tener cuidado con la revisión del dominio.

  - El razonador puede descubrir inferencias no deseadas, por ejemplo: *padres con al menos tres hijos* es una subclase de *padres con al menos dos hijos*, incluso si no existe la relación explícita.
  - Pérdidas del tipo en las declaraciones, es decir, se utiliza un recurso pero no se declara de forma explícita, por ejemplo, si se tiene la relación $<$*John hasParent Mary*$>$, se sabe que *hasParent* es una Propiedad de Objetos OWL y se puede inferir que *John* y *Mary* son ambos individuos OWL. En estos casos, el razonador inferirá la correspondiente implicación, pero la ausencia de información explícita se puede considerar un defecto.
  - En algunos casos, las redundancias pueden existir en la base de conocimiento, por ejemplo: un axioma se vincula con otros conjuntos de axiomas de la misma base. Este caso depende de si se desea o no considerar la redundancia como un defecto.

– Hay muchos casos de clases atómicas sin usar, propiedades sin referencias en la base de conocimiento que se pueden considerar como datos extraños (por ejemplo, un término no se usa explícitamente en un axioma en la base).

No es nuestra misión recorrer todas las inconsistencias posibles de un modelado con ontologías sino aportar algún ejemplo que nos ayude a la hora de realizar los experimentos.

Hasta aquí, presentada la parte teórica de nuestro ensayo, la apoyaremos con un ejemplo práctico en el que mostraremos el funcionamiento del algoritmo de Búsqueda de Contradicciones y, posteriormente en el capítulo 5, expondremos los experimentos en los que se demuestra la eficiencia del mismo.

4.5 Ejemplo de utilización del Algoritmo de Búsqueda de Contradicciones en Textos

Para finalizar este capítulo, mostraremos un ejemplo de utilización del Algoritmo de Búsqueda de Contradicciones dando una visión más práctica, desde el nivel de lenguaje OWL.

Continuando con nuestra metodología representada gráficamente en la figura 20 y desarrollada en la sección 4.3 elaboraremos el proceso sobre la siguiente frase:

Born in Philadelphia in 1884, Thomas Eakins is one of America's few indisputably great painters.

---

1. **Paso 1: Construcción de un Corpus Inicial**

   Utilizando una herramienta de etiquetado semántico del texto, los autores Fensel y Morozova ([Fensel and Morozova, 2010]), convierten el párrafo anterior en la ontología mostrada en la imagen (32).

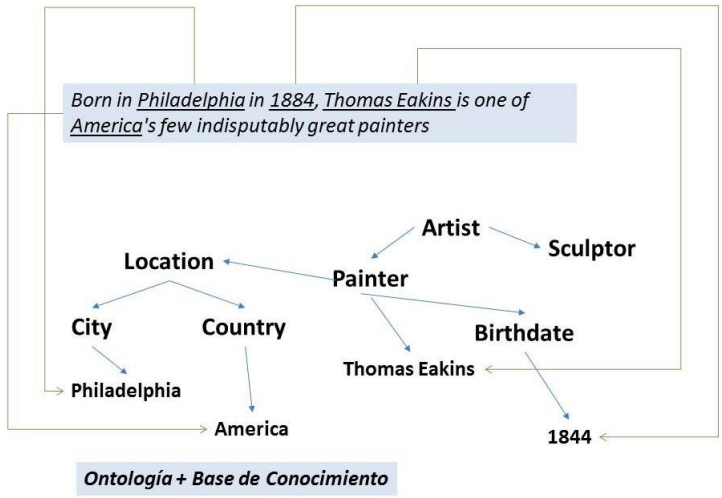

Figura 32: Ontología extraída a partir de texto ([Fensel and Morozova, 2010])

Reconstruimos la ontología de la imagen 32 en Protégé y añadimos una serie de elementos imprescindibles para que la expresividad sea SHOIN(D). Si sólo representamos los elementos de la imagen: el conjunto de clases C = *(Artist, Painter, Sculptor, Location, Birthdate, City, Country)*, el conjunto de instancias I = *(Philadelphia, America, Thomas Eakins, 1844)*, las subclases de *Artist*, *Painter, Sculptor*, las subclases de *Painter*, *Birthdate* y, por último, las subclases de *Location*, *City, Country*. La expresividad que obtendríamos sería AL, insuficiente para nuestro trabajo.

La sintaxis básica OWL para nuestra ontología *o_1* proporcionada por Protégé, sería:
Ontology(<http:www.semanticweb.orgontologies2015 9 o_1.owl>

Declaration(Class(:Artist))
Declaration(Class(:Birthdate))
SubClassOf(:Birthdate :Painter)
Declaration(Class(:City))
SubClassOf(:City :Location)
Declaration(Class(:Country))
SubClassOf(:Country :Location)
Declaration(Class(:Location))
SubClassOf(:Location :Painter)
Declaration(Class(:Painter))

SubClassOf(:Painter :Artist)
Declaration(Class(:Sculptor))
SubClassOf(:Sculptor :Artist)
Declaration(NamedIndividual(:1844))
ClassAssertion(:Birthdate :1844)
Declaration(NamedIndividual(:America))
ClassAssertion(:Birthdate :America)
ClassAssertion(:Country :America)
Declaration(NamedIndividual(:Philadelphia))
ClassAssertion(:Birthdate :Philadelphia)
ClassAssertion(:City :Philadelphia)
Declaration(NamedIndividual(:Thomas_Eakins))
ClassAssertion(:Painter :Thomas_Eakins)

A dichas declaraciones, añadiremos una serie de elementos para conseguir aumentar la expresividad, hasta conseguir SHOIN(D), según las explicaciones del anexo B.1:

- ALC: Introducimos una clase disjunta, nos servirá para representar la negación compleja
  DisjointClasses(:Location :Painter)
  DisjointClasses(:Painter :Location)
- S: Propiedades transitivas, creamos una propiedad de objeto y la etiquetamos como transitiva
  ObjectPropertyDomain(:hasCapital :Country)
  Declaration(ObjectProperty(:hasCapital))
  TransitiveObjectProperty(:hasCapital)
- H: creamos una subpropiedad de nuestra propiedad de objeto
  Declaration(ObjectProperty(:hasFinancialDistrict))
  SubObjectPropertyOf(:hasFinancialDistrict :hasCapital)
- O: para conseguir este símbolo, añadiremos, por ejemplo, un nuevo individuo *Washington* y un valor a una propiedad de objeto

  Declaration(NamedIndividual(:Washington))
  ClassAssertion(:City :Washington)
- I: ahora, añadiremos una propiedad inversa a *:hasCapital*,
  Declaration(ObjectProperty(:isPlacedAt))
  InverseObjectProperties(:hasCapital :isPlacedAt)
  ObjectPropertyDomain(:isPlacedAt :City)

ObjectPropertyRange(:isPlacedAt :Country)

- *N*: restricciones de cardinalidad, utilizaremos la restricción *al menos 1*, *EquivalentClasses(:City DataMinCardinality(1 :hasZipCode))*
- *(D)*: Dominio, añadimos un rango a una propiedad *SubClassOf(:Country ObjectHasValue(:hasCapital :Washington))* *Declaration(DataProperty(:hasZipCode))* *DataPropertyRange(:hasZipCode xsd:int)*

Una vez construida nuestra ontología ejemplo, que nos servirá como base para el corpus inicial sobre el que iteraremos el algoritmo, necesitaremos un segundo documento inicial al que le añadiremos una *contradicción* respecto de la primera ontología.

Caso 1: La contradicción que añadiremos será la que represente al siguiente fragmento textual:

*Born in **Chicago** in 1884, Thomas Eakins is one of America's few indisputably great painters.*

Para la construcción de la ontología *o_2* que presenta la contradicción repetiremos el proceso descrito en este **Paso1**.

Caso 2: La contradicción, esta vez, la realizaremos directamente sobre OWL, añadiendo una misma subclase *Studio* como hija de dos clases disjuntas:

DisjointClasses(:Location :Painter)
DisjointClasses(:Painter :Location) SubClass(:Studio :Painter)
SubClass(:Studio :Location)

Para la construcción de la ontología *o_2*, añadiremos estos axiomas al modelo descrito en el **Paso1**.

2. **Paso 2: Comprobación de la Consistencia**

   Construido nuestro conjunto inicial de forma robusta, se comprobará la consistencia de ambas ontologías, es decir, la existencia o no de contradicciones.

   El razonador *pellet*, al que le pasaremos ambas ontologías, devuelve el conjunto de clases insatisfacibles (si existe alguna clase dependiente de *owl:Nothing* indicará que existen contradicciones en ellas.

Para ambas ontologías (*o_1* y *o_2*), las clases insatisfacibles devueltas por el razonador son el conjunto vacío *owl:Nothing*, lo que quiere decir que no presentan contradicciones.

Para detectar posibles incongruencias en el texto, objetivo de este trabajo, detectaremos no sólo las inconsistencias definidas en la sección 4.4.5 sino que, además, incluiremos un nuevo tipo de inconsistencias que podrán ser evaluadas por un humano como son aquellas clases, que, siendo la misma, contienen diferentes individuos.

3. **Paso 3: Fusión de los Candidatos**

   Realicemos la fusión de los candidatos *o_1* y *o_2* mediante una clase Java de la librería *owlapi* ([API, 2011]) que nos devolverá una nueva ontología con la mezcla de los axiomas de ambas ontologías.

   Caso 1: siendo ambas ontologías consistentes, nuestro algoritmo no devolverá una contradicción, puesto que no existen clases insatisfacibles según el razonador y las premisas establecidas en la sección 4.4.5, pero sí devolverá una serie de clases incongruentes:

   Ontología o_1, Clases: City, Individuos: ***Philadelphia, Ohio***
   Ontología o_2, Clases: City, Individuos: ***Chicago, Ohio***

   Caso 2: al aplicar el razonador sobre el resultado de la mezcla de los candidatos *o_1* y *o_2* (*Ontología o_1o_2*) obtendremos un conjunto de clases insatisfacibles, es decir, una contradicción:

   Ontología o_1o_2, Clases: Location, Painter, Studio

4. **Paso 4: Finalización**

   Nuestro algoritmo devolverá, en el primer supuesto, las ontologías: *o_1, o_2 y o_1o_2* todas ellas sin contradicciones, si el conjunto inicial es el **Caso 1**. Mientras que si se trata del **Caso 2**, el resultado de candidatos será *o_1, o_2* y de contradicciones *o_1o_2*.

4.6 Conclusión

En este capítulo hemos propuesto un algoritmo para la búsqueda de contradicciones en textos, y lo hemos aplicado sobre fragmentos textuales reales, traducidos a ontologías, algunos de los cuales presentan contradicciones (otros, por el contrario o

bien no presentan contradicción o bien no se puede afirmar que exista tal incoherencia). Hemos descrito tanto la generación del corpus inicial como el razonamiento que devuelve si existen o no contradicciones en el modelo, a lo que hemos llamado *consistencia*. También hemos descrito la fusión de ontologías, necesaria para completar el proceso de búsqueda de contradicciones.

La ejemplificación de búsqueda de contradicciones también la hemos llevado al ámbito del lenguaje OWL, elegido para la representación de las ontologías, y sobre un nuevo fragmento de texto hemos aplicado la metodología de búsqueda de contradicciones.

En el siguiente capítulo presentaremos una batería de experimentos sintéticos realizados sobre el algoritmo de Búsqueda de Contradicciones en Textos.

Capítulo 5 Experimentación

You know nothing, Jon Snow
(George RR Martin [Martin, 2000])

El objetivo principal de este estudio será presentar varios experimentos para probar el algoritmo del capítulo 4 ejemplificando algunos de los problemas que hemos planteado en este trabajo. Tal y como se ha mencionado anteriormente, expondremos una técnica de Minería de Textos y mostraremos la capacidad inherente de las ontologías para constituir Formas Intermedias de representación textual.

La aplicación desarrollada para responder a dichas preguntas la hemos construido con ciertos prerrequisitos para su correcto funcionamiento, entre ellos, la existencia de un corpus inicial de ontologías, bien obtenidas de un repositorio, bien generadas automáticamente. A partir de dicho corpus, nuestra aplicación localizará información textual desconocida previamente en forma de *contradicciones*.

En nuestra experimentación propondremos, asimismo, una aplicación que genera automáticamente ontologías de expresividad SHOIN(D) según unos parámetros de entrada iniciales, las cuales servirán como corpus inicial. Este conjunto inicial de ontologías está basado en la ontología *pizza.owl* ([University, 2006]) y en los ejemplos construidos en el capítulo 4, las cuales cumplen con los requisitos de expresividad y consistencia deseados según la aplicación Protégé 4.1.

La consistencia del conjunto inicial (en este capítulo nos referiremos a conjunto consistente como aquel libre de contradicciones) y un algoritmo eficiente y robusto constituirán este capítulo en el que presentaremos un marco de trabajo desarrollado en Java para trabajar con ontologías escritas en OWL DL.

Lo estructuraremos presentando nuestra aproximación a la Minería de Textos,

describiremos la arquitectura de nuestro sistema y discutiremos los experimentos con los que hemos probado nuestra aproximación. Al final del capítulo mostraremos dichos experimentos y su evaluación. En las siguientes secciones describiremos los experimentos realizados.

5.1 Descripción de la Experimentación

La Minería de Textos, tal y como hemos repasado en los capítulos 2 y 4, tiene como requisito imprescindible, proporcionar, a la información textual, una representación intermedia que la haga computable de un modo asequible.

Hemos mencionado, asimismo en el capítulo 4, que las técnicas de extracción de representaciones intermedias sobre texto, en concreto, ontologías, no están aún desarrolladas en su totalidad.

Según hemos visto en la sección 4.4.5, para probar la consistencia del conjunto inicial, debe existir un algoritmo que finalice en un tiempo finito, por lo que probaremos la eficiencia y mediremos el tiempo de nuestro **algoritmo de Búsqueda de Contradicciones en Textos**. Más aún, una de las tareas de nuestro algoritmo es la de localizar contradicciones en el conjunto inicial, de este modo, al comprobar la existencia de contradicciones en dicho corpus, se demostraría la insatisfacibilidad del elemento que la contenga.

Para esta segunda misión, mostraremos las inconsistencias que se detecten (como puede ser el hecho que un individuo pertenezca o no a una clase). Debido a la enorme cantidad de factores a tener en cuenta, nos basaremos en las inconsistencias que detectan las clases Java de las librerías *owlapi* ([API, 2011]). Para la parte experimental de este trabajo hemos desarrollado una aplicación; *ContradictionsFinder*: un marco de trabajo desarrollado en Java en un entorno *software* con ontologías escritas en OWL DL. Las características de esta representación se amplían en el anexo B.1.

En base a estas premisas, realizaremos una serie de asunciones en nuestros experimentos (algunas de ellas propias y otras basadas en las de las presentadas por Fernández en su artículo ([Fernández-López et al., 2006])):

1. El origen de datos de nuestra aplicación de Minería de Textos será un corpus de ontologías, los cuales proporcionan estructura al texto.
2. Asumiremos que nuestra ontología será un conjunto de clases y de sus com-

ponentes básicos (propiedades de objetos, propiedades de tipos de datos e instancias).

3. El resultado de la mezcla de ontologías sólo será consistente si no presenta una contradicción.
4. Las relaciones no tendrán atributos.

Entre las características de nuestros experimentos se incluyen:

- Las ontologías generadas automáticamente se nombrarán como *o_i* siendo *i= 1..n* y siendo *n* el tamaño del corpus inicial.
- El formato de la mezcla será el definido por las librerías *owlapi* ([API, 2011]).
- La consistencia la gestionará el razonador *pellet* ([Clark and LLC, 2011]).
- La cardinalidad de la mezcla será 1:1, un término de la primera ontología se emparejará, al menos, con un término de la segunda ontología. Se pueden ampliar más los detalles de la integración de ontologías en este trabajo, en el apartado 4.3.3.
- Asumiremos que si una relación entre dos componentes no existe, esos componentes no están relacionados.

Consideraremos las siguientes formas de construcción / obtención de ontologías para completar nuestro corpus inicial:

- Tomar las ontologías de un repositorio consistente y bien construido.
- Creación de elementos aleatorios a partir de una ontología inicial tomada como base.
- Extracción de los elementos de una ontología y la posterior generación de elementos para la construcción de una ontología.
- Creación manual de una ontología, en los ejemplos desarrollados en el capítulo 4, para cada elemento marcado durante la extracción de términos se creaba un concepto en la ontología, y además, debíamos completar el modelo con nuevos conceptos y elementos para conseguir la expresividad deseada.

El método de construcción del corpus inicial que adoptaremos será aquel en el que crearemos una serie de ontologías ejemplo a las cuales se les realizará una *inserción*

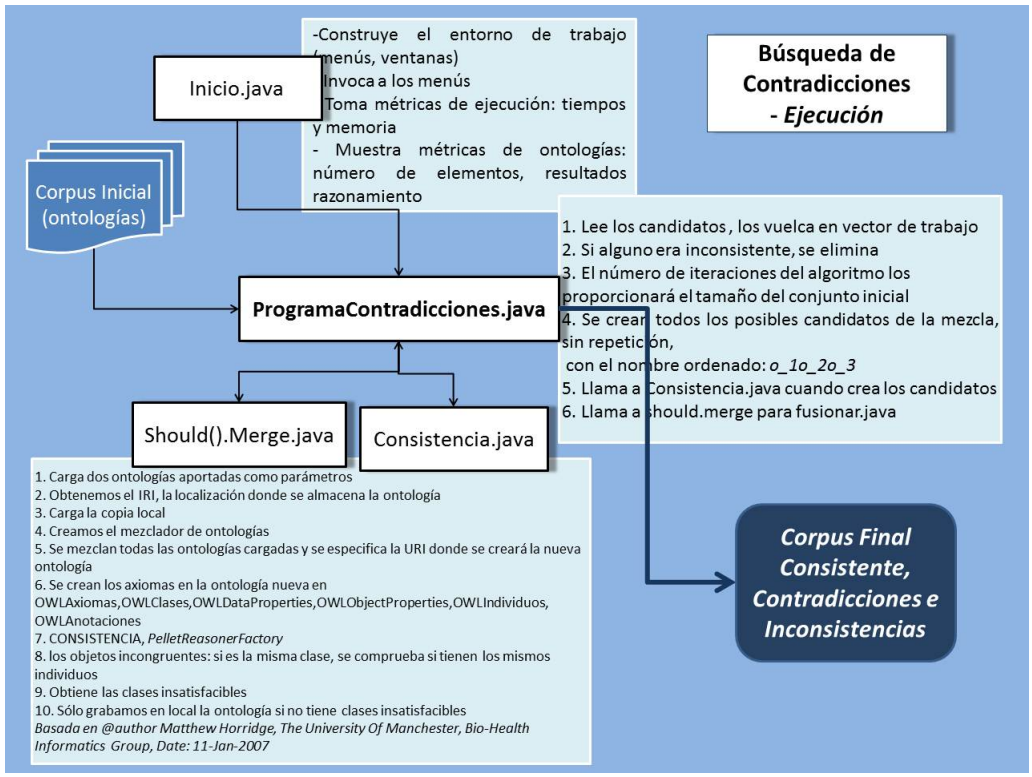

Figura 35: Proceso de Búsqueda de Contradicciones en Textos - Ejecución

de contradicciones, proceso durante el cual el usuario podrá indicar el número de contradicciones que desee que contenga el corpus, y se manifestará como el hecho de que un porcentaje de las taxonomías iniciales contendrán contradicciones.

5.2 Arquitectura del Sistema

Hemos implementado un sistema gracias al cual tanto la **Búsqueda de Contradicciones** como la **generación del corpus inicial** se hacen realidad. Los diferentes módulos implementados en Java y su funcionalidad se puede ver en la figura 35.

En el diagrama observamos cómo, un módulo inicial (*Inicio.java*) que construye el entorno de trabajo (menús, ventanas, directorios, ficheros de parámetros), invoca a los menús, toma métricas de ejecución (tiempos y memoria) y métricas de ontologías (número de clases, de propiedades de objetos, de propiedades de tipos de

datos, de instancias...) y muestra los resultados del razonamiento.

La clase *ProgramaContradicciones.java* será el núcleo central del algoritmo de **Búsqueda de Contradicciones**: leerá los candidatos objeto de la Minería de Textos y los volcará en estructuras intermedias de trabajo; si alguno de ellos es inconsistente, hecho que se constatará mediante la utilización del razonador *pellet* desde la clase *Consistencia.java*, se elimina de la colección pero se conserva para ser evaluado a posteriori. Nuestro algoritmo iterará, creando todos los posibles candidatos de la mezcla, sin repetición, y proporcionando un nombre al resultado de la mezcla, el cual ordenaremos alfabéticamente (por ejemplo, *o_1o_2o_3*).

Para realizar la mezcla de ontologías, se invoca a la clase *shouldMergeOntologies* ([Horridge et al., 2004]) a la que se le han realizado algunas modificaciones para ajustarlas a nuestras necesidades. Esta clase realiza la fusión de ontologías basándose en la implementación de la mezcla de ontologías que proporciona las librerías *owlapi* ([API, 2011]), el cual crea una ontología que contiene la unión de los axiomas de las dos ontologías padre. Los diferentes tipos de integración de ontologías se han presentado en el capítulo 4.

Por último, recordemos según lo mencionado en el capítulo anterior, que el número de iteraciones del algoritmo los proporcionará el tamaño del conjunto inicial.

A partir de este planteamiento software, realizaremos nuestros estudios de eficiencia, tiempo, memoria y medidas de elementos de ontologías que presentaremos a partir de ahora. La evaluación de las ontologías la realizaremos en función del costo que supone su implementación, la utilización de métricas y el análisis de la ontología desde el punto de vista de su consistencia semántica.

5.3 Generación de un Corpus Inicial

Mediante la construcción del corpus inicial de ontologías no pretendemos que las ontologías generadas, si se analizan con exhaustividad, alcancen el grado de perfección que se puede requerir en una ontología real con la que se represente un área concreta del conocimiento. A estas ontologías les pediremos que, al evaluarlas con una herramienta de gestión de ontologías, como puede ser Protégé, sean legibles y tengan una expresividad exacta a la requerida, la que hemos mencionado a lo largo de nuestro trabajo, SHOIN(D).

La aplicación que genera el corpus inicial, en la figura 36, permite generar un conjunto de ontologías ejemplo en base a una serie de parámetros:

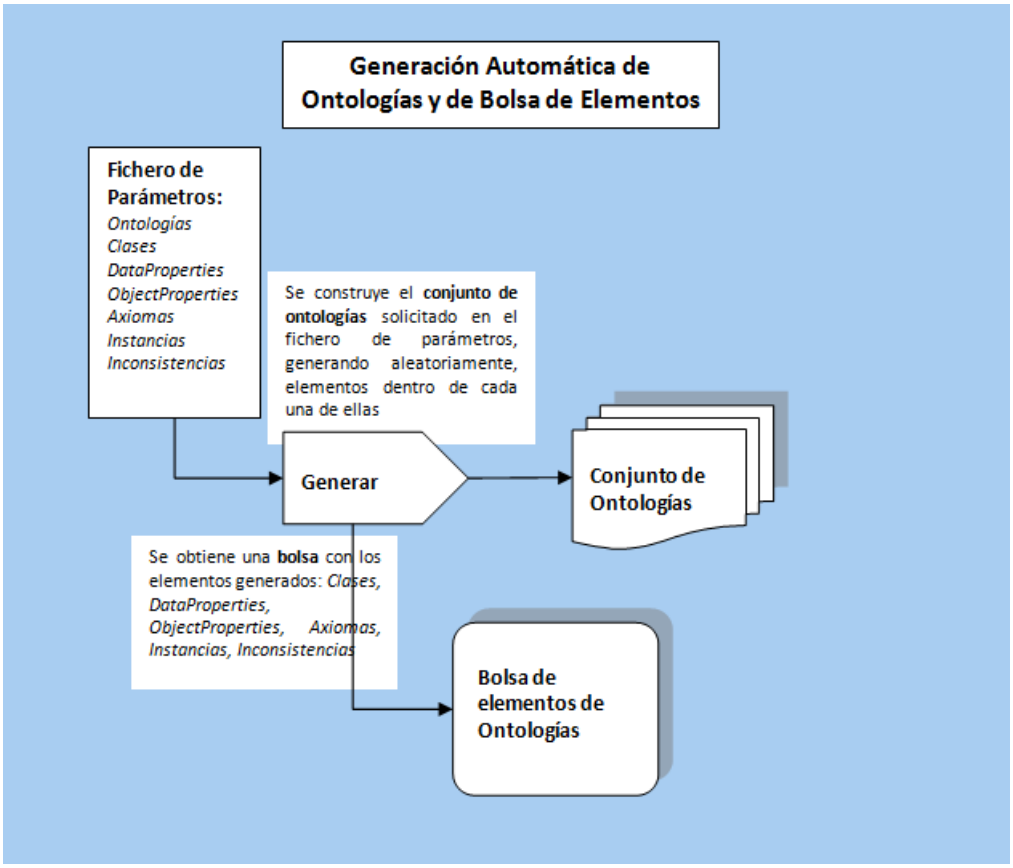

Figura 36: Generación automática de ontologías

- Nombre de Ontologías Generadas: Nomenclatura que llevará cada una de las ontologías del conjunto ejemplo, seguido por un número correlativo. Por defecto, *o_*.owl*.
- Número de Ontologías, *n*: el número de ontologías que se generará será el valor introducido por parámetro en cada experimento. Coincidirá con el número de iteraciones del algoritmo.
- Número de Clases, *c*: El número de Clases de cada una de las ontologías generadas se corresponderá con un valor aleatorio entre 1 y el número introducido aquí por parámetro. El número aleatorio devuelto estará entre dos valores dados (*desde, hasta*), con una semilla aleatoria. Utilizamos la clase *java.util.random* por ser más flexible que la clase *Math.random* ([Franco García, 2004]), véase Anexo E.
- Número de Tipos de Objeto, *b*: El número de *ObjectProperties* de cada una de las ontologías generadas se corresponderá con un valor aleatorio entre 1 y el valor introducido.
- Número de Tipos de Datos, *d*: El número de *DataProperties* de cada una de las ontologías generadas se corresponderá con un valor aleatorio entre 1 y el valor introducido.
- Número de Instancias, *i*: El número de Instancias dentro de cada ontología generada se corresponderá con un valor aleatorio entre 1 y el valor introducido.
- Número de Inconsistencias, *t*: El conjunto inicial de ontologías podrá o no contener inconsistencias. Por defecto, 0, lo que indicará que no presenta inconsistencias. Cualquier otro número se corresponderá con el % de ontologías que se desea que presenten inconsistencias.

Presentamos los conjuntos iniciales, el conjunto O_i, con un máximo de 10 ontologías, estará basado en la ontología *pizza.owl ([University, 2006])*, de expresividad SHOIN, a la que le añadiremos una serie de elementos hasta completar la expresividad SHOIN(D), la deseada en OWL DL. En este caso, añadiremos una propiedad de datos para alcanzar dicha expresividad. En la tabla 18 veremos el número de elementos de la ontología *pizza.owl* comparada con la ontología *o_1.owl* construida en la capítulo 4. El número de clases de la ontología *o_1.owl* es considerablemente inferior que el de *pizza.owl*, la razón, además de agilizar los tiempos de preprocesamiento y la memoria empleada, es la de simplificar los procesos.

Elementos	***pizza.owl***	***o_1.owl***
No. de Clases	100	7
No. de Prop. Objeto	8	3
No. de Prop. Datos	1	2
No. de Instancias	5	5
Expresividad	SHOIN(D)	SHOIN(D)

Tabla 18: Elementos de las ontologías base de nuestro corpus *pizza.owl* y *o_1.owl*

Ontología	**Número de Clases**	**Número de Propiedades de Objeto**	**Número de Propiedades de Datos**	**Número de Instancias**
$\mathbf{o}_0$	7	4	4	6
$\mathbf{o}_1$	7	3	3	6
$\mathbf{o}_2$	7	4	4	6
$\mathbf{o}_3$	7	4	4	6
$\mathbf{o}_4$	7	4	4	5
$\mathbf{o}_5$	7	4	4	6
$\mathbf{o}_6$	7	3	4	6
$\mathbf{o}_7$	7	4	4	6
$\mathbf{o}_8$	6	4	4	6
$\mathbf{o}_9$	7	4	4	6

Tabla 19: Elementos de las ontologías del corpus inicial generado automáticamente

Los elementos del conjunto inicial que servirá para alimentar nuestro algoritmo los podemos ver en la tabla 19. Esta tabla se ha construido a partir de una serie de parámetros establecidos por el usuario. Gracias a unos cómputos aleatorios y unos porcentajes se han generado los elementos descritos para cada una de las ontologías, de expresividad SHOIN(D). El recuento de elementos de las ontologías (número de clases, instancias, propiedades de datos y de objetos) lo hemos realizado con Protégé [University, 2015].

En la figura 37 podemos observar de forma gráfica la distribución de los elementos dentro de cada ontología.

Para la construcción de los conjuntos iniciales, que, tal y como hemos comentado podemos observar en la tabla 19, y dados los parámetros iniciales introducidos a solicitud del usuario, a saber, 20 ontologías, 12 clases o 4 propiedades de objeto, nuestra aplicación generará un valor aleatorio entre 1 y el número indicado, poblando las ontologías con un número comprendido entre el inicial y el número elegido. Por lo tanto, en la tabla 19 observamos que se han generado automáticamente y a partir de los datos introducidos por el usuario, diez ontologías con un número de clases

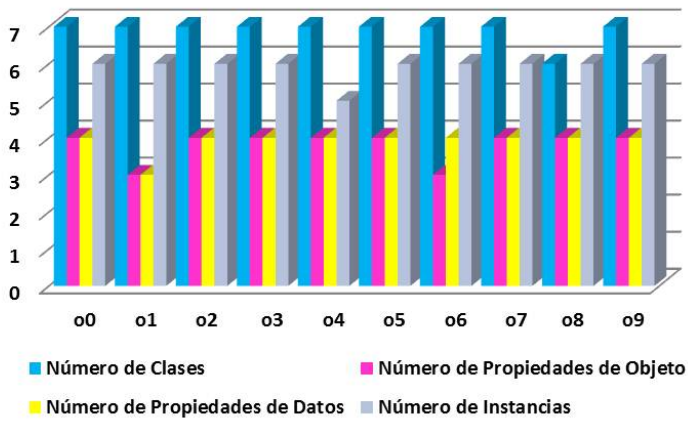

Figura 37: Elementos de las ontologías del corpus inicial generado automáticamente

comprendido entre 6 y 7, 3 ó 4 propiedades de objeto y de datos y entre 5 y 6 instancias por ontología.

Un ejemplo de cómo se comportaría el programa de generación del corpus inicial de ontologías lo podemos ver a continuación, donde, una vez fijado el número de ontologías por la constante (20 en este caso), el número de inconsistencias representará que sólo un tanto por ciento de las ontologías origen contendrán inconsistencias;

1. Nombre del Fichero de Salida: o_.
2. Número de Ontologías: 20.
3. Número de Clases, valor aleatorio entre 2 y 12: 2.
4. Número de Propiedades de Objetos, valor aleatorio entre 2 y 4: 3.
5. Número de Propiedades de Tipo de Datos, valor aleatorio entre 1 y 3: 3.
6. Número de Instancias: valor aleatorio entre 3 y 9: 4.
7. Número de Inconsistencias, 30% de 20 = 6.

Diferentes ejecuciones del programa de generación de corpus generarán diferentes valores en nuestros parámetros iniciales y, por ende, diferentes elementos resultado. Veamos la tabla 20 en la que se muestra la variabilidad de los conjuntos en las diferentes iteraciones, por ejemplo, si aplicamos el fichero de generación de parámetros anterior sobre el primer conjunto de datos (CD 1) *solicitado*, el resultado del fichero sería el conjunto de datos 1 *obtenido*:

	Solicitado						Obtenido					
Cjtos de Datos	**No. Onto**	**No. Clas**	**No. Prop Obj**	**No. Prop-Dat**	**No. Inst**	**No. Inc**	**No. Onto**	**No. Clas**	**No. Prop Obj**	**No. Prop-Dat**	**No. Inst**	**No. Inc**
CD 1	10	12	4	2	9	30	10	8	4	3	5	2
CD 2	10	10	4	2	9	20	10	7	3	3	8	2
CD 3	7	10	4	5	6	10	7	7	4	3	3	0
CD 4	7	10	4	5	6	50	7	8	2	4	3	3
CD 5	5	4	4	2	9	2	5	2	2	3	3	0
CD 6	5	10	4	2	9	50	5	7	4	3	6	2
CD 7	3	12	4	5	6	0	3	5	3	4	3	0
CD 8	3	12	4	5	9	100	3	3	3	3	4	3
CD a	20	10	4	5	6	10	20	7	2	3	4	2
CD b	20	20	4	5	6	10	20	15	3	3	5	2

Tabla 20: Elementos de las ontologías Solicitados / Obtenidos

1. Nombre del Fichero de Salida: o_.
2. Número de Ontologías: 10.
3. Número de Clases, valor aleatorio entre 2 y 12: 8.
4. Número de Propiedades de Objetos, valor aleatorio entre 2 y 4: 4.
5. Número de Propiedades de Tipo de Datos, valor aleatorio entre 1 y 2: 2.
6. Número de Instancias: valor aleatorio entre 3 y 9: 5.
7. Número de Inconsistencias, 30% de 10:valor entre 0 y 30% de 10 = 2.

En la secuencia de imágenes 38, 39, 40, 41, 42 vemos también cómo se optimiza el número de elementos conseguidos en las ontologías al aplicar una variable aleatoria en su construcción. En las imágenes se aprecia cómo, para los conjuntos de datos seleccionados en nuestra experimentación de (CD1 - CD8, CDa y CDb) con diferentes ontologías, las series de datos representadas en la tabla 20 varían del valor seleccionado al obtenido mediante la aleatoriedad anteriormente expuesta, así, en la figura 38, para los conjuntos de datos CD7 y CD8, ambos con un número de clases estimado de 12, se obtiene con nuestra experimentación entre 5 y 3 clases, respectivamente, siendo ambos conjuntos de 3 ontologías. Si analizamos el número de instancias, figura 41, la variabilidad es mayor, mientras que inicialmente se esperaban entre 6 y 9 para cualquiera de los conjuntos de datos, se han obtenido cifras variables entre 3 y 8, todas en el rango inicial. También es remarcable el cambio producido entre la monotonía de los datos iniciales que se presentaban como propiedades de objetos, figura 39, 4 para todos los conjuntos, y los obtenidos, entre 2 y 4, independientemente del número de ontologías de cada conjunto.

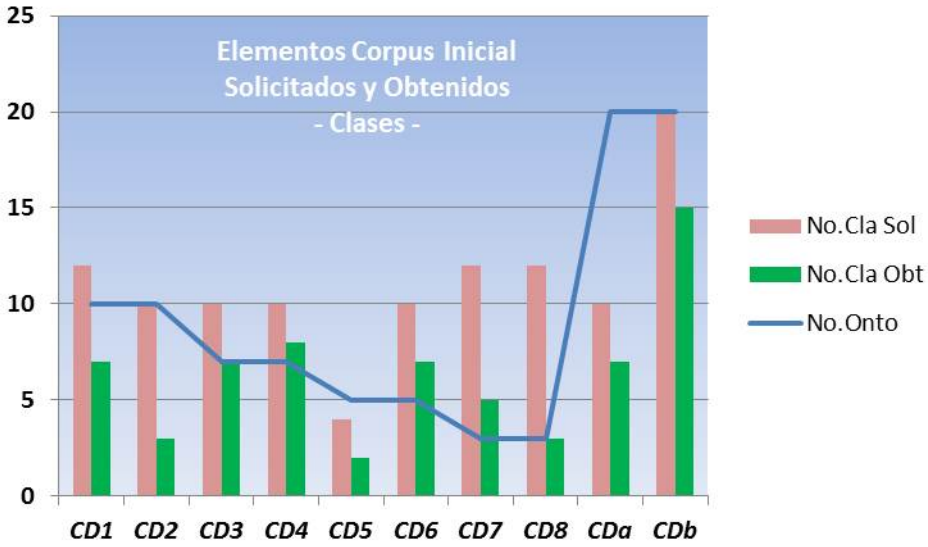

Figura 38: Elementos de las ontologías del corpus inicial - Clases

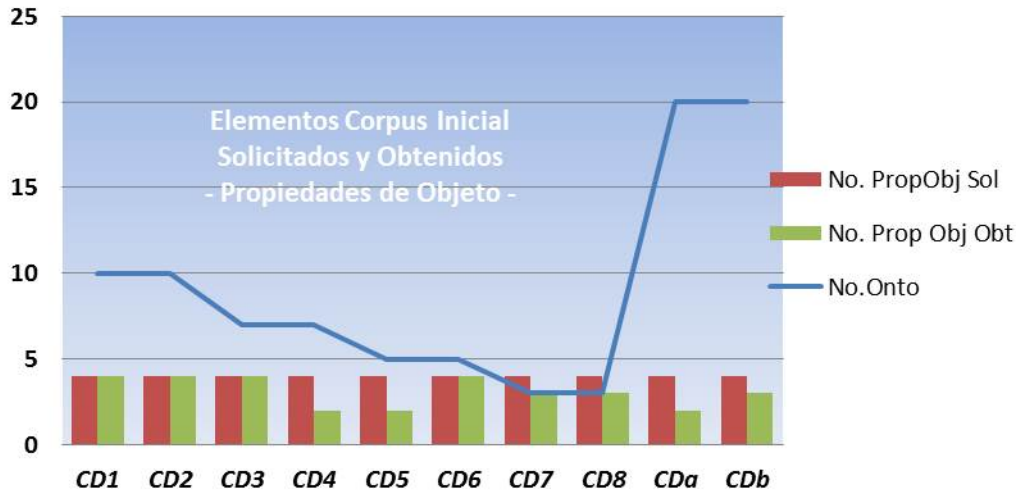

Figura 39: Elementos de las ontologías del corpus inicial - Propiedades de Objetos

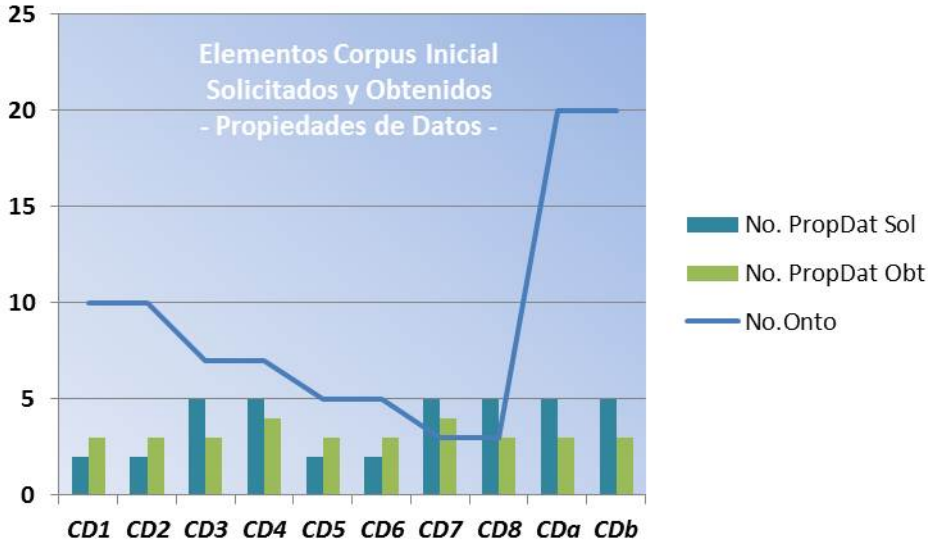

Figura 40: Elementos de las ontologías del corpus inicial - Propiedades de Datos

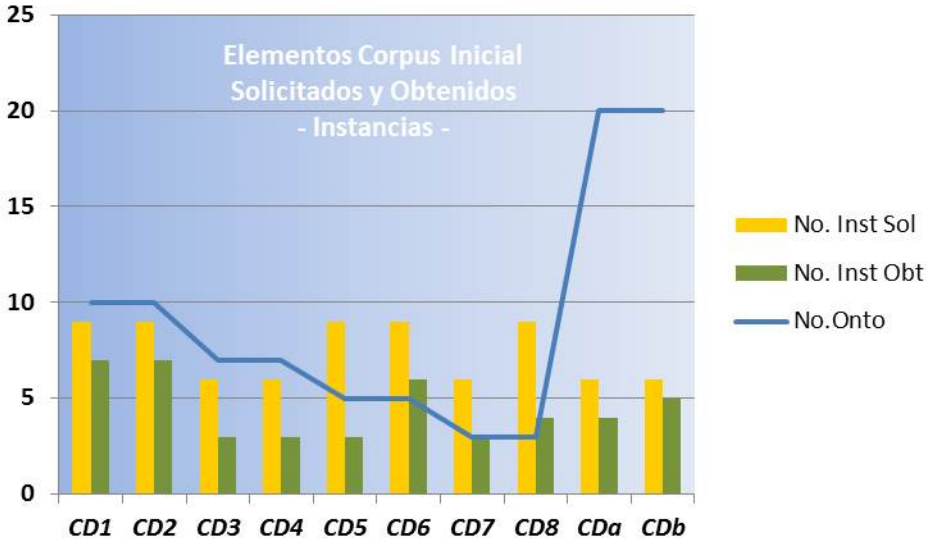

Figura 41: Elementos de las ontologías del corpus inicial - Instancias

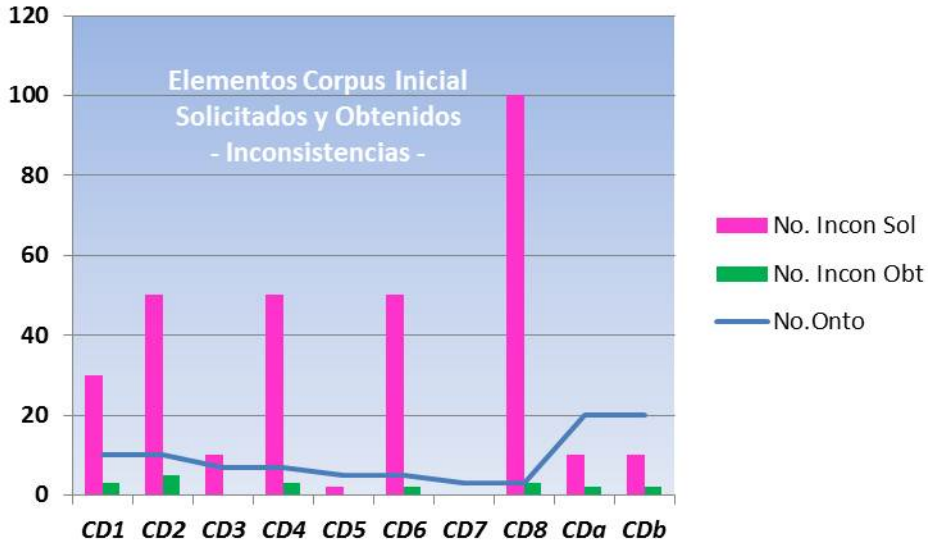

Figura 42: Elementos de las ontologías del corpus inicial - Inconsistencias

Considerando nuestra implementación, la evaluación general de la aproximación se puede ver como una tarea cambiante, de hecho, no hay una medida estándar de evaluación que se ajuste al proceso de extracción de ontologías ([Gaeta et al., 2011]), por lo que trataremos de evaluar nuestro estudio midiendo tiempo, memoria y métricas de ontología basadas en la incrementalidad del razonador.

Otro posible análisis de los resultados consiste en comprobar cómo afecta la inclusión de ontologías inconsistentes en el corpus inicial al conjunto último de ontologías generado. Según podemos comprobar en la figura 43, el tamaño de corpus disminuye drásticamente al introducir contradicciones. Veamos, por ejemplo el caso de un conjunto inicial de diez ontologías, para las que las combinaciones posibles, una vez tomados los elementos de 2 en 2, de 3 en 3, ... hasta de 10 en 10 sería 1,023 si todos los elementos del conjunto fueran consistentes, se queda en 127 elementos si se introducen 3 ontologías con contradicciones o sólo en 13 si se introducen 5. No debemos olvidar que nos encontramos ante la generación de un corpus inicial en el que hemos recogido los tiempos medios y la memoria media de tres diferentes ejecuciones de nuestro constructor de ontologías.

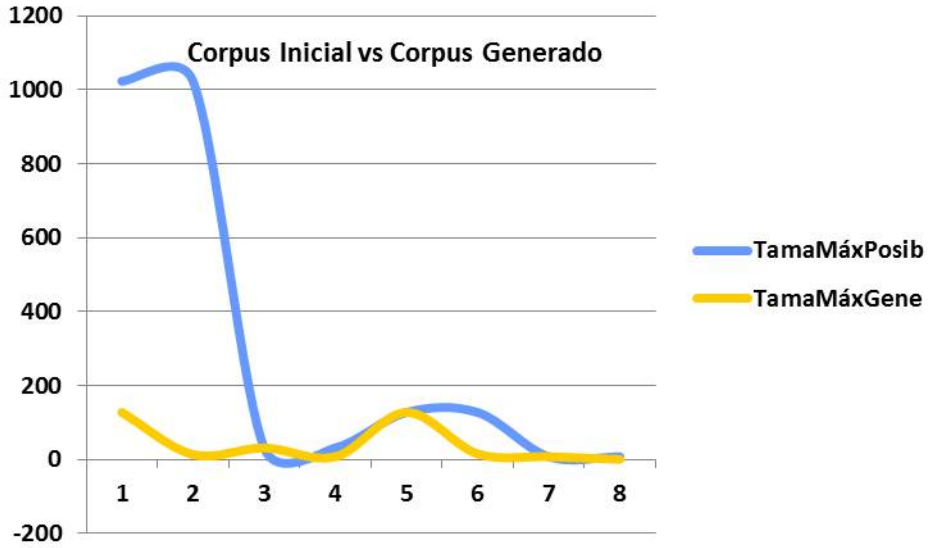

Figura 43: Diferencias entre Corpus Inicial y Corpus Generado sin Contradicciones

5.4 Análisis de los Factores Temporal y Memoria en la Búsqueda de Contradicciones

Se han realizado una serie de iteraciones del algoritmo sobre un número diferente de ontologías para mostrar el rendimiento del mismo bajo diferentes parámetros, el tiempo consumido de CPU será el objetivo a estudiar en esta sección. El equipo utilizado tanto para el desarrollo de la aplicación como para los experimentos con un microprocesador Intel (R) Core(TM)2 Duo P8400 2,26GHz 2,27GHz de 64 bits con 4 GB de memoria RAM y sistema operativo Windows 7 Enterprise. La versión de Java es la 1.7 y la versión del razonador es *pellet 2.1*.

En la tabla 21 vemos los resultados de los conjuntos de datos y los tiempos y memoria medios medidos tras tres ejecuciones del algoritmo sobre el mismo conjunto de ontologías. Lo que nos interesa en este experimento es comprobar el comportamiento de tiempo y memoria, si permanecen o no constantes en las diferentes ejecuciones del algoritmo, sobre un determinado número de ontologías.

El máximo de ontologías con el que vamos a trabajar es diez, conjunto inicial que consideramos representativo para un corpus. Debido a que una ontología real suele estar muy poblada, el tiempo de razonamiento sobre un corpus mayor a diez

crecería exponencialmente, a lo que habría que añadirle, además, el número de posibles combinaciones que nuestro algoritmo debería inspeccionar.

Con estos experimentos comprobaremos que, incluso con un conjunto de textos que pueda parecer pequeños, tanto la complejidad como el tiempo son bastante altos.

Concretando esta idea, si volvemos a la tabla 20 donde se muestran los elementos de las ontologías del corpus inicial generado automáticamente y comprobamos el conjunto de datos número 10 cuyo tamaño es veinte ontologías, el número de posibles candidatos, si todos fueran consistentes, sería 1,048,555.

					Raz. No Incremental	
Conjuntos de Datos	**No. Onto**	**Tama Posib**	**No. Onto-Cont**	**Tama Obt**	**Tiempo Med (ms)**	**Memoria Med (MB)**
CD 1	10	1023	3	127	133060,33	53
CD 2	10	1023	5	31	3821,00	10,00
CD 3	7	127	0	127	131489,33	41,66
CD 4	7	127	3	15	3206,11	7,65
CD 5	5	31	0	31	3115,67	9,00
CD 6	5	31	2	7	2160,33	9,00
CD 7	3	7	0	7	2133,33	8,00
CD 8	3	7	3	0	1608,00	10,00

Tabla 21: Métricas con Razonamiento No Incremental

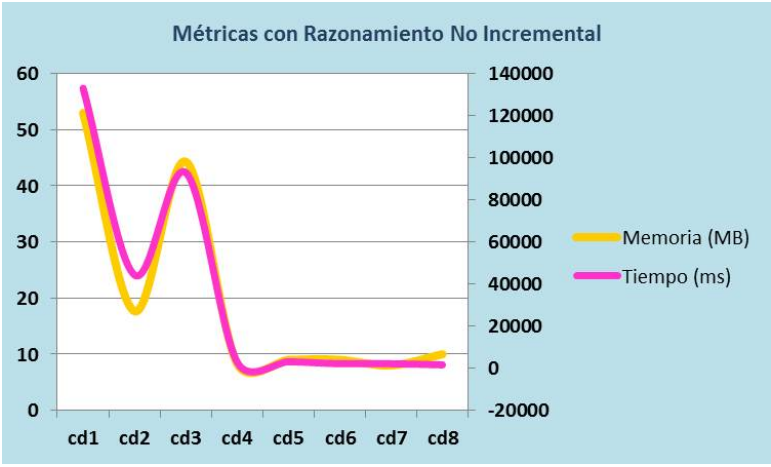

Figura 44: Métricas con Razonamiento No Incremental

Pasemos a la evaluación de los resultados reflejados en la tabla 21, y en la figura 44, basándonos en los tiempos medios de ejecución, en milisegundos, y en la memoria media de ejecución en megabytes, y modificando el número de elementos contradictorios en cada grupo de elementos, observamos cómo, para un razonamiento no incremental, el tiempo de ejecución para un conjunto inicial de 10 ontologías es mayor al doble del que se necesita para un conjunto de 5 elementos. Sobre ese conjunto de 10 ontologías, 3 de ellas resultan contradictorias y el tamaño obtenido final es de 127, en lugar de los 1023 elementos esperados al principio, sin embargo, si no existen ontologías con contradicciones en el conjunto de datos, como es el caso del conjunto 3, el tamaño obtenido es igual al tamaño supuesto, 127.

Como norma general, el gasto computacional en tiempo medio es mayor en aquellos conjuntos de datos donde el tamaño final obtenido es superior, es decir, para los conjuntos CD1 y CD2, ambos con 10 ontologías, el tiempo es mayor para el CD1, donde el tamaño final obtenido es superior a CD2.

No es relevante respecto al parámetro de tiempo, el número de contradicciones, puesto que para los conjuntos CD2, con mayor número de contradicciones, el tiempo no ha sufrido un incremento significativo. Sí se incrementa en el CD1 porque el tamaño final es mayor.

Respecto a la memoria empleada, en general, es mayor en aquellas experimentaciones en las que el número de contradicciones es mayor. Este esfuerzo suponemos se corresponde con el trabajo del razonador y con el de la clase Java que comprueba la consistencia y almacena las contradicciones encontradas para un posterior estudio.

Observemos cómo, si varía el tamaño obtenido, aunque el conjunto inicial fuese el mismo, el gasto de memoria es superior, en general, cuando el tamaño obtenido es mayor como ocurre en los conjuntos de datos CD1, CD3 y CD5. Si el tamaño posible es pequeño, así como no aparecen muchas contradicciones y el conjunto de datos también es pequeño, puede que la memoria no varíe, observemos el caso de los conjuntos CD5 y CD6. Por último, destaquemos, que, puede haber un incremento no demasiado significativo para los conjuntos de datos CD7 y CD8 donde, para pocas ontologías, el rendimiento de memoria puede asumentar ya que, si no se obtiene un conjunto final, el trabajo computacional es más alto que en caso contrario.

					Raz. Incremental	
Conjuntos de Datos	**No. Onto**	**Tama Posib**	**No. Onto-Cont**	**Tama Obt**	**Tiempo Med (ms)**	**Memoria Med (MB)**
CD 1	10	1023	3	127	102839,67	57,67
CD 2	10	1023	5	31	3974,00	11,00
CD 3	7	127	0	127	100648,95	47,19
CD 4	7	127	3	15	2137,48	9,00
CD 5	5	31	0	31	3401,67	9,00
CD 6	5	31	2	7	1998,00	9,00
CD 7	3	7	0	7	2015,33	8,00
CD 8	3	7	3	0	1537,33	10,00

Tabla 22: Métricas con Razonamiento Incremental

5.5 Análisis del Factor Incremental en el Razonamiento en la Búsqueda de Contradicciones

Según presentamos en el capítulo 3, el uso de razonadores incrementales puede reducir el tiempo empleado en el razonamiento puesto que no será necesario realizar todo el razonamiento desde el inicio sino que se podrán reutilizar los razonamientos previos sobre una ontología, realizando el trabajo sólo sobre los cambios que haya sufrido la misma.

En la tabla 22 introducimos un nuevo parámetro para validar nuestro algoritmo, el razonamiento incremental. Según vimos en la sección 4.4.5, una de las características por las que nos hemos decantado por la ontologías OWL DL para nuestros experimentos es porque son susceptibles de ser tratadas por un razonador, el cual nos permitirá detectar inconsistencias en las ontologías y algunos de ellos, además, permiten realizar razonamiento incremental y no incremental.

La diferencia entre el razonamiento incremental y el no incremental radica, según la propia página de *pellet* ([Clark and LLC, 2011]) en la capacidad del razonador para procesar actualizaciones aplicadas a una ontología sin tener que realizar todos los pasos de razonamiento a partir de cero.

Pellet admite dos técnicas de razonamiento incrementales diferentes: la comprobación de la *consistencia incrementales* y *clasificación incremental*.

La *clasificación incremental* se utiliza para actualizar resultados de la clasifi-

	Raz. No Incremental		Raz. Incremental	
Conjuntos de Datos	**Tiempo Med (ms)**	**Memoria Med (MB)**	**Tiempo Med (ms)**	**Memoria Med (MB)**
CD 1	133.060,33	53,00	102839,67	57,67
CD 2	3.821,00	10,00	3974,00	11,00
CD 3	131.489,33	41,66	100648,95	47,19
CD 4	3.206,11	7,65	2137,48	9,00
CD 5	3.115,67	9,00	3401,67	9,00
CD 6	2.160,33	9,00	1998,00	9,00
CD 7	2.133,33	8,00	2015,33	8,00
CD 8	1.608,00	10,00	1537,33	10,00

Tabla 23: Comparación Razonamiento No Incremental vs Incremental

cación de forma incremental cuando la jerarquía de clases cambia. Esta función se activa mediante la utilización de función de extracción del módulo previsto en *pellet*. La primera vez que una ontología se clasifica, *pellet* también calcula módulos para cada clase. La clasificación inicial y la extracción del módulo se pueden realizar en serie o en paralelo. Cuando la ontología se cambia de una manera que afecta a la jerarquía de clases, *pellet* determinará qué módulo se ve afectado y reclasificará solamente ese módulo que suele ser mucho menor que la ontología inicial.

La *clasificación incremental* es más adecuada para los casos en que la ontología contiene una jerarquía de clases grandes y / o complejas.

La comprobación de *consistencia incrementales* se utiliza para actualizar la comprobación de coherencia de los resultados incrementalmente cuando hay cambios en las instancias. Los únicos cambios que sugieren los creadores de *pellet* son la adición o el traslado de la instancia. Si la ontología se cambia mediante la adición de nuevos axiomas de clase o de propiedad, no se utilizará la comprobación de consistencia incremental. Tampoco se utilizará si la ontología contiene enumeraciones, propiedades inversas, o reglas.

La comprobación de *consistencia incremental* es más adecuada para los casos en que la comprobación de la consistencia toma mucho tiempo o hay cambios muy frecuentes a los datos de instancia. Esta función se desactiva por defecto y necesita ser activada manualmente.

En nuestro caso, hemos utilizado la *consistencia incremental* puesto que nuestro ejemplo no es muy populoso.

En las tablas 22, 23 y en la figura 45 observamos el comportamiento del algoritmo de **Búsqueda de Contradicciones en Textos** con el razonador *pellet* en modo incremental.

El comportamiento de las gráficas representadas en las figuras 44 y 45 (podemos comprobar los datos en la tabla 23) es, en general, similar. Las diferencias entre ambos razonamientos se ponen de manifiesto en los puntos máximos que alcanzan las líneas de tiempo y de memoria; éstos no son tan acusados en el razonamiento incremental: mientras que el conjunto de datos 3 en el razonamiento no incremental supera los 120000ms, en el incremental la media de tiempo se queda rondando los 100000ms. Asimismo, tanto las medidas de tiempo como de memoria evolucionan de forma más suave en el razonamiento incremental para aquellos conjuntos con menor cantidad de ontologías, véase los conjuntos del 4 al 8.

El tiempo que tarda en ejecutarse, es inferior en casi todas las ejecuciones salvo en los conjuntos de datos 2 y 5. Una posible explicación a este comportamiento puede encontrarse en que para el conjunto 2 el número de ontologías con contradicciones es muy alto (5 de ellas presentan contradicciones), tienen menor número de ontologías candidato susceptibles de ser mezcladas pero el razonador debe comprobar la consistencia de mayor número de candidatos. Respecto al comportamiento en la conjunto de datos 5, al no presentar, por el contrario, contradicciones, todas las posibles combinaciones de candidatos deberán ser evaluadas.

Observamos también que la memoria utilizada, en tiempos medios, por el razonamiento incremental es mayor en todos conjuntos de datos con más de 5 ontologías, para el resto de conjuntos, el comportamiento es igual que el razonamiento no incremental.

Los autores de *pellet* ([Clark and LLC, 2011]) nos indican que, el *clasificador incremental* es robusto, bien probado y adecuado para los sistemas de producción mientras que la comprobación de *coherencia incrementales* no es sólida, no está bien probada y no es adecuada para los sistemas de producción.

Los propios ejemplos planteados en las librerías *pellet* validan la consistencia incremental, mediante el establecimiento del parámetro del razonador a *verdadero* si se desea esta característica del mismo. Esta es la que utilizaremos nosotros, debido, en gran parte, porque nuestros ejemplos no son muy extensos.

En resumen, el método incremental no aporta ventajas claras para los conjuntos que se han estudiado en este capítulo. En trabajos futuros se ampliará el estudio incremental del razonador sobre los corpus propuestos y corpus adicionales, propondremos un razonador incremental propio que mejore sustancialmente los tiempos de

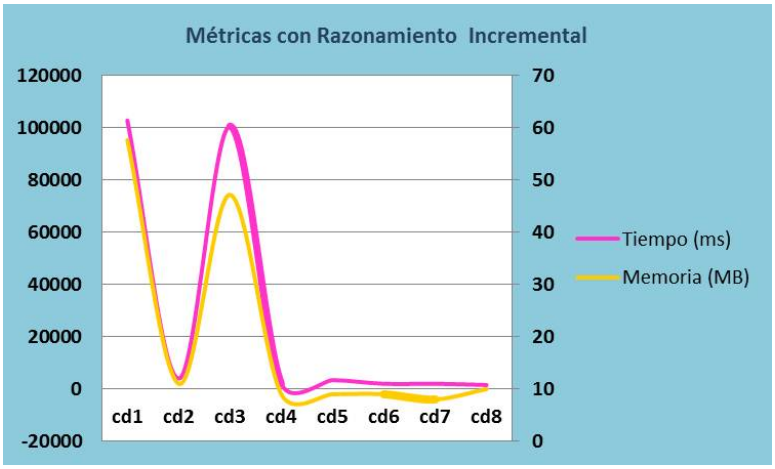

Figura 45: Métricas con Razonamiento Incremental

ejecución y que nos permita el acceso a los elementos que hayan sido objeto de razonamiento para poder proporcionarles una utilidad adicional.

5.6 Conclusión

En este capítulo hemos intentado ilustrar por medio de ejemplos prácticos cómo, utilizando ontologías, se puede desarrollar una técnica de Minería de Textos.

Se ha implementado como técnica de Minería de Textos el **algoritmo de Búsqueda de Contradicciones en Textos** presentado en el capítulo anterior desde un punto de vista teórico midiendo la complejidad del mismo en función del tamaño de las ontologías, el tiempo y la memoria consumidos. La implementación del algoritmo se ha realizado mediante una aplicación Java.

La Búsqueda de Contradicciones es de utilidad a la hora de evaluar la consistencia de una colección de documentos.

Hemos presentado una herramienta para la generación automática de un corpus de ontologías según la expresividad OWL DL, implementada en Java, con la finalidad de que sirva como conjunto inicial para nuestro algoritmo. Esta generación automática construirá un número aleatorio de ontologías sobre el total deseado, rellenándola con un porcentaje de elementos entre clases, propiedades, objetos e instancias, así como una serie de contradicciones a evaluar, de tal forma que se

pueda construir un corpus suficiente para realizar la Búsqueda de Contradicciones en Textos.

El **algoritmo de Búsqueda de Contradicciones en Textos** se ha ejecutado sobre las ontologías ejemplos generadas automáticamente y sobre un ejemplo real de ontologías.

Capítulo 6 Conclusiones

Una contradicción es una contradicción sólo si está allí
(Wittgenstein and Schlick [Wittgenstein and Schlick, 1979])

6.1 Conclusiones

Llegados a este capítulo, realizaremos un balance de lo expuesto en las secciones anteriores y describiremos las principales conclusiones que hemos alcanzado.

Las preguntas a las que se les ha proporcionado respuesta a lo largo de este trabajo tienen que ver con los problemas de análisis textual presentados en las secciones iniciales: ¿es posible encontrar una representación intermedia para el texto, adecuada desde un punto de vista computacional, y que conserve la semántica del texto? ¿es posible emplear la Minería de Textos para una tarea útil?

Lo que pretendía ser un acercamiento inicial a la Minería de Textos ha desembocado en un estudio pormenorizado de las diferentes corrientes de descubrimiento de conocimiento de las formas de representación de la información textual, en general y de las ontologías con una determinada expresividad, en particular.

La tesis hasta aquí expuesta ha dado lugar los siguientes resultados, los cuales cumplen los objetivos planteados al inicio de esta disertación:

- En los últimos años ha habido un significativo crecimiento del número de publicaciones relacionadas con la Minería de Textos, sin embargo, el estado del arte de la Minería de Textos se ha diversificado sustancialmente. Mientras que hace diez años era relativamente frecuente encontrar en la literatura términos

como: *Knowledege Discovery in Texts (KDT), Text Data Mining (TDM)* o *Text Mining (TM)*, en los últimos tiempos es cada vez más frecuente encontrar conceptos como *Information Mining*, *Web Mining* o *Sentiment Analysis* en artículos que realizan minería sobre información textual. Si bien dichos conceptos presentan diferencias respecto a la Minería de Textos, existen autores que llaman al proceso de descubrimiento de conocimiento sobre textos indistintamente con cualquiera de estos nombres.

- La Minería de Textos intenta resolver una particularidad del texto escrito en lenguaje natural: la ausencia de estructura del mismo. Por lo tanto, una solución basada en Minería de Textos tendrá que proporcionar cierto formato o estructura a los documentos elegidos como objetivo del análisis para, posteriormente, aplicar las técnicas de minería. Hemos presentado nuestra definición de Minería de Textos, así como las diferentes fases necesarias en el proceso de Descubrimiento de Conocimiento en Textos.

- Se ha realizado un estudio a través de los artículos más representativos de la literatura que abordan el tema de la Minería de Textos bien desde el punto de vista de representación del texto, bien desde las técnicas propuestas para la minería o por las aplicaciones propuestas. Se ha aportado información estadística que apoya la idea de que el concepto de Minería de Textos se diluye entre la bibliografía científica, fusionándose y confundiéndose a veces con otros conceptos propios de otros ámbitos entre los que se encuentran la Recuperación de Información o de Extracción de Información.

- Hemos abogado por la necesidad de proporcionar una representación intermedia al texto que conserve la mayor carga semántica posible sobre la cual poder utilizar técnicas deductivas o abductivas, además, hemos remarcado la importancia del conocimiento de *background* para seguir conservando la riqueza del texto escrito en lenguaje natural.

- El rango de áreas en el que se puede aplicar la Minería de Textos es amplio, abarca desde sistemas inteligentes de mercado hasta filtrado de emails. Hemos aportado un conjunto de soluciones de minería existentes, presentando tanto la descripción del problema como los métodos propuestos para su solución.

- Además de las aplicaciones ya existentes, parece claro que las perspectivas de crecimiento en este campo son tan amplias y variadas como permita la imaginación. Hemos aportado varias ideas sobre el camino que pueden seguir las investigaciones en Minería de Textos, en concreto, hemos aportado la idea de *Text Knowledge Mining* como un caso particular de *knowledge mining*, el cual se define como *la obtención de conocimiento potencialmente útil, no trivial y previamente desconocido a partir de repositorios de información.* Nos apoyamos en la idea de que las técnicas tradicionales de Minería de Datos que se han

ido utilizando para realizar Minería de Textos son inductivas y no explotan todas las posibilidades que puede ofrecer la extracción de conocimiento. La Minería de Textos habitual emplea técnicas inductivas propias de la Minería de Datos, mientras que la Minería de Textos basada en conocimiento utiliza inferencia deductiva y abductiva, entre otras.

- Hemos presentado una novedosa aplicación de *Text Knowledge Mining* para la búsqueda de contradicciones en texto. En dicha aplicación proporcionamos una estructura intermedia al texto intentando conservar la mayor carga semántica posible, utilizando ontologías como Forma Intermedia, y realizamos la búsqueda de contradicciones con un algoritmo de aprendizaje no supervisado.
- Se ha realizado una experimentación con baterías de ejemplos sintéticos para comprobar la fortaleza del algoritmo de Minería de Textos.

6.2 Trabajos Futuros

Cada nuevo experimento realizado nos ha proporcionado un nuevo campo susceptible de ser analizado, estudiado y abordado, dispuesto a seguir evaluándose en un futuro.

Entre los temas, problemas y situaciones que han suscitado nuestro interés y pueden dar lugar a nuevas líneas de investigación presentamos, aún a riesgo de dejarnos alguna de ellas en el tintero, las siguientes:

- La extensión de nuestro modelo al caso de la incertidumbre inspirándonos en los trabajos de Trillas en los que extiende sus modelos a estos casos.
- Utilización del algoritmo de Búsqueda de Contradicciones sobre problemas reales de forma masiva. La localización de corpus ontológicos sobre los que iterar nuestro algoritmo será uno de los puntos principales de este trabajo.
- Probar la utilidad de nuestra técnica de minería utilizando otros tipos de razonamiento no presentados en esta disertación y que pueden ser igualmente útiles a la hora de aplicarlos a la vida cotidiana.
- Extender el razonamiento ontológico para incluir conjeturas.
- Comprobar la fortaleza de la aplicación sobre otros tipos de formas intermedias y si los resultados para la detección de contradicciones se aproximan a las arrojadas por ontologías.

- Implementación y uso de ontologías difusas como formas intermedias en los casos de uso. Se prevé tanto la construcción del corpus inicial difuso como el uso de ontologías difusas existentes.
- Generación de explicaciones posibles a las contradicciones encontradas durante la minería.

Anexo A Gramáticas y Lenguajes

A.1 Gramáticas

> *El estudio del lenguaje, en cuanto a forma, estructura y significado y también el manual en el que se recogen una serie de normas de la lengua, es lo que llamamos gramática ([Llopart and Huber, 1986]).*

Una gramática es un conjunto de reglas que describen las estructuras de una lengua e intentan ofrecer generalizaciones sobre la lengua en cuestión ([Llopart and Huber, 1986]).

Existen distintos tipos de gramáticas, según su orientación en el campo de la lingüística: *Gramática Pedagógica*, cuando se centra en el uso de la lengua, *Gramática Descriptiva*, la que refleja aquello que la gente habla, *Gramática Prescriptiva*, fija las bases de lo que es correcto y de lo que no, la *Gramática Formal* que observa la lengua como un mecanismo lingüístico innato que existe en todo ser humano (en este caso se habla de la psicolingüística y se desarrolla en relación con la escuela formal de Noam Chomsky [wikipedia, 2015])), y la *Gramática Funcional* que se centra especialmente en la relación entre el sistema y el uso de dicho sistema (sociolingüísitca) ([Española, 2010]).

A.1.1 Definición de Gramática

Dejando aparte la visión lingüista de la gramática, definiremos ahora una gramática útil para el procesamiento del lenguaje natural.

Sea $G = (VT, VN, S, P)$ una gramática donde

- VT es un conjunto de símbolos terminales,
- VN es un conjunto de símbolos no terminales,
- $VT \bigcap VN = \phi$,
- S es el símbolo inicial tal que $S \epsilon VN$,
- P es un conjunto finito de pares o producciones tal que cada producción (α, β) se representa como $\alpha \rightarrow \beta$, de manera que $\alpha \epsilon V^+$ y $\beta \epsilon V^*$ siendo V un vocabulario, V^* el conjunto de todas las cadenas compuestas de símbolos de V y V^+ el conjunto V^* exceptuando la cadena vacía ([Sánchez Dueñas and Valverde Andreu, 1989])

Ejemplo: *Sea G una gramática con los siguientes elementos, G=({a,b,c,d), {S,A,B),S,P), podemos obtener los conjuntos anteriores:*

- $VT = \{a,b,c,d)$
- $VN = \{S,A,B)$
- el conjunto P sería
  - $S \rightarrow ASB$ - $S \rightarrow d$
  - $A \rightarrow b$ - $A \rightarrow aA$
  - $aaA \rightarrow aBB$ - $B \rightarrow dcb$

Veamos ahora algunos elementos fundamentales íntimamente relacionados con la idea de gramática:

- **Derivación:**

  Sea $\alpha \rightarrow \beta$ una producción y sea $\delta\alpha\beta$ una cadena. Llamamos a $\delta\alpha\mu \rightarrow \delta\beta\mu$ **derivación inmediata**.

  Una **derivación inmediata** es una secuencia de cadenas α_0,α_1,...,α_n donde $n \geq 0$ tal que $\alpha_0 \rightarrow \alpha_1$, $\alpha_1 \rightarrow \alpha_2$, ..., $\alpha_{n-1} \rightarrow \alpha_n$

  La notación de la derivación es $\alpha_0 \rightarrow \alpha_n$ *ó* $\alpha_0 \rightarrow \alpha_n$ si $n \geq 1$.

  Ejemplo: *Sea la gramática G =({a,b,c,d), {S,A,B),S,P), una derivación posible es* $SASB \rightarrow aASB \rightarrow abSB \rightarrow abdB \rightarrow abddcb$

- **Sentencia**:

  Se denomina sentencia (α_n) a cualquier cadena compuesta sólo por símbolos terminales y que sea resultado de una última derivación a partir del símbolo inicial S: $S \rightarrow \alpha_n$.

Según Chomsky, existen cuatro tipos de gramáticas:

- **Tipo 0**. Es el tipo más general de gramática y su definición es la definición dada anteriormente. Generará un lenguaje tipo 0.
- **Tipo 1** o sensibles al contexto: su característica principal es que sus reglas de producción son $\|\beta\| \geq \|\alpha\|$
- **Tipo 2** o libres de contexto: las reglas de producción de esta gramática son del tipo $A \rightarrow \alpha$ donde α es una cadena con 0 o más símbolos terminales y no terminales, por ejemplo: $S \rightarrow SS|(S)|()$
- **Tipo 3** o regulares: una gramática regular es una gramática formal *(VT, VN, S, P)* que sólo puede generar a los lenguajes regulares de manera similar a los autómatas finitos y las expresiones regulares. Además, una gramática regular es una gramática libre de contexto.

  Las gramáticas regulares pueden ser derechas o izquierdas, si $G = (VT, VN, S, P)$ es **derecha**, las reglas de producción P son de la siguiente forma:

  - $A \rightarrow a$
  - $A \rightarrow aB$
  - $A \rightarrow \varepsilon$

  Si la gramática es regular **izquierda**, las reglas son las siguientes:

  - $A \rightarrow a$
  - $A \rightarrow Ba$
  - $A \rightarrow \varepsilon$

  Las reglas están restringidas y son de la forma $A \rightarrow tN|t$, por ejemplo: $B \rightarrow 0B|1B|0|1$

En la figura 42 vemos la jerarquía descrita por Chomsky para la representación de las gramáticas.

Gramática	Lenguaje	Autómata	Normas de producción
Tipo-0	recursivamente enumerable (LRE)	Máquina de Turing (MT)	Sin restricciones
Tipo-1	dependiente del contexto (LSC)	Autómatas Linealmente Acotados	$\alpha A\beta \rightarrow \alpha\gamma\beta$
Tipo-2	independiente del contexto (LLC)	Autómata a Pila	$A \rightarrow \gamma$
Tipo-3	regular (RL)	Autómata Finito	$A \rightarrow aB$ $A \rightarrow a$

Figura 42: Gramáticas según Chomsky ([wikipedia, 2015])

A.2 Lenguajes

El **lenguaje** es la capacidad del ser humano para comunicarse mediante un sistema de signos, lengua natural o literaria. El *lenguaje natural* es un compendio formado por todas las comunicaciones animales y el lenguaje humano. En ciencias exactas como matemáticas o informática, se utilizan *lenguajes artificiales* o formales como pueden ser los lenguajes de programación. Una de las características fundamentales del lenguaje humano que lo diferencia de los demás, es la diversidad.

Formalmente, *definimos un lenguaje L generado por una gramática G, L(G), del siguiente modo: L(G) = ($\mu\epsilon$ $VT^*|s \rightarrow \mu$) y contendrá todas las sentencias que genera G.*

Por lo tanto, una cadena pertenecerá al lenguaje L(G) si está compuesta sólo por símbolos terminales y, además, puede derivarse de S.

Tipos de Lenguajes según la clasificación de Chomsky ([wikipedia, 2015]):

- **Lenguajes Recursivamente Enumerables (de tipo 0)**. Como son los lenguajes naturales, en los que pueden existir reglas compresoras.
- **Lenguajes Dependientes del Contexto (sensibles al contexto, de tipo 1)**. No existen reglas compresoras, salvo, opcionalmente, la que deriva el axioma a la palabra vacía.

  Existen reglas en las que un símbolo no terminal puede derivar a sentencias distintas, según los símbolos que aparezcan a su alrededor.
- **Lenguajes Independientes del Contexto (de contexto libre, de tipo**

2). La mayoría de los lenguajes de programación.

- **Lenguajes Regulares (de tipo 3)**. Aquellos que se pueden expresar también mediante expresiones regulares.

Anexo B Tecnologías Ontológicas

B.1 Ontologías

A la hora de elegir una forma de representación del texto que permita extraer *conocimiento* previamente desconocido a partir del texto y que no pierda de vista el contexto en el que se escribieron los documentos, coexisten una amplia variedad de representaciones que son susceptibles de ser evaluadas: desde una representación intermedia de los documentos basada en palabras en la que, por su sencillez, pierde información semántica importante, hasta redes semánticas que aumentan el detalle de la representación pero que también incrementan la complejidad del tratamiento del mismo. Este estudio lo hemos presentado detalladamente a lo largo de esta disertación y en concreto en el capítulo 2.

También hemos visto que, dentro de las Formas Intermedias que trabajan la semántica del corpus textual, las más utilizadas en la literatura son: los grafos semánticos, diferentes tipos de redes semánticas como pueden ser las redes IS-A, los grafos conceptuales, las dependencias conceptuales o las ontologías, entre otros.

Y, ¿qué es *conocimiento*? en la figura 43 se aprecia la definición tradicional de conocimiento, como la intersección de hechos o verdades (*truth*) y creencias (*beliefs*). Si nuestro deseo, como es el objeto de este trabajo, es descubrir conocimiento a partir de una representación del texto manejable computacionalmente, las **ontologías** nos proporcionarán el nexo entre la representación del texto y la definición de conocimiento, porque una ontología posee entre sus elementos: objetos, propiedades y relaciones, mediante los cuales seremos capaces de dar forma a la semántica, y nos servirá como sustento para la obtención de conocimiento nuevo.

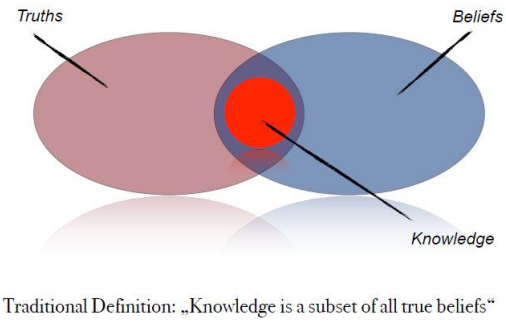

Figura 43: Definición de Conocimiento ([Shack, 2013])

Las ontologías son una de las Formas Intermedias que aúnan, según los objetivos perseguidos en este trabajo, un punto adecuado de simplicidad de tratamiento y de preservación de la semántica de los textos escritos en lenguaje natural. Este anexo se ocupa de revisar diferentes definiciones de ontologías encontradas en la literatura así como de estudiar en profundidad qué es una ontología, desde el lenguaje de construcción hasta la expresividad de la misma.

B.1.1 Definición de Ontología

Cuando se presenta la necesidad de encontrar una definición formal de ontología, puede parecer obvio remitirse a la literatura en busca de una definición genérica, el problema surge cuando casi cada uno de los autores que trabajan con ontologías aporta su propia definición y decidirse por sólo una puede resultar una tarea difícil. Algunas de las definiciones de ontologías más referenciadas en la literatura científica son las siguientes:

> Una especificación formal y explícita de una conceptualización compartida ([Gruber, 1993]).

> Un conjunto de términos estructurados que describen algún dominio o tópico. La idea es que una ontología proporcione una estructura esquelética para una base de conocimiento ([Swartout et al., 1996]).

> Una especificación explícita a nivel de conocimiento de una conceptualización, por ejemplo, el conjunto de distinciones que son significativas

> para un agente. La conceptualización y por ende la ontología, pueden estar afectadas por el dominio particular y la tarea particular a la que está destinada ([Heijst et al., 1997]).
>
> Las ontologías son teorías de contenido sobre tipos de objetos, propiedades de objetos y relaciones entre objetos que son posibles en un dominio de conocimiento especificado. Proporcionan términos potenciales para describir nuestro conocimiento sobre el dominio ([Chandrasekaran et al., 1999]).
>
> Una descripción formal de entidades y sus propiedades, relaciones, restricciones, comportamientos. Proporciona una terminología que captura distinciones de clave y son genéricas en muchos dominios ([Grüninger et al., 2000]).
>
> Una descripción de conceptos formal y explícita en un dominio de discurso (las clases en ocasiones se llaman conceptos), las propiedades de cada concepto describen varias características y a los atributos de los conceptos (los *slots* algunas veces se nombran como roles o propiedades), y las restricciones en slots (las facetas en ocasiones llamadas restricciones de rol) ([Noy and McGuinness, 2001]).

Una de las definiciones más extendidas es la de Gruber ([Gruber, 1993]): *una ontología constituye una especificación formal y explícita de una conceptualización compartida* (*a formal, explicit specification of a shared conceptualization*), donde conceptualización es el sinónimo utilizado para referirse a una abstracción de algún suceso, hecho o evidencia del mundo del cual se identificarán los conceptos relevantes ([Hernández, 2002]), que sea *explícita* implica que se debe definir el significado de todos los conceptos, que sea *formal* significa que debe ser comprensible por una máquina y que sea *compartida* tiene que ver con los consensos sobre ontologías ([Shack, 2013]). Pero, debido a la necesidad imperiosa de formalizar una definición de ontología, se ha elegido la aportada por ([Schorlemmer and Kalfoglou, 2005]):

Definición: Una ontología es una tupla

$$O = \langle C, \leq, \perp, | \rangle$$

donde ([Schorlemmer and Kalfoglou, 2005]),

- C es un conjunto finito de conceptos

- $\leq$ es una relación reflexiva, transitiva y anti-simétrica en C (orden parcial)
- $\perp$ es una relación en C simétrica e irreflexiva (disyunción)
- | es una relación simétrica en C (cobertura)

Parece claro, sin pretender ser exhaustivos con la búsqueda de una definición definitiva de ontología, que muchos de los autores aquí mencionados coinciden en que los elementos básicos de una ontología son los **conceptos** (predicados unarios que representan entidades o clases), los **roles** (relaciones binarias, propiedades) que representan propiedades o relaciones y que estarán formados a su vez por un conjunto de conceptos y roles atómicos a los que se les aplicará una serie de constructores, los **indiviudos** (instancias) y los **axiomas** (operadores o constructores que servirán para construir representaciones complejas de conceptos o roles).

Es interesante humanizar el concepto de ontología, acercándola a objetos reales, como hace Stevens en su artículo ([Stevens et al., 2010]), así, cada uno de los individuos de la ontologías se corresponderán con objetos materiales e inmateriales. Como humanos que somos, otorgaremos categorías o clases a cada uno de los objetos de la ontología, estas categorías describirán los datos, conceptualizándolos, otorgándoles etiquetas, lo que formará un vocabulario, esencial también en las ontologías. Cada uno de dichos objetos se podrá relacionar con los demás, apareciendo aquí los conceptos de clase, subclase y las relaciones *is-a*, *partOf*, *deriveFrom*, *participateIn*, entre otras.

Veamos gráficamente un ejemplo de ontología en la figura 44, la información de base que ha servido para construir dicho ejemplo se ha obtenido del recurso *David Livingston* de la *dbpedia* ([DBpedia, 2015]).

La utilización de ontologías en las aplicaciones puede resultar costosa a nivel computacional pero presenta, a su vez, diversas *ventajas*:

1. Una ontología es independiente de la plataforma, puede servir de lenguaje común entre diferentes versiones de software.
2. La reutilización es una de las grandes ventajas del uso de ontologías, puede ayudar a reducir costes, validar aplicaciones...
3. Diferentes ontologías pueden representar el mismo conocimiento. Punto que se pude ver gráficamente en la figura 45 ([Shack, 2013]). Podemos comprobar cómo, para un mismo modelo de bloques, existen dos ontologías, al menos, que pueden representar dicho estado.

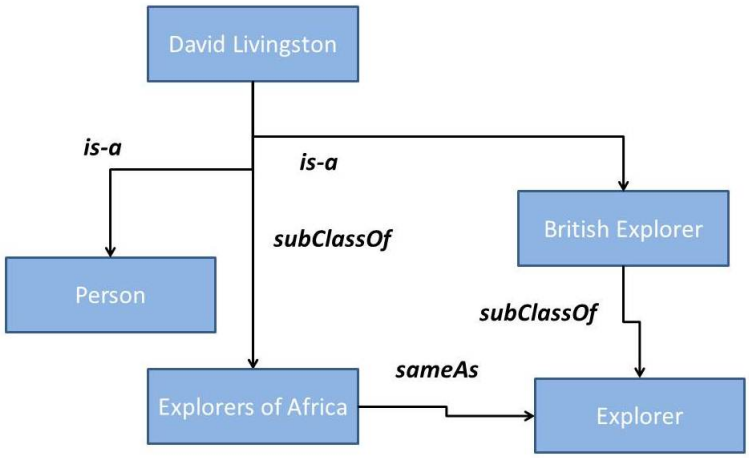

Figura 44: Ejemplo de ontología: Recurso de DBpedia - David Livingston

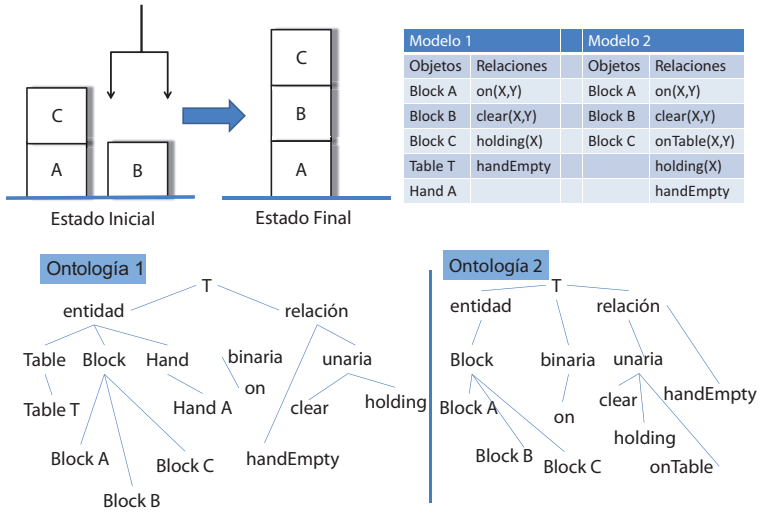

Figura 45: Diferentes Ontologías, igual modelo de conocimiento ([Shack, 2013])

Una vez definida la ontología y teniendo en cuenta que uno de los objetivos de esta memoria es utilizar las ontologías como representación del texto, resulta imprescindible comprender con qué lenguajes se confeccionan dichas ontologías y, a su vez, elegir el más conveniente.

Una ontología se basa en un lenguaje ontológico, lo que le permite procesar inferencias o deducciones, dicho lenguaje ontológico estará basado en Lógicas Descriptivas (OIL, OWL, ...), lógica de primer orden (KIF *Knowledege Interchange Format*), en reglas (RuleML, LP/Prolog), lógica difusa o en cualquier otro tipo como: LBase, F-Logic ...

En este trabajo nos hemos decantado por las Lógicas Descriptivas, que se desarrollarán ampliamente en la siguiente sección.

B.1.2 Lógicas Descriptivas

Las Lógicas Descriptivas son una familia de lenguajes que se utilizan para la representación del conocimiento. En general, son un subconjunto de las Lógicas de Primer Orden y, además, es posible realizar deducciones lógicas basándose en este tipo de lógicas. Algunos de los lenguajes que se utilizan para representar ontologías forman parte de las Lógicas Descriptivas, entre ellos: los derivados de las lógicas atributivas (AL, SHIQ, SHOIN, ...), OLCD (*Object Language with Complements allowing Descriptive cycles* [Beneventano et al., 2003]), DAML-ONT (*DARPA Agent Markup Language Ontologies* [w3c, 2001]), OWL (*Web Ontology Language* [w3c, 2012a]).

En esta sección, se describirán los principales componentes de un sistema basado en Lógica Descriptiva, en nuestro caso, una ontología, como son TBox y ABox y algún que otro componente más, ([Calvanese et al., 2001]):

- TBox (*Terminological Box*). TBox contiene el conjunto de aserciones universalmente cuantificadas. Expresa el conocimiento intensional (el significado o connotación) o terminológico sobre los conceptos de un dominio (clases, atributos relaciones, ...), es decir, la descripción formal del esquema de la ontología.
- ABox (*Assertional Box*). ABox contiene las aserciones sobre los objetos individuales, llamados instancias. Expresa el conocimiento extensional sobre los objetos (el conjunto de elementos sobre los que se aplica).
- El conjunto de constructores para conseguir los conceptos usados en la TBox y en la ABox.

	Conceptos	**Roles**	**Individuos**	**Operadores / Constructores**
Definición	Predicados unarios	Predicados binarios	Constantes, entidades, aserciones de conceptos	Ayudan a construir representaciones complejas de conceptos / roles
Representan	Entidades / Clases	Propiedades / Relaciones	Elementos	
Ejemplo	Persona, Estudiante	participaEn	Alicia, Seminario	
Sintaxis	Estudiante:{x | Estudiante(x)}	participaEn: {(x,y) | participaEn(x,y)}	Estudiante(Alicia)	participaEn (Alicia,Seminario)

Tabla 24: Elementos básicos de la Lógica Descriptiva ([Shack, 2013])

- Las aserciones de la TBox: también llamadas definiciones, donde una definición es una aserción que indica que la extensión de un concepto identificado por su nombre es igual a la extensión de otro concepto (a ésto también se le llama concepto complejo. Estas definiciones son acíclicas, si construyéramos un grafo donde conectáramos conceptos atómicos (nodos) éste sería acíclico. Y para cada concepto C, al menos una definición de C aparecerá en la TBox.
- Las aserciones de la ABox: un individuo es la instancia de cierto concepto.
- Mecanismos de inferencia que se utilizarán para razonar.

En la tabla 24, se resumen los elementos básicos de la Lógica Descriptiva ([Shack, 2013]), de los que dependerá el nivel expresivo de la lógica.

Los lenguajes basados en AL (lenguaje de descripción llamado AL (*Attributive Language*), germen de una familia de lenguajes que se han extendido en el ámbito de las Lógicas Descriptivas) ganarán en expresividad según se les vaya añadiendo sintaxis y semántica.

B.1.2.1 Elección de una familia de lenguajes para Lógicas Descriptivas *AL*

Tomando como punto de partida el lenguaje de descripción AL, cuyo modelo sintáctico es el siguiente ([García, 2009]):

Sean C, D dos descripciones de conceptos

Sea A un concepto atómico

Sea R un rol atómico

Las reglas sintácticas para AL serán:

C, $[$ D $\rightarrow A \mid]$ (concepto atómico)
$[\top \mid]$ (concepto universal)
$[\bot \mid]$ (concepto base)
$[\neg A \mid]$ (negación atómica)
$[C \sqcap D \mid]$ (intersección)
$[\forall R.C \mid]$ (restricción de valores)
$[\exists R.T \mid]$ (cuantificación existencia limitada)

En este trabajo, se pretende alcanzar una capacidad expresiva suficiente para utilizar el lenguaje OWL DL2 (basado en la expresividad SHOIN(D) en owl 2, recomendación w3c desde 2009) en la construcción de ontologías, para ello, se añadirá semántica al lenguaje AL hasta llegar a la expresividad SHOIN(D). El conjunto de constructores empleados y la semántica utilizada se detalla en la sección B.1.3.

A lo largo de este trabajo, se ha optado por el empleo de las ontologías desarrolladas en OWL DL como forma intermedia del proceso de descubrimiento, por lo tanto, la detección de contradicciones deberá realizarse dentro de la propia ontología.

Para poder discriminar qué una contradicción en el lenguaje OWL, se estudiarán los conceptos que la delimitan en lógica proposicional, más concretamente en: El Principio de No Contradicción, la Satisfacibilidad y la Consistencia. Hemos repasado estos conceptos en el capítulo 4.

En la siguiente sección, extenderemos el concepto de lenguaje de marcado OWL.

B.1.3 Lenguaje OWL

Los lenguajes de marcado son aquellos que permiten codificar y etiquetar un documento añadiendo marcas y anotaciones al texto. Estos lenguajes establecen también una serie de reglas para realizar dicho etiquetado. Una de las formas de representar las ontologías es utilizando un lenguaje de marcado que permitirá mostrar explícitamente la estructura de la misma, así como su información tanto sintáctica como semántica.

Son lenguajes de marcado: SGML(siglas correspondientes a *Standard Generalized Markup Language*), HTML (*Hypertext Markup Language*), XML (*Extensible*

Markup Language), XHTML (*Extensible Hypertext Markup Language*), RDF (*Resource descriptión framework*) y OWL (*Web Ontology Language*), entre otros.

Entrar en detalle en cada uno de los lenguajes de marcado mencionados no es un objetivo de este trabajo, sin embargo, sí lo es elegir uno de ellos para construir la ontología final, se ha optado por **OWL DL** debido a que su expresividad nos permite conservar la semántica del texto. OWL se ha diseñado para trabajar con necesidades específicas del lenguaje de ontologías web y para que vaya más allá de la semántica básica del esquema RDF.

La pila de tecnologías w3c se basa, entre otras, en ([w3c, 2004]):

1. XML: proporciona la interfaz de la sintaxis para los documentos estructurados, pero no impone restricciones semánticas a dichos documentos.
2. XML Schema: es el lenguaje que restringe la estructura de los documentos XML y extiende el XML gracias a los tipos de datos.
3. RDF: modelo de datos para los objetos y las relaciones entre ellos. RDF sí proporciona semántica (aunque sencilla) y se puede representar mediante sintaxis XML.
4. RDF Schema: vocabulario que describe propiedades y clases de los recursos RDF. Contiene semántica para la generalización de las jerarquías de las propiedades y las clases.
5. OWL añade más vocabulario para describir propiedades y clases: por ejemplo, relaciones entre clases, cardinalidad, igualdad, tipos de propiedades más ricas, características de propiedades como pueden ser la simetría y enumeraciones de clases.

El lenguaje OWL se caracteriza por tener varios niveles de expresividad que se representan mediante símbolos. Cada uno de esos símbolos se corresponde con una característica de dicho lenguaje. En la figura 46 se pueden ver estos símbolos y su significado, según Protégé DL Métrics 4.1.

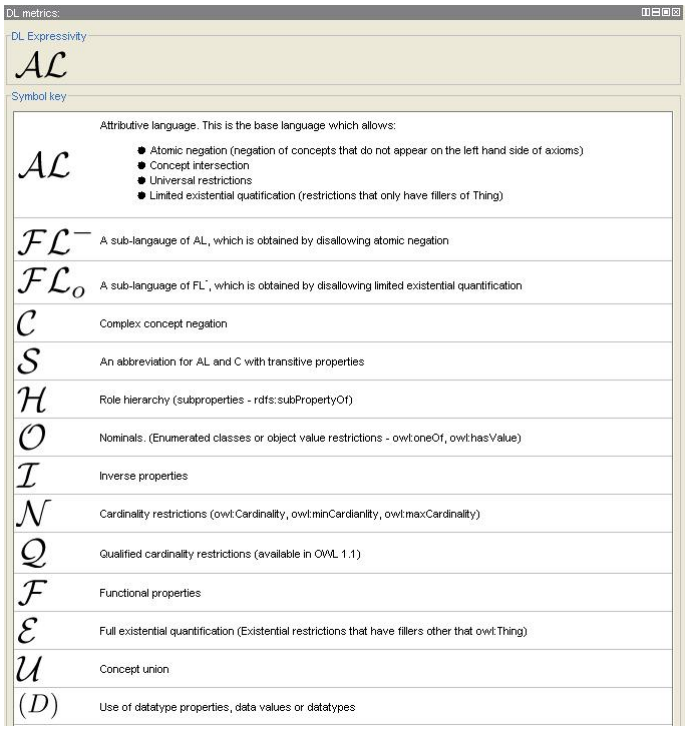

Figura 46: Expresividad según Protégé 4.1

El lenguaje OWL DL se basa en la lógica de descripción SHOIN (D) ([Llop, 2008]), que se caracteriza por una semántica muy expresiva pero el razonamiento menos eficiente. Para alcanzar esta expresividad, nuestra ontología debe cumplir los siguientes requisitos ([García, 2009], [Horridge et al., 2004]):

1. El operador negación sólo se aplicará a los conceptos atómicos (restricción propia del lenguaje AL).
2. Negación compleja de conceptos, ALC.
3. Obtención de conceptos resultantes de la unión de otros dos, $ALCU$.
4. Definición de propiedades o relaciones mediante axiomas complejos, $ALCUR$.
5. Roles Transitivos y Jerarquías de conceptos, SH (extensión de ALC) ([Tortosa, 2013]).
6. Construcción de clases o conceptos a partir de enumeraciones, SHO.
7. Propiedades Inversas, $SHOI$.

8. Nominales, roles inversos, restricciones de cardinalidad y tipos de datos, *SHOIQ*.

9. Restricciones de cardinalidad cualificadas, roles disjuntos, roles reflexivos o no, *SROIQ*. Este tipo de expresividad corresponden con el lenguaje OWL DL 2 ([Tortosa, 2013]).

En la siguiente sección, se enumerarán los factores a tener en cuenta en el razonamiento con el lenguaje elegido OWL DL de expresividad SHOIN(D).

B.1.4 Razonamiento con Lenguaje Owl

B.1.4.1 Sintaxis

La sintaxis que se empleará para construir las ontologías ejemplo es la *Sintaxis Manchester* ([Horridge et al., 2006]), que se utiliza para escribir expresiones de clase OWL. Se basa en *OWL Abstract Syntax* y en *DL Style Syntax*, los cuales usan cualificadores existenciales ($\exists$) y universales ($\forall$). En este tipo de sintaxis, los tipos de datos pueden o no tener tipo, y los tipos disponibles dependerán de las herramientas de apoyo y las que se incluyan en las XSD (*XML Schema Definition*).

Entre las restricciones del lenguaje OWL DL ([Horridge et al., 2004]) encontramos:

1. Las instancias serán atómicas, dos instancias con diferentes nombres serán diferentes.

2. Las propiedades: las relaciones entre los individuos serán binarias, se usarán axiomas para los dominios y los rangos (por ejemplo, *hasTopping* y *helados*), podrán tener subpropiedades, podrán tener como características la transitividad, ser simétricas y funcionales (algo así como propiedades booleanas).

3. Las clases serán un conjunto de instancias.

4. Existirán clases especiales (*[$\top$] owl:Thing y [$\bot$] conjunto vacío*), se podrán combinar utilizando un conjunto de operadores (subconjuntos, disjuntos, unión, intersección, complemento), podrá haber una aproximación intensional en la que se describirán las características de una clase y el sistema completará automáticamente reconociendo a un individuo como una instancia de él y reconociendo subclases y superclases.

OWL	**Símbolo DL**	**Manchester OWL**	**Ejemplo**
someValuesFrom	$\exists$	some	hasChild some Man
allValuesFrom	$\forall$	only	hasSibling only Woman
hasValue	$\ni$	value	hasCountryOfOrigin value England
minCardinality	$\geq$	min	hasChild min 3
cardinality	$=$	exactly	hasChild exactly 3
maxCardinality	$\leq$	max	hasChild max 3
intersectionOf	$\cap$	and	Doctor and Female
unionOf	$\cup$	or	Man or Woman
complementOf	$\neg$	not	not Child

Tabla 25: Restricciones Sintaxis Mánchester, OWL-DL ([Horridge et al., 2006])

Anexo C Integración de Ontologías

La fusión de los candidatos, también llamada, **integración de ontologías** es el proceso de generación de una ontología sencilla, de un tema determinado, a partir de dos o más ontologías existentes de diferentes temas, aunque pueden estar relacionadas ([Choi et al., 2006]). No es una tarea sencilla, son muchos los problemas con los que habrá que enfrentarse tanto a nivel de lenguaje, como a nivel ontológico. Para realizar la comparación entre conceptos se podrán utilizar algoritmos de similaridad, herramientas de integración semántica, buscar equivalencias entre conceptos ...

A priori, podría pensarse en utilizar herramientas de bases de datos para desarrollar esta tarea puesto que las ontologías contienen la mayor parte de las características de un modelo Entidad - Relación (modelo ER) ([Chen, 1981]): los datos están organizados en tablas (clases) y sus relaciones... la diferencia principal con un modelo ER es que una ontología es un modelo de teoría semántica. Dicha semántica proporciona las reglas para la interpretación de la sintaxis que, aparentemente, no tiene un significado directo, pero que restringe las posibles interpretaciones.

El proceso de integración de ontologías (fusión, mapeo, integración y alineamiento) se puede considerar como un proceso de reutilización de ontologías ([Choi et al., 2006]) A la hora de obtener candidatos para realización de la integración, habrá que elegir no sólo la respresentación sino también toda la documentación disponible posible, asimismo, las ontologías fuente deberían elegirse según se ajusten a nuestras necesidades o propósitos ([Pinto and Martins, 2001]).

Dicho proceso puede encontrarse diversos problemas que harán difícil la integración, algunos de los desajustes que presentan las ontologías pueden ser ([Noy, 2005]):

- Los mismos términos describen diferentes conceptos. Pueden ser errores ter-

minológicos como sinónimos (*explorador* ≡ *expedicionario*), homónimos (*vela (cilindro de parafina) - vela (mantenerse despierto)*).

- Diferentes términos describen el mismo concepto, aquí entran los errores de explicación como (*elipse y círculo*).
- Diferentes paradigmas de modelado, como por ejemplo, intervalos para describir aspectos temporales.
- Diferentes convenciones de modelado: (*Revista puede ser clase o propiedad*).
- Diferentes niveles de granularidad: (*empleado público ó funcionario, interino, profesor titular...*).
- Diferente cobertura.
- Diferentes puntos de vista.
- Errores de conceptualización: Diferentes formas de interpretar un dominio.
- Errores de explicación: Diferentes formas de especificar un concepto (*elipse - círculo, círculo - elipse*).
- Errores de codificación: $ddmmaa - mmddaa$.

Definamos cada una de las opciones posibles de mezcla o integración de ontologías y decantémonos por la que mejor se ajuste a nuestras necesidades. No es uno de los objetivos de este trabajo resolver y profundizar en cada uno de los métodos de integración de ontologías sino aportar una visión de cada una de ellas y decantarnos por una.

- **Emparejamiento** y **Alineamiento**

  El *emparejamiento* es el proceso de encontrar relaciones o correspondencias entre entidades de diferentes ontologías.

  El **alineamiento de ontologías** (o *alignment*) es la tarea de crear enlaces o equivalencias entre dos ontologías originales. Normalmente se realiza cuando existen dominios complementarios ([Choi et al., 2006]). La alineación será la salida del proceso de emparejamiento de ontologías.

  Las correspondencias entre ambas ontologías puede ser entre 1 o más clases de la primera ontología a la segunda.

  Sean *o* y *o'* dos ontologías, el **proceso de emparejamiento** (*matching*), que podemos apreciar gráficamente en la figura 47, se puede definir como una función f que, a partir de dicho par de ontologías y de una serie de recursos

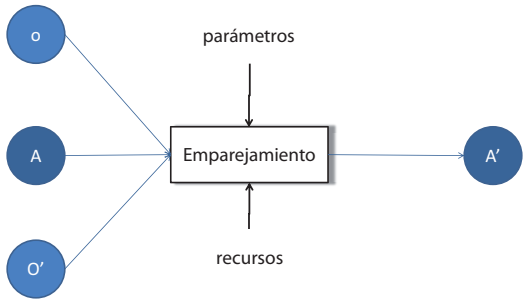

Figura 47: Proceso de emparejamiento de ontologías ([Euzenat and Shvaiko, 2013])

r, parámetros *p* y alineaciones *A*, devuelve una alineación *A'* entre *o* y *o'* ([Euzenat and Shvaiko, 2013]): *A' = f(o, o', A, p, r)*.

- **Fusión (*merge*)**

  ***Merge* o fusión de ontologías** es el proceso de generación una ontología coherente y simple a partir de dos ontologías existentes, o más de dos, y diferentes entre sí, que estén relacionadas con el mismo sujeto. Una ontología coherente y sencilla resultado de un *merge* incluirá información de todas las ontologías fuente sin apenas alterar las originales ([Choi et al., 2006]).

  Para realizar una fusión de ontologías, será necesario realizar una serie de operaciones, no sólo las habituales a la hora de editar ontologías, también algunas específicas de integración de ontologías como pueden ser ([Noy and Musen, 2000]):

  - Mezclar clases.
  - Mezclar *slots*.
  - Mezclar enlaces existentes entre *slots* y clases.
  - Desarrollar una copia en profundidad de una clase desde una ontología a otra, lo cual incluirá todos los padres de una clase hasta la raíz en la jerarquía y de todas las clases y *slots* implicados.
  - Desarrollar una copia superficial de una clase, sólo de ella misma.

Estas operaciones conllevan diferentes conflictos, los estudiados por Noy en su artículo ([Noy and Musen, 2000]) los veremos a continuación, en nuestro estudio hemos detectados otros que mostraremos en el capítulo 5:

- Conflictos de nombres.
- Referencias que se quedan descolgadas.
- Redundancias en la jerarquía de clases, puede haber más de un camino de una clase padre.
- Restricciones que violan la herencia de clases.

Un ejemplo muy conocido sobre integración de ontologías es la aproximación PROMPT ([Fridman Noy and Musen, 2000], [Noy and Musen, 2000]).

En la mezcla de ontologías (figura 48), sea *o"* el resultado de emparejar dos ontologías *o* y *o'*, tendrán un conjunto de entidades relacionadas que se prescribirán gracias a la alineación. La mezcla se puede representar, según ([Euzenat and Shvaiko, 2013]) como:

Mezcla (o, o', A) = o", donde sería conveniente que
Mezcla (o, o', A) $\models$ *o*
Mezcla (o, o', A) $\models$ *o'*
Mezcla (o, o', A) $\models \alpha(A)$,
siendo α(A) la alineación ontológica y
Mezcla (o, o', A) $\models$ *o'*

- **Mapeo (*mapping*)**

  Trata de descubrir similaridades entre dos ontologías fuente. El resultado del emparejamiento será una especificación de similaridades entre dos ontologías.

  Es un método eficiente para la comparación de ontologías. La salida del mapeo será una especificación de la semántica superpuesta entre dos ontologías sin alterar la estructura original.

  Para hallar la correspondencia entre entidades diferentes de dos ontologías, se expresará utilizando axiomas formulados en un lenguaje específico de mapeo. Para conseguir un *mapping* eficiente, algunos autores ([de Bruijn et al., 2006]) sugieren un proceso con tres pasos: descubrimiento, representación y explotación o ejecución. Algunos ejemplos de mapeo de ontologías son: MAFRA y RDFT.

  Dadas dos ontologías *o* y *o'*, el *mapeo* de *o* en *o'* que podemos ver gráficamente en la figura 49, significa para ([Ehrig and Staab, 2004]) que para cada entidad bien sea concepto C, relación R o instancia I de la ontología *o*, se intentará encontrar una entidad correspondiente con el mismo significado, en la ontología *o'*.

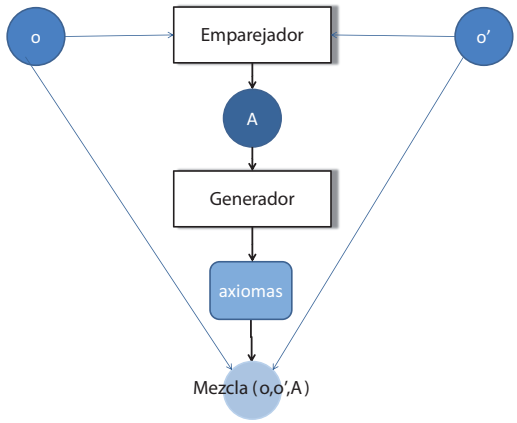

Figura 48: Proceso de mezcla de ontologías ([Euzenat and Shvaiko, 2013])

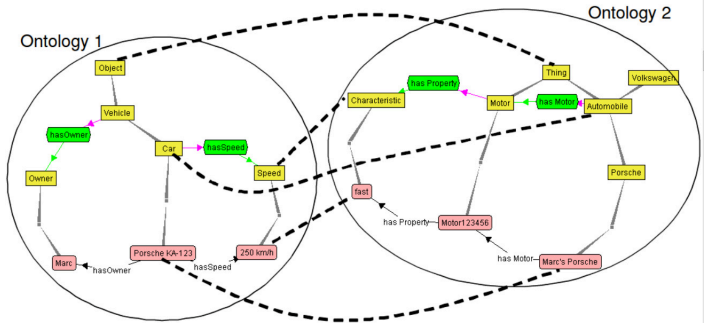

Figura 49: Mapeo de ontologías ([Ehrig and Staab, 2004])

Anexo D Corpus Textuales. Sistemas que Generan Conocimiento

D.1 Corpus Textuales

En los últimos tiempos, existe una tendencia generalizada a la recopilación de datos, recolección de textos y creación de corpus de referencia con el firme propósito de descubrir el conocimiento que se almacena dentro de ellos. Aunque tendremos que saber si realmente estos corpus nos proporcionan la información que necesitamos o si es conveniente el uso de otro tipo de información externa que aporte el conocimiento que falta dentro de la colección.

¿Qué es un **corpus de texto**?

Un corpus puede verse como un cuerpo textual, es decir, como una colección donde exista, al menos, un documento, o bien, como un corpus lingüístico donde el número de documentos no es relevante pero sí otras connotaciones como pueden ser la autoría de los documentos (por ejemplo, dos de los documentos estén escritos por el mismo autor) o el tipo de documentos que alberga la colección ([Hernández, 2002]).

Sin embargo, nosotros estamos interesados en el tratamiento, análisis y representación de grandes colecciones de documentos, ya que éste es uno de los principales problemas que existen en las empresas actuales.

Un corpus puede ser cualquier colección que contenga más de un documento, pero desde el punto de vista lingüístico, Chantal enuncia que un corpus debe estar compuesto por textos producidos en situaciones reales (*pieces of language*) y la selección de los textos que componen el corpus debe corresponder a una serie

de criterios lingüísticos explícitos para asegurar que pueda usarse como muestra representativa de una lengua ([Hernández, 2002]).

Existen dos tipos de clasificaciones (excluyentes) de corpus de texto; la primera de ellas la aporta Atkins ([Atkins, 1992]) y se trata de cuatro tipos genéricos de colecciones de texto mientras que la segunda clasificación, según Chantal ([Hernández, 2002]), es más completa porque sí puede servir para representar a cualquier variedad de corpus de texto. A continuación presentamos una breve descripción de ambas clasificaciones de corpus:

- **Archivos**: se denomina así al conjunto de textos sin relación aparente entre sí, que se almacenan en formato magnético en un lugar común.
- **Bibliotecas de texto en formato magnético (ETL ó *Electronic Text Library*)**: se trata del conjunto de archivos en formato magnético que presentan conatos de estandarización pero sin criterios de selección definidos.
- **Corpus**: es aquel subconjunto de una ETL que se ha creado con unos criterios definidos y con un propósito determinado.
- **Subcorpus**: cualquier subconjunto de un corpus cuya selección se realiza de forma dinámica durante las consultas a dicho corpus.

Respecto a la selección presentada por Chantal, destacamos los siguientes tipos de corpus textuales:

- **Corpus de referencia** (*Reference corpus*): es aquel conjunto de textos que constituye una muestra representativa de las variedades/ estructuras / vocabulario más importantes de una lengua. Su principal misión es construir obras de referencia, gramáticas y / o diccionarios de tal forma que aporten una amplia visión sobre la lengua que se está estudiando.

  Ejemplos: *British National Corpus, el Bank of English y el CREA.*
- **Corpus monitor** (*Monitor corpus*): llamamos así al conjunto de documentos que se almacenan en soporte magnético, sin límite de tamaño y que mantienen una tasa de circulación (*rate of flow*) más o menos constante.
- **Corpus oral** (*Spoken corpus*): aunque no exista una definición unánime para este tipo de corpus, podríamos identificarlo con aquel conjunto de documentos que recoge conversaciones no formales o bien documentos escritos con la finalidad de ser utilizados después para comunicación oral.

- **Corpus de fragmentos textuales** (*Sample corpus*): este tipo de colección de documentos recoge únicamente fragmentos de documentos de idéntico tamaño. Esta política de construcción de corpus se debía a una falta de dispositivos de almacenamiento con capacidad suficiente como para albergar todos los documentos en su totalidad.

  Estos corpus de texto están en contraposición con la gran mayoría de las colecciones que se utilizan en la actualidad, en los que sí se almacena el documento completo (*whole text corpus*).

  Ejemplo: *corpus Brown y corpus LOB, compuestos por 500 fragmentos textuales de 2.000 palabras cada uno.*

- **Corpus especiales, especializados y corpus diseñados con fines especiales**: un corpus especial es aquel tipo de corpus construido para almacenar información específica, por ejemplo, lenguaje infantil. Son especiales porque no tienen todas las características del corpus general, ya que suelen ser de tamaño pequeño y son útiles para un determinado grupo de usuarios.

  Los corpus especializados son un tipo especial de corpus, construidos para que representen una lengua especializada y que representen los términos de esta lengua.

- **Corpus bilingüe (o multilingüe)**: Hay que diferenciar dos tipos de corpus bilingües: los *corpus paralelos* o *bi-texts* que almacenan el texto y su traducción al menos a un idioma distinto y los *corpus comparables* o *paired texts* que albergan tipos similares de textos en más de una lengua para hacer comparaciones interlingüísticas.

  El primer tipo se suele usar en organismos oficiales y / o administrativos de países y comunidades bilingües como puede ser Canadá. El segundo tipo, se está usando en el proyecto NERC (*Network of European Reference Corpora*), donde se construye un corpus en todas las lenguas europeas con idénticas características.

Desde un punto de vista general, identificaremos los siguientes conceptos: corpus, colecciones de texto, colecciones de documentos y corpus textuales. Los utilizaremos indistintamente para referirnos a las colecciones de documentos que van a ser procesadas para extraer información.

D.2 Sistemas que *Generan* Conocimiento

La extracción de conocimiento a partir de textos suele requerir un aporte extra de información que ayude al sistema a ubicarse en el ámbito de trabajo o que le asesore sobre algunos conceptos desconocidos. Para ello, existen una serie de herramientas que, aunque no realizan las mismas funciones ni tienen la misma apariencia ni complejidad, sí que aportan información. Presentamos como ejemplo, herramientas tan diversas como: las bases de conocimiento WordNet y EuroNet, los tesauros y las ontologías.

D.2.1 Lexicón

Es una lista de palabras enriquecida con alguna información extra como puede ser el etiquetado de las mismas, diferentes significados de las palabras o sinónimos ([Hearst, 1999a]).

D.2.2 Ontología

Se entiende por ontología una estructura conceptual que implica diversos procesos de análisis textual, incorpora conocimiento de dominio y, dependiendo del caso, puede devolver el texto etiquetado como XML ([Maedche and Staab, 2000a]).

Entre sus principales características se encuentran: la capacidad de representación de conceptos y de sus relaciones, que suelen ser independientes de contexto y que trabajan con reglas de inferencia ([Hearst, 1999a]).

Se distinguen tres tipos de entidades dentro de las ontologías que merecen ser destacadas, las entidades de Primer Orden, de Segundo y de Tercer Orden ([Lyons, 1977]):

- Entidades de Primer Orden: son aquellas entidades concretas que pueden ser percibidas en cualquier momento y lugar, por ejemplo, *objetos y sustancias.*
- Entidades de Segundo Orden: son aquellas situaciones tanto estáticas como dinámicas que no pueden ser percibidas en sí mismas, por ejemplo: *ser, estar, continuar ...*

- Entidades de Tercer Orden: son las proposiciones que existen pero que no pueden ser observadas aunque no tienen nada que ver con el tiempo y el espacio, por ejemplo, *pensamiento, información, teoría, idea ...*

Dentro de la ontología existen diversos procesos a tener en cuenta, como pueden ser la adquisición de conceptos, el establecimiento de una taxonomía de conceptos o bien descubrir las relaciones que no pertenecen a esa taxonomía ([Maedche and Staab, 2000a]), aunque, a menudo, cuando se habla de la construcción de una ontología sólo se tienen en cuenta la adquisición de conceptos y su relación en lo referente al dominio de la aplicación, pero existen otros tipos de ontologías:

- **Ontología de Dominio** Es aquella ontología en la que sólo se tiene en cuenta el dominio de la aplicación.
- **Ontología de Alto Nivel** En este tipo de ontología, se describen estructuras conceptuales aplicables, basadas en puntos de vista filosóficos o lógicos más allá de las aplicaciones de dichas estructuras.

Las ontologías son útiles en la integración de información inteligente y en el procesamiento del lenguaje natural.

Podemos decir que una ontología es un sistema para la generación de conocimiento, de forma similar a lo que hacen las bases de conocimiento **WordNet**, **GermanNet** o **EuroNet**.

D.2.3 WordNet

Es una base de datos léxica que contiene el número de referencias cruzadas entre nombres y verbos ([WordNet, 2005]). Se trata de un sistema léxico *on-line* de referencia inspirado en las teorías psicolingüísticas de la memoria léxica humana. Los nombres, verbos, adjetivos y adverbios del habla inglesa se organizan dentro de conjuntos de sinónimos, cada uno de los cuales representa un concepto léxico subyacente.

D.2.4 EuroNet

EuroNet, también llamada **EuroWordNet** es una base de datos multilingüe (danés, italiano, español, alemán, francés, checo y estonio) basada en WordNet ([Vossen, 1998]).

Se estructura, al igual que WordNet, en conjuntos de palabras sinónimas con relaciones de sinonimia. Cada uno de los lenguajes es soportado por una WordNet y todas ellas se conectan mediante un índice, el cual, facilita la navegación entre palabras de diferentes idiomas al tiempo que proporciona una ontología de alto nivel con una gran cantidad de distinciones semánticas (la cual aporta el marco de trabajo semántico común para todos los lenguajes).

Esta base de datos se puede utilizar como herramienta para la recuperación de información sea o no sobre documentos monolingües.

D.2.5 GermanNet

Una red semántica de léxico para el alemán que se basa en WordNet; en ella, cada concepto semántico se representa por un grupo de palabras ([Hamp and Feldwig, 1997]).

D.2.6 Tesauros

Un tesauro es una estructura de datos compuesta de una lista precompilada de palabras importantes en un dominio de conocimiento dado y de una lista de sinónimos para cada palabra de esa lista. Estos sinónimos puede que no se encuentren en un diccionario habitual porque los tesauros suelen abarcar un campo de trabajo muy técnico.

Formalmente, un tesauro se define como *'lenguaje documental controlado formado por un conjunto de descriptores normalizados, agrupados en familias semánticas y relacionados entre sí de forma jerárquica y asociativa'* ([Prada, 1996]).

Entre las funciones del tesauro, encontramos tanto la de indexación como la de recuperación de datos o documentos ([Maedche and Staab, 2000a]). Una de las utilidades del mismo es la traducción del lenguaje natural utilizado en consultas en un lenguaje más técnico, es decir, es una herramienta de control terminológico

	Unidades Léxicas	
Descriptores	Términos Equivalentes	Identificadores

Tabla 26: Elementos de un Tesauro (I)

		Relaciones		
Definitoria	Equivalencia	Alternativa	Jerárquica	Asociativa

Tabla 27: Elementos de un Tesauro (II)

porque utiliza un vocabulario, que también se puede llamar controlado, formado por una lista de descriptores normalizados fijados por expertos antes del análisis. Los elementos básicos de un tesauro pueden verse en la tabla 27 ([Prada, 1996]).

D.2.7 Twitter

El *microblogging* se utiliza por una cantidad de usuarios en todo momento para expresar su opinión sobre diferentes tópicos. Una de las herramientas de *microblogging* de gran éxito en los últimos años es *Twitter* que contiene una enorme cantidad de *posts*. El tamaño de dicho corpus puede ser arbitrariamente largo ([Pak and Paroubek, 2010]). Cada post constaba hasta hace poco de un número máximo de 140 caracteres. La variedad tanto de idiomas, tendencias, sentimientos y opiniones es amplísima.

Anexo E Medidas

Las capacidades de precisión y de respuesta de las ontologías se pueden controlar con diferentes medidas, bien tomando tiempos y memoria, bien contando el número de elementos clasificados. En este anexo revisaremos un tipo de medida utilizado en los trabajos con ontologías, como es la **Medida Heurística** ([Hellmann et al., 2009]). Esta medida servirá para:

- Ver cómo de bien se ajusta una ontología a un algoritmo de aprendizaje.
- Medir la precisión de la ontología, como

$$\frac{EjemplosClasificadosCorrectamente}{TotalEjemplos} \tag{E.1}$$

Existen diversas heurísticas procedentes de la Recuperación de Información que nos pueden ser de utilidad a la hora de representar los resultados, nos quedaremos sólo con algunas, véase:

- *F-Measure* ([Li et al., 2008]): se utiliza en algoritmos de búsqueda y en clasificación de documentos. Se mide la precisión y la exhaustividad (del inglés *recall*) del algoritmo:

$$F_Measure = 2 * \frac{cobertura * exhaustividad}{cobertura + exhaustividad} \tag{E.2}$$

  La cobertura o precisión es la proporción de casos en los que el elemento clasificado como positivo también fueron positivos, mientras que la exhaustividad es la proporción de casos positivos en el estándar de oro que fueron clasificados como positivos por un sujeto.

- Coeficiente de Jaccard ([Jaccard, 1912]): que proporciona la similaridad y diversidad de dos conjuntos de ejemplos. Sean A y B dos conjuntos de datos, el

coeficiente de Jaccard, *J(A,B)*, será

$$J(A,B) = \frac{\mid A \cap B \mid}{\mid A \cup B \mid} \tag{E.3}$$

Si A y B son vacíos, $J(A,B) = 1$

- Distancia Jaccard (Medida de Similaridad de Jaccard) ([Jaccard, 1912]), es la complementaria al Coeficiente de Jaccard:

$$d_J(A,B) = 1 - J(A,B) \tag{E.4}$$

Y otras que se pueden aplicar directamente a las ontologías como pueden ser:

- Medida del número de Equivalencias.
- Medida del número de Redundancias: si una clase A es más grande que una clase B y ambas tienen las mismas características (*slots*), entonces podemos asegurar que A=B.
- Grado de precisión: igual nombre y distintos objetos.

Bibliografía

[Abdessalem et al., 2016] Abdessalem, W. B., Dridi, K., and Alkhammash, E. (2016). Fuzzy decision trees for text document clustering. In *Communication, Management and Information Technology: Proceedings of the International Conference on Communication, Management and Information Technology (Iccmit 2016)*, pages 439–446. CRC Press.

[Abney and Abney, 1991] Abney, S. and Abney, S. P. (1991). Parsing by chunks. In *Principle-Based Parsing*, pages 257–278. Kluwer Academic Publishers.

[Aggarwal and Zhai, 2012] Aggarwal, C. C. and Zhai, C. (2012). *Mining text data*. Springer Science & Business Media.

[Agrawal and Shafer, 1996] Agrawal, R. and Shafer, J. C. (1996). Parallel mining of association rules. *Ieee Trans. On Knowledge And Data Engineering*, 8:962–969.

[Agrawal et al., 1994] Agrawal, R., Srikant, R., et al. (1994). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499.

[Ahmad and Gillam, 2005] Ahmad, K. and Gillam, L. (2005). Automatic ontology extraction from unstructured texts. In *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, pages 1330–1346. Springer.

[Ahonen et al., 1998] Ahonen, H., Heinonen, O., Klemettinen, M., and Verkamo, A. I. (1998). Applying data mining techniques for descriptive phrase extraction in digital document collections. In *Research and Technology Advances in Digital Libraries, 1998. ADL 98. Proceedings. IEEE International Forum on*, pages 2–11. IEEE.

[Ahonen et al., 1997] Ahonen, H., Heinonen, O., Klemettinen, M., and Verkamo, I. (1997). Applying data mining techniques in text analysis. Technical Report C-1997-23, University of Helsinki, Departament of Computer Science.

[Ahonen-Myka et al., 1999] Ahonen-Myka, H., Heinonen, O., Klemettinen, M., and Verkamo, A. I. (1999). Finding co-occurring text phrases by combining sequence and frequent set discovery. In *Proceedings of 16th International Joint Conference on Artificial Intelligence IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications*, pages 1–9.

[Allan et al., 1998] Allan, J., Papka, R., and Lavrenko, V. (1998). On-line new event detection and tracking. In *Research and Development in Information Retrieval*, pages 37–45.

[Ananiadou and Mcnaught, 2005] Ananiadou, S. and Mcnaught, J. (2005). *Text Mining for Biology And Biomedicine*. Artech House, Inc., Norwood, MA, USA.

[Ananiadou et al., 2010] Ananiadou, S., Pyysalo, S., Tsujii, J., and Kell, D. B. (2010). Event extraction for systems biology by text mining the literature. *Trends in biotechnology*, 28(7):381–390.

[Andreasen and Nilsson, 2014] Andreasen, T. and Nilsson, J. F. (2014). A case for embedded natural logic for ontological knowledge bases. In Filipe, J., Dietz, J. L. G., and Aveiro, D., editors, *KEOD 2014 - Proceedings of the International Conference on Knowledge Engineering and Ontology Development, Rome, Italy, 21-24 October, 2014*, pages 423–427. SciTePress.

[API, 2011] API, T. O. (2011). Owlontologymerger api. Online; last access 07-September-2015.

[Ari Pirkola and Järvelin, 2003] Ari Pirkola, J. T. H. K. K. V. and Järvelin, K. (2003). Fuzzy translation of cross-lingual spelling variants. In *SIGIR2003*, pages 345–352.

[Aristotle, 1994] Aristotle (1994). *Aristotle Metaphysics*. Oxford University Press.

[Arotaritei and Mitra, 2004] Arotaritei, D. and Mitra, S. (2004). Web mining: a survey in the fuzzy framework. *Fuzzy Sets and Systems*, 148(1):5–19.

[Atkins, 1992] Atkins, B. (1992). Tools for computer-aided lexicography: the {HECTOR} project. *F. Kiefer, G. Kiss and J. Pajzs (eds.)*, pages 1–61.

[Atkinson-Abutridy et al., 2004] Atkinson-Abutridy, J., Mellish, C., and Aitken, S. (2004). Combining information extraction with genetic algorithms for text mining. *Intelligent Systems, IEEE*, 19(3):22–30.

[Baeza-Yates and Tiberi, 2007] Baeza-Yates, R. and Tiberi, A. (2007). Extracting semantic relations from query logs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 76–85. ACM.

[Barzilay and Elhadad, 2002] Barzilay, R. and Elhadad, N. (2002). Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, pages 35–55.

[Baudelaire, 1988] Baudelaire, C. (1988). Edgar a. poe: Su vida y sus obras. *Edgar Allan Poe*, pages 43–79.

[Becker, 2005] Becker, C. (2005). Contextual vocabulary acquisition of verbs. *(online). Visited on Dec*, 11.

[Beneventano et al., 2003] Beneventano, D., Bergamaschi, S., Guerra, F., and Vincini, M. (2003). Building an integrated ontology within sewasie system. In *SWDB*, pages 91–107. Citeseer.

[Berendt et al., 2002] Berendt, B., Hotho, A., and Stumme, G. (2002). Towards semantic web mining. In *The Semantic Web—ISWC 2002*, pages 264–278. Springer.

[Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., Lassila, O., et al. (2001). The semantic web. *Scientific american*, 284(5):28–37.

[Berry and Castellanos, 2004] Berry, M. W. and Castellanos, M. (2004). Survey of text mining. *Computing Reviews*, 45(9):548.

[Bezdek et al., 1984] Bezdek, J. C., Ehrlich, R., and Full, W. (1984). Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203.

[Biemann, 2005] Biemann, C. (2005). Ontology learning from text: A survey of methods. *LDV Forum*, 20(2):75–93.

[Bifet and Frank, 2010] Bifet, A. and Frank, E. (2010). Sentiment knowledge discovery in twitter streaming data. In *Discovery Science*, pages 1–15. Springer. Online; last access 30-March-2015.

[Bisbal et al., 2004] Bisbal, E., Molina, A., and Moreno, L. (2004). Podado y lexicalización de reglas gramaticales y su aplicación al análisis sintáctico parcial. *Procesamiento del lenguaje natural*, 33.

[Bloom, 1991] Bloom, A. D. (1991). *The republic of Plato*. Basic Books.

[Bobrow et al., 2009] Bobrow, D. G., Condoravdi, C., Karttunen, L., and Zaenen, A. (2009). Learning by reading: Normalizing complex linguistic structures onto a knowledge representation. In *Learning by Reading and Learning to Read, Papers from the 2009 AAAI Spring Symposium, Technical Report SS-09-07, Stanford, California, USA, March 23-25, 2009*, pages 3–6.

[Bonillo and Betanzos, 1998] Bonillo, V. M. and Betanzos, A. A. (1998). *Fundamentos de Inteligencia Artificial. Servicio de Publicaciones*. Universidad Da Coruña.

[Brachman et al., 1993] Brachman, R. J., Selfridge, P. G., Terveen, L. G., Altman, B., Halper, F., Kirk, T., Lazar, A., and Mcguinness, D. L. (1993). Integrated support for data archaeology. In *In Proceedings of KDD-9J Workshop, Menlo Park*. AAAI Press.

[Brau, 1972] Brau, J.-L. (1972). *Biografía de Antonin Artaud*. Anagrama.

[Brewster and Miller, 2000] Brewster, M. E. and Miller, N. E. (2000). Information retrieval system utilizing wavelet transform. US Patent 6,070,133.

[Buitelaar et al., 2005] Buitelaar, P., Cimiano, P., and Magnini, B. (2005). *Ontology learning from text: methods, evaluation and applications*, volume 123. IOS press.

[Buitelaar et al., 2003] Buitelaar, P., Olejnik, D., and Sintek, M. (2003). Ontolt: A protege plug-in for ontology extraction from text. In *Proceedings of the International Semantic Web Conference (ISWC)*.

[Calvanese et al., 2001] Calvanese, D., De Giacomo, G., Lenzerini, M., and Nardi, D. (2001). Reasoning in expressive description logics. *Handbook of Automated Reasoning*, 2:1581–1634.

[Caracciolo, 1979] Caracciolo, R. A. (1979). Contradictions in the legal system. *ARSP: Archiv für Rechts- und Sozialphilosophie / Archives for Philosophy of Law and Social Philosophy*, 65(4):pp. 457–473.

[Cardie and Mooney, 1999] Cardie, C. and Mooney, R. J. (1999). Guest editors' introduction: Machine learning and natural language. *Machine Learning*, 34(1-3):5–9.

[Cavnar, 1994] Cavnar, W. B. (1994). Using an n-gram based document representation with a vector processing retrieval model. In *NIST Special Publication 500-226: Overview of the 3th Text Retrieval Conference (TREC-3)*, pages 269–278.

[Chandrasekaran et al., 1999] Chandrasekaran, B., Josephson, J., and Benjamins, V. (1999). What are ontologies, and why do we need them? *IEEE Intelligent Systems*, 14:20–26.

[Chang et al., 1996] Chang, K. H., Raman, P., Carlisle, W. H., and Cross, J. H. (1996). A self-improving helpdesk service system using case-based reasoning techniques. *Computers in Industry*, 30(2):113–125.

[Chein and Mugnier, 2004] Chein, M. and Mugnier, M.-L. (2004). Concept types and coreference in simple conceptual graphs. In *Conceptual Structures at Work: 12th International Conference on Conceptual Structures, ICCS 2004*, volume 3127 / 2004, pages 303–318.

[Chen, 2005] Chen, C. (2005). Top 10 unsolved information visualization problems. *Computer Graphics and Applications, IEEE*, 25(4):12–16.

[Chen et al., 2005] Chen, L., Liu, H., and Friedman, C. (2005). Gene name ambiguity of eukaryotic nomenclatures. *Bioinformatics*, 21(2):248–256.

[Chen, 1981] Chen, P. P. (1981). A preliminary framework for entity-relationship models. In *ER*, pages 19–28.

[Choi et al., 2006] Choi, N., Song, I.-Y., and Han, H. (2006). A survey on ontology mapping. *ACM Sigmod Record*, 35(3):34–41.

[Cimiano, 2006] Cimiano, P. (2006). *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Clark and LLC, 2011] Clark and LLC, P. (2011). Pellet. Online; last access 19-October-2015.

[Company, 2014] Company, S. W. (2014). Pool party. Online; last access 12-October-2015.

[Conde et al., 2015] Conde, A., Larrañaga, M., Arruarte, A., Elorriaga, J. A., and Roth, D. (2015). litewi: A combined term extraction and entity linking method for eliciting educational ontologies from textbooks. *Journal of the Association for Information Science and Technology*.

[Corcho et al., 2003] Corcho, O., Fernández-López, M., and Gómez-Pérez, A. (2003). Methodologies, tools and languages for building ontologies. where is their meeting point? *Data & knowledge engineering*, 46(1):41–64.

[Datcu et al., 2003] Datcu, M., Daschiel, H., Pelizzari, A., Quartulli, M., Galoppo, A., Colapicchioni, A., Pastori, M., Seidel, K., Marchetti, P. G., and d'Elia, S. (2003). Information mining in remote sensing image archives: system concepts. *Geoscience and Remote Sensing, IEEE Transactions on*, 41(12):2923–2936.

[Dave et al., 2003] Dave, K., Lawrence, S., and Pennock, D. M. (2003). Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th international conference on World Wide Web*, pages 519–528. ACM.

[DBpedia, 2015] DBpedia (2015). The dbpedia knowledge base. Online; last access 8-September-2015.

[de Bruijn et al., 2006] de Bruijn, J., Ehrig, M., Feier, C., Martín-Recuerda, F., Scharffe, F., and Weiten, M. (2006). Ontology mediation, merging and aligning. *Semantic web technologies*, pages 95–113.

[de Marneffe et al., 2008] de Marneffe, M., Rafferty, A. N., and Manning, C. D. (2008). Finding contradictions in text. In McKeown, K., Moore, J. D., Teufel, S., Allan, J., and Furui, S., editors, *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA*, pages 1039–1047. The Association for Computer Linguistics.

[de Paiva et al., 2007] de Paiva, V., Bobrow, D. G., Condoravdi, C., Crouch, D., King, T. H., Karttunen, L., Nairn, R., and Zaenen, A. (2007). Textual inference logic: Take two. In *Proceedings of the International Workshop on Contexts and Ontologies: Representation and Reasoning (C&O:RR) Collocated with the 6th International and Interdisciplinary Conference on Modelling and Using Context (CONTEXT-2007), Roskilde, Denmark, August 21st, 2007*.

[De Weerdt et al., 2012] De Weerdt, J., Vanden Broucke, S. K., Vanthienen, J., and Baesens, B. (2012). Leveraging process discovery with trace clustering and text mining for intelligent analysis of incident management processes. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8. IEEE.

[Delgado et al., 2005] Delgado, M., Marín, N., Martín-Bautista, M., Sanchez, D., and Vila, M.-A. (2005). Mining fuzzy association rules: an overview. In *Soft Computing for Information Processing and Analysis*, pages 351–373. Springer.

[Delgado et al., 2002a] Delgado, M., Martín-Bautista, M., Sánchez, D., and Vila, M. (2002a). Mining text data: Special features and patterns. pattern detection and discovery. In *Proceedings ESF Exploratory Workshop 2002*.

[Delgado et al., 2003] Delgado, M., Marín, N., Sánchez, D., and Vila, M. A. (2003). Fuzzy association rules: General model and applications. *IEEE Transactions on Fuzzy systems. Vol. 11. Nº 2*.

[Delgado et al., 2008] Delgado, M., Ruíz, M., and Sánchez, D. (2008). Reglas de asociación difusas: Nuevos retos. In *Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTYLF08) Cuencas Mineras (Mieres-Langreo)*.

[Delgado et al., 2001] Delgado, M., Sánchez, D., Martín-Bautista, M. J., and Vila, M.-A. (2001). Mining association rules with improved semantics in medical databases. *Artificial Intelligence in Medicine*, 21(1):241–245.

[Delgado et al., 2002b] Delgado, M., Sanchez, D., and Vila, M. (2002b). Acquisition of fuzzy association rules from medical data. In *Fuzzy Logic in Medicine. Studies in Fuzziness and Soft Computing Series. Vol. 83, pp. 286-310*, Alemania: Physica Verlag.

[Deng et al., 2010] Deng, J., Hu, J., Chi, H., and Wu, J. (2010). An improved fuzzy clustering method for text mining. In *Networks Security Wireless Communica-*

tions and Trusted Computing (NSWCTC), 2010 Second International Conference on, volume 1, pages 65–69. IEEE.

[Do Prado, 2007] Do Prado, H. A. (2007). *Emerging Technologies of Text Mining: Techniques and Applications: Techniques and Applications*. IGI Global.

[Dubois and Quafafou, 2002] Dubois, V. and Quafafou, M. (2002). Incremental and dynamic text mining. *ISMIS 2002 265-273*.

[Ehrig and Staab, 2004] Ehrig, M. and Staab, S. (2004). Qom–quick ontology mapping. In *The Semantic Web–ISWC 2004*, pages 683–697. Springer.

[Española, 2010] Española, R. A. (2010). Manual de la nueva gramática de la lengua española. *Madrid. Asociación de Academias de la Lengua Española*.

[Etzioni, 1996] Etzioni, O. (1996). The world-wide web: quagmire or gold mine? *Communications of the ACM*, 39(11):65–68.

[Euzenat and Shvaiko, 2013] Euzenat, J. and Shvaiko, P. (2013). *Ontology Matching, Second Edition*. Springer.

[Everett et al., 2002] Everett, J. O., Bobrow, D. G., Stolle, R., Crouch, R., de Paiva, V., Condoravdi, C., van den Berg, M., and Polanyi, L. (2002). Making ontologies work for resolving redundancies across documents. *Commun. ACM*, 45(2):55–60.

[Fan et al., 2006] Fan, W., Wallace, L., Rich, S., and Zhang, Z. (2006). Tapping the power of text mining. *Communications of the ACM*, 49(9):76–82.

[Faure et al., 1998] Faure, D., Nédellec, C., and Rouveirol, C. (1998). Acquisition of semantic knowledge using machine learning methods: The system" asium". In *Universite Paris Sud*.

[Fayyad et al., 1996] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). The kdd process for extracting useful knowledge from volumes of data. In *Communication of the ACM*, volume 29, pages 27–34.

[Federico et al., 2011] Federico, P., Aigner, W., Miksch, S., Windhager, F., and Zenk, L. (2011). A visual analytics approach to dynamic social networks. In *Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies*, page 47. ACM.

[Feldman et al., 2002] Feldman, R., Aumann, Y., Schler, J., Landau, D., Lipshtat, O., and Ben-Yehuda, Y. (2002). Term-level text with mining with taxonomies. US Patent 6,442,545.

[Feldman and Dagan, 1995] Feldman, R. and Dagan, I. (1995). Knowledge discovery in textual databases (kdt). In *KDD*, volume 95, pages 112–117.

[Feldman et al., 1998a] Feldman, R., Fresko, M., Hirsh, H., Aumann, Y., Liphstat, O., Schler, Y., and Rajman, M. (1998a). Knowledge management: A text mining approach. In *PAKM*, volume 98, page 9.

[Feldman et al., 1998b] Feldman, R., Fresko, M., Kinar, Y., Lindell, Y., Liphstat, O., Rajman, M., Schler, Y., and Zamir, O. (1998b). Text mining at the term level. In *Principles of Data Mining and Knowledge Discovery*, pages 65–73.

[Feldman and Hirsh, 1996] Feldman, R. and Hirsh, H. (1996). Mining associations in text in the presence of background knowledge. In *Knowledge Discovery and Data Mining*, pages 343–346.

[Feldman and Sanger, 2007] Feldman, R. and Sanger, J. (2007). *The text mining handbook: advanced approaches in analyzing unstructured data*. Cambridge University Press.

[Fensel and Morozova, 2010] Fensel, D. and Morozova, O. (2010). Generating semantic annotations.

[Fernandez et al., 2006] Fernandez, V. F., Unanue, R. M., Herranz, S. M., and Rubio, A. C. (2006). Naive bayes web page classification with html mark-up enrichment. In *Computing in the Global Information Technology, 2006. ICCGI'06. International Multi-Conference on*, pages 48–48. IEEE.

[Fernández-López et al., 2006] Fernández-López, M., Gómez-Pérez, A., and Ramos, J. (2006). Oegmerge: a case-based model for merging ontologies. *MDAI 2006: Modeling Decisions for Artificial Intelligence*.

[Fox, 2014] Fox (2014). Federated knowledge extraction framework. Online; last access 12-October-2015.

[Franco García, 2004] Franco García, A. (2004). Física con ordenador. Online; last access 19-October-2015.

[Frawley et al., 1992] Frawley, W. J., Piatetsky-Shapiro, G., and Matheus, C. J. (1992). Knowledge discovery in databases: An overview. *AI magazine*, 13(3):57.

[Fridman Noy and Musen, 2000] Fridman Noy, N. and Musen, M. A. (2000). Prompt: algorithm and tool for automated ontology merging and alignment. *AAAI/IAAI*, 2000:450–455.

[Fung et al., 2005] Fung, G. P. C., Yu, J. X., Yu, P. S., and Lu, H. (2005). Parameter free bursty events detection in text streams. In *Proceedings of the 31st international conference on Very large data bases*, pages 181–192. VLDB Endowment.

[Furnkranz, 1998] Furnkranz, J. (1998). A study using n-gram features for text categorization. *Austrian Research Institute for Artificial Intelligence Technical Report OEFAI-TR-98-30 Schottengasse*, 3.

[Fürnkranz and Hüllermeier, 2016] Fürnkranz, J. and Hüllermeier, E. (2016). Special issue on discovery science. *Information Sciences: an International Journal*, 329(C):849–850.

[Gaeta et al., 2011] Gaeta, M., Orciuoli, F., Paolozzi, S., and Salerno, S. (2011). Ontology extraction for knowledge reuse: The e-learning perspective. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 41(4):798–809.

[Gamon et al., 2005] Gamon, M., Aue, A., Corston-Oliver, S., and Ringger, E. (2005). Pulse: Mining customer opinions from free text. In *Advances in Intelligent Data Analysis VI*, pages 121–132. Springer.

[García-Honrado and Trillas, 2012] García-Honrado, I. and Trillas, E. (2012). On an attempt to formalize guessing. In [Seising and Sanz, 2012], pages 237–255.

[García, 2009] García, M. N. (2009). Modelado y análisis de sistemas cscw siguiendo un enfoque de ingeniería dirigida por ontologías.

[G.A.Ringland, 1988] G.A.Ringland, D. D. (1988). *Approaches to Knowledge Representation- An Introduction*. Research Studies Press ltd.

[Gelfand and Wulfekuhler, 1998] Gelfand, B. and Wulfekuhler, M. (1998). Automated concept extraction from plain text.

[Gelfand et al., 1998] Gelfand, B., Wulfekuhler, M., and Punch, W. (1998). Discovering concepts in raw texts: Building semantic relationship graphs. In *ICML/AAAI workshop on learning for text categorization*.

[George Chang and Wang, 2001] George Chang, Marcus J. Healey, J. A. M. M. and Wang, J. T. L. (2001). *Mining the World Wide Web*. Kluwer Academic Publishers.

[Ghose and Ipeirotis, 2011] Ghose, A. and Ipeirotis, P. G. (2011). Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics. *IEEE Transactions on Knowledge and Data Engineering*, 23(10):1498–1512.

[Ghosh et al., 2012] Ghosh, S., Roy, S., and Bandyopadhyay, S. K. (2012). A tutorial review on text mining algorithms. *International Journal of Advanced Research in Computer and Communication Engineering*, 1(4):7.

[Giannotti et al., 1999] Giannotti, F., Manco, G., Nanni, M., Pedreschi, D., and Turini, F. (1999). Integration of deduction and induction for mining supermarket sales data. In *SEBD*, pages 117–131.

[Goebel and Gruenwald, 1999] Goebel, M. and Gruenwald, L. (1999). A survey of data mining and knowledge discovery software tools. *ACM SIGKDD Explorations Newsletter*, 1(1):20–33.

[Goswami and Shishodia, 2013] Goswami, S. and Shishodia, M. S. (2013). A fuzzy based approach to text mining and document clustering. *arXiv preprint arXiv:1306.4633*.

[Grau et al., 2007] Grau, B. C., Halaschek-Wiener, C., and Kazakov, Y. (2007). History matters: Incremental ontology reasoning using modules. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, pages 183–196.

[Greco et al., 2001] Greco, S., Masciari, E., and Pontieri, L. (2001). Combining inductive and deductive tools for data analysis. *AI Commun.*, 14(2):69–82.

[Gruber, 1993] Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220.

[Grüninger et al., 2000] Grüninger, M., Atefi, K., and Fox, M. S. (2000). Ontologies to support process integration in enterprise engineering. *Computational and Mathematical Organization Theory*, 6(4):381–394.

[Gupta and Lehal, 2009] Gupta, V. and Lehal, G. S. (2009). A survey of text mining techniques and applications. *Journal of emerging technologies in web intelligence*, 1(1):60–76.

[H. Karanikas and Theodoulidis, 2000] H. Karanikas, C. T. and Theodoulidis, B. (2000). An approach to text mining using information extraction. In *Knowledge Management Theory Applications Workshop, (KMTA 2000)*.

[Hajic et al., 2010] Hajic, J., Carberry, S., and Clark, S., editors (2010). *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*. The Association for Computer Linguistics.

[Hamp and Feldwig, 1997] Hamp, B. and Feldwig, H. (1997). GermaNet — A lexical-semantic net for German. In Vossen, P., Adriaens, G., Calzolari, N., Sanfilippo, A., and Wilks, Y., editors, *Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 9–15. Association for Computational Linguistics, New Brunswick, New Jersey.

[Han and Kamber, 2000] Han, J. and Kamber, M. (2000). Data mining - concepts and techniques. In *ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, Madrid.

[He et al., 2013] He, W., Zha, S., and Li, L. (2013). Social media competitive analysis and text mining: A case study in the pizza industry. *International Journal of Information Management*, 33(3):464–472.

[Hearst, 1999a] Hearst, M. A. (1999a). http://www.sims.berkeley.edu/courses/is296a-4/f99/. Online; last access 10-September-2015.

[Hearst, 1999b] Hearst, M. A. (1999b). Untangling text data mining. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 3–10. Association for Computational Linguistics.

[Heijst et al., 1997] Heijst, G. V., Schreiber, A., and Wielinga, B. (1997). Using explicit ontologies in KBS development. *International Journal of Human-Computer Studies*, pages 183–191.

[Hellmann et al., 2009] Hellmann, S., Lehmann, J., and Auer, S. (2009). Learning of owl class descriptions on very large knowledge bases. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 5(2):25–48.

[Hernández, 2002] Hernández, C. P. (2002). Explotación de los córpora textuales informatizados para la creación de bases de datos terminológicas basadas en el conocimiento. *Estudios de lingüística del español*, 18:1.

[Holt and Chung, 2001] Holt, J. D. and Chung, S. M. (2001). Multipass algorithms for mining association rules in text databases. *Knowledge and Information Systems*, 3(2):168–183.

[Horridge et al., 2006] Horridge, M., Drummond, N., Goodwin, J., Rector, A. L., Stevens, R., and Wang, H. (2006). The manchester owl syntax. In *OWLed*, volume 216.

[Horridge et al., 2004] Horridge, M., Knublauch, H., Rector, A., Stevens, R., and Wroe, C. (2004). A practical guide to building owl ontologies using the protégé-owl plugin and co-ode tools edition 1.0. *University of Manchester*.

[Horrocks and Sattler, 2002] Horrocks, I. and Sattler, U. (2002). Logical foundations for the semantic web. *Reasoning with Expressive Description Logics: Theory and Practice CADE-18 Invited Talk*.

[Hotho et al., 2005] Hotho, A., Nürnberger, A., and Paaß, G. (2005). A brief survey of text mining. In *Ldv Forum*, volume 20, pages 19–62.

[Hsu and Yih, 1997] Hsu, J. Y.-j. and Yih, W.-t. (1997). Template-based information mining from html documents. In *AAAI/IAAI*, pages 256–262.

[Hu and Liu, 2004] Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

[Hu and Chen, 2006] Hu, Y.-H. and Chen, Y.-L. (2006). Mining association rules with multiple minimum supports: a new mining algorithm and a support tuning mechanism. *Decision Support Systems*, 42(1):1–24.

[Iiritano and Ruffolo, 2001] Iiritano, S. and Ruffolo, M. (2001). Managing the knowledge contained in electronic document: Clustering method for text mining. *IEEE 2001*.

[ISO704, 2000] ISO704 (2000). Terminology work principles and methods. Online; last access 18-August-2015.

[Jaccard, 1912] Jaccard, P. (1912). The distribution of the flora in the alpine zone. *New phytologist*, 11(2):37–50.

[Jin and Srihari, 2007] Jin, W. and Srihari, R. K. (2007). Graph-based text representation and knowledge discovery. In *Proceedings of the 2007 ACM symposium on Applied computing*, pages 807–811. ACM.

[Jipkate and Gohokar, 2012] Jipkate, B. R. and Gohokar, V. (2012). A comparative analysis of fuzzy c-means clustering and k means clustering algorithms. *International Journal Of Computational Engineering Research*, 2(3):737–739.

[Justicia de la Torre et al., 2006] Justicia de la Torre, C., Martın-Bautista, M., Sánchez, D., and Blanco, I. (2006). Un enfoque deductivo para la mineria de textos. *Actas del Congreso Español sobre Tecnologías y Lógica Fuzzy. Estylf 2006*, pages 241–246.

[Justicia de la Torre et al., 2008] Justicia de la Torre, M., Sánchez Fernández, D., Blanco Medina, I. J., and Martín-Bautista, M. J. (2008). Text knowledge mining: An approach to text mining. *Spain, ESTYLF*.

[Kalyanpur et al., 2005a] Kalyanpur, A., Parsia, B., Cuenca-Grau, B., and Sirin, E. (2005a). Tableaux tracing in shoin. Technical report, Tech. Rep. 2005-66, University of Maryland.

[Kalyanpur et al., 2005b] Kalyanpur, A., Parsia, B., Sirin, E., and Hendler, J. (2005b). Debugging unsatisfiable classes in owl ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(4):268–293.

[Kantardzic, 2011] Kantardzic, M. (2011). *Data mining: concepts, models, methods, and algorithms*. John Wiley & Sons.

[Kargupta et al., 1997] Kargupta, H., Hamzaoglu, I., and Stafford, B. (1997). Scalable, distributed data mining-an agent architecture. In *KDD*, pages 211–214.

[Kaski et al., 1998] Kaski, S., Honkela, T., Lagus, K., and Kohonen, T. (1998). Websom—self-organizing maps of document collections. *Neurocomputing*, 21:101–117.

[Kaya and Alhajj, 2005] Kaya, M. and Alhajj, R. (2005). Genetic algorithm based framework for mining fuzzy association rules. *Fuzzy sets and systems*, 152(3):587–601.

[Kazakov and Klinov, 2013] Kazakov, Y. and Klinov, P. (2013). Incremental reasoning in owl el without bookkeeping. In Alani, H., Kagal, L., Fokoue, A., Groth, P. T., Biemann, C., Parreira, J. X., Aroyo, L., Noy, N. F., Welty, C., and Janowicz, K., editors, *International Semantic Web Conference (1)*, volume 8218 of *Lecture Notes in Computer Science*, pages 232–247. Springer.

[Kešelj et al., 2003] Kešelj, V., Peng, F., Cercone, N., and Thomas, C. (2003). N-gram-based author profiles for authorship attribution. In *Proceedings of the conference pacific association for computational linguistics, PACLING*, volume 3, pages 255–264.

[Khomra, 2005] Khomra, T. (2005). Ai: Artificial intelligence. it4307 / 5307.

[Kiddon and Domingos, 2012] Kiddon, C. and Domingos, P. (2012). Knowledge extraction and joint inference using tractable markov logic. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX '12, pages 79–83, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Kim and Zhai, 2009] Kim, H. D. and Zhai, C. (2009). Generating comparative summaries of contradictory opinions in text. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 385–394, New York, NY, USA. ACM.

[Kim et al., 2008] Kim, J.-D., Ohta, T., and Tsujii, J. (2008). Corpus annotation for mining biomedical events from literature. *BMC bioinformatics*, 9(1):10.

[Kim et al., 2011] Kim, K., Ko, S., Elmqvist, N., and Ebert, D. S. (2011). Wordbridge: Using composite tag clouds in node-link diagrams for visualizing content and relations in text corpora. In *System Sciences (HICSS), 2011 44th Hawaii International Conference on*, pages 1–8. IEEE.

[Klösgen, 1996] Klösgen, W. (1996). Explora: A multipattern and multistrategy discovery assistant. In *Advances in knowledge discovery and data mining*, pages 249–271. American Association for Artificial Intelligence.

[Klösgen, 1998] Klösgen, W. (1998). Deviation and association patterns for subgroup mining in temporal, spatial, and textual data bases. In *Rough sets and current trends in computing*, pages 1–18. Springer.

[Knublauch et al., 2004] Knublauch, H., Fergerson, R. W., Noy, N. F., and Musen, M. A. (2004). The protégé owl plugin: An open development environment for semantic web applications. In *The Semantic Web–ISWC 2004*, pages 229–243. Springer.

[Kodratoff, 2001] Kodratoff, Y. (2001). Rating the interest of rules induced from data and within texts. In *Proceedings of the 12th International Workshop on Database and Expert Systems Applications*, DEXA '01, pages 265–, Washington, DC, USA. IEEE Computer Society.

[Kontopoulos et al., 2013] Kontopoulos, E., Berberidis, C., Dergiades, T., and Bassiliades, N. (2013). Ontology-based sentiment analysis of twitter posts. *Expert systems with applications*, 40(10):4065–4074.

[Kontos et al., 2002] Kontos, J., Elmaoglou, A., and Malagardi, I. (2002). ARISTA causal knowledge discovery from texts. In Lange, S., Satoh, K., and Smith, C. H., editors, *Discovery Science, 5th International Conference, DS 2002, Lübeck, Germany, November 24-26, 2002, Proceedings*, volume 2534 of *Lecture Notes in Computer Science*, pages 348–355. Springer.

[Kosala and Blockeel, 2000] Kosala, R. and Blockeel, H. (2000). Web mining research: A survey. *ACM Sigkdd Explorations Newsletter*, 2(1):1–15.

[Kostoff et al., 2001] Kostoff, R. N., del Rio, J. A., Humenik, J. A., Garcia, E. O., and Ramirez, A. M. (2001). Citation mining: Integrating text mining and bibliometrics for research user profiling. *Journal of the American Society for Information Science and Technology*, 52(13):1148–1156.

[Kraft and Buell, 1993] Kraft, D. and Buell, D. (1993). Fuzzy sets and generalized boolean retrieval systems. In *D. Dubois and H. Prade (Eds.), Readings in Fuzzy Sets for Intelligent Systems*, San Mateo. CA: Morgan Kaufmann Publishers.

[Krallinger et al., 2008] Krallinger, M., Valencia, A., and Hirschman, L. (2008). Linking genes to literature: text mining, information extraction, and retrieval applications for biology. *Genome Biol*, 9(Suppl 2):S8.

[Kroeze et al., 2003] Kroeze, J. H., Matthee, M. C., and Bothma, T. J. D. (2003). Differentiating data- and text-mining terminology. In *Proceedings of the 2003*

Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on Enablement Through Technology, SAICSIT '03, pages 93–101, Republic of South Africa. South African Institute for Computer Scientists and Information Technologists.

[Kruse et al., 1999] Kruse, R., Borgelt, C., and Nauck, D. (1999). Fuzzy data analysis: challenges and perspectives. In *Fuzzy Systems Conference Proceedings, 1999. FUZZ-IEEE'99. 1999 IEEE International*, volume 3, pages 1211–1216. IEEE.

[Kumar et al., 2012] Kumar, S., Kathuria, M., Gupta, A. K., and Rani, M. (2012). Fuzzy clustering of web documents using equivalence relations and fuzzy hierarchical clustering. In *Software Engineering (CONSEG), 2012 CSI Sixth International Conference on*, pages 1–5. IEEE.

[Larsen and Aone, 1999] Larsen, B. and Aone, C. (1999). Fast and effective text mining using linear-time document clustering. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 16–22. ACM.

[Leahey and Jackson, 1998] Leahey, T. H. and Jackson, R. (1998). *Aprendizaje y cognición*. Madrid. Prentice Hall.

[Lehmann and Voelker, 2014] Lehmann, J. and Voelker, J. (2014). An introduction to ontology learning. In Lehmann, J. and Voelker, J., editors, *Perspectives on Ontology Learning*, pages ix–xvi. AKA / IOS Press.

[Lenat, 1995] Lenat, D. B. (1995). Cyc: A large-scale investment in knowledge infrastructure. *Commun. ACM*, 38(11):33–38.

[Lent et al., 1997] Lent, B., Agrawal, R., and Srikant, R. (1997). Discovering trends in text databases. In Heckerman, D., Mannila, H., Pregibon, D., and Uthurusamy, R., editors, *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining, KDD*, pages 227–230. AAAI Press.

[Lewis, 1992] Lewis, D. D. (1992). An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 37–50. ACM.

[Li and Wu, 2010] Li, N. and Wu, D. D. (2010). Using text mining and sentiment analysis for online forums hotspot detection and forecast. *Decision Support Systems*, 48(2):354–368.

[Li et al., 2008] Li, X., Wang, Y.-Y., and Acero, A. (2008). Learning query intent from regularized click graphs. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 339–346. ACM.

[Lima et al., 2015] Lima, F., de Oliveira, H. T. A., and do Nascimento Salvador, L. (2015). An unsupervised method for ontology population from textual sources on the web. In *Proceedings of the annual conference on Brazilian Symposium on Information Systems: Information Systems: A Computer Socio-Technical Perspective-Volume 1*, page 23. Brazilian Computer Society.

[Lin et al., 2011] Lin, K.-C., Liao, I.-E., and Chen, Z.-S. (2011). An improved frequent pattern growth method for mining association rules. *Expert Systems with Applications*, 38(5):5154–5161.

[Liu, 2007] Liu, B. (2007). *Web data mining: exploring hyperlinks, contents, and usage data*. Springer Science & Business Media.

[Liu and Wong, 2009] Liu, W. and Wong, W. (2009). Web service clustering using text mining techniques. *International Journal of Agent-Oriented Software Engineering*, 3(1):6–26.

[Liu et al., 2010] Liu, Z., Geng, X., and Deng, G. (2010). Nwfnn: A clustering algorithm based on fuzzy neural network with its application in text mining. *JOURNAL OF INFORMATION &COMPUTATIONAL SCIENCE*, 7(1):41–47.

[Llop, 2008] Llop, R. R. (2008). *Especificación owl de una ontología para teleeducación en la web semántica*. PhD thesis, Universidad Politécnica de Valencia.

[Llopart and Huber, 1986] Llopart, M. M. and Huber, W. (1986). *Lingüística Computacional*. Editorial Teide Barcelona.

[Lopes et al., 2007] Lopes, A., Pinho, R., Paulovich, F. V., and Minghim, R. (2007). Visual text mining using association rules. *Computers & Graphics*, 31(3):316–326.

[Lu et al., 2004] Lu, W.-H., Chien, L.-F., and Lee, H.-J. (2004). Anchor text mining for translation of web queries: A transitive translation approach. *ACM Transactions on Information Systems (TOIS)*, 22(2):242–269.

[Lyons, 1977] Lyons, J. (1977). *Semantics*. Cambridge University Press, London.

[MacCartney and Manning, 2009] MacCartney, B. and Manning, C. D. (2009). An extended model of natural logic. In *Proceedings of the Eighth International Conference on Computational Semantics*, IWCS-8 '09, pages 140–156, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Mack and Hehenberger, 2002] Mack, R. and Hehenberger, M. (2002). Text-based knowledge discovery: search and mining of life-sciences documents. *DDT Vol. 7, No. 11 (Suppl.)*.

[Madria et al., 1999] Madria, S. K., Bhowmick, S. S., Ng, W.-K., and Lim, E.-P. (1999). Research issues in web data mining. In *DataWarehousing and Knowledge Discovery*, pages 303–312. Springer.

[Maedche and Staab, 2000a] Maedche, A. and Staab, S. (2000a). Mining ontologies from text. In *Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling and Management*, pages 189–202. Springer-Verlag.

[Maedche and Staab, 2000b] Maedche, A. and Staab, S. (2000b). Semi-automatic engineering of ontologies from text. In *Proceedings of the 12th international conference on software engineering and knowledge engineering*, pages 231–239.

[Maedche and Staab, 2001] Maedche, A. and Staab, S. (2001). Ontology learning for the semantic web. *IEEE Intelligent systems*, 16(2):72–79.

[Mahgoub et al., 2008] Mahgoub, H., Rösner, D., Ismail, N., and Torkey, F. (2008). A text mining technique using association rules extraction. *International Journal of Computational Intelligence*, 4(1):21–28.

[Manco et al., 2008] Manco, G., Baglioni, M., Giannotti, F., Kuijpers, B., Raffaetà, A., and Renso, C. (2008). Querying and reasoning for spatiotemporal data mining. In *Mobility, Data Mining and Privacy - Geographic Knowledge Discovery*, pages 335–374.

[Mansilla et al., 2014] Mansilla, P. S., Costaguta, R., and Missio, D. (2014). Aplicación de algoritmos de clasificación de minería de textos para el reconocimiento de habilidades de e-tutores colaborativos. *Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial*, 17(53):57–67.

[Martin, 2000] Martin, G. R. (2000). *A storm of swords*. Bantam Dell Publishing Group.

[Martín-Bautista et al., 2008] Martín-Bautista, M. J., Martínez-Folgoso, S., and Vila, M. (2008). A new semantic representation for short texts. In *Data warehousing and knowledge discovery*, pages 347–356. Springer.

[Martín-Bautista et al., 2003] Martín-Bautista, M., Sánchez, D., Serrano, J., and Vila, M. (2003). Mining web documents to find additional query terms using fuzzy association rules. *Elsevier Science)*.

[Martín-Bautista, 2000] Martín-Bautista, M. J. (2000). *Soft Computing Models for Information Retrieval*. Universidad de Granada, PhD Thesis, Universidad de Granada, Spain (in Spanish).

[Medelyan et al., 2009] Medelyan, O., Milne, D., Legg, C., and Witten, I. H. (2009). Mining meaning from wikipedia. *International Journal of Human-Computer Studies*, 67(9):716–754.

[Mei and Zhai, 2005] Mei, Q. and Zhai, C. (2005). Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 198–207. ACM.

[Michalski, 2003] Michalski, R. S. (2003). Knowledge mining: A proposed new direction. Invited talk at the Sanken Symposium on Data Mining and Semantic Web, Osaka University.

[Milne and Witten, 2013] Milne, D. and Witten, I. H. (2013). An open-source toolkit for mining wikipedia. *Artificial Intelligence*, 194:222–239.

[Minsky, 1975] Minsky, M. (1975). A framework for representing knowledge. *Winston, Patrick ed.; The Psicology of computer vision; Nueva York: Mc Graw-Hill.*

[Mishne, 2003] Mishne, G. (2003). Source code retrieval using conceptual graphs. master of logic thesis. *Institute for Logic, Language and Computation, University of Amsterdam.*

[Mizoguchi et al., 2006] Mizoguchi, R., Shi, Z., and Giunchiglia, F. (2006). The semantic web-aswc 2006. In *First Asian Semantic Web Conference*, number 4183.

[Molina and Pla, 2002] Molina, A. and Pla, F. (2002). Shallow parsing using specialized hmms. *The Journal of Machine Learning Research*, 2:595–613.

[Montes-y Gómez et al., 2002] Montes-y Gómez, M., Gelbukh, A., and López-López, A. (2002). Text mining at detail level using conceptual graphs. *Lecture Notes in Artificial Intelligence 2393.*

[Mooney and Bunescu, 2005] Mooney, R. J. and Bunescu, R. (2005). Mining knowledge from text using information extraction. *SIGKDD Explor. Newsl.*, 7(1):3–10.

[Moreno Ortiz, 2000] Moreno Ortiz, A. (2000). Diseño e implementación de un lexicón computacional para lexicografía y traducción automática. In *Estudios de lingüística del español*, volume 9.

[Mulhem et al., 2001] Mulhem, P., Leow, W. K., and Lee, Y. K. (2001). Fuzzy conceptual graphs for matching images of natural scenes. In *IJCAI*, pages 1397–1404.

[Munezero et al., 2014] Munezero, M., Montero, C. S., Kakkonen, T., Sutinen, E., Mozgovoy, M., and Klyuev, V. (2014). Automatic detection of antisocial behaviour in texts. *Informatica (Slovenia)*, 38(1).

[Nahm and Mooney, 2000] Nahm, U. Y. and Mooney, R. J. (2000). Using information extraction to aid the discovery of prediction rules from texts. In *Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000) Workshop on Text Mining*, pages 51–58, Boston.

[Nahm and Mooney, 2001] Nahm, U. Y. and Mooney, R. J. (2001). Mining soft-matching rules from textual data. In *IJCAI 2001*, pages 979–986.

[Nahm and Mooney, 2002] Nahm, U. Y. and Mooney, R. J. (2002). Text mining with information extraction. aaai. In *Spring Symposium on Mining Answers from Texts and Knowledge Bases*.

[Nastase et al., 2009] Nastase, V., Milne, D., and Filippova, K. (2009). Summarizing with encyclopedic knowledge. In *2nd Text Analysis Conference, National Institute of Standards and Technology, Gaithersburg, MA*.

[Nasukawa and Nagano, 2001] Nasukawa, T. and Nagano, T. (2001). Text analysis and knowledge mining system. *IBM systems journal*, 40(4):967–984.

[Nasukawa and Yi, 2003] Nasukawa, T. and Yi, J. (2003). Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd international conference on Knowledge capture*, pages 70–77. ACM.

[Navigli et al., 2003] Navigli, R., Velardi, P., and Gangemi, A. (2003). Ontology learning and its application to automated terminology translation. *Intelligent Systems, IEEE*, 18(1):22–31.

[Neto et al., 2000] Neto, J. L., Santos, A. D., Kaestner, C. A., Alexandre, N., Santos, D., et al. (2000). Document clustering and text summarization. In *Proceedings of the 4th International Conference Practical Applications of Knowledge Discovery and Data Mining (PADD-2000)*, pages 41–55. The Practical Application Company.

[Nguyen and Kan, 2007] Nguyen, T. D. and Kan, M.-Y. (2007). Keyphrase extraction in scientific publications. In *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, pages 317–326. Springer.

[Noy, 2005] Noy, N. (2005). Ontology mapping and alignment. In *Fifth International Workshop on Ontology Matching collocated with the 9th International Semantic Web Conference ISWC-2010, Shangai, China*.

[Noy and McGuinness, 2001] Noy, N. F. and McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology. Technical report, Manchester University.

[Noy and Musen, 2000] Noy, N. F. and Musen, M. A. (2000). Algorithm and tool for automated ontology merging and alignment. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00). Available as SMI technical report SMI-2000-0831*.

[O'Connor et al., 2010] O'Connor, B., Balasubramanyan, R., Routledge, B. R., and Smith, N. A. (2010). From tweets to polls: Linking text sentiment to public opinion time series. *ICWSM*, 11:122–129.

[of Manchester, 2011] of Manchester, T. U. (2011). Owl patch. Online; last access 07-September-2015.

[Orallo, 2002] Orallo, J. H. (2002). Transparencias del seminario "exploración de datos masivos ('data mining')".

[Pak and Paroubek, 2010] Pak, A. and Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In *LREC*.

[Pang and Lee, 2008] Pang, B. and Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

[Paralic, 2001] Paralic, J. (2001). Knowledge discovery in databases. In *2nd Internal Scientific Conference of the Faculty of Electrical Enginering and Informatics, TU Kosice, pp. 33-34*.

[Paralic and Bednar, 2002] Paralic, J. and Bednar, P. (2002). Knowledge discovery in texts supporting e-democracy. In *6th IEEE International Conference on Intelligent Engineering Systems, INES 2002, pp. 327-332*, Opatija, Croatia.

[Paralic and Bednar, 2003] Paralic, J. and Bednar, P. (2003). Text mining for document annotation and ontology support. *Intelligent Systems at the Service of Mankind, Ubooks*, pages 237–248.

[Patil and Dongre, 2015] Patil, D. B. and Dongre, Y. V. (2015). A fuzzy approach for text mining. *IJ Mathematical Sciences and Computing*, 4:34–43.

[Peng et al., 2003] Peng, F., Schuurmans, D., Keselj, V., and Wang, S. (2003). Automated authorship attribution with character level language models. In *10th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2003)*, pages 267–274.

[Pinto and Martins, 2001] Pinto, H. S. and Martins, J. P. (2001). A methodology for ontology integration. In *Proceedings of the 1st international conference on Knowledge capture*, pages 131–138. ACM.

[Polovina, 2008] Polovina, S. (2008). Conceptual graphs: An overview, software and issues.

[Prabha et al., 2013] Prabha, K., Vivekanandan, K., and Sukumaran, S. (2013). Efficient keyword based document clustering using fuzzy c–means algorithm. *Data Mining and Knowledge Engineering*, 5(12):460.

[Prada, 1996] Prada, J. J. (1996). Un algoritmo para la extracción de rasgos morfológicos a partir de descriptores.

[Priest et al., 2004] Priest, G., Beall, J. C., and Armour-Garb, B. P. (2004). *The law of non-contradiction: New philosophical essays*. Oxford University Press.

[Puente et al., 2015] Puente, C., Sobrino, A., Garrido, E., and Olivas, J. A. (2015). Summarizing information by means of causal sentences through causal questions. In *10th International Conference on Soft Computing Models in Industrial and Environmental Applications, SOCO 2015, Burgos, Spain, June 2015*, pages 353–363.

[Puente et al., 2012] Puente, C., Sobrino, A., and Olivas, J. A. (2012). Retrieving crisp and imperfect causal sentences in texts: From single causal sentences to mechanisms. In [Seising and Sanz, 2012], pages 175–194.

[Puente et al., 2010] Puente, C., Sobrino, A., Olivas, J. A., and Merlo, R. (2010). Extraction, analysis and representation of imperfect conditional and causal sentences by means of a semi-automatic process. In *FUZZ-IEEE 2010, IEEE International Conference on Fuzzy Systems, Barcelona, Spain, 18-23 July, 2010, Proceedings*, pages 1–8.

[Qiu, 2007] Qiu, D. (2007). A note on trillas' chc models. *Artificial Intelligence*, 171:239–254.

[Quillian, 1968] Quillian, M. (1968). Semantic memory. In *Semantic Information Processing, Ed. Minsky. MIT Press: Cambridge, MA. Pp 216-270.*

[Rajman and Besançon, 1998] Rajman and Besançon (1998). Text mining — knowledge extraction from unstructured textual data. In *6th Conference of International Federation of Classification Societies (IFCS-98), Rome.*

[Rajman and Besançon, 1997] Rajman, M. and Besançon, R. (1997). Text mining: Natural language techniques and text mining applications. In *Proceedings of the seventh IFIP 2.6 Working Conference on Database Semantics (DS-7), Chapam & Hall IFIP Proceedings serie, (1997) Oct 7-10.*

[Ramshaw and Marcus, 1995] Ramshaw, L. A. and Marcus, M. P. (1995). Text chunking using transformation-based learning. pages 82–94.

[Rios-Alvarado et al., 2015] Rios-Alvarado, A. B., Lopez-Arevalo, I., Tello-Leal, E., and Sosa-Sosa, V. J. (2015). An approach for learning expressive ontologies in medical domain. *Journal of medical systems*, 39(8):1–15.

[Rodrigues and Sacks, 2004] Rodrigues, M. M. and Sacks, L. (2004). A scalable hierarchical fuzzy clustering algorithm for text mining. In *Proceedings of the 5th international conference on recent advances in soft computing*, pages 269–274.

[Rodríguez, 1999] Rodríguez, H. (1999). Tutorial de extracción y recuperación de información. sepln 99.

[Rojas and Villegas, 2012] Rojas, W. C. and Villegas, C. M. (2012). Una revisión comparativa de esquemas de visualización multidimensional para técnicas de minería de datos. *Infonor-chile*. Online; last access 12-September-2015.

[Roy and Toshniwal, 2010] Roy, R. S. and Toshniwal, D. (2010). Fuzzy clustering of text documents using naïve bayesian concept. In *Recent Trends in Information, Telecommunication and Computing (ITC), 2010 International Conference on*, pages 55–59. IEEE.

[Russell, 1998] Russell, L. (1998). Deductive data mining: Uncertainty measures for banding the search space. In Borgida, A., Chaudhri, V. K., and Staudt, M., editors, *KRDB*, volume 10 of *CEUR Workshop Proceedings*, pages 15.1–15.5. CEUR-WS.org.

[S. Iiritano and Rullo, 2003] S. Iiritano, M. R. and Rullo, P. (2003). Preprocessing method and similarity measures in clustering-based text mining: a preliminary study. *Data Mining IV*, pages 73–79.

[Sahami and Heilman, 2006] Sahami, M. and Heilman, T. D. (2006). A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the 15th international conference on World Wide Web*, pages 377–386. AcM.

[Sakurai and Suyama, 2005] Sakurai, S. and Suyama, A. (2005). An e-mail analysis method based on text mining techniques. *Applied Soft Computing*, 6(1):62–71.

[Sánchez et al., 2008] Sánchez, D., Martín-Bautista, M. J., Blanco, I., and Justicia de la Torre, C. (2008). Text knowledge mining: an alternative to text data mining. In *Data Mining Workshops, 2008. ICDMW'08. IEEE International Conference on*, pages 664–672. IEEE.

[Sánchez et al., 2009] Sánchez, D., Vila, M., Cerda, L., and Serrano, J.-M. (2009). Association rules applied to credit card fraud detection. *Expert Systems with Applications*, 36(2):3630–3640.

[Santana Mansilla et al., 2013] Santana Mansilla, P., Costaguta, R., and Missio, D. (2013). Construcción de clasificadores automáticos de habilidades de e-tutores de aprendizaje colaborativo. In *XVIII Congreso Argentino de Ciencias de la Computación*.

[Saxena et al., 2016] Saxena, N., Gupta, B. K., Pandey, H., and Singh, P. K. (2016). Web mining for personalization: A survey in the fuzzy framework. *Asian Journal of Computer and Information Systems*, 4(1).

[Schank, 1972] Schank, R. C. (1972). Conceptual dependency: A theory of natural language understanding. *Cognitive psychology*, 3(4):552–631.

[Schorlemmer and Kalfoglou, 2005] Schorlemmer, W. M. and Kalfoglou, Y. (2005). Progressive ontology alignment for meaning coordination: an information-theoretic foundation. In Dignum, F., Dignum, V., Koenig, S., Kraus, S., Singh, M. P., and Wooldridge, M., editors, *AAMAS*, pages 737–744. ACM.

[Scott and Matwin, 1999] Scott, S. and Matwin, S. (1999). Feature engineering for text classification. In *ICML*, volume 99, pages 379–388.

[Seising and Sanz, 2012] Seising, R. and Sanz, V., editors (2012). *Soft Computing in Humanities and Social Sciences*, volume 273 of *Studies in Fuzziness and Soft Computing*. Springer.

[Sha and Pereira, 2003] Sha, F. and Pereira, F. (2003). Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 134–141. Association for Computational Linguistics.

[Shack, 2013] Shack, H. (2013). *Semantic Web Technologies*. Open Hasso Plattner Institut. Online; last access 10-September-2015.

[Shen, 1992] Shen, W. (1992). Discovering regularities from knowledge bases. *Int. J. Intell. Syst.*, 7(7):623–635.

[Silvestri, 2010] Silvestri, F. (2010). Mining query logs: Turning search usage data into knowledge. *Foundations and Trends in Information Retrieval*, 4(1—2):1–174.

[Simoudis et al., 1994] Simoudis, E., Livezey, B., and Kerber, R. (1994). Integrating inductive and deductive reasoning for database mining. In *Knowledge Discovery in Databases: Papers from the 1994 AAAI Workshop, Seattle, Washington, July 1994. Technical Report WS-94-03*, pages 37–48.

[Sirin et al., 2007] Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., and Katz, Y. (2007). Pellet: A practical owl-dl reasoner. *Web Semant.*, 5(2):51–53.

[Smrž et al., 2007] Smrž, P., Paralič, J., Smatana, P., and Furdík, K. (2007). Text mining services for trialogical learning. In *Proc. of the 6th annual conference Znalosti*, pages 97–108.

[Sobrino et al., 2014] Sobrino, A., Puente, C., and Olivas, J. A. (2014). Extracting answers from causal mechanisms in a medical document. *Neurocomputing*, 135:53–60.

[Song and Shepperd, 2006] Song, Q. and Shepperd, M. (2006). Mining web browsing patterns for e-commerce. *Computers in Industry*, 57(7):622–630.

[Sowa and Way, 1986] Sowa and Way (1986). Implementing a semantic interpreter using conceptual graphs. *IBM Journal of Research and Development 30:1*.

[Spasic et al., 2005] Spasic, I., Ananiadou, S., McNaught, J., and Kumar, A. (2005). Text mining and ontologies in biomedicine: making sense of raw text. *Briefings in bioinformatics*, 6(3):239–251.

[Stavrianou et al., 2007] Stavrianou, A., Andritsos, P., and Nicoloyannis, N. (2007). Overview and semantic issues of text mining. *ACM Sigmod Record*, 36(3):23–34.

[Stevens et al., 2010] Stevens, R., Rector, A., and Hull, D. (2010). What is an ontology? *Ontogenesis*.

[Svátek et al., 2004] Svátek, V., Labskỳ, M., and Vacura, M. (2004). Knowledge modelling for deductive web mining. In *Engineering Knowledge in the Age of the Semantic Web*, pages 337–353. Springer.

[Swanson and Smalheiser, 1999] Swanson, D. R. and Smalheiser, N. R. (1999). Implicit text linkages between medline records: Using arrowsmith as an aid to scientific discovery. *Library trends*, 48(1):48–59.

[Swartout et al., 1996] Swartout, B., Patil, R., Knight, K., and Russ, T. (1996). Towards distributed use of large-scale ontologies. In *Proceedings of the 10th. Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Canada.

[Sánchez Dueñas and Valverde Andreu, 1989] Sánchez Dueñas, G. and Valverde Andreu, J. (1989). *Compiladores e Intérpretes. Un enfoque pragmático*. Ediciones Díaz de Santos, S.A., Madrid.

[Takeda et al., 2000] Takeda, M., Fukuda, and Nanri, I. (2000). Mining from literary texts: Pattern discovery and similarity computation. *Progress in Discovery Science 2002: 518-531*.

[Tan, 1999] Tan, A.-H. (1999). Text mining: Promises and challenges. In *Pacic Asia Conf on Knowledge Discovery and Data Mining PAKDD'99 workshop on Knowledge Discovery from Advanced Databases*, pages 65–70.

[Tang et al., 2014] Tang, J., Chang, Y., and Liu, H. (2014). Mining social media with social theories: A survey. *ACM SIGKDD Explorations Newsletter*, 15(2):20–29.

[Tang and Liu, 2010] Tang, L. and Liu, H. (2010). Community detection and mining in social media. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 2(1):1–137.

[TheBritishLibrary, 2007] TheBritishLibrary (2007). The british library's collections on flickr. Online; last access 20-March-2017.

[Thelwall, 2000] Thelwall, M. (2000). Web impact factors and search engine coverage. *Journal of documentation*, 56(2):185–189.

[Thomas et al., 2011] Thomas, J., McNaught, J., and Ananiadou, S. (2011). Applications of text mining within systematic reviews. *Research Synthesis Methods*, 2(1):1–14.

[Thorleuchter et al., 2010] Thorleuchter, D., den Poel, D. V., and Prinzie, A. (2010). Mining ideas from textual information. *Expert Systems with Applications*, 37(10):7182–7188.

[Thorleuchter and Van Den Poel, 2012] Thorleuchter, D. and Van Den Poel, D. (2012). Predicting e-commerce company success by mining the text of its publicly-accessible website. *Expert Systems with Applications*, 39(17):13026–13034.

[Thuraisingham, 1999] Thuraisingham, B. (1999). *Data Mining: Technologies, Techniques, Tools, and Trends*. CRC Press.

[Tian et al., 2009] Tian, J., Gao, M., and Sun, Y. (2009). Study on web classification mining method based on fuzzy neural network. In *Automation and Logistics, 2009. ICAL'09. IEEE International Conference on*, pages 1781–1785. IEEE.

[Tjong Kim Sang and Buchholz, 2000] Tjong Kim Sang, E. F. and Buchholz, S. (2000). Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*, pages 127–132. Association for Computational Linguistics.

[Tortosa, 2013] Tortosa, M. M. (2013). *Modelos de representación de arquetipos en sistemas de información sanitarios*. PhD thesis, PhD Thesis, Universidad de Murcia.

[Trillas et al., 1999] Trillas, E., Alsina, C., and Jacas, J. (1999). On contradiction in fuzzy logic. *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, 3(4):197–199.

[Trillas et al., 2002] Trillas, E., Alsina, C., and Pradera, A. (2002). Searching for the roots of non-contradiction and excluded-middle. *International journal of general systems*, 31(5):499–513.

[Trillas et al., 2000] Trillas, E., Cubillo, S., and Castiñeira, E. (2000). On conjectures in orthocomplemented lattices. *Artif. Intell*, 117(2):255–275.

[Tse-Tung, 1937] Tse-Tung, M. (1937). Sobre la contradicción. *Pekín, Ediciones en lenguas extranjeras, Mao Tsetung. Obras*.

[Tseng et al., 2007] Tseng, Y.-H., Lin, C.-J., and Lin, Y.-I. (2007). Text mining techniques for patent analysis. *Information Processing & Management*, 43(5):1216–1247.

[Tsytsarau and Palpanas, 2011] Tsytsarau, M. and Palpanas, T. (2011). Towards a framework for detecting and managing opinion contradictions. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, Vancouver, BC, Canada, December 11, 2011*, pages 1219–1222.

[Turney, 2000] Turney, P. D. (2000). Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4):303–336.

[University, 2015] University, P. S. (2015). Protégé Standford University). Online; last access 19-October-2015.

[University, 2006] University, S. (2006). Pizza.owl Standford University). Online; last access 19-October-2015.

[Van Eck and Waltman, 2011] Van Eck, N. J. and Waltman, L. (2011). Text mining and visualization using vosviewer. *ISSI Newsletter*, 7(3):50–54.

[Vossen, 1998] Vossen, P. (1998). Eurowordnet: A multilingual database with lexical semantic networks. kluwer academic press. *Computers and the Humanities, 32*, pages 2–3.

[w3c, 2001] w3c (2001). Daml+oil (march 2001) reference description. Online; last access 31-August-2015.

[w3c, 2004] w3c (2004). Owl web ontology language. Online; last access 20-August-2015.

[w3c, 2012a] w3c (2012a). Owl 2 web ontology language. Online; last access 06-September-2015.

[w3c, 2012b] w3c (2012b). Structural specification and functional-style syntax (second edition). Online; last access 07-September-2015.

[Wahiba and Ahmed, 2016] Wahiba, B. A. and Ahmed, B. E. F. (2016). New fuzzy decision tree model for text classification. In *The 1st International Conference on Advanced Intelligent System and Informatics (AISI2015), November 28-30, 2015, Beni Suef, Egypt*, pages 309–320. Springer.

[Wang and Wang, 2005] Wang, Y. and Wang, Z.-o. (2005). Text categorization rule extraction based on fuzzy decision tree. In *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, volume 4, pages 2122–2127. IEEE.

[Wason, 2006] Wason, T. D. (2006). Dr. Tom's taxonomy guide: Description, use and selections. *IMS Global Learning Consortium, Inc., Version*, 1:15.

[Weiss et al., 2010] Weiss, S. M., Indurkhya, N., and Zhang, T. (2010). *Fundamentals of predictive text mining*, volume 41. Springer.

[Weiss et al., 2000] Weiss, S. M., White, B. F., Apte, C. V., and Damerau, F. J. (2000). Lightweight document matching for help-desk applications. *Intelligent Systems and their Applications, IEEE*, 15(2):57–61.

[Wells, 2005] Wells, H. G. (2005). *La guerra de los mundos*, volume 227. Edaf.

[Weng and Lee, 2011] Weng, J. and Lee, B.-S. (2011). Event detection in twitter. *ICWSM*, 11:401–408.

[Whitehead, 2011] Whitehead, A. N. (2011). *Science and the modern world*. Cambridge University Press.

[wikipedia, 2015] wikipedia (2015). Jerarquía de chomsky. Online; last access 20-August-2015.

[Winchkler, 1997] Winchkler, G. (1997). Artefacto e instrumento en un diccionario de arqueología de base textual. 2º Coloquio Latinoamericano de Analistas del Discurso. Buenos Aires-La Plata, 25 al 29 de agosto.

[Witte et al., 2010] Witte, R., Khamis, N., and Rilling, J. (2010). Flexible ontology population from text: The owlexporter. In *LREC*, volume 2010, pages 3845–3850.

[Wittgenstein and Schlick, 1979] Wittgenstein, L. and Schlick, M. (1979). *Wittgenstein and the Vienna Circle*. Basil Blackwell.

[Wong et al., 2000] Wong, P. C., Cowley, W., Foote, H., Jurrus, E., and Thomas, J. (2000). Visualizing sequential patterns for text mining. In *Information Visualization, 2000. InfoVis 2000. IEEE Symposium on*, pages 105–111.

[Wong et al., 1999] Wong, P. C., Whitney, P., and Thomas, J. (1999). Visualizing association rules for text mining. In *Information Visualization, 1999.(Info Vis' 99) Proceedings. 1999 IEEE Symposium on*, pages 120–123. IEEE.

[Wong et al., 2012] Wong, W., Liu, W., and Bennamoun, M. (2012). Ontology learning from text: A look back and into the future. *ACM Computing Surveys (CSUR)*, 44(4):20.

[WordNet, 2005] WordNet (2005). Wordnet (Princeton University).

[Wu et al., 2004] Wu, S.-T., Li, Y., Xu, Y., Pham, B., and Chen, P. (2004). Automatic pattern-taxonomy extraction for web mining. In *Web Intelligence, 2004. WI 2004. Proceedings. IEEE/WIC/ACM International Conference on*, pages 242–248. IEEE.

[Xu et al., 2002] Xu, F., Kurz, D., and Piskorski, J. (2002). Term extraction and mining of term relations from unrestricted texts in the financial domain. In *Proceedings of BIS 2002*.

[Yang et al., 1999] Yang, Y., Carbonell, J. G., Brown, R. D., Pierce, T., Archibald, B. T., and Liu, X. (1999). Learning approaches for detecting and tracking news events. *IEEE Intelligent Systems*, 14(4):32–43.

[Yang et al., 1998] Yang, Y., Pierce, T., and Carbonell, J. (1998). A study on retrospective and on-line event detection. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 28–36, Melbourne, AU.

[Zendesk, 2014] Zendesk (2014). On-demand and low-cost smart data management. Online; last access 12-October-2015.

[ZenTut, 2015] ZenTut (2015). The cross-industry standard process for data mining (crisp-dm). Online; last access 23-August-2015.

[Zhai et al., 2004] Zhai, C., Velivelli, A., and Yu, B. (2004). A cross-collection mixture model for comparative text mining. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 743–748. ACM.

[Zhao and Bhowmick, 2003] Zhao, Q. and Bhowmick, S. S. (2003). Association rule mining: A survey. *Nanyang Technological University, Singapore*.

[Zhao and Karypis, 2005] Zhao, Y. and Karypis, G. (2005). Topic-driven clustering for document datasets. In *SDM*, pages 358–369.

[Zhong et al., 2012] Zhong, N., Li, Y., and Wu, S.-T. (2012). Effective pattern discovery for text mining. *Knowledge and Data Engineering, IEEE Transactions on*, 24(1):30–44.

Nos reconocemos entre nosotros, y sólo entre nosotros, en una cierta irreductibilidad calificada de todos los demás que nos permite prescindir de la interpretación pública de nuestras acciones. Nuestras contradicciones deben ser consideradas como el signo de esa enfermedad del espíritu que puede pasar por nuestra más alta dignidad. Repetimos que creemos en la fuerza de la contradicción (Brau [Brau, 1972])

ugr Universidad de Granada

Nuevas Técnicas de Minería de Textos: Aplicaciones

memoria que presenta

María del Consuelo Justicia de la Torre

para optar al grado de

Doctor en Informática

2017

DIRECTORES

Daniel Sánchez Fernández María José Martín Bautista

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
E INTELIGENCIA ARTIFICIAL

- 2017 -