

BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN



EXTRACCIÓN AUTOMÁTICA DE RELACIONES
TAXONÓMICAS EN DOCUMENTOS NO ESTRUCTURADOS A
TRAVÉS DE UN ALGORITMO DE AGRUPAMIENTO

TESIS PRESENTADA PARA OBTENER EL TÍTULO DE:
LICENCIATURA EN INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN

Presenta:

Emmanuel Santos Rodríguez

Director de tesis:

Dra. Mireya Tovar Vidal

Asesor de Tesis:

Dr. José Alejandro Reyes Ortiz

Octubre 2021

Agradecimientos

Agradezco a mi familia infinitamente por el apoyo durante estos años. Especialmente, le agradezco a mi mamá todo su apoyo, comprensión y paciencia. También le agradezco a mis abuelos, Dolores y Simón, por todas sus enseñanzas y cariño.

Agradezco a la Doctora Mireya Tovar Vidal y al Doctor José Alejandro Reyes Ortiz por el apoyo, conocimiento y tiempo que me brindaron durante el desarrollo de este trabajo.

Agradecimientos especiales al *Consejo Nacional de Ciencia y Tecnología* (CONACYT) por el apoyo otorgado durante el desarrollo de este trabajo de investigación bajo el proyecto número 257357 de Ciencia Básica, registro 25547.

Resumen

Una ontología es una representación formal del conocimiento de un conjunto de conceptos en un dominio y las relaciones entre esos conceptos. Están compuestas por clases o conceptos, relaciones, funciones, axiomas e instancias. Las ontologías permiten razonar acerca de las propiedades de un dominio dado y describir al mismo. Asimismo, proveen un vocabulario compartido que sirve para modelar el dominio [34]. Debido a esto, la ingeniería ontológica se centra en las actividades relevantes al proceso de desarrollo de una ontología, el ciclo de vida de ésta, así como las herramientas y metodologías que permiten crear las ontologías. Además de las ontologías, existe otra forma de jerárquica de estructurar el conocimiento que se denominan taxonomías, que se representan como un árbol cuya raíz es un concepto universal y los hijos son conceptos más específicos. Los nodos representan los conceptos y las aristas la conexión con el concepto más general. Sin embargo, la mayoría de la información que poseemos actualmente no se encuentra en un formato estandarizado, por lo que no se puede obtener conocimiento directamente a partir de ésta. A causa de esto, se han creado formas de lidiar efectivamente con este problema. La extracción de información es la tarea de extraer, de forma automática, información de un texto no estructurado con el objetivo de darle una estructura que permita su uso en aplicaciones específicas. El agrupamiento o agrupamiento se define como la tarea de agrupar un conjunto de objetos en un grupo de tal forma que queden agrupados los objetos con características similares. Este tipo de análisis es valioso porque permiten ver la estructura de los datos y las relaciones entre estos, son útiles para utilizarse en conjunto con otras técnicas de IA y son especialmente útiles para el descubrimiento de conocimiento. Por otra parte, la creación automática de ontologías y taxonomías es de vital importancia debido a las extensas aplicaciones, basadas en

conocimiento, de éstas en campos como la farmacéutica donde ayuda a la minería de causalidad al categorizar las relaciones explícitas identificadas con una ontología de relación causal, enriquecen la minería en la web semántica, para obtener información en la minería de registros médicos, detección de fraudes, sistemas de gestión de conocimiento, entre otros. Asimismo, las ontologías son importantes puesto que nos permiten reducir la ambigüedad, ayudan a estructurar el conocimiento de un dominio, representarlo e incluso compartirlo y reutilizarlo de forma efectiva. En la actualidad existen distintas metodologías para la extracción de taxonomías como las basadas en patrones léxico-sintácticos, LSA, word embeddings, etc. En este proyecto se pretende implementar un sistema basado en un algoritmo de agrupamiento que permita la obtención automática de relaciones taxonómicas de un corpus de dominio o documentos no estructurados y permita construir la taxonomía.

Índice general

Agradecimientos	I
Resumen	II
Índice de figuras	VI
Índice de tablas	VII
Índice de algoritmos	IX
1. Introducción	2
1.1. Antecedentes	2
1.2. Objetivos	5
1.2.1. Objetivo General	5
1.2.2. Objetivos Específicos	6
1.3. Metodología	6
2. Estado del Arte	7
3. Marco teórico	13
3.1. Lenguaje	13
3.2. Procesamiento de Lenguaje Natural	15
3.2.1. Niveles de Procesamiento de Lenguaje Natural	15
3.2.2. Word embeddings	16
3.2.3. Aplicaciones	18
3.3. Aprendizaje automático	19
	IV

3.3.1.	Aprendizaje supervisado	20
3.3.2.	Aprendizaje no supervisado	20
3.3.3.	Aprendizaje semi-supervisado	20
3.3.4.	Aprendizaje por refuerzo	21
3.3.5.	Grupos de algoritmos de aprendizaje automático	21
3.4.	Minería de datos	24
3.4.1.	Fases de la minería de datos	25
3.4.2.	Análisis de agrupamiento	27
3.4.3.	Minería de texto	30
3.5.	Extracción de Información	32
3.5.1.	Fases de la extracción de información	34
3.5.2.	Extracción de información como un mapeo de texto-ontología	35
3.6.	Ingeniería ontológica	35
3.6.1.	Ontología	35
3.6.2.	Definición de ingeniería ontológica	37
3.6.3.	Aprendizaje de ontologías	38
3.6.4.	Aspectos	42
3.6.5.	Criterios	43
3.6.6.	Evaluación de ontologías	44
3.6.7.	Herramientas	45
4.	Propuesta de solución	47
4.1.	Sistema propuesto para la extracción automática de relaciones taxonómicas	47
4.1.1.	Extracción de relaciones de hiponimia	48
4.1.2.	Extracción de características	51
4.1.3.	Agrupamiento jerárquico	53
4.2.	Sistema propuesto para la clasificación de relaciones taxonómicas	58
4.3.	Sistema propuesto para la clasificación de relaciones de hiperonimia	59
4.3.1.	Características utilizadas	60
4.3.2.	Entrenamiento	60

5. Resultados	61
5.1. Medidas de evaluación	61
5.1.1. Exactitud	61
5.1.2. Precisión	62
5.1.3. Exhaustividad	62
5.1.4. F_1	62
5.1.5. Área bajo la curva ROC	62
5.2. Conjuntos de datos	62
5.3. Extracción y validación de relaciones taxonómicas	64
5.4. Agrupamiento jerárquico y representación	66
5.5. Clasificación y validación de relaciones de hiperonimia	70
6. Conclusiones	74
Bibliografía	77

Índice de figuras

3.1. Niveles de procesamiento de lenguaje natural [18].	17
3.2. Niveles de la minería de macrodatos [27].	26
3.3. Proceso general de la minería de datos [3].	27
3.4. Niveles de granularidad en el agrupamiento jerárquico [2, 3].	29
3.5. Arquitectura conceptual del aprendizaje de ontologías [32].	38
3.6. Capas del aprendizaje de ontologías [7, 32]	39
4.1. Vista general del sistema propuesto.	49
4.2. Ejemplo de frases sustantivas para una oración.	51
4.3. Esquema del modelo de clasificación de nuevas relaciones taxonómicas.	58
4.4. Vista general de la arquitectura propuesta.	59
5.1. Dendrograma para el dominio de IA.	68
5.2. Dendrograma para el dominio de SCORM.	69

Índice de tablas

3.1. Grupo de algoritmos de aprendizaje automático [48]	23
3.2. Algoritmos de aprendizaje automático y su uso en aprendizaje de ontologías [32].	40
3.3. Patrones léxico-sintácticos propuestos por Hearst para la extracción de relaciones de hiponimia [23].	42
4.1. Patrones obtenidos de [57]	50
4.2. Patrones, propuestos por Giovannetti et al. [21], obtenidos de [57] . .	51
5.1. Corpus de dominio	63
5.2. Elementos que conforman las ontologías	63
5.3. Relaciones totales obtenidas mediante el algoritmo 3.	64
5.4. Ejemplos de relaciones obtenidas para la ontología de IA.	64
5.5. Ejemplos de relaciones obtenidas para la ontología de SCORM. . . .	65
5.6. Relaciones depuradas obtenidas mediante el algoritmo 3.	65
5.7. Exactitud de las relaciones encontradas para cada dominio.	66
5.8. Dimensiones de las matrices de características durante la segunda fase del sistema.	67
5.9. Dimensiones de las matrices de características durante la tercera fase del sistema.	67
5.10. Reporte de clasificación conforme a las anotaciones del primer experto para la ontología de IA.	71
5.11. Reporte de clasificación conforme a las anotaciones del segundo experto para la ontología de IA.	71

5.12. Reporte de clasificación conforme a las anotaciones del tercer experto para la ontología de IA.	71
5.13. Reporte de clasificación conforme a las anotaciones del primer experto para la ontología de SCORM.	71
5.14. Reporte de clasificación conforme a las anotaciones del segundo experto para la ontología de SCORM.	72
5.15. Reporte de clasificación conforme a las anotaciones del tercer experto para la ontología de SCORM.	72
5.16. Exactitud y área debajo de la curva ROC para las predicciones para la ontología de SCORM conforme a las anotaciones de los expertos. .	72
5.17. Exactitud y área debajo de la curva ROC para las predicciones para la ontología de SCORM conforme a las anotaciones de los expertos. .	73

Índice de algoritmos

1.	Algoritmo de fusión aglomerativo genérico con criterio de fusión no especificado [2, 3]	29
2.	Algoritmo genérico de agrupamiento de arriba hacia abajo [2, 3] . . .	30
3.	Algoritmo para la extracción de relaciones hipónimo-hiperónimo . . .	55
4.	Algoritmo para la extracción de características a partir de las tuplas de hipónimo, hiperónimo y oración.	56
5.	Algoritmo para agrupamiento jerárquico y representación.	57
6.	Algoritmo general del primer sistema.	57

Capítulo 1

Introducción

Actualmente, poseemos una cantidad ingente de información y ésta no para de crecer a cada minuto. Pese a esto, es difícil aprovecharla puesto que no se encuentra estructurada en su mayoría. Debido a esto, es importante la creación de soluciones que permitan explotar el conocimiento contenido en dicha información de forma automática, ya que sería complicado hacerlo mediante agentes humanos. Con base en lo anterior, el objetivo de esta investigación es crear un modelo que permita extraer taxonomías automáticamente a partir de texto no estructurado y evaluar la taxonomía producida.

1.1. Antecedentes

En el trabajo realizado por Chavez y Tovar [9], que sirve como antecedente para esta investigación, se presenta una metodología para la extracción de relaciones taxonómicas. En este trabajo se hace una propuesta para la tarea 13 de SemEval 2016, en el que se extraen las relaciones de hiperonimia y sinonimia mediante patrones léxico-sintácticos y análisis formal en textos no estructurados. El interés de esta investigación se centra en la identificación de hiperónimos y la construcción de la taxonomía. A continuación, se presentan los trabajos que sirven de apoyo y como antecedente para este proyecto:

- Tovar et al. [60]. En este artículo se propone la evaluación automática de relaciones en ontologías de dominio restringido usando patrones léxico-sintácticos

y análisis de dependencias sintácticas para evaluar relaciones de tipo class-inclusion así como otras relaciones ontológicas incluidas. El método descrito utiliza la evaluación basada en corpus considerando el criterio de exactitud. Este método es capaz de proporcionar una medida de exactitud para cada ontología y dicha puntuación tiene un grado de confiabilidad obtenido a través de la comparación de la evaluación realizada por expertos y un baseline. Los resultados indican que este enfoque supera al baseline propuesto.

- Chavez y Tovar [9]. En este artículo se desarrolla una metodología para la extracción automática de relaciones mediante patrones y métodos de agrupamiento como el análisis formal de conceptos para las dos primeras subtarefas de la tarea 13 de SemEval 2016, llamada Evaluación de Extracción de Taxonomías. Este modelo consta de tres fases: la fase de preprocesamiento de la información (remover puntuación, símbolos especiales, convertir a minúsculas, etc), la de modelos propuestos (desarrollo e implementación de los modelos para la extracción de relaciones de hiperonimia y sinonimia) y la evaluación (medida mediante precisión, recall y validación manual hecha por expertos). Finalmente, gracias a dicha metodología es posible detectar y extraer relaciones taxonómicas de hiperonimia e hiponimia en textos no estructurados de forma automática.
- Tovar et al. [58]. Los autores presentan un método basado en patrones léxico-sintácticos (obtenidos de la subtarea 3 de la tarea 10 de SemEval 2017 así como de la literatura) para la validación de relaciones semánticas de sinonimia entre conceptos ontológicos; también se consideran los acrónimos como tipos de sinónimo. Este método se divide en cuatro etapas: preprocesamiento, creación de una lista de patrones léxico sintácticos, obtención de una lista de acrónimos de un corpus de dominio y la evaluación de las relaciones semánticas, que se divide en evaluación utilizando patrones léxico-sintácticos y utilizando acrónimos. La evaluación se realiza usando un subcorpus creado por el sistema que fue validado por tres expertos en el dominio. Los resultados muestran que el sistema provee resultados fiables que coinciden con el criterio de los expertos.

- Tovar et al. [59]. En este artículo se aborda un enfoque para la evaluación automática de relaciones en ontologías de dominio restringido que utiliza la evidencia encontrada en un corpus asociado al mismo dominio de la ontología para definir la validez de las relaciones encontradas. Este enfoque usa Análisis Semántico Latente (LSA) y tiene dos variantes para evaluar las relaciones semánticas y conceptos de la ontología objetivo (similitud de cosenos y Agrupamiento By Committees (CBC)). Los resultados indican que se obtienen resultados satisfactorios al aplicar LSA y similitud de cosenos, pero usando CBC no es posible encontrar todas las relaciones ontológicas en los conceptos agrupados.
- Tovar et al. [61]. En este artículo los autores proponen un enfoque que utiliza el análisis formal de conceptos para la identificación automática de relaciones en ontologías de dominio restringido usando la evidencia hallada en un corpus del mismo dominio para así determinar la validez de las relaciones ontológicas. Los conceptos formales se usan para evaluar las relaciones ontológicas de la ontología objetivo. Este enfoque considera dos formas de seleccionar los atributos para la construcción de la matriz de incidencia que utiliza el análisis formal de conceptos para obtener los conceptos formales. La diferencia entre estas variantes es el analizador de dependencias sintácticas utilizado (minipar tagger y Stanford tagger). Es importante resaltar que la evaluación de las relaciones ontológicas fue realizada por humanos mediante la medida de precisión (accuracy). De acuerdo con los resultados descritos por los autores, este enfoque obtiene una precisión muy alta para relaciones taxonómicas y una puntuación perfecta para las relaciones no taxonómicas.
- Tovar et al. [63]. En este artículo se aborda un método que utiliza patrones léxico-sintácticos y un corpus de referencia para la evaluación (a través de precisión) de relaciones taxonómicas en ontologías de dominio restringido. Se utilizan los patrones para evaluar relaciones taxonómicas en las que los conceptos son totalmente distintos y usa una técnica que se basa en la suposición de que para esas relaciones una está completamente contenida en la otra; gracias a esto, se pudieron evaluar las relaciones taxonómicas de dos ontologías de dominio, un de inteligencia artificial y otra de e-learning. La evaluación se

lleva acabo encontrando evidencia (mediante los patrones) en un corpus de referencia utilizando el criterio de exactitud. Además, se muestra que existe una alta relación en la frecuencia de ocurrencia de los patrones utilizados en el proceso de evaluación, independientemente del dominio ontológico. Finalmente, se evaluó este sistema, mediante expertos humanos, y consiguió una confiabilidad promedio de 80 %, aunque esto también refleja la calidad de la ontología utilizada.

- Tovar et al. [62]. El enfoque presentado en este trabajo se basa en utilizar la evidencia hallada en un corpus asociado al mismo dominio de la ontología para determinar las relaciones ontológicas, haciendo uso de análisis formal de conceptos, para la identificación automática de relaciones en ontologías de dominio restringido. Este enfoque utiliza dos variantes para construir la matriz de incidencia que usa el análisis formal de conceptos, cuya diferencia entre ambos radica en el analizador utilizado (minipar o Stanford). De acuerdo con el artículo, la variante que usa Stanford es más exacta que la que utiliza minipar; sin embargo, ésta última obtiene una buena precisión en los dos tipos de relaciones a evaluar, taxonómica y no taxonómicas, mientras que la versión de Stanford obtiene mejores resultados para relaciones no taxonómicas. Sin embargo, la versión que usa minipar es más rápida.

1.2. Objetivos

Nuestros objetivos planteados, se dividen en dos partes, el objetivo general que define el alcance de la investigación y los objetivos específicos que determinan las tareas a realizar.

1.2.1. Objetivo General

Desarrollar un modelo que permita la extracción de relaciones taxonómicas a partir de textos no estructurados usando un algoritmo de agrupamiento jerárquico.

1.2.2. Objetivos Específicos

1. Analizar los algoritmos de agrupamiento jerárquico para la construcción de una taxonomía..
2. Analizar técnicas para la extracción de relaciones hipónimo-hiperónimo..
3. Implementar un método para la extracción de relaciones hipónimo-hiperónimo a partir de un texto no estructurado.
4. Implementar un método para la construcción de una taxonomía a partir de las relaciones extraídas.
5. Evaluar el método propuesto.

1.3. Metodología

Con las intenciones de cumplir los objetivos y crear un sistema que cumpla las expectativas, se usará el modelo clásico para desarrollo de software, iniciando con la parte de obtención de requerimientos, análisis, especificación, diseño de la solución, implementación y pruebas.

Para tener un punto de referencia en el cual basar el trabajo, un rastreo exhaustivo del estado del arte nos permitirá conocer las propuestas de soluciones que se han planteado, cuáles fueron sus bases y los resultados obtenidos. Asimismo, una revisión general de conceptos básicos de fundamentos de Procesamiento de Lenguaje Natural, técnicas de extracción de relaciones semánticas y taxonómicas, así como características de estas y conceptos ontológicos serán necesarios. Durante la implementación, el uso de una metodología de programación adecuada para crear el sistema y los modelos, nos permitirán obtener resultados que funcionarán para someter a validación lo propuesto.

Capítulo 2

Estado del Arte

Para la revisión del estado del arte, se partirá desde el estudio de artículos relacionados con la ingeniería ontológica, aprendizaje de ontologías, aprendizaje automático y aplicaciones de éstas en diversos campos. A continuación, se presentan trabajos relacionados con las áreas mencionadas previamente que servirán para el desarrollo de esta investigación. Particularmente estamos interesados en conocer las diferentes metodologías para la extracción de relaciones taxonómicas y la creación automática de ontologías. Mahmoud et al. [33] proponen un método para aprendizaje de ontologías, basado en aprendizaje profundo, word embeddings y un algoritmo de agrupamiento avanzado (BIRCH). Los resultados muestran que esta metodología es apta para grandes volúmenes de texto debido a cómo se representan las palabras, así como al algoritmo de agrupamiento jerárquico utilizado pues éste permite la reducción del tiempo de procesamiento asociado al aprendizaje de la ontología. Finalmente, se concluye que, en relación tiempo/calidad, este método propuesto es aplicable para el aprendizaje de ontologías en grandes volúmenes de texto (big data).

Di Martino y Cantiello [15] presentan una técnica para crear ontologías que utiliza agrupamiento jerárquico (mediante el algoritmo de unión de vecinos). ésta se aplica a un corpus y a partir de él se genera un árbol binario en el que cada nodo tiene asociado un conjunto de términos derivados de los vectores característicos del documento. Dichos términos son conceptos candidatos para la construcción de una taxonomía de dominio, de la cual pueden extraerse las relaciones jerárquicas entre las clases de la ontología de dominio. Después, el árbol binario se convierte en una taxonomía

mediante el uso del concepto de iso-cohesión. Cabe destacar que el algoritmo descrito se encuentra disponible mediante un prototipo denominado OntoClust.

Novelli y de Oliveira [42] describen un método, simple y rápido, para la extracción automática de ontologías de documentos o colecciones de estos (sólo de texto o estructurados). Dicho método es independiente del tipo de documento y se basa en técnicas como análisis semántico latente para proponer los conceptos iniciales, agrupamiento jerárquico, WordNet y medidas de similitud para así obtener la correlación entre los conceptos. Los conceptos iniciales y sus relaciones se obtienen de los términos de los documentos; otros conceptos son creados a partir del análisis de los términos usando análisis semántico latente y agrupamiento. Por otra parte, las relaciones entre conceptos se generan del análisis de un tesauro u ontología (Wordnet). Los resultados son suficientemente satisfactorios para el entendimiento y manipulación de los conceptos del documento sin necesitar técnicas avanzadas, intervención de un experto o conocimiento del dominio de la ontología. Finalmente, las ontologías generadas mediante este método permiten representar adecuadamente los conceptos de los documentos. Es necesario señalar que este algoritmo no es supervisado y puede analizar texto multidisciplinario.

Sarencheh y Schiffauerova [50] plantean un framework que combina técnicas de minería de texto y minería web para crear una ontología más exhaustiva; éste se para el proceso de hallar los conceptos del análisis lingüístico para así extraer más relaciones taxonómicas y no taxonómicas. Dicho framework utiliza una versión modificada de TF-IDF para extraer los conceptos de una enciclopedia en línea. También usa etiquetado semántico de clases para conectar conceptos, categorizar atributos y etiquetarlos; asimismo, se utiliza la enciclopedia mencionada anteriormente para crear una estructura para el conocimiento de un dominio dado. Por otra parte, se usa etiquetado gramatical y un árbol de dependencias de las oraciones para obtener los conceptos y sus relaciones. De acuerdo con los autores, este método puede utilizarse para construir ontologías de documentos de texto plano que puede incluir conceptos complejos, como nanotecnología, o bien relaciones no taxonómicas.

Sarkar et al. [51] exponen un método que utiliza aprendizaje supervisado para combinar múltiples características, las cuales pueden ser obtenidas a través de etiquetas o un corpus asociado, con una máquina de soporte vectorial como clasificador.

La metodología propuesta consiste en utilizar el baseline de la tarea TExEval para saber si una cadena es subcadena de otra para así relacionar estas cadenas en el grafo de la taxonomía. Para extraer las relaciones entre pares de términos se utiliza la similitud de Jaccard par así construir el árbol que contendrá la taxonomía. Asimismo, utiliza word embeddings para hallar la similitud entre dos palabras mediante dos enfoques: min-max y promedio. Luego se utiliza descomposición en valores singulares para las matrices de vectores obtenidas mediante Word embeddings mencionados anteriormente. Después, se utiliza una máquina de soporte vectorial (SVM) para clasificar las relaciones. Finalmente se realiza la taxonomía construyendo, vorazmente, un árbol tomando los vértices con la puntuación más alta y añadiéndolos al árbol. La técnica descrita supera significativamente al baseline propuesto para TExEval en la medida F, pero no en la puntuación Fowlkes-Mallows debido a la estructura de la red.

Maynard et al. [36] discuten el uso de técnicas de extracción de información, incluyendo patrones léxico-sintácticos como técnicas para generar información ontológica a partir de texto no estructurado, ya sea para crear una nueva ontología o extender una existente con nuevas entidades. Estos patrones se refinan usando una herramienta de extracción de términos y restricciones semánticas derivadas de WordNet y VerbNet para evitar un problema relacionado con el uso de patrones de diseño ontológico (ODP). Utilizan ODP ya que, a diferencia de los patrones de Hearst, estos son más generales, pero son más ambiguos. Además, se reduce el problema de sobre generación mediante restricciones semánticas. De acuerdo con los autores, los patrones utilizados proveen una buena base, pero aún existe margen de mejora como incorporar combinaciones de patrones de Hearst con información colocacional derivada estadísticamente.

Cimiano et al. [11] presentan un enfoque basado en clasificación para aprendizaje de relaciones taxonómicas entre términos mediante múltiples y heterogéneas fuentes de información como corpus de texto, la Web, WordNet, etc. Se utiliza aprendizaje computacional para hallar una combinación óptima que, a su vez, represente estas fuentes como características de primer orden. Por otra parte, se utilizar diferentes clasificadores (Naive Bayes, C4.5, perceptrón y perceptrón multicapa) y distintas técnicas para lidiar con los conjuntos de datos desbalanceados (oversampling y un-

dersampling). Sin embargo, este enfoque utiliza aprendizaje supervisado, por lo que provee cierta independencia de dominio. Finalmente, es necesario resaltar que una correcta combinación de fuentes de información y técnicas de aprendizaje permite mejorar los resultados, a comparación de utilizar una solamente.

Pocostales [44] muestra un sistema semi-supervisado, llamado NUIG-UNL, que halla los hiperónimos candidatos para sustantivos representándolos como vectores. Esta propuesta asume que los hiperónimos pueden obtenerse sumando un vector de desplazamiento al hiperónimo correspondiente. El vector de desplazamiento se obtiene como el promedio del desplazamiento entre 200 pares de hipónimo-hiperónimo. Por otra parte, es necesario resaltar que este sistema alcanzó el segundo lugar para la conectividad y categorización en la construcción de ontologías en inglés.

Cleuziou y Moreno [12] diseñó un método semi-supervisado y auto-supervisado para obtener taxonomías léxicas usando la representación de vectores semánticos. Esta propuesta calcula las similitudes entre conceptos usando vectores semánticos para después construir una estructura preliminar gracias a un espacio pre-topológico definido. Finalmente, se usa un algoritmo genético para optimizar la calidad de las relaciones añadidas, así como la calidad de la estructura.

Doing-Harris et al. [16] proponen un sistema llamado SEAM (SEmi-Automated ontology Maintenance) para la extracción de información y relaciones para apoyar al proceso de desarrollo de una ontología mediante procesamiento de lenguaje natural. Este sistema combina diferentes tipos de texto y técnicas de extracción de relaciones. Usa semántica basada en corpus (a través de tf-idf y vectores de contexto) y patrones léxico-sintácticos para identificar sinónimos y grupos de jerarquías. SEAM se utiliza para recomendar términos clínicos, sinónimos y relaciones jerárquicas sin importar el dominio médico (estado mental, síndromes médicos sin explicación y resumen de ecocardiograma).

Singhal et al. [54] proponen un modelo de aprendizaje automático supervisado para la identificación automática de relaciones de tipo enfermedad-mutación, obtenidas a partir de texto biomédico con el objetivo de curación de bases de datos para la medicina de precisión. El modelo propuesto utiliza características estadísticas, de distancia y de sentimientos para obtener las relaciones enfermedad-mutación. Este enfoque demuestra ser útil para la extracción de relaciones enfermedad-mutación no

curadas en escenarios reales. Además, el enfoque obtiene mejores puntuaciones con respecto al estado del arte para medidas F.

Qian et al. [46] discuten un enfoque para determinar dinámicamente la extensión del árbol en métodos para extracción de la relación semántica basada en el núcleo del árbol. El método propuesto usa las dependencias para mantener los nodos y las cabezas de los nodos hijos en el camino para conectar dos entidades, mientras remueve el ruido del árbol sintáctico. Por otra parte, exploran la combinación de características para extraer relaciones mediante análisis unificado y el árbol semántico. La evaluación de este sistema se realizó en el corpus ACE RDC y los resultados demuestran que el trabajo propuesto supera a los demás basados en extensión del árbol mediante la combinación del núcleo del árbol con un núcleo lineal basado en características.

El trabajo de Shen y Huang [52] se centra en el uso de una red neuronal convolucional basada en atención, llamada Attention-CNN, para la extracción de relaciones semánticas. Este modelo usa embeddings a nivel de palabras, etiquetas gramaticales e información posicional y para realizar la clasificación utiliza WordNet y las palabras cerca de los nominales. Además, el mecanismo de atención usado (a nivel de palabras) permite determinar las parte más importantes en una oración con respecto a las entidades de interés. Este modelo fue probado con el conjunto de datos de la tarea 8, clasificación de relaciones, del SemEval-2010. Sin embargo, este modelo no contempla la direccionalidad de las relaciones. Se demostró que la red neuronal propuesta obtiene mejores resultados en términos de F1 respecto a otros enfoques basados en máquinas de soporte vectorial o redes neuronales recurrentes.

Xu et al. [68] desarrollan un modelo de red neuronal convolucional para el aprendizaje de representaciones más robustas a partir del camino de dependencia más corto, pues los modelos neuronales enfocados a la identificación de relaciones sufren de información irrelevante cuando los objetos y sujetos se encuentran a largas distancias (direccionalidad de las relaciones). Asimismo, proponen solucionar este problema mediante muestreo negativo. Esta propuesta fue evaluada usando el conjunto de datos para la tarea 8 del SemEval-2010 y se demostró que se obtienen buenos resultados en términos de F1; sin embargo, al usar el muestreo negativo, esta puntuación aumenta. Así, supera a métodos basados en máquinas de soporte vectorial,

redes recurrentes o redes convolucionales simples. Sin embargo, Attention-CNN lo supera por 0.05

Capítulo 3

Marco teórico

En este capítulo, se presentan y explican los conceptos teóricos usados en el presente trabajo de tesis.

Se inicia con la definición de lenguaje y lenguaje natural, después se habla del procesamiento de lenguaje natural y las distintas áreas que éste comprende. Luego, se habla del aprendizaje automático y se discuten los procesos de minería de datos y recuperación de información, poniendo especial énfasis en los procesos relacionados con las ontologías y taxonomías. Posteriormente, se trata la ingeniería ontológica así como las tareas fundamentales de la misma. Finalmente, se indican las herramientas usadas en el desarrollo de la propuesta de solución.

3.1. Lenguaje

Existen diversas definiciones de lenguaje, cada una con diferentes connotaciones según el autor y la época. A continuación se muestran algunas interpretaciones de este concepto, a partir de las cuales se derivaran los elementos en común entre cada una de ellas.

Según el Diccionario de la Real Academia Española, el lenguaje es la facultad del ser humano de expresarse y comunicarse con los demás a través del sonido articulado o de otros sistemas de signos ¹.

De acuerdo con Chomsky, el lenguaje es un conjunto finito o infinito de oraciones,

¹<https://dle.rae.es/lenguaje>

cada una de ellas de longitud finita y construida a partir de un conjunto finito de elementos [1, 10, 41].

Pavio y Begg definen el lenguaje como un sistema de comunicación biológico de información significativa e interindividualmente, a través de signos lingüísticos [24].

Coseriu, por su parte, dice que el lenguaje está fundado en la asociación arbitraria de un significado (contenido semántico) y un significante (imagen acústica). La unidad resultante es el signo lingüístico. Por lo tanto, para establecer comunicación se necesita que el hablante y el oyente acepten los mismos símbolos para los mismos objetos y que la combinación de dichos símbolos se efectúe según unas reglas [40].

De lo anteriormente expuesto, se deben destacar los elementos más relevantes de cada definición. Es decir, el lenguaje tiene un componente biológico, dota de la capacidad de comunicación entre individuos y que se trata de un conjunto generado a partir de unidades básicas [24, 40]. Finalmente, de acuerdo con Herrera [35], podríamos concluir que el lenguaje es un sistema de signos a través del cual el hombre elabora, expresa y comunica sus pensamientos.

Asimismo, es importante destacar que existen 2 categorías en cuanto a lenguajes: los naturales y los formales. Dentro de los lenguajes formales se encuentran los lenguajes matemáticos, lógicos, de programación, entre otros. En los lenguajes naturales, se hallan el inglés, alemán, francés, etc. Sin embargo, en este estudio sólo se abordarán los lenguajes naturales [45, 56].

Por otra parte, un lenguaje natural puede definirse como aquel lenguaje, ya sea escrito o hablado, que permite comunicarnos con las demás personas de forma cotidiana. Este tipo de lenguajes nos permiten expresar sentimientos, ideas, emociones, hechos, etc. Es crucial destacar que se designan como naturales debido a que no son lenguajes artificiales o creados, por ejemplo, a partir de ciertas reglas como una sintaxis o semántica. A pesar de esto, un lenguaje natural puede ser modelado a través de un lenguaje formal o bien ser formalizado a través de una teoría [45, 56].

De acuerdo con Polanco Fernández [45], los lenguajes naturales poseen las siguientes propiedades:

- Se desarrollan y organizan a partir de la experiencia humana. Esto es, sufren un enriquecimiento progresivo mediante la práctica y ésta precede a la teoría, la cual se desarrolla a través del tiempo.

- Gran expresividad. Permiten expresar y razonar sobre situaciones complejas
- Polisemia. Una oración puede tener diferentes significados dependiendo del contexto en el cual se utilicen ciertas palabras.
- Formalización compleja. A pesar de que la sintaxis puede ser expresada formalmente, la polisemia impide una formalización completa de cualquier lenguaje natural.

3.2. Procesamiento de Lenguaje Natural

El Procesamiento de Lenguaje Natural (PLN) es un subcampo de investigación enmarcado dentro de diferentes disciplinas como la lingüística, las ciencias de la computación, la inteligencia artificial, etc., cuyo objetivo es encontrar técnicas que permita a las computadoras la manipulación de texto en lenguaje natural. Liddy lo define como una gama de técnicas computacionales, teóricamente motivadas, para analizar y representar textos de ocurrencia natural a uno o más niveles de análisis lingüístico con el propósito de alcanzar procesamiento de lenguaje similar al de un humano para una variedad de tareas o aplicaciones [70]. Existen diferentes fines para el PLN, relacionados estrechamente con el dominio de la aplicación para la cual se utilice. Por ejemplo, en un sistema de recuperación de información que use PLN, la meta es representar el significado e intención de la consulta del usuario, la cual puede ser expresada en lenguaje natural como si se hablara a una bibliotecaria.

3.2.1. Niveles de Procesamiento de Lenguaje Natural

Un sistema de Procesamiento de Lenguaje Natural está dividido en cinco niveles, los cuales se presentan a continuación [38, 56]:

- Nivel fonológico. La relación de las palabras con el sonido que las representan. Para un análisis a este nivel se requiere de reglas fonéticas (sonido de las palabras y variación cuando se utilizan ciertas palabras en conjunto) y prosódicas (acentos, tonos, entonación).

- Nivel morfológico. En este nivel se define cómo se construyen las palabras a partir de morfemas (unidad mínima de significado que no puede dividirse más). Así, podemos conocer el significado de una palabra desconocida mediante la descomposición en morfemas.
- Nivel sintáctico. Este nivel trata de cómo las palabras pueden unirse para formar oraciones, fijando el papel estructural que cada palabra juega en la oración y que sintagmas son parte de otros sintagmas. Esto es, analizar las palabras en una oración para descubrir la estructura gramática de las mismas.
- Nivel semántico. En este nivel se determina el posible significado de las oraciones usando la interacción de las palabras, pero considerando el significado independiente del contexto, es decir, solamente las palabras.
- Nivel pragmático. En este nivel se considera el propósito con el cual se usan las oraciones y cómo este uso afecta al significado de las mismas.

En la figura 3.1 se aprecian los distintos niveles del PLN y las aplicaciones de cada uno de estos.

3.2.2. Word embeddings

Word embeddings son un conjunto de modelos para lenguaje y selección de características, cuyo fin es asignar palabras o frases a un espacio continuo de baja dimensión [28], a diferencia de otros modelos como n -gramas o vectores *one-hot*. Asimismo, este conjunto de técnicas permiten preservar la información semántica (significado) y sintáctica (estructura) de las palabras.

Existen distintas formas de agrupar los métodos para generar embeddings, entre las que se encuentran las siguientes.

- Modelos sintagmáticos y paradigmáticos. Estos se basan en la distribución de información de las palabras.
- Métodos de factorización de matrices y muestreo de ventana deslizante. Se dividen de acuerdo a cómo se genera el embedding, ya sea con descomposición de matrices o prediciendo la siguiente palabra mediante ventana deslizante.

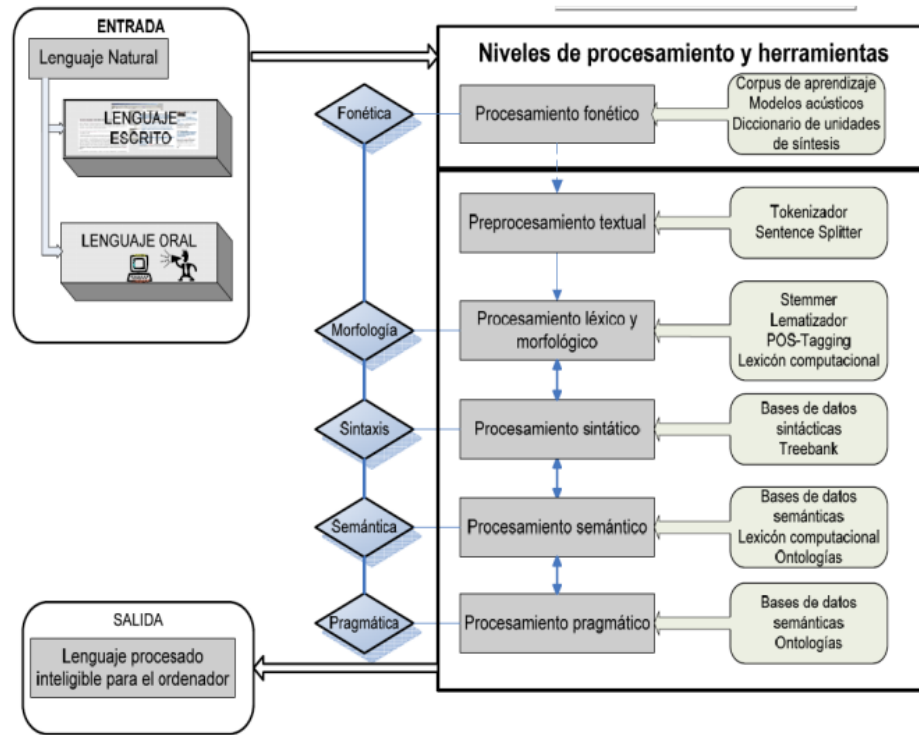


Figura 3.1: Niveles de procesamiento de lenguaje natural [18].

De acuerdo con Li y Yang [28], los *embeddings* pueden utilizarse en tareas de PLN donde pueden ser usados como datos de entrada o características para datos textuales. Así, pueden ser utilizados en los siguientes escenarios.

- Análisis semántico. Relación de la estructura sintáctica de las palabras con su significado, independientemente del lenguaje.
- Análisis sintáctico. Extracción de la estructura sintáctica a partir de oraciones.
- Análisis de idiomaticidad. Predicción de la composición de expresiones de palabras múltiples.
- Etiquetado gramatical. Etiquetar palabras en diferentes categorías gramaticales.
- Análisis de sentimientos. Extracción de opinión, sentimiento así como subjetividad de un texto.

- Reconocimiento de entidades nombradas. Reconocimiento de nombres de personas, organizaciones, valores monetarios, lugares, etc.
- Traducción automática. Sustitución de textos en un lenguaje dado por textos en otro correspondiente.
- Vinculación textual. Deducción del significado de un nuevo texto a partir de fuentes conocidas.

Por otra parte, es importante destacar que también existen embeddings a nivel de frase y documento. A nivel de frase pueden ser utilizados para administración de documentos web, identificación de paráfrasis y traducción automática, mientras que a nivel de documento se aplican en traducción automática y análisis de sentimientos Li y Yang [28].

3.2.3. Aplicaciones

El Procesamiento de Lenguaje Natural tiene distintas aplicaciones, entre las cuales se encuentran las siguientes [19, 25]:

- Recuperación de Información. Es la ciencia de buscar documentos o información en documentos. Estos documentos pueden ser de texto o multimedia así como pueden residir en la Web.
- Extracción de Información. Esta tarea tiene como objetivo la extracción automática de información a partir de documentos no estructurados o semi-estructurados. Ya que se ha obtenido la información, ésta se representa de forma estructurada y puede usarse en otras aplicaciones.
- Búsqueda de Respuestas. Se centra en proveer al usuario una respuesta concreta para la pregunta planteada.
- Resúmenes. Ciertos niveles del PLN, permiten reducir un texto largo en uno más corto pero mantiene la representación narrativa, de forma abreviada, del documento original.

- Traducción automática. Se encarga de las técnicas computacionales, las cuales varían desde las basadas en palabras hasta las que usan niveles altos de análisis, las cuales pueden ser utilizadas para traducir texto de un idioma a otro
- Sistemas de diálogo. Sistemas pensados para aplicaciones en las que hay interacción con el usuario final. El usuario se comunica con el sistema mediante diversas formas, por ejemplo, voz o texto.

Sin embargo, de acuerdo con Liddy [29], cualquier aplicación que utilice texto es candidata a usar Procesamiento de Lenguaje Natural. Por lo tanto, no está limitado a las mencionadas previamente.

3.3. Aprendizaje automático

El aprendizaje automático es una disciplina científica que se encarga del desarrollo de capacidades de aprendizaje para sistemas computacionales. El aprendizaje automático es una de las áreas centrales de la Inteligencia Artificial (IA).

De acuerdo con Camastra y Vinciarelli [8], el aprendizaje automático se desarrolla a partir de las siguientes líneas de investigación:

- Estudios orientados a tareas. Se encarga del desarrollo de sistemas de aprendizaje para mejorar el desempeño en ciertas tareas.
- Simulación cognitiva. Investigación y simulación de procesos de aprendizaje humanos.
- Análisis teórica. Es la investigación teórica de métodos y algoritmos posibles de aplicarse sin importar el dominio.

Por otra parte, de acuerdo con Ramasubramanian y Singh [48], existen distintas formas de agrupar los algoritmos de aprendizaje automático. Así, hay dos tipos de grupos, los cuales se muestran seguidamente:

- Agrupación basada en el tipo de aprendizaje. En este aprendizaje se consideran los tipos de variables de respuesta que están presentes en el conjunto de entrenamiento. Bajo esta clasificación se encuentran el aprendizaje supervisado, no supervisado, semi-supervisado y por refuerzo.

- Agrupación basada en tipo de problema. En este grupo se considera qué es lo que el modelo trata de lograr.

A continuación, se presentan las definiciones de los elementos que componen el grupo basado en el tipo de aprendizaje [48].

3.3.1. Aprendizaje supervisado

El aprendizaje supervisado se aplica para conjuntos de datos que tengan una variable de respuesta (es decir, una etiqueta), sin importar el tipo de la misma. El algoritmo debe aprender una variable de respuesta a partir del conjunto de variables predictorias proporcionado. En otras palabras, en este tipo de aprendizaje, el algoritmo recibe un conjunto de entrada pre-etiquetado e intenta converger al mejor clasificador posible $f : X \rightarrow Y$ que permita predecir nuevos ejemplos con una exactitud alta [17].

Este enfoque puede usarse para problemas de regresión o clasificación, por ejemplo, predecir el precio de una vivienda o determinar si un producto es o no defectuoso.

3.3.2. Aprendizaje no supervisado

Por otra parte, si el conjunto de datos no posee variables de respuesta, no puede usarse aprendizaje supervisado. El aprendizaje no supervisado se basa en alguna medida para cada instancia del conjunto que permita caracterizar los datos. Sin embargo, deben analizarse los datos de entrada para encontrar similitudes entre los mismos. A diferencia del aprendizaje supervisado, donde se trata de hallar una función que permita relacionar las variables predictoras con las de respuesta, en el aprendizaje no supervisado se debe analizar la organización espacial de los datos.

Este enfoque suele usarse en problemas asociación y agrupamiento, siendo éste último más común. Entre sus aplicaciones se encuentra la segmentación de usuarios, detección de fraudes, extracción de reglas de asociación, etc.

3.3.3. Aprendizaje semi-supervisado

En el aprendizaje supervisado, existen etiquetas para cada instancia del conjunto de datos mientras que en el aprendizaje no supervisado, dichas etiquetas no existen

o no se tiene acceso a ellas. En este tipo de aprendizaje, solamente se poseen variables de respuestas para una parte del conjunto. Sin embargo, usando técnicas de agrupamiento, se pueden propagar las etiquetas a todas las instancias de un mismo grupo [20].

3.3.4. Aprendizaje por refuerzo

El aprendizaje por refuerzo suele usarse en escenarios donde sólo se dispone de un estado inicial de los datos y hay diferentes respuestas posibles. Este tipo de algoritmos aprenden continuamente de su entorno de forma iterativa y, de esta manera, logran explorar todos los posibles resultados. Normalmente se usan en la robótica, conducción autónoma o en juegos de mesa como Go o videojuegos como StarCraft II [66].

3.3.5. Grupos de algoritmos de aprendizaje automático

Existen diversos algoritmos, cada uno con aplicaciones particulares. Sin embargo, pueden agruparse de acuerdo a los datos de entrada y a la salida que producen. A continuación se presenta una forma de agrupar estos algoritmos [48]:

- Métodos basados en regresión. En este tipo de métodos se aplica regresión para estimar la relación entre un conjunto dependiente de variables y uno independiente a través de minimización de error o un método probabilístico.
- Algoritmos basados en distancia. Estos algoritmos se utilizan para aprender representaciones y crear métricas que permitan determinar si un objeto pertenece o no a una clase.
- Métodos de regularización. Estos métodos penalizan la función de pérdida con el objetivo de balancear la complejidad y efectividad del modelo, ya que modelos muy complejos para un problema simple tienden al sobreajuste. Suelen utilizarse en problemas con grandes volúmenes de datos y gran cantidad de características.

- Algoritmos basados en árboles. Estos algoritmos aplican una serie de reglas condicionales a los datos; se decide una vez que se hayan cumplido las condiciones. Son algoritmos rápidos y de tipo distribuido.
- Algoritmos bayesianos. Los algoritmos bayesianos usan distribuciones de Bayes a priori y posteriori. A diferencia de los algoritmos de aprendizaje que aprenden iterativamente, los algoritmos basados en Bayes usan inferencia a partir de la distribución de una variable.
- Algoritmos de agrupamiento. La idea principal de este algoritmo es encontrar la mejor forma de formar grupos mediante una medida de similitud. En otras palabras, deben maximizar la similitud dentro de los grupos y minimizar la similitud entre grupos. Como se mencionó en la sección anterior, se trata de aprendizaje no supervisado.
- Minería de reglas de asociación. Se observan las relaciones entre variables y se cuantifica dicha relación con el fin de usarse para exploración y predicción. Son útiles para la construcción y extracción de reglas de asociación en grandes conjuntos de datos.
- Redes Neuronales Artificiales. Son modelos que permiten reconocer y aprender relaciones no lineales así como relaciones de alto orden entre múltiples variables. Permiten implementar dos de los tipos de aprendizaje automático mencionados, el supervisado y el no supervisado.
- Aprendizaje profundo. Son redes neuronales de gran tamaño y poseen múltiples capas ocultas. Son computacionalmente complejas, pero ofrecen la ventaja de ser aptas para conjuntos de datos enormes.
- Aprendizaje conjunto. Este paradigma de aprendizaje se basa en la combinación de resultados obtenidos mediante otros algoritmos de aprendizaje automático. Uno de los aspectos más importantes es ofrece la posibilidad de dividirse y ser distribuido.
- Minería de textos. Es una subárea del PLN, que a su vez es subárea del aprendizaje automático. Estos algoritmos proveen de herramientas para tratar infor-

mación en forma de texto (normalmente no estructurado) o bien crear características a partir de ésta para ser usadas por otros algoritmos de aprendizaje.

Tabla 3.1: Grupo de algoritmos de aprendizaje automático [48]

Grupo de algoritmos	Ejemplos
Regresión	Regresión lineal, logística, paso a paso, polinomial, mínimos cuadrados ordinarios (OLSR) y local (LOESS).
Basados en distancia	K-Vecinos más cercanos (k-NN), aprendizaje de cuantificación vectorial y mapa auto-organizado (SOM).
Regularización	Regresión Ridge, Elastic Net, regresión de ángulo mínimo y LASSO
Basados en árboles	Árbol de clasificación y regresión (CART), dicotomizador iterativo 3 (ID3), C4.5 y C5.0, bosque aleatorio, árboles de decisión condicional y detección de la interacción automática Chi-cuadrado (CHAID).
Bayesianos	Naive Bayes, Naive Bayes Gaussiano, Naive Bayes Multinomial, red de creencia Bayesiana y redes Bayesianas.
Agrupamiento	K-Means, k-Medians, particionamiento alrededor de medoides (PAM) y clustering jerárquico
Minería de reglas de asociación	Algoritmo de Apriori, de Eclat, minería basada en contexto y algoritmo de crecimiento FP.
Redes neuronales artificiales	Perceptron, retropropagación, redes de Hopfield y redes de base radial.

Aprendizaje profundo	Máquinas profundas de Boltzmann, redes de creencia profunda, redes convolucionales y autocodificadores apilados.
Aprendizaje conjunto	Boosting, agregación de Bootstrap, AdaBoost, generalización apilada y máquinas de potenciación de gradiente
Minería de texto	Resúmenes automáticos, reconocimiento de entidades nombradas, reconocimiento óptico de caracteres, etiquetado gramatical, análisis de sentimientos, reconocimiento de habla y modelado de tópicos.

3.4. Minería de datos

Existe una cantidad abismal de datos en la actualidad. Sin embargo, la mayoría de estos datos no están disponibles en una forma que permita procesarlos para obtener información útil. Debido a esto, surge la minería de datos. La minería de datos es el estudio de la recolección, limpieza, procesamiento, análisis y ganancia de perspectivas útiles a partir de los datos [3].

Esta tarea es especialmente difícil debido a la disparidad entre los tipos de datos y los problemas a resolver; incluso en problemas similares, las diferencias llegan a ser grandes. A pesar de esto, las aplicaciones de minería de datos se relacionan con alguno de los cuatro *super problemas* de minería de datos: minería de patrones de asociación, agrupamiento, clasificación y detección de valores atípicos. Estos problemas son de suma importancia ya que fungen como bloques de construcción para distintas aplicaciones [3].

Por otra parte, la minería de texto se clasifica de acuerdo a la fuente de los datos. A continuación, se describen los tipos de minería de texto [27]:

- Minería de datos relacional. La minería de datos relacional se encarga de los datos relacionales (bases de datos); los datos son colecciones de registros con

donde cada campo tiene un valor y cada registro está formado por un número fijo de valores. Es decir, tablas. Las tareas típicas en este tipo de minería son la clasificación, regresión y agrupamiento.

- **Minería web.** En esta tarea los datos a procesar proviene de la web; los datos que se proporcionan son un documento HTML e información adicional como texto, registros de acceso, hipervínculos, etc. Se subdivide en minería de contenido web, minería de web estructurada y minería de uso web. La minería de contenido web utiliza los contenidos de los documentos web para clasificarlos en uno o varios tópicos o bien realizar resúmenes automáticos. La minería de estructura web considera la estructura de los documentos web para categorizar un documento dada sus configuraciones de etiqueta; es decir, se ignora el contenido para concentrarse en plantillas y estructuras. Por otra parte, la minería de uso web considera las tendencias de acceso a los documentos web. Estas tendencias se representan como un grafo donde cada nodo indicala ruta de acceso desde una URL hacia otra.
- **Minería de macrodatos.** La minería de macrodatos supone un mayor reto respecto a las anteriormente mencionadas debido a que posee un gran volumen, velocidad (respecto a la generación de los datos), variabilidad (inconsistencia o ruido), variedad (datos en distintos formatos) y veracidad (calidad) respecto a los datos. En la figura 3.2 se observan los tres niveles de la minería de macrodatos. El nivel más básico y el que sirve de base para los demás niveles está formado por las técnicas de minería relacional; el nivel intermedio se forma por las técnicas de minería de texto, minería web y bio minería. Finalmente, el nivel más externo se compone de la minería de macrodatos, la cual es más compleja en estructura y tiende a estar incompleta o sufrir actualizaciones de datos de forma frecuente.

3.4.1. Fases de la minería de datos

De acuerdo con Aggarwal [3], la minería de datos involucra actividades como la limpieza, extracción de características, entre otras. A continuación, se describe el proceso general que se sigue:

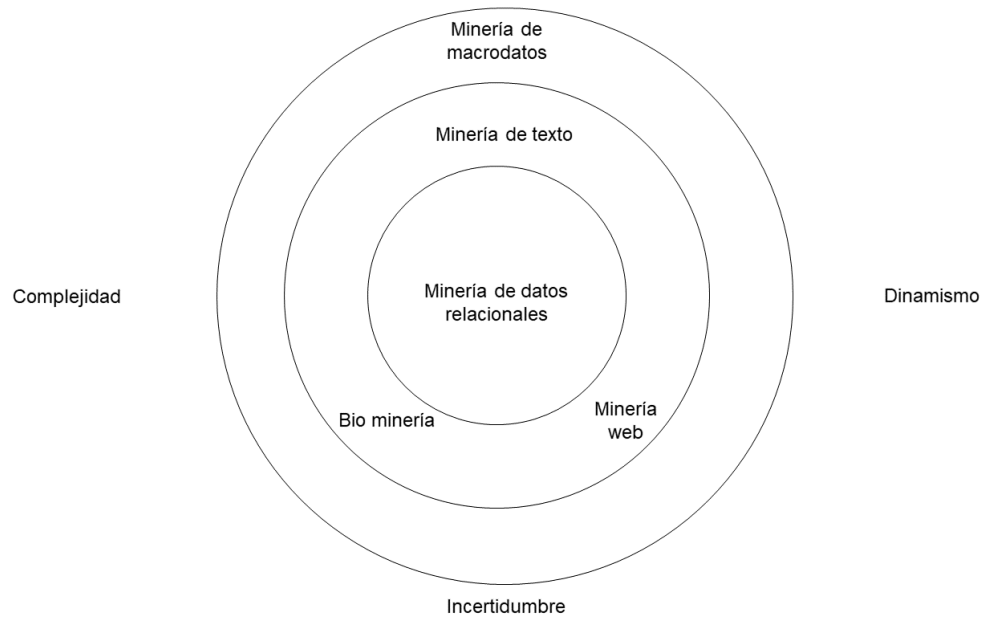


Figura 3.2: Niveles de la minería de macrodatos [27].

- **Recolección de datos.** La información puede provenir de diferentes fuentes como redes de sensores, encuestas de usuarios o de la Web. A pesar de esta dificultad, deben realizarse decisiones óptimas para evitar afectar este proceso. Al final de la recolección, los datos se guardan en bases de datos o almacenes de datos, para ser procesadas posteriormente.
- **Extracción de características y limpieza de datos.** Los datos necesitan volverse procesables, ya que la mayor parte del tiempo los datos recolectados se encuentran en formatos que no son aptos para ser procesados mediante algoritmos de minería de datos. Debido a esto, los datos deben ser convertidos a formatos semiestructurados o multidimensionales o series de tiempo. Esta etapa también involucra la limpieza de los datos, donde estos son corregidos o estimados en caso de que no estén presentes; si los datos son de fuentes heterogéneas, estos se integran y unifican. El producto final de esta fase es un conjunto de datos estructurado que será guardado, para ser procesado después.
- **Procesamiento analítico y algoritmos.** Esta es la última fase del proceso y

se encarga de diseñar métodos eficaces para los datos obtenidos en la etapa anterior. En ocasiones, los problemas no se ajustan a los super problemas, pero estos se pueden utilizar como bloques de construcción para crear una solución.

En la figura 3.3, se pueden observar las tres fases que conforman el proceso de minería de datos. También se indica que puede haber retroalimentación entre las fases de la minería de datos.

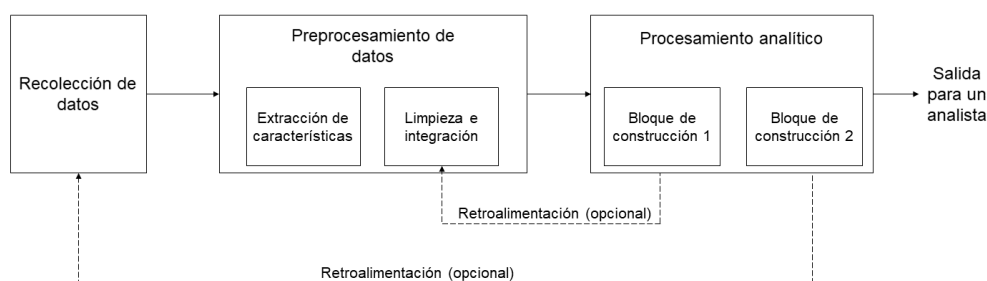


Figura 3.3: Proceso general de la minería de datos [3].

3.4.2. Análisis de agrupamiento

El análisis de agrupamiento agrupa objetos o individuos en grupos de tal forma que los objetos en un mismo grupo sean más similares entre sí que con los objetos de otros grupos. Se intenta maximizar la homogeneidad de los objetos dentro de los grupos y maximizar la heterogeneidad entre grupos [73].

Este análisis tiene diversas aplicaciones como resúmenes de datos, segmentación de clientes, análisis de redes sociales y sirve de herramienta a otros problemas de minería de datos como fase de preprocesamiento.

Existen diferentes modelos para realizar agrupamiento, los cuales tienen diferente rendimiento dependiendo del problema y el tipo de datos. La diferencia principal entre los modelos es el cómo definen la similitud entre los grupos de datos. Asimismo, deben seleccionarse características que sean informativas y debe eliminarse el ruido en los datos: este proceso es conocido como selección de características.

La selección de características es esencial ya que permite remover las características que puedan entorpecer el agrupamiento; determinan el subconjunto de características inherentes a la tendencia de agrupamiento subyacente. Hay dos modelos para la selección de características [2]:

- Modelos de filtro. Cada característica está asociada con una puntuación con el uso de un criterio de similitud. Este criterio provee una condición que permite filtrar las características, pues los datos que no alcancen cierta puntuación no son considerados. Los criterios utilizados en estos modelos son la fuerza de un término, dependencia predictiva de atributos, entropía y estadística de Hopkins.
- Modelos de envoltura. Este método evalúa la calidad de un subconjunto de características mediante un algoritmo de agrupamiento. Es útil debido a que los distintos métodos de agrupamiento pueden funcionar mejor con ciertos subconjuntos. Sin embargo, se pierde la información que pueden proveer algunas características. Se pueden utilizar criterios como la entropía de clase o la puntuación de Fisher.

Una diferencia significativa entre ambos modelos es que los modelos de filtro pueden usarse sólo para preprocesamiento mientras que los de envoltura se pueden aplicar al realizar el agrupamiento.

A pesar de la existencia de distintos tipos de algoritmos de agrupamiento, a continuación se presentarán los algoritmos de agrupamiento jerárquico debido a su importancia en la realización del presente trabajo.

Los algoritmos jerárquicos típicamente agrupan los datos basados en la distancia y usan otros métodos de agrupamiento para construir una jerarquía. Este tipo de agrupamiento es muy útil, pues provee una taxonomía de grupos, que puede usarse para obtener perspectivas semánticas.

En la figura 3.4 se puede observar una taxonomía creada a partir de *Open Directory Project* en la cual, a un alto nivel, los tópicos están organizados en tres categorías: arte, ciencia y salud. Luego, en el siguiente nivel se dividen en subtemas para cada área, como películas en el caso de arte o biología en ciencia.

De acuerdo a cómo se construye el árbol jerárquico de grupos, existen dos tipos de algoritmos de agrupamiento jerárquico, los cuales se presentan en seguida.

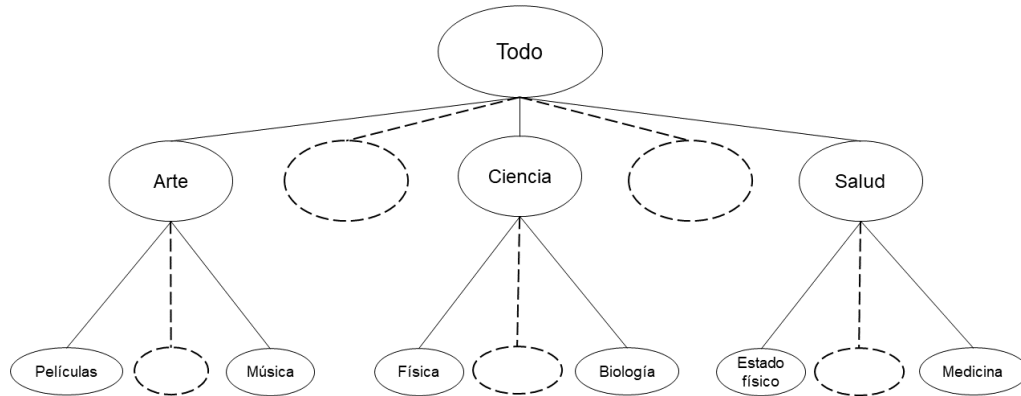


Figura 3.4: Niveles de granularidad en el agrupamiento jerárquico [2, 3].

- Métodos aglomerativos. Los puntos individuales, sucesivamente, se aglomeran en grupos más grandes. La diferencia entre métodos radica en qué función decide si los grupos se unen.
- Métodos divisivos. Se utiliza un enfoque de arriba hacia abajo para dividir sucesivamente los datos en una estructura parecida a un árbol. Debido a este enfoque es posible tener un mayor control del árbol en cuanto a grados y balance de las distintas ramas.

Algoritmo 1 Algoritmo de fusión aglomerativo genérico con criterio de fusión no especificado [2, 3]

Entrada: Datos: D

- 1: Inicializar una matriz de distancia M de $n \times n$ usando D
 - 2: **repetir**
 - 3: Elegir el par de grupo i y j usando M
 - 4: Fusionar los grupos i y j
 - 5: Eliminar filas/columnas i y j de M y crear una nueva fila y columna para el grupo creado
 - 6: Actualizar las entradas de la nueva fila y columna de M
 - 7: **hasta que** se alcance criterio de paro
 - 8: **devolver** Conjunto actual de grupos fusionados
-

El algoritmo 1 ilustra el procedimiento de agrupamiento aglomerativo con un criterio sin especificar. Las distancias se guardan en una matriz m de dimensiones

$n_t \times n_t$. Esta matriz provee de las distancia de los grupos calculadas por pares con base en un criterio de fusión. La fusión de dos grupos correspondientes a filas i y j requiere la distancia entre los objetos que componen a estos grupos. Es decir, que para dos grupos que contienen m_i y M_j objetos, respectivamente, existen $m_i \cdot m_j$ pares de distancias entre los objetos que los componen. Finalmente, la distancia se calcula como función de los pares $m_i \cdot m_j$.

El algoritmo 2 usa un algoritmo de agrupamiento plano A como subrutina; inicializa la raíz del árbol con todos los datos y divide los nodos recursivamente mediante un enfoque de arriba hacia abajo hasta que se alcanza cierta altura o cada nodo contiene un número menor de datos que el predeterminado. Es importante mencionar que A puede ser cualquier algoritmo de agrupación.

Algoritmo 2 Algoritmo genérico de agrupamiento de arriba hacia abajo [2, 3]

Entrada: Datos: D , Algoritmo plano: A

- 1: Inicializar el árbol T con una raíz que contenga a D
 - 2: **repetir**
 - 3: Seleccionar un nodo hoja L en T basado en un criterio predefinido
 - 4: Usar el algoritmo A para dividir L en $L_1 \dots L_k$
 - 5: Añadir $L_1 \dots L_k$ como hijos de L en T
 - 6: **hasta que** se alcance criterio de paro
-

3.4.3. Minería de texto

La minería de texto se define como el proceso de extracción de conocimiento implícito a partir de datos textuales [27]. La minería de texto es un caso especial de la minería de datos. La minería de texto tiene como objetivo principal analizar y descubrir cualquier patrón interesante, incluyendo tendencias y valores atípicos en un texto; asimismo, la noción de búsqueda no se consdiera esencial.

Las técnicas de minería aplicables son afectadas debido a las características del texto con respecto a otros datos como los relacionales, por ejemplo. A continuación, se presentan las características que diferencian al texto.

- El texto es disperso y de gran dimensionalidad.
- Existen diferentes niveles de análisis para la representación de un texto.

- Dificultad para tratar con dominios no restringidos.
- En dominios específicos, se necesitan técnicas especiales (por ejemplo, PLN) para obtener representaciones.

De acuerdo con Aggarwal y Zhai [4] existen distintas tareas en la minería de datos las cuales, a su vez, poseen algoritmos apropiados para lograr sus objetivos. Dichas tareas se presentan en seguida.

- Extracción de información para minería de texto. Esta tarea se encarga de extracción de información como entidades nombradas, relaciones entre entidades o bien de representaciones del texto.
- Resúmenes automáticos. Para esta tarea deben realizarse resúmenes cortos a partir de una gran colección de texto.
- Métodos de aprendizaje no supervisado para texto. Ya que no requieren datos para entrenamiento, son fácilmente aplicables al texto. Normalmente se usan métodos como agrupamiento o modelado de tópicos.
- Indexación semántica latente y reducción de dimensionalidad para minería de texto. Son los métodos para crear una representación comprimida de los datos subyacentes para fines de indexación y recuperación.
- Métodos de aprendizaje supervisado para texto. Son los métodos que necesitan datos con el fin de aprender una función para predecir nuevos datos. Se usan métodos tradicionales de aprendizaje como árboles de decisión, vecinos más cercanos, etc.
- Aprendizaje por transferencia. Son métodos en los cuales se intenta transferir el conocimiento de un dominio hacia otro. Es aplicable cuando la cantidad de datos es escasa.
- Técnicas probabilísticas. Métodos probabilísticos para modelado de tópicos no supervisado (análisis latente semántico probabilístico o asignación latente de Dirichlet) y métodos supervisados como campos aleatorios condicionales.

- Minería de flujos de texto. Se trata de minería de texto continua debido al ritmo de creación de datos.
- Minería de texto interlengua. Aprovecha la existencia de recursos en distintos lenguajes para realizar minado y aplicar el conocimiento obtenido de un lenguaje a otro.
- Minería de texto en redes sociales. Se encarga de explotar las plataformas de medios sociales mediante el contenido y los enlaces disponibles en el texto.
- Minería de opinión. Esta aplicación se centra en obtener resúmenes de las opiniones de consumidores o usuarios de productos o servicios con fines de optimizar las decisiones.
- Minería de datos biomédicos. Son métodos para proveer a los investigadores de conocimiento disponible en la literatura o complementar otros datos.

3.5. Extracción de Información

La Extracción de Información (EI) es la identificación y consecuente, o concurrente, clasificación y estructuración en clases semánticas de información específica encontrada en fuentes de datos no estructuradas, como textos en lenguaje natural, haciendo la información más adecuada para tareas de procesamiento de información. También es el nombre dado a cualquier proceso que estructura y combina selectivamente los datos que se encuentran, explícita o explícitamente, en uno o más textos [13]. La salida final del proceso de extracción varía, pero puede ser transformada de tal forma que sea usada para poblar una base de datos de algún tipo.

Por otra parte, la Extracción de Información no debe ser confundida con la Recuperación de Información, en la cual se selecciona un subconjunto relevante de documentos a partir de un conjunto más grande mientras que la Extracción de Información realmente extrae información del texto. Sin embargo, cualquier aplicación que utilice esta tecnología, normalmente es precedida por una etapa de Recuperación de Información en la que se seleccionan documentos acordes a alguna criterio que identifique la información relevante.

La EI se divide en 5 tareas, de acuerdo con Madalina Croitoru [31]:

- Reconocimiento de entidades nombradas. Encontrar entidades (por ejemplo, sustantivos nombres propios) y clasificarlas como nombres de personas, lugares, organizaciones, etc. A pesar de la gran exactitud alcanzada, esta tarea está limitada por el dominio al que se aplique, por lo que cambiar de dominio puede suponer cambios a un sistema que realice reconocimiento de entidades nombradas.
- Resolución de co-referencia. Hallar qué entidades y referencias (por ejemplo, pronombres) son idénticas, es decir, que se refieren a lo mismo. Esta tarea es menos relevante que las demás, pero su importancia principal es ser un bloque de construcción para la construcción de elementos de plantilla y la producción de escenarios de plantilla ya que permite la asociación de información descriptiva dispersa a través de varios textos a los que las entidades se refieren.
- Construcción de elementos de plantilla. Encontrar información adicional acerca de entidades de plantillas, por ejemplo, sus atributos. Es decir, asocian la información descriptiva con las entidades.
- Construcción de relación de plantilla. Encontrar relaciones entre entidades de plantilla. La extracción de relaciones entre entidades es una característica central de cualquier tarea de extracción. Sin embargo, es débilmente dependiente del dominio.
- Producción de escenarios de plantilla. Llenar la plantilla de eventos con relaciones y entidades especificadas. Normalmente son la salida prototípica de un sistema de EI. Unen la construcción de elementos de plantilla y la construcción de relación de plantilla. Es una tarea compleja ya que incluso las puntuaciones obtenidas por personas son relativamente bajas. Además, es dependiente de un dominio y está sujeta a los escenarios de interés del usuario.

La relación entre la Extracción de Información y las ontologías se puede caracterizar de dos formas: la primera es el uso de EI para aprendizaje de ontologías, así como métodos de poblamiento y enriquecimiento de las mismas; la segunda es la

explotación de ontologías para la interpretación de contenido textual en documentos con la finalidad de extraer información de estos.

Es importante destacar que la mayoría de estas tareas son dependientes del dominio en mayor o menor medida. Por lo tanto, en ciertos casos, cambiar de dominio involucraría cambios al sistema que las realice. La profundidad de estos cambios está relacionada con el dominio: si se cambia a un dominio similar, los cambios pueden ser menores mientras que si son dominios totalmente diferentes, los cambios pueden ser sustanciales.

Con base en lo anteriormente expuesto, se prestará mayor importancia a la tarea de Construcción de Relación de Plantilla, puesto que el descubrimiento y extracción de relaciones taxonómicas es acorde al objetivo de la presente investigación.

3.5.1. Fases de la extracción de información

La Extracción de Información utiliza el preprocesamiento de documentos y reglas de extracción (patrones o expresiones regulares) para identificar e interpretar el texto objetivo.

Normalmente, los sistemas de EI tienen tres pasos [39]:

- **Análisis de texto:** El análisis lingüístico incluye análisis de tipo morfológico, sintáctico y semántico. El análisis morfológico se centra en la forma de las unidades textuales, la segmentación del texto en secuencias de palabras, lematización y el etiquetado morfológico; el análisis sintáctico se encarga de etiquetado gramatical, mientras que el análisis semántico lidia con la representación formal del significado del texto. Este proceso depende de conocimiento lingüístico y del dominio del texto
- **Selección de reglas.** Las reglas de extracción están asociados con disparadores. Estos disparadores activan la verificación de las reglas de asociación.
- **Aplicación de reglas.** Una vez que se ha activado un disparador, se revisan todas las condiciones y se llena una plantilla de acuerdo a las reglas cumplidas.

3.5.2. Extracción de información como un mapeo de texto-ontología

Una de las metas de la EI es el mapeo de un texto a una ontología. La EI selecciona los datos más relevantes del texto para usarlos como valores para una plantilla; los valores se obtienen mediante un análisis semántico y la plantilla representa la estructura de la ontología [39].

Por otra parte, el mapeo de texto a ontología no es directo. Debido a esto, deben existir niveles intermedios de conocimiento. Así, el conocimiento léxico juega el papel de mediador. Este rol no es fácil de formalizar, pues no hay una relación uno a uno entre los conceptos en un texto, recursos léxicos o entidades ontológicas debido a fenómenos como la variación, polisemia o elipsis. Sin embargo, mediante la ayuda de lexicones se pueden construir relaciones entre el texto y el lexicon y entre el lexicon y la ontología.

3.6. Ingeniería ontológica

Antes de abordar la ingeniería ontológica, se discute la definición de ontología ya que ésta es la base sobre la que se construye el concepto de ingeniería ontológica. Posteriormente, se define la ingeniería ontológica, las tareas que comprende (poniendo especial énfasis en el aprendizaje de ontologías) y se detalla la evaluación de ontologías.

3.6.1. Ontología

La palabra ontología posee distintas definiciones, sin embargo, sólo se proporcionarán las competentes al área de las ciencias de la computación.

Una primera definición, propuesta por Gruber [30] en 1993, define una ontología como una especificación explícita de una conceptualización. Luego, en 1997, Borst definió una ontología como una especificación formal de una conceptualización compartida. Posteriormente, de acuerdo con [22], y Rudi Studer [71] al combinaron las definiciones propuestas por Borst y Gruber para determinar que una ontología es una especificación formal y explícita de una conceptualización compartida.

Por otra parte, en [30], una ontología se define como un conjunto de primitivas representacionales con las cuales modelar un dominio de conocimiento o discurso. Las primitivas representacionales son típicamente clases, atributos y relaciones. Las definiciones de las primitivas incluyen información acerca del significado y restricciones de su aplicación lógicamente consistente. Normalmente se especifican en lenguajes que permiten la abstracción separada de estructuras de datos o estrategias de implementación. Se dice que son semánticas ya que tienen poder expresivo similar al de la lógica de primer orden. Asimismo, permiten la integración de fuentes de datos heterogéneas la interoperabilidad entre sistemas distintos y la especificación de interfaces independendientes para servicios basados en conocimiento, gracias a la independencia de modelos de datos de bajo nivel.

Una ontología posee un conjunto mínimo de componentes principales: clases, relaciones, axiomas formales e instancias. Cada uno de estos se detalla a continuación [43]:

- Clases. Representan conceptos y abarcan una gran cantidad de estos, dependiendo del dominio. Se organizan en taxonomías. Asimismo, permiten la definición de metaclases, las cuales permiten establecer distintos grados de significado.
- Relaciones. Representan alguna asociación entre conceptos del dominio. Formalmente se define de la siguiente manera: $R \subset C_1 \times C_2 \times \dots \times C_n$. Es decir, una relación R es un subconjunto cualquiera del producto de n conjuntos C . Constan de un dominio y un rango. En una ontología, existen relaciones taxonómicas y no taxonómicas; normalmente se encuentran relaciones binarias, sin embargo, se pueden expresar relaciones n-arias fácilmente. Estas relaciones son usadas para expresar atributos de conceptos por lo que poseen un tipo de datos (entero, cadena, etc.) como rango. Por su parte, las relaciones poseen un concepto como rango.
- Axiomas formales. Permiten formar oraciones que siempre son verdaderas. Ayudan a representar conocimiento que no está definido mediante los otros componentes, permiten inferir nuevo conocimiento y verificar el conocimiento de la ontología e incluso la consistencia de la misma.

- Instancias. Son los individuales particulares de una ontología.

3.6.2. Definición de ingeniería ontológica

La ingeniería ontológica se refiere al conjunto de actividades concernientes al proceso de desarrollo de la ontología, ciclo de vida de la ontología, principios, métodos y metodologías para construir ontologías, y el conjunto de herramientas y lenguajes que los respaldan [43].

A continuación, se proporcionan los detalles con respecto a algunas de estas actividades.

- Métodos y metodologías. En 1990 aparecieron las primeras guías para el ciclo de desarrollo de ontologías. Con el paso de los años, surgieron distintas metodologías [39]. Asimismo, se crearon métodos para atacar distintas tareas relacionadas con el proceso de creación de una ontología, como la reingeniería, evaluación, entre otras. Entre los métodos y metodologías existentes se encuentran los métodos de Cyc, de Uschold y King y el método Sensus; el enfoque KAKTUS y las metodologías METHONTOLOGY, de Grüninger y Fox y On-To-Knowledge.
- Herramientas. Se dividen en 2 grupos: el primer grupo es de aquellas cuyo conocimiento puede ser asignado a un lenguaje de ontología y son usados como editores para dicho lenguaje; el segundo grupo son los conjuntos de herramientas integradas, las cuales tienen una arquitectura ampliable y cuyo conocimiento es independiente del lenguaje. Protégé, WebODE, OntoEdit y KAONI son ejemplos de conjuntos de herramientas para apoyar al desarrollo de ontologías.
- Lenguajes. Estos lenguajes tienen distintas bases, sin embargo, todos están basados en lógica de primer orden. Hubo intentos de estandarización, pero no fueron exitosos. Posteriormente, gracias al auge de la web, surgieron nuevos lenguajes de marcado para ontologías. Estos lenguajes aprovechaban otras tecnologías existentes. Entre los lenguajes con soporte activo se encuentran RDF, RDF Schema y OWL. Sin embargo, no se limitan a los anteriormente mencionados.

3.6.3. Aprendizaje de ontologías

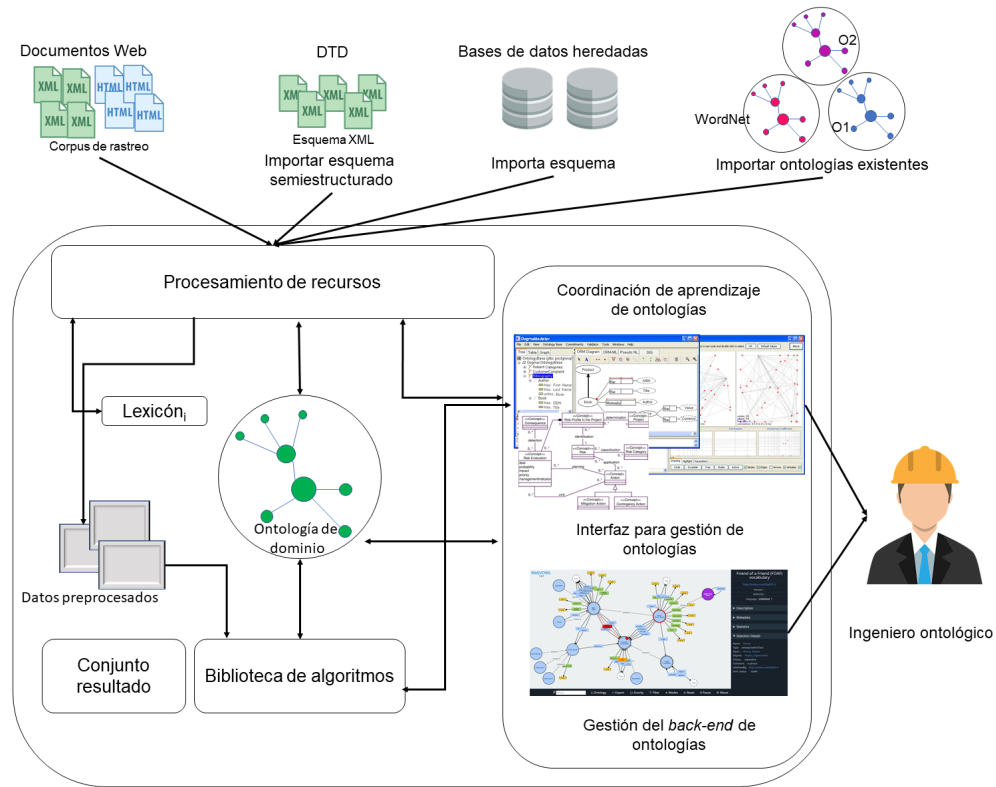


Figura 3.5: Arquitectura conceptual del aprendizaje de ontologías [32].

El aprendizaje de ontologías puede definirse como los métodos que sirven de apoyo a la ingeniería ontológica mediante técnicas de minería de datos, con el fin de proveer al ingeniero ontológico de relaciones que le permitan modelar un dominio. Existen tres tipos de datos a los que se les puede aplicar el aprendizaje de ontologías: datos estructurados, semi-estructurados y no estructurados [32].

El proceso de un sistema de aprendizaje de ontologías es similar al de minería de datos, pues siguen las fases de comprensión del negocio y problema, preparación de datos, modelado, evaluación y despliegue. Asimismo, los componentes cruciales para el proceso de aprendizaje de ontologías son la gestión de la ontología, coordinación, procesos de recursos y una biblioteca de algoritmos. A continuación, se describe cada

uno de estos componentes [32]].

- Componente de gestión de ontologías. Provee una interfaz entre la ontología y los algoritmos de aprendizaje. Las herramientas diseñadas para esta tarea permiten importar, navegar, modificar, versionar y evolucionar ontologías.
- Componente de coordinación. Permite la interacción de los componentes tanto para el procesamiento de recursos como para la biblioteca de algoritmos.
- Componente de procesamiento de recursos. Contiene técnicas para el descubrimiento, importación, análisis, y transformación de datos de entrada. Debido a esto, un subcomponente esencial es el Procesamiento de Lenguaje Natural, ya que mediante éste se puede procesar texto sin formato. Los componentes de este subsistema pueden ser un analizador léxico, un analizador morfológico, un etiquetador gramatical, una lista de expresiones regulares, un analizador parcial y un analizador sintáctico.
- Componente de biblioteca de algoritmos. Una biblioteca de algoritmos de aprendizaje automático aplicados al aprendizaje de ontologías; también permiten el mantenimiento y extracción de las primitivas de las ontologías.

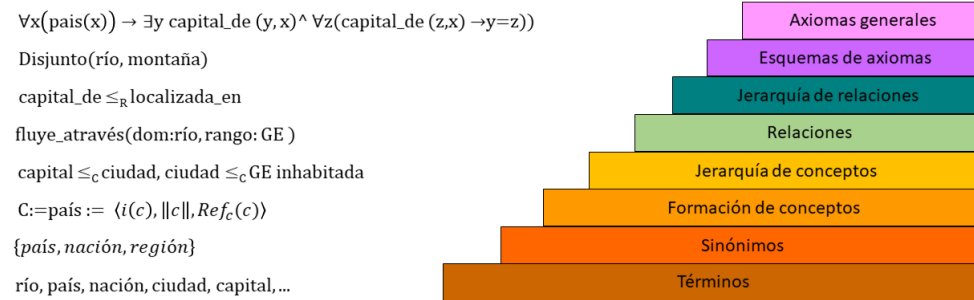


Figura 3.6: Capas del aprendizaje de ontologías [7, 32]

El modelo que aparece en la Figura 3.6 fue introducido por Maedche et al. [32] para mostrar la dependencia conceptual entre distintas tareas y cada capa se construye con los resultados de las anteriores. De esta manera, las 2 capas inferiores

Tabla 3.2: Algoritmos de aprendizaje automático y su uso en aprendizaje de ontologías [32].

Algoritmo	Uso genérico	Uso en aprendizaje de ontologías
Descubrimiento de reglas de asociación	Descubrimiento de grupos en datos (no supervisado)	Descubrimiento de asociaciones interesantes entre palabras
Agrupamiento jerárquico	Descubrimiento de grupos en datos (no supervisado)	Agrupamiento de palabras
Clasificación (p.e., SVM, Naive Bayes, kNN, etc.)	Predicción (supervisado)	Clasificación de nuevos conceptos en una jerarquía existente
Programación lógica inductiva	Inducción de reglas a partir de datos (supervisado)	Descubrimiento de nuevos conceptos a partir de datos extensionales
Agrupamiento conceptual (p.e., Análisis Formal de Conceptos)	Descubrimiento de conceptos (extensión e intensión)	Aprendizaje de conceptos y jerarquías conceptuales

corresponden al nivel léxico del aprendizaje de ontologías y su objetivo es determinar la terminología relevante así como los grupos de sinónimos, los cuales servirán para la formación de conceptos. Los conceptos se definen de la siguiente forma: $c := \langle i(c), \|c\|, Ref_i \rangle$, donde $c(i)$ es una descripción intensional, $\|c\|$ es una extensión y Ref_i es una función que representa cómo se realiza simbólicamente un concepto en un corpus. Similarmente, los dos niveles superiores son las más difíciles debido a que no hay límite para la complejidad de los axiomas que se desarrollan en ésta. Sin embargo, dicha complejidad se limita mediante el lenguaje para el desarrollo de la ontología.

Para descubrir relaciones taxonómicas en un texto, existen diversos enfoques. Dichos enfoques son los basados en patrones léxico-sintácticos, clustering, análisis de frases y clasificación, los cuales se describen en seguida [32].

- Patrones léxico-sintácticos. Este enfoque se basa en la definición de patrones que permitan extraer relaciones de tipo hiponimia. Hearst [23] definió una lista

de patrones léxico-sintácticos, los cuales pueden apreciarse en la tabla 3.3, para el aprendizaje automático de relaciones de hiponimia a partir de un corpus. Por ejemplo, uno de los patrones propuestos es: *such NP₀ as NP₁, ..., NP_{n-1} (or/and) other NP_n*, donde NP_i es un sintagma nominal. Si se encuentra en el texto una coincidencia con cualquiera de los patrones se tiene que $0 < i \leq n$ *hiponimo*(NP_i, NP_0) ; es decir, NP_0 es hiperónimo de todos los NP_i mayores que 0 y menores o iguales que n . Estos patrones muestran una buena precisión para relaciones de tipo *es un* y *parte de*.

- **Agrupamiento.** Puede definirse como el método de agrupar objetos similares, basados en alguna representación. Existen 3 tipos de agrupamiento: aglomerativo, divisivo y conceptual. El agrupamiento aglomerativo construye un grupo para cada término en la fase inicial, en las siguientes fases combina los grupos más similares hasta que alcanza un criterio de parada. El agrupamiento divisivo agrupa todos los términos en un mismo grupo para que posteriormente, iterativamente, se generen grupos más pequeños mediante la división de los grupos más grandes en subgrupos. El agrupamiento conceptual crea una *lattice* que indica qué tanto se traslapan los atributos descriptivos representados por 2 términos. Es importante destacar que los métodos aglomerativos y divisivos se basan en una medida de similaridad; por otra parte, el agrupamiento conceptual puede ser computacionalmente intensivo, por lo que deben aplicarse heurísticas para reducir la *lattice*.
- **Análisis de frases.** Este enfoque utiliza la estructura interna de los sintagmas nominales para descubrir relaciones taxonómicas. Se basan en la idea de que un modificador para un sustantivo define una clase denotada por el sustantivo.
- **Clasificación.** Se crea un clasificador a partir de un corpus de entrenamiento y un conjunto de conceptos predefinidos con sus respectivas entradas léxicas. Posteriormente, se puede usar para clasificar los términos no clasificados. Puede utilizarse para refinar una taxonomía al clasificar los términos nuevos en una jerarquía dada.

Por otra parte, el descubrimiento de colocaciones se encarga de hallar asociaciones entre palabras. En otras palabras, se debe descubrir las palabras que co-ocurren de

manera estadísticamente significativa. Al aprender estas relaciones, debe tomarse en cuenta el nivel exacto de abstracción con respecto a la jerarquía de conceptos para el dominio y rango de la tarea en cuestión.

Tabla 3.3: Patrones léxico-sintácticos propuestos por Hearst para la extracción de relaciones de hiponimia [23].

Patrón	Estructura
Hearst 1	$NP_{hyper} \text{ such as } \{NP_{hypo},\} * \{(and or)\} NP_{hypo}$
Hearst 2	$\text{such } NP_{hyper} \text{ as } \{NP_{hypo},\} * \{(and or)\} NP_{hypo}$
Hearst 3	$NP_{hypo} \{, NP\} * \{, \} \text{ or other } NP_{hyper}$
Hearst 4	$NP_{hypo} \{, NP\} * \{, \} \text{ and other } NP_{hyper}$
Hearst 5	$NP_{hyper} \text{ including } \{NP_{hypo},\} * NP_{hypo} \{(and or)\} NP_{hypo}$
Hearst 6	$NP_{hyper} \text{ especially } \{NP_{hypo},\} * \{(and or)\} NP_{hypo}$

3.6.4. Aspectos

Existen distintos aspectos que son aptos para la evaluación automática de una ontología (de dominio e independiente a las tareas). Sin embargo, para obtener una evaluación de calidad, el evaluador debe considerar la integración de resultados obtenidos mediante la evaluación de distintos aspectos de la ontología. Es importante destacar que los aspectos deben mostrar un grado de libertad y revelan las decisiones de diseño tomadas para la construcción de una ontología.

A continuación, se listan los aspectos a evaluar de una ontología [67]:

- Vocabulario. El vocabulario en una ontología es el conjunto de todos los nombres (URI o literales) en esa ontología.
- Sintaxis. Las ontologías pueden ser descritas en distintas sintaxis. Este aspecto es acerca de la serialización en varias sintaxis.
- Estructura. Este aspecto contempla la varianza en cuanto la estructura de una ontología, la cual es representada como un grafo, debido a que la estructura puede variar aunque semánticamente sea la misma ontología.
- Semántica. Este aspecto es acerca de las características semánticas de la ontología.

- Representación. Este aspecto captura la relación entre la estructura y la semántica. Usualmente son evaluados comparando las métricas calculadas en el grafo RDF con las características de los posibles modelos especificados mediante la ontología.
- Contexto. Este aspecto es acerca de las características de la ontología comparada con otros artefactos en su ambiente, por ejemplo, una aplicación que use la ontología, la fuente de datos que describe la ontología, etc.

Sin embargo, debe destacarse que la ontología debe ser consistente y coherente para poder evaluarla. No obstante, el vocabulario, la sintaxis, el contexto (ciertas partes), y la estructura de la ontología pueden ser evaluados aún cuando la ontología sea inconsistente. Por otra parte, la semántica, la representación y algunos aspectos conceptuales no pueden ser evaluados.

3.6.5. Criterios

Además de los aspectos, también deben considerarse una serie de criterios que determinan la calidad de una ontología. Estos criterios se detallan a continuación [47, 67]:

- Exactitud. Correcta representación de los aspectos del mundo real.
- Adaptabilidad. Fácilidad para realizar cambios.
- Claridad. Comunicación efectiva del significado deseado de los términos.
- Completitud/Competencia. Cobertura apropiada del dominio de interés.
- Eficiencia computacional. Velocidad de procesamiento para los razonadores.
- Concisión. Ausencia de conceptos innecesarios o inútiles.
- Consistencia/Coherencia. Incapacidad de obtener conclusiones contradictorias a partir de la entrada.
- Aptitud organizacional/Accesibilidad comercial. Determina la facilidad de uso de la ontología para una organización y dice si el proceso de creación fue el adecuado.

Los criterios descritos son deseables y son metas en la creación y evaluación de una ontología, pero no son medibles directamente. Debido a esto se necesitan métodos concretos para evaluar ontologías [67].

3.6.6. Evaluación de ontologías

De acuerdo a y Asunción Gómez-Pérez [69], la evaluación de ontologías se refiere a la actividad de verificar la calidad técnica de una ontología contra un marco de referencia. Por otra parte, existen 2 conceptos para describir la evaluación de ontologías: verificación y validación. La primera se encarga de la correcta construcción de la ontología, mientras que la segunda se refiere a si realmente las definiciones modelan el mundo real para el cual fue creada la ontología [49].

Existen distintas formas para evaluar ontologías, las cuales se basan en medidas y métodos para examinar un conjunto de criterios [47, 55]:

- Evaluación basada en métricas. Presenta un conjunto de procesos que se esperan que el usuario lleve a cabo para obtener medidas de estabilidad de una ontología existente.
- Evaluación basada en limpieza. Provee una visión estructural y fundamental del modelo considerando aspectos como rigidez, unidad, identidad y dependencia. Se utiliza para ayudar a los usuarios a limpiar taxonomías.
- Evaluación basada en lexicón. Se enfoca en el alcance del vocabulario, el bienestar de la taxonomía y la adecuación de las relaciones no taxonómicas. Se aplica a resultados de técnicas de minado automáticas cuyo objetivo es crear ontologías.
- Evaluación basada en corpus. Son usados para evaluar qué tan suficientemente una ontología cubre un dominio dado. Existen 2 enfoques: el primero es realizar una extracción de términos automáticos en el corpus y después contar el número de conceptos que se traslapan entre el corpus y la ontología a evaluar. El segundo, usa una representación vectorial para el corpus y la ontología con el fin de medir el ajuste entre las representaciones.

- Evaluación basada en tareas. Intentan medir qué tanto ayuda una ontología a mejorar los resultados de cierta tarea. Se evalúa solamente en la tarea para la que está diseñada y no se toman en cuenta sus características estructurales.

3.6.7. Herramientas

La implementación de la propuesta de solución desarrollada en esta investigación se realizó utilizando el lenguaje de programación Python y distintas bibliotecas para este lenguaje.

Python [64] es un lenguaje creado en 1991 por Guido van Rossum. Este lenguaje es de alto nivel, interpretado, de tipado dinámico, de propósito general y orientado a objetos. Sin embargo, se considera multiparadigma pues permite programación procedural o funcional.

Existen diversas bibliotecas que fueron útiles durante la implementación del enfoque propuesto para la extracción de relaciones taxonómicas. Entre las bibliotecas usadas, se encuentran las siguientes:

- NLTK [5]. Es un conjunto de bibliotecas orientada al procesamiento de lenguaje natural en distintos lenguajes; sin embargo, no todas las características están disponibles en todos los lenguajes. Provee de herramientas para el análisis léxico, colocaciones, reconocimiento de entidades nombradas, etiquetador gramatical, entre otras. Esta biblioteca se usa principalmente en investigación².
- Spacy [26]. Es una biblioteca de código abierto diseñada para procesamiento de lenguaje natural avanzado. Ofrece modelos neuronales estadísticos para lenguajes como francés, inglés, etc. Además de provee reconocimiento de entidades nombradas, lematización, etiquetado gramatical, compatibilidad con modelos de aprendizaje profundo, entre otras funciones. A diferencia de NLTK, Spacy se centra en el uso en ambientes de producción³.
- Scikit-Learn. Es una biblioteca que ofrece algoritmos de aprendizaje automático listos para usar para tareas como clasificación, regresión y agrupamiento.

²<https://www.nltk.org/>

³<https://spacy.io/>

Está implementado en Cython, lo que le permite realizar operaciones rápidamente. Asimismo, facilita su integración con otras bibliotecas de Python⁴.

- SciPy. Es una biblioteca de código abierto que se proporciona un ecosistema de software para ciencia, ingeniería y matemáticas. Se compone de paquetes que permiten la manipulación de vectores, visualización de datos, matemáticas simbólicas, análisis y estructuras de datos, entre otros⁵.
- Tensorflow. Es una plataforma de código abierto, desarrollada por Google, que brinda un ecosistema de herramientas y bibliotecas para el desarrollo de aprendizaje automático⁶.
- Gensim. Es una librería para modelado de tópicos, indexación de documentos y recuperación de similitud en corpora⁷.

⁴<https://scikit-learn.org/stable/>

⁵<https://www.scipy.org/>

⁶<https://www.tensorflow.org/>

⁷<https://radimrehurek.com/gensim/>

Capítulo 4

Propuesta de solución

Usando los conceptos teóricos y el estudio del estado del arte se propone una solución para la extracción de relaciones taxonómicas así como dos sistemas adicionales para el apoyo de esta solución. El primer sistema descrito se utiliza para la extracción de automática de relaciones taxonómicas, el segundo sistema se usa para la clasificación de relaciones taxonómicas a partir de los datos obtenidos mediante el primer sistema y el último sistema tiene como objetivo la clasificación de relaciones de hiperonimia.

En este capítulo contiene tres secciones en la cuales se describe el comportamiento general de los sistemas propuestos y se detalla el funcionamiento de cada uno de las componentes que conforman los sistemas.

4.1. Sistema propuesto para la extracción automática de relaciones taxonómicas

Para la extracción de relaciones taxonómicas en documentos no estructurados se propuso un enfoque que se explica a continuación. Este enfoque está formado por tres etapas. La primera etapa se basa en patrones léxico-sintácticos para la extracción de relaciones de hiponimia en documentos no estructurados; en la segunda etapa se aplica ingeniería de características para extraer una matriz de características, para cada documento, que sea adecuada para aplicar el algoritmo de agrupamiento jerárquico y finalmente en la última etapa se obtiene una taxonomía mediante un

algoritmo de agrupamiento jerárquico. En la figura 4.1 se puede observar una vista general del sistema propuesto y en el algoritmo 6 se describe su funcionamiento.

Un aspecto importante de la implementación del algoritmo 6 es que fue realizada utilizando multiprocesamiento asíncrono mediante un paquete nativo de Python ¹. Así, se evita el problema de bloqueo del intérprete global y se permite el uso de los núcleos disponibles de un procesador. Es decir, cada proceso se encarga de hallar el conjunto de las relaciones de hipónimos (un archivo de texto) y escribirlo a un archivo; luego se obtiene una matriz de características para este conjunto de relaciones y finalmente se ejecuta el algoritmo de agrupamiento y se obtiene una segunda matriz con la cual se genera un archivo que contendrá el dendrograma de ésta.

4.1.1. Extracción de relaciones de hiponimia

En la primera etapa del sistema, se hace una extracción de relaciones de hiponimia mediante patrones léxico-sintácticos (tablas 3.3, 4.1 y 4.2) para cada documento. Asimismo, se extrae la oración donde se encuentra esta relación con el fin de obtener la similitud en la segunda etapa de este sistema.

Los pasos que se realizan en esta etapa, que se detallan en el algoritmo 3, son los siguientes:

1. Preprocesamiento del texto. El texto se pasa a minúsculas y se conservan los signos de puntuación como paréntesis, puntos, comas, punto y coma, entre otros. Además, el texto se divide en oraciones.
2. Formación de fragmentos de texto. De cada oración se obtienen las frases sustantivas (mediante una llamada a la biblioteca Spacy), que se definen como las frases que tienen un sustantivo como cabeza (en la Figura 4.2 se muestran un ejemplo de frases sustantivas). Luego, se dividen las frases sustantivas en tokens (secuencia de caracteres que forman una unidad semántica útil para procesamiento) para eliminar caracteres no alfanuméricos y estos se añaden a una lista.
3. Discriminación de candidatos. A partir de la lista de patrones léxico-sintácticos

¹<https://docs.python.org/2/library/multiprocessing.html>

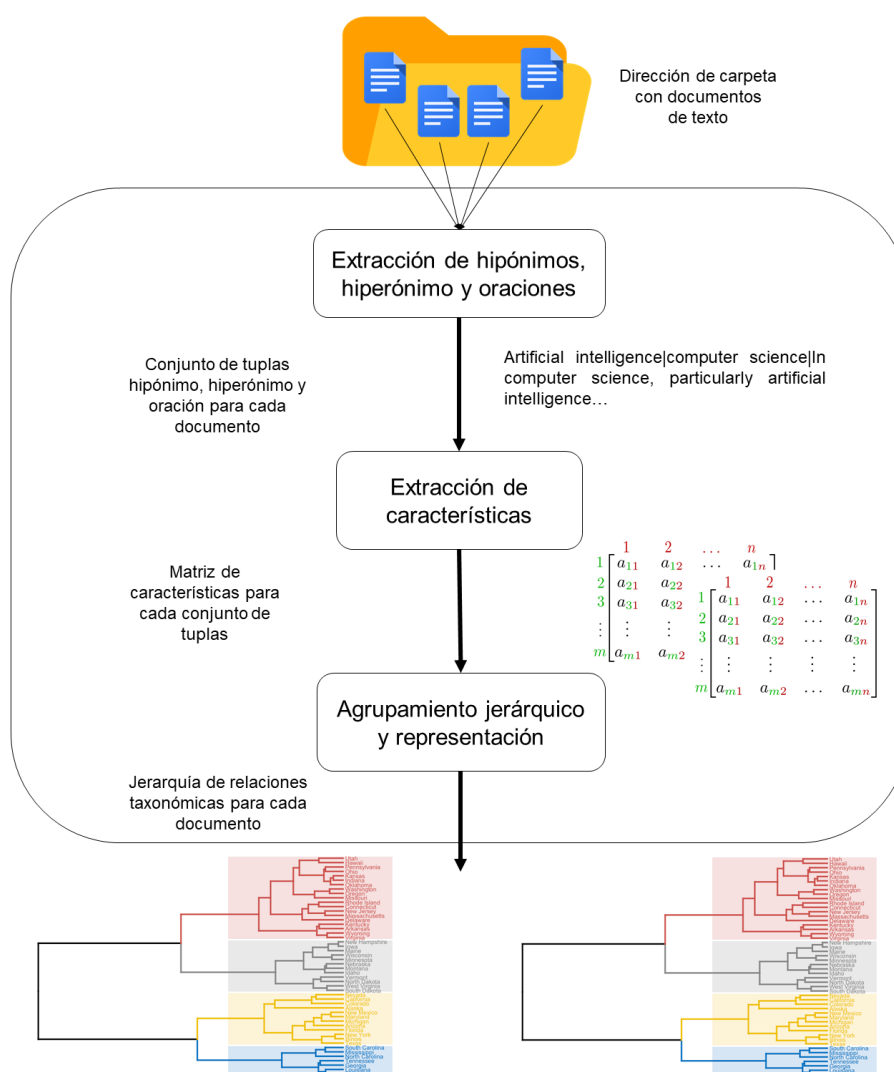


Figura 4.1: Vista general del sistema propuesto.

definidos se buscan candidatos a relaciones de hiponimia en la lista de texto obtenida en el paso anterior. En este paso existen dos casos:

- a) Existe una coincidencia. Si en el texto se encuentra algún patrón léxico-sintáctico, se procede a obtener los términos específicos, el término general y la oración donde se hallan, se forma una tupla con estos y se guarda en una lista de relaciones.
- b) No existe una coincidencia. Si no se encuentra el patrón, se procede a la siguiente oración en el texto o el siguiente patrón en la lista.

Cuando se haya finalizado la búsqueda de todos los patrones en las oraciones, las listas de relaciones se pasan a la segunda etapa del sistema.

Tabla 4.1: Patrones obtenidos de [57]

Patrones
NP including NP
NP such as NP
by such NP as NP
NP like NP
NP except NP
NP e.g. NP
NP i.e. NP
NP, (a) kind(s) type(s) form(s) of NP
NP: (a) kind(s) type(s) form(s) of NP
NP other than NP
There (are is)(could would) be two types of NP(: ,) NP
NP especially NP
NP notably NP
NP particularly NP
NP usually NP
NP mostly NP
NP mainly NP

NP principally NP

Tabla 4.2: Patrones, propuestos por Giovannetti et al. [21], obtenidos de [57]

Relación	Patrón en inglés	Ejemplo
Hiponimia	"A and other B"	"duck and another animals"
	"A is a type of B"	"duck is a type of animals"
	"A or other B"	"duck or another animals"
	"B like the A"	"animals like the duck"
	"B such as the A"	"animals such as the duck"
Meronomia	"A is made up of B"	"bicycle is made up of wheel"
Cercana-sinonimia	"A also called B"	"puma also called cougar"
Co-hiponimia	"A and B are"	"thymine and adenine are nucleobases..."

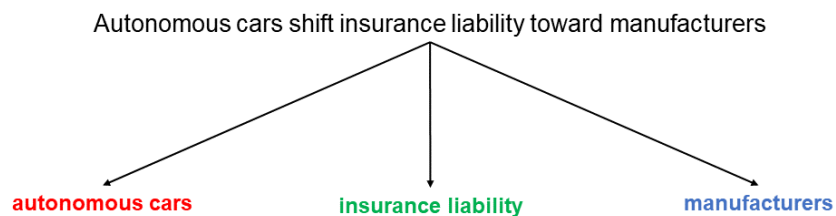


Figura 4.2: Ejemplo de frases sustantivas para una oración.

4.1.2. Extracción de características

La segunda etapa recibe como entrada el conjunto de relaciones para cada documento obtenidas en la primera etapa. A partir de cada conjunto de relaciones se

genera una matriz de características con el fin de usarla en la etapa de agrupamiento jerárquico. Como se mencionó anteriormente, cada conjunto de relaciones tiene tuplas que contienen el hipónimo, el hiperónimo y la oración donde se hallaron. Con base en lo hallado en la literatura, se proponen como características la similitud entre el hipónimo e hiperónimo, la similitud entre el hipónimo y la oración, la etiqueta gramatical del hipónimo, la similitud del hipónimo con la oración después de reducirla a su forma raíz, la similitud del hipónimo con la oración después de reducirla a su forma lematizada y la posición donde se encuentran los hipónimos e hiperónimos en la oración, respectivamente.

Los pasos que se realizan para la obtención de las características enumeradas, que se muestran en el algoritmo 4, son los que se muestran a continuación:

1. Reducción de oración y asignación de etiqueta gramatical. Para cada tupla, se reduce la oración donde se encontró la relación de hiponimia y se asigna la etiqueta gramatical al hiperónimo y estos datos se añaden a la tupla correspondiente. La reducción de la oración se realiza a su forma raíz (*stemming*) y a su forma lematizada (*lemmatisation*) usando NLTK y Spacy, respectivamente. Asimismo, se asigna la etiqueta gramatical mediante NLTK.
2. Obtener representación vectorial. Con el fin de aplicar una medida de similitud, se debió obtener una vectores de los datos obtenidos durante la fase anterior, pues no se puede cuantificar directamente sólo con el texto. Así, se utilizó Spacy para obtener una representación vectorial de dimensión 300 de cada dato de cada tupla en el conjunto de relaciones.
3. Cálculo de similitud. Una vez obtenida la representación vectorial, se midió la similitud usando similitud coseno a través de *Scikit-Learn*. Posteriormente, a cada tupla se le añaden las siguientes medidas de similitud: entre el hipónimo y el hiperónimo, entre el hipónimo y la oración, entre el hipónimo y la oración reducida a forma raíz, entre el hipónimo y la oración reducida a forma lematizada.
4. Cálculo de posición de hipónimo-hiperónimo y transformación de la etiqueta gramatical. Para cada hipónimo e hiperónimo se busca en la oración en qué posición se encuentra cada uno de ellos. Por otra parte, se transforma la etiqueta

gramatical de cada hipónimo, asignada en el paso anterior, a un valor numérico utilizando *Scikit-Learn*. Esta información también se agrega a cada tupla del cada conjunto de relaciones para cada uno de los documentos.

Finalmente, se genera una matriz que contiene las características descritas anteriormente para cada conjunto de relaciones. Estas matrices pasan a la etapa final del sistema.

4.1.3. Agrupamiento jerárquico

La última etapa de la propuesta de sistema recibe como entrada las matrices de características de cada documento, aplica un algoritmo de agrupamiento jerárquico aglomerativo (algoritmo 1) mediante SciPy y obtiene una representación a través de un dendrograma de las relaciones taxonómicas encontradas.

A continuación, se provee una explicación de la función utilizada para el agrupamiento ²: Se necesita calcular la distancia $d(s, t)$ entre dos grupos s y t . El algoritmo comienza con un bosque de grupos que aún no han sido agrupados. Cuando dos grupos s y t se combinan en un grupo u , estos se remueven y éste último se añade al bosque; el algoritmo finaliza cuando sólo queda un grupo en el bosque y este grupo se vuelve la raíz. En cada iteración se mantiene una matriz de distancia donde la entrada $d[i, j]$ corresponde a los grupos i y j del bosque original; asimismo, debe actualizarse esta matriz al formar un nuevo grupo u respecto a los grupos que quedan en el bosque. Suponiendo que hay $|u|$ observaciones en u y $|v|$ objetos originales en v y sea v cualquier grupo restante diferente de u se usan métodos de enlace para calcular la distancia entre el grupo u y cada v . El método de enlace usado en el agrupamiento es el de enlace completo, también denominado algoritmo de punto más lejano o algoritmo Voor Hees, que asigna $d(u, v) = \max(\text{dist}(u[i], v[j]))$ a todos los puntos i en el grupo u y j en el grupo v .

Asimismo, se necesita una medida de distancia, por lo cual se utilizó la distancia Jaccard que indica, dados dos vectores u y v , la proporción de los elementos de estos vectores que corresponden ³. Se seleccionó esta medida debido a que permite una mejor representación de las relaciones.

²<https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html>

³<https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.pdist.html>

Esta etapa consta de 3 pasos, los cuales se describen en seguida:

1. Eliminación de datos no numéricos. Para cada matriz se eliminan los datos textuales y se mantienen las similitudes calculadas, las posiciones y la etiqueta gramatical transformada.
2. Aplicación de agrupamiento. A cada matriz de características, se le aplica el algoritmo de agrupamiento jerárquico aglomerativo y se obtiene otra matriz.
3. Generación dendrograma. A partir de la matriz obtenida en el paso anterior, se genera un dendrograma (mediante SciPy) que permite representar el agrupamiento como una taxonomía.

Algoritmo 3 Algoritmo para la extracción de relaciones hipónimo-hiperónimo

Entrada: Texto T **Salida:** Conjunto de tuplas que contienen hipónimo-hiperónimo y la oración donde se encuentran

```

1: Inicializar lista  $L$  con los patrones léxico-sintácticos
2: texto := { }
3: relaciones := { }
4: TextoPreprocesado := Preprocesamiento( $T$ )
5: Oraciones := dividirEnOraciones(TextoPreprocesado)
6: frasesSustantivas := obtenerFrasesSustantivas(Oraciones)
7: para fraseSustantivas en frasesSustantivas hacer
8:   para token en fraseSustantiva hacer
9:     texto := texto + token
10:    si token no es alfanumérico entonces
11:      texto := texto + obtenerAlfanumérico(token)
12:    fin si
13:  fin para
14: fin para
15: para oracion en texto hacer
16:   para patrón en  $L$  hacer
17:    si patrón está en oración entonces
18:      terminosgenerales := obtenerTerminosGenerales(oracion)
19:      terminosespecificos := obtenerTerminosEspecificos(oracion)
20:      relaciones := relaciones + (terminosespecificos, terminosgenerales, oración)
21:    fin si
22:  fin para
23: fin para
24: devolver relaciones

```

Algoritmo 4 Algoritmo para la extracción de características a partir de las tuplas de hipónimo, hiperónimo y oración.

Entrada: Conjunto de tuplas S que contiene hipónimo-hiperónimo y la oración donde se encuentran

Salida: Matriz M de características

```

1:  $M := \{ \}$ 
2: para tupla en  $S$  hacer
3:   posTag := asignarEtiquetaGramatical(tupla(oracion))
4:   oracionLematizada := lematizer(tupla(oracion))
5:   oracionRaiz := stemmer(tupla(oracion))
6:   vectorHiponimo := calcularVector(tupla(hiponimo))
7:   vectorHiperonimo := calcularVector(tupla(hiperonimo))
8:   vectorOracion := calcularVector(tupla(oracion))
9:   oracionLematizadaVector := calcularVector(oracionLematizada)
10:  oracionRaizVector := calcularVector(oracionRaiz)
11:  similitudHiponimoHiperonimo := calcularsimilitud(vectorHiponimo, vectorHiperonimo)
12:  similitudHiponimoOracion := calcularsimilitud(vectorHiponimo, vectorOracion)
13:  similitudHiponimoRaiz := calcularsimilitud(vectorHiponimo, oracionRaizVector)
14:  similitudHiponimoLematizada := calcularsimilitud(vectorHiponimo, oracionLematizadaVector)
15:  posicionHiponimo := buscarenOracion(tupla(hiponimo), tupla(oracion))
16:  posicionHiperonimo := buscarenOracion(tupla(hiperonimo), tupla(oracion))
17:  posTagNumérico := convertirANumérico(posTag)
18:   $M := M + \{ \text{tupla(hiponimo), tupla(hiperonimo), tupla(oracion), similitudHiponimoHiperonimo, similitudHiponimoOracion, similitudHiponimoRaiz, similitudHiponimoLematizada, posicionHiponimo, posicionHiperonimo, posTagNumérico} \}$ 
19: fin para
20: devolver  $M$ 

```

Algoritmo 5 Algoritmo para agrupamiento jerárquico y representación.

Entrada: Matriz M de características, nombreDendrograma

Salida: Archivo de dendrograma

para fila en M **hacer**

 eliminar hiponimo, hiperonimo, oracion, posTag, oracionLematizada, oracion-Raiz

fin para

$Z := \text{agrupamientoJerarquicoAglomerativo}(M)$

 archivoDendrograma := crearDendrograma(Z , nombreDendrograma)

Algoritmo 6 Algoritmo general del primer sistema.

Entrada: Dirección de directorio D

Salida: Archivos con tuplas de hipónimo, hiperónimo y oración y dendrogramas para cada archivo en el directorio

```

1: hiperonimos := {}
2: tuplas := {}
3: características := {}
4: para nombreArchivo en dirección hacer
5:   para linea en nombreArchivo hacer
6:     para oracion en linea hacer
7:       hiperonimos := hiperonimos + buscarHiperonimos(oracion)
8:       si hiperonimos no está vacío entonces
9:         abrirArchivo nombreArchivo_salida.txt en modo añadir
10:        para hiperonimo en hiperonimos hacer
11:          escribirArchivo nombreArchivo_salida.txt, hiperonimo
12:        fin para
13:      fin si
14:    fin para
15:  fin para
16:  abrirArchivo nombreArchivo_salida.txt en modo leer
17:  para linea en nombreArchivo_salida.txt hacer
18:    tuplas := tuplas + linea
19:  fin para
20:  características=extraerCaracteristicas(tuplas)
21:  agrupamientoRepresentacion(características, nombreArchivo)
22: fin para
```

4.2. Sistema propuesto para la clasificación de relaciones taxonómicas

Después de haber aplicado el algoritmo de agrupamiento, se puede saber a qué grupo pertenece cada dato y, a su vez, obtener el número de grupos que existen en los datos. Esto puede lograrse mediante diferentes técnicas como el método del codo o el método de inconsistencia. Una vez que se ha obtenido el número de clases existentes y se conoce a qué clase pertenecen los datos, estos datos pueden ser utilizados para alimentar un modelo de aprendizaje supervisado. En el trabajo de Vidal et al. [65] se analiza el uso de diferentes algoritmos de aprendizaje automático para clasificación. Con base en lo hallado por los autores, se propone el uso de árboles de decisión para clasificar cualesquiera nuevas relaciones dentro de las jerarquías propuestas.

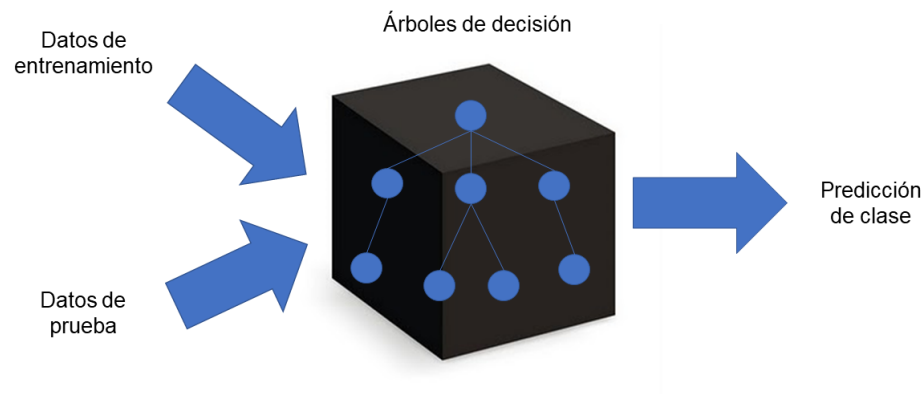


Figura 4.3: Esquema del modelo de clasificación de nuevas relaciones taxonómicas.

En la figura 4.3 se muestra el esquema general del modelo que se propone para clasificar nuevas relaciones taxonómicas. Los datos de entrenamiento para el algoritmo son las matrices obtenidas mediante el algoritmo de agrupamiento (algoritmo 1) y los datos de prueba son generados a través de las dos primeras fases del primer sistema propuesto, sin necesidad de aplicar la etapa de agrupamiento. De esta forma, se predice a qué clase pertenece cada una de las relaciones en el conjunto de datos de prueba.

4.3. Sistema propuesto para la clasificación de relaciones de hiperonimia

Para la clasificación de relaciones de hiperonimia se utilizó un enfoque basado en el trabajo de Yildirim y Yildiz [72], donde el problema de determinar si existe una relación de tipo hiperonimia entre un par de términos se aborda como clasificación binaria. El aspecto fundamental detrás de esta propuesta es que existe una fuerte relación entre los desplazamientos de vectores (que se definen como la diferencia del vector del hiperónimo y el vector del hipónimo) y las relaciones semánticas, de hiperonimia en este caso. En la figura 4.4 se puede observar una vista general de la arquitectura para la red neuronal utilizada en esta propuesta. La implementación de este algoritmo fue realizada en Python mediante módulos como Tensorflow⁴ para la creación del modelo de aprendizaje y Gensim⁵ para la utilización de embeddings de fastText⁶.

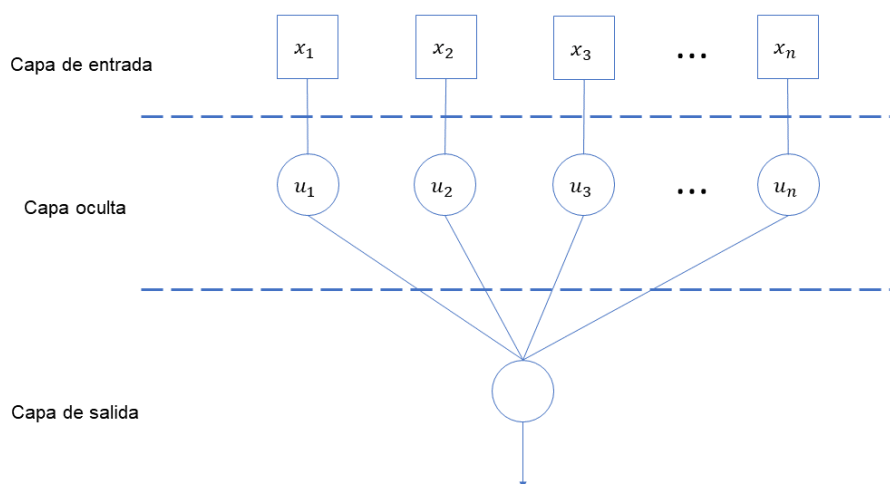


Figura 4.4: Vista general de la arquitectura propuesta.

⁴<https://www.tensorflow.org/>

⁵<https://radimrehurek.com/gensim/>

⁶<https://fasttext.cc/>

4.3.1. Características utilizadas

El modelo utilizado para la clasificación de relaciones de hiperonimia recibe una entrada de dimensión 300 y la capa oculta posee el mismo número de neuronas. Debido a esto, se decidió utilizar una representación vectorial de dimensión 300 (mediante *word embeddings* de fastText) para los pares hipónimo-hiperónimo. Esta representación está formada por 2 millones de vectores de palabras pre-entrenados en el corpus Common Crawl. El formato del archivo de los vectores contiene el número de palabras en el vocabulario y el tamaño de los vectores; las palabras están ordenadas de forma descendente de acuerdo a su frecuencia. Asimismo, cada línea en el archivo consta de una palabra y su vector asociado [37]. De esta forma, para cada palabra se obtiene un vector de dimensión 300 y al obtener la diferencia de los vectores de hiperónimo e hipónimo, se conserva la misma dimensión. Este proceso se repite para cada uno de los pares presentes en el conjunto de entrenamiento y se utilizan para entrenar el modelo.

4.3.2. Entrenamiento

Se abordó la tarea como un problema de clasificación binaria, donde se determina si para un par dado de términos existe o no una relación de hiperonimia. Para el desarrollo de la solución se eligió una red neuronal para realizar la clasificación, cuya arquitectura se muestra en la figura 4.4. Para implementarla, se utilizó Keras con Tensorflow 2.0, lo que permitió realizar el entrenamiento usando una GPU (unidad de procesamiento de gráficos). Los experimentos fueron realizados con un CPU Intel I5 y una GPU GeForce GTX 1050 Ti. La red fue entrenada para minimizar la cross-entropía binaria, utilizando optimización ADAM con tamaño de lote 32 durante 300 épocas.

Capítulo 5

Resultados

En este capítulo se muestran los resultados obtenidos mediante los enfoques propuestos para la extracción de relaciones taxonómicas en documentos no estructurados y la clasificación de relaciones de hiperonimia. Primero se discute el método de evaluación utilizado, se describen los conjuntos de datos usados, luego se muestran los resultados alcanzados mediante los sistemas propuestos.

5.1. Medidas de evaluación

Para definir el rendimiento del enfoque propuesto para la extracción de relaciones taxonómicas se usó una forma de evaluación, la cual se explica a continuación.

5.1.1. Exactitud

La exactitud, del inglés *accuracy*, puede definirse como la evaluación del sesgo de las predicciones; es decir, indica qué porcentaje de las predicciones son correctas. Se define de la siguiente forma: $Exactitud = \frac{CantidadDeCasosCorrectos}{TotalCasos}$

En este trabajo se utilizó la exactitud para determinar la cantidad de relaciones obtenidas mediante este sistema, comparándolas con las relaciones taxonómicas totales existentes en los conjuntos de datos usados. Por otra parte, se usó para medir el desempeño del clasificador de relaciones de hiperonimia.

5.1.2. Precisión

La precisión permite indicar al clasificador que un ejemplo negativo no debe ser marcado como positivo y se utilizó para medir el desempeño del clasificador de relaciones de hiperonimia.

Se define de la siguiente forma: $Precision = \frac{verdaderosPositivos}{verdaderosPositivos+falsosPositivos}$

5.1.3. Exhaustividad

La exhaustividad permite hallar los ejemplos positivos y se define de la siguiente manera: $Exhaustividad = \frac{verdaderosPositivos}{verdaderosPositivos+falsosNegativos}$ Se usó para medir el desempeño del clasificador de relaciones de hiperonimia.

5.1.4. F_1

Esta medida se define como la media ponderada de la precisión y la exhaustividad; la precisión y la exhaustividad contribuyen igualmente a esta medida. Se define de la siguiente forma: $F_1 = 2 * \frac{precision*exhaustividad}{precision+exhaustividad}$. La medida F_1 se usó para medir el desempeño del clasificador de relaciones de hiperonimia.

5.1.5. Área bajo la curva ROC

La curva ROC se define como una gráfica en la que se representa el rendimiento de un modelo de clasificación en todos los umbrales de clasificación con base en la tasa de verdaderos positivos (exhaustividad) y falsos positivos. En esta gráfica, la tasa de verdaderos positivos se encuentra en el eje y mientras que la tasa de falsos positivos está en el eje x. Así, el área bajo la curva ROC proporciona una medición agregada del rendimiento en todos los umbrales de clasificación posibles¹.

5.2. Conjuntos de datos

Para la realización de este trabajo se consideró un conjunto de datos, obtenido del trabajo de Zouaq et al. [74], que consta de dos ontologías de dominio: la primera de

¹<https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc>

Inteligencia Artificial (IA) y la segunda de estándar e-Learning SCORM (SCORM). En la tabla 5.1 se muestra la cantidad de documentos, tokens y vocabulario que conforman cada uno de los corpus de estas ontologías.

Tabla 5.1: Corpus de dominio

Dominio	Número de documentos	Tokens	Vocabulario
IA	8	11370	1510
SCORM	36	34497	1325

Para extraer los elementos de las ontologías se utilizó Jena ², una herramienta que permite crear aplicaciones basadas en ontologías. En la tabla 5.2 se encuentra el total de conceptos y relaciones taxonómicas para cada ontología de dominio.

Tabla 5.2: Elementos que conforman las ontologías

Dominio	Número de conceptos	Relaciones taxonómicas
IA	276	205
SCORM	1461	1038

Por otra parte, se utilizó un conjunto de datos diferente para la creación del clasificador de relaciones de hiperonimia, obtenido del trabajo de Shwartz et al. [53]. Este conjunto está formado por pares de términos positivos y negativos de relaciones de hiperonimia obtenidos de fuentes como WordNet, DBPedia, Wikidata y Yago; relaciones diferentes a la de hiperonimia se consideran ejemplos negativos. Los autores proveen dos divisiones (70 % para entrenamiento, 25 % para validación y 5 % para prueba) de los conjuntos de datos: una aleatoria y otra léxica; en este trabajo se usó la aleatoria para el desarrollo del modelo de clasificación de relaciones de hiperonimia.

²<https://jena.apache.org/>

5.3. Extracción y validación de relaciones taxonómicas

En esta sección se muestran los resultados obtenidos a través del sistema propuesto para la extracción de relaciones taxonómicas en documentos no estructurados. Primero se presentan los resultados de la extracción de hipónimos-hiperónimos que serán considerados como candidatos a relaciones taxonómicas y posteriormente se indica el porcentaje de relaciones taxonómicas encontradas con respecto a las presentes en la ontología.

Para obtener las relaciones candidatas se utilizó la primera fase del sistema propuesto, la cual consta de lo expresado en el algoritmo 3. Después de aplicarlo a los corpus, se obtuvieron 102 relaciones para el dominio de inteligencia artificial y 172 para el dominio de SCORM (Tabla 5.3). En las tablas 5.4 y 5.5 se encuentran ejemplos de las relaciones obtenidas para el dominio de inteligencia artificial y SCORM, respectivamente.

Tabla 5.3: Relaciones totales obtenidas mediante el algoritmo 3.

Dominio	Número de relaciones
IA	102
SCORM	172

Tabla 5.4: Ejemplos de relaciones obtenidas para la ontología de IA.

Hipónimo	Hiperónimo
STRIPS	formal language
Resource Description Framework RDF	standard
Web Ontology Language OWL	standard
truth table	logic operation
computer vision	machine learning

consciousness	human trait
---------------	-------------

Tabla 5.5: Ejemplos de relaciones obtenidas para la ontología de SCORM.

Hipónimo	Hiperónimo
text	medium
data management	service
SCORM	reference model
ARIADNE	group
menu	user interface navigation control
xml	internet technology

Posteriormente, se hizo una verificación para comprobar si todas las relaciones identificadas eran adecuadas para continuar con la siguiente etapa del sistema. Se hallaron 4 y 15 relaciones en el dominio de inteligencia artificial y SCORM, respectivamente, que no eran relevantes debido a que eran unigramas como *the* o *a*. Esto se debe a que no se eliminaron las palabras cerradas durante la etapa de preprocesamiento con el fin de evitar que se perdiera información necesaria para que los patrones léxico-sintácticos pudieran coincidir. Por ejemplo *of* se considera una palabra cerrada, pero es importante en un patrón como *NP*, *(a) kind of NP*. Una vez depuradas, el total de relaciones para IA es de 98 y para SCORM es de 157 (Tabla 5.6).

Tabla 5.6: Relaciones depuradas obtenidas mediante el algoritmo 3.

Dominio	Número de relaciones
IA	98

SCORM	157
-------	-----

Finalmente, se hizo una evaluación de la exactitud de las relaciones obtenidas. Para el dominio de IA se obtuvieron 98 relaciones candidatas, pero sólo se hallaron 14 que existen en relaciones taxonómicas de la ontología de inteligencia artificial. Para el dominio de SCORM, a pesar de que se proponen 157 relaciones candidatas, no se encontró ninguna de éstas en las relaciones taxonómicas de la ontología, por lo que se obtuvo una exactitud de 0 en tal dominio. En la tabla 5.7 se pueden observar los puntajes de exactitud alcanzados mediante esta propuesta.

Tabla 5.7: Exactitud de las relaciones encontradas para cada dominio.

Dominio	Exactitud
IA	0.068
SCORM	0.0

5.4. Agrupamiento jerárquico y representación

En esta sección se muestran los resultados obtenidos al aplicar las fases dos y tres del sistema propuesto al conjunto de relaciones obtenido en la primera etapa del sistema. Primero, se muestran las matrices de características (resultado de la extracción de característica) y luego se muestra la representación de los hipónimos mediante un dendrograma resultado de aplicar el algoritmo de agrupamiento jerárquico.

Para obtener las matrices de características que se utilizarán en la última etapa de la propuesta de sistema, se necesita el conjunto de relaciones de hiponimia. A este conjunto de relaciones, que contiene 3 características iniciales, se le aplica el algoritmo 4 para obtener una matriz que contendrá 10 características más. En la tabla 5.8 se observan las dimensiones de las estas matrices antes y después de aplicar el algoritmo de extracción de características.

Tabla 5.8: Dimensiones de las matrices de características durante la segunda fase del sistema.

Dominio	Dimensión inicial	Dimensión final
IA	(98,3)	(98,13)
SCORM	(157,3)	(157,13)

Una vez finalizada la segunda fase del sistema, se aplica el algoritmo 5 a las matrices obtenidas. Por lo tanto, se eliminan 6 columnas de estas matrices y se reducen las dimensiones a las mostradas en la tabla 5.9.

Tabla 5.9: Dimensiones de las matrices de características durante la tercera fase del sistema.

Dominio	Dimensión inicial	Dimensión final
IA	(98,13)	(98,7)
SCORM	(157,13)	(157,7)

Luego se continúa con la aplicación del algoritmo de agrupamiento aglomerativo, el cual recibe cada una de estas matrices y devuelve otra matriz de dimensiones $((n - 1), 4)$ para cada una de ellas. Esta matriz obtenida se utilizará en el paso final del algoritmo para crear un dendrograma que ilustre el proceso de agrupamiento de cada conjunto de relaciones.

Finalmente, las matrices producto del algoritmo de agrupamiento se utilizan para crear una representación estructurada en forma de árbol en la que pueden observarse las subcategorías en las que están organizados los datos. En las figuras 5.1 y 5.2 se observan los dendrogramas creados para las relaciones obtenidas mediante el sistema para los dominios de inteligencia artificial (IA) y SCORM. En los nodos se encuentran los hipónimos obtenidos mediante los patrones sintácticos y se puede observar que los conceptos similares se encuentran agrupados juntos.

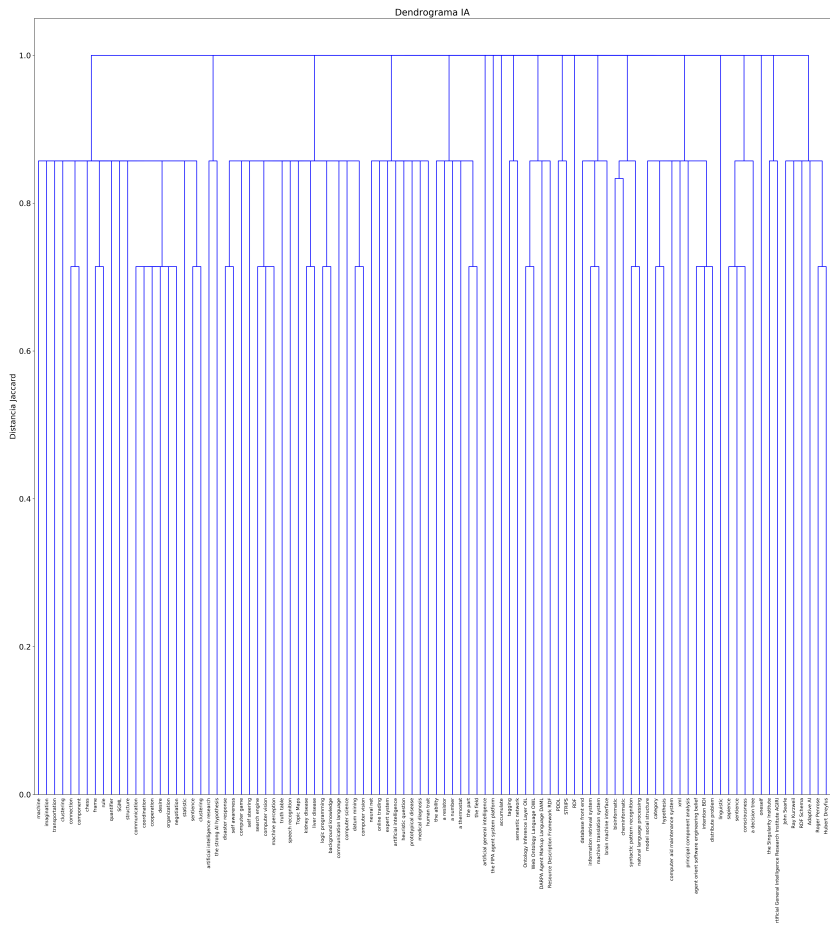


Figura 5.1: Dendrograma para el dominio de IA.

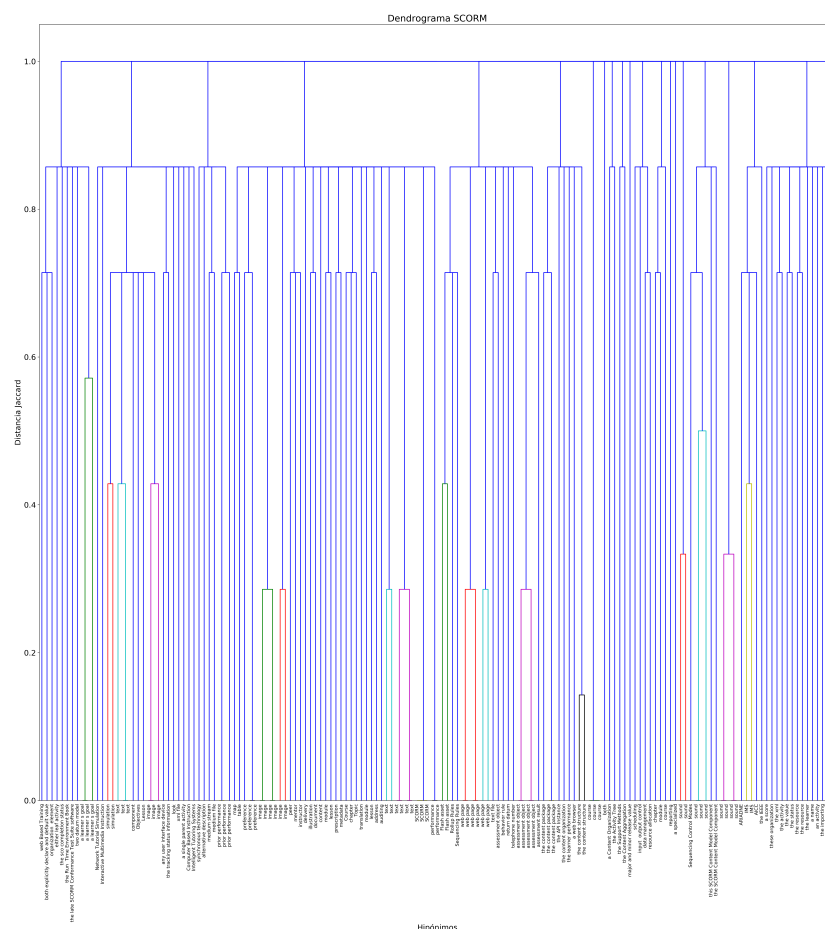


Figura 5.2: Dendrograma para el dominio de SCORM.

5.5. Clasificación y validación de relaciones de hiperonimia

En esta sección se muestran los resultados obtenidos a través del sistema propuesto para la extracción de relaciones de hiperonimia. Los resultados de las predicciones del modelo usado en este trabajo fueron medidas utilizando el área bajo la curva ROC, precisión, exhaustividad y F_1 ; como gold estándar se utilizó la evaluación hecha por expertos para determinar si las relaciones existentes en las ontologías de IA y SCORM son, efectivamente, relaciones de hiperonimia.

Para obtener las predicciones sobre las relaciones anotadas por los expertos para ambas ontologías, se utilizó el sistema para la clasificación de relaciones de hiperonimia. En las tablas 5.10, 5.11 y 5.12 se muestran los reportes de clasificación para las predicciones realizadas por el modelo respecto a lo anotado por el primer, segundo y tercer experto, respectivamente para la ontología de IA. Con base en estas tablas, el modelo obtiene una precisión muy alta para los ejemplos donde sí existe una relación de hiperonimia; cuando no existe una relación de hiperonimia, la precisión es muy baja, especialmente para las anotaciones del tercer experto. Sin embargo, este comportamiento puede deberse al bajo soporte que tiene esta clase en este caso. Por otra parte, debe destacarse que las predicciones tienen un rendimiento medio en términos de F_1 y exhaustividad si consideramos el promedio ponderado. En las tablas 5.13, 5.14 y 5.15 se observan los reportes de clasificación para las predicciones realizadas por el modelo respecto a lo anotado por el primer, segundo y tercer experto, respectivamente para la ontología de SCORM. En general, la precisión es más baja, pero los valores de F_1 y exhaustividad son más cercanos, exceptuando el valor de F_1 para la clase falsa, anotada por el primer experto. Por otra parte, si consideramos los promedios ponderados, las predicciones con mejor rendimiento son las hechas respecto a las anotaciones del primer experto para la ontología de SCORM. Finalmente, si consideramos lo expuesto en las tablas 5.16 y 5.17, se puede ver que todas las predicciones obtienen una exactitud relativamente similar en el caso de IA, pero las predicciones respecto a lo anotado por el tercer experto, tiene mayor área bajo la curva. En el caso de las predicciones para SCORM, se puede apreciar que la peor exactitud y área bajo la curva ROC son para las predicciones hechas por el

modelo para las anotaciones del tercer experto, mientras que las predicciones de las anotaciones restantes tienen puntuaciones casi idénticas.

Tabla 5.10: Reporte de clasificación conforme a las anotaciones del primer experto para la ontología de IA.

Clase	Precisión	Exhaustividad	F_1	Soporte
Falso	15	70	25	23
Verdadero	93	51	66	181

Tabla 5.11: Reporte de clasificación conforme a las anotaciones del segundo experto para la ontología de IA.

Clase	Precisión	Exhaustividad	F_1	Soporte
Falso	11	69	18	16
Verdadero	95	51	66	181

Tabla 5.12: Reporte de clasificación conforme a las anotaciones del tercer experto para la ontología de IA.

Clase	Precisión	Exhaustividad	F_1	Soporte
Falso	4	80	7	5
Verdadero	99	50	66	199

Tabla 5.13: Reporte de clasificación conforme a las anotaciones del primer experto para la ontología de SCORM.

Clase	Precisión	Exhaustividad	F_1	Soporte
-------	-----------	---------------	-------	---------

Falso	16	50	24	18
Verdadero	79	40	53	82

Tabla 5.14: Reporte de clasificación conforme a las anotaciones del segundo experto para la ontología de SCORM.

Clase	Precisión	Exhaustividad	F_1	Soporte
Falso	28	50	36	32
Verdadero	32	38	47	68

Tabla 5.15: Reporte de clasificación conforme a las anotaciones del tercer experto para la ontología de SCORM.

Clase	Precisión	Exhaustividad	F_1	Soporte
Falso	33	48	39	33
Verdadero	45	30	36	47

Tabla 5.16: Exactitud y área debajo de la curva ROC para las predicciones para la ontología de SCORM conforme a las anotaciones de los expertos.

Experto	Exactitud	Área bajo la curva ROC
1	53.43	60.47
2	51.96	59.64
3	50.49	64.87

Tabla 5.17: Exactitud y área debajo de la curva ROC para las predicciones para la ontología de SCORM conforme a las anotaciones de los expertos.

Experto	Exactitud	Área bajo la curva ROC
1	42	45.12
2	42	44.11
3	37.5	39.13

Capítulo 6

Conclusiones

En el presente trabajo de investigación fueron implementadas tres propuestas de sistemas para la extracción y validación de relaciones taxonómicas en documentos no estructurados, clasificación de relaciones taxonómicas y clasificación de relaciones de hiperonimia. A continuación se detalla cada uno de los sistemas.

El primer sistema está compuesto de tres etapas (extracción de hipónimos-hiperónimos, extracción de características y agrupamiento jerárquico y representación), cuyo objetivo es la extracción de relaciones taxonómicas en documentos no estructurados. La primera fase del primer sistema, consta de la extracción de hipónimos-hiperónimos mediante patrones léxico sintácticos, obtuvo dos listas de 98 y 157 relaciones taxonómicas para los dominios de inteligencia artificial y SCORM, respectivamente. Sin embargo, sólo 14 de las relaciones obtenidas para IA se encontraban en las relaciones taxonómicas de la ontología de este dominio; en el caso del estándar de e-Learning SCORM, ninguna de las relaciones encontradas existe en la ontología SCORM. Esto se refleja en una exactitud muy baja para IA y nula para SCORM. Esto puede deberse a que la cobertura de los patrones de Hearst es baja [6] y a que el algoritmo busca los patrones en las frases nominales de las oraciones, lo cual puede limitar la búsqueda. La lista de hipónimos-hiperónimos obtenida en este paso se utilizará para la segunda fase. En la segunda fase del primer sistema se obtiene una matriz de características para los conjuntos de relaciones que sea apta de usarse en la siguiente fase de este sistema. Mediante Spacy y NLTK se representan vectorialmente los hipónimos-hiperónimos y las oraciones con el fin de calcular medidas de simili-

tud. Asimismo, se asignan etiquetas gramaticales y posiciones donde se encuentran los hipónimos e hiperónimos, con el objetivo de añadir información suplementaria para el algoritmo de aprendizaje automático no supervisado. En la fase final del primer sistema se eliminan características no numéricas de las matrices obtenidas en la fase anterior con el objetivo de aplicar agrupamiento jerárquico aglomerativo usando distancia Jaccard. Después de aplicar el algoritmo de agrupamiento, se usa la matriz devuelta por éste para realizar una representación por medio de un dendrograma de la jerarquía de relaciones obtenidas a partir de un documento no estructurado. Esta representación permite visualizar de forma efectiva cómo se agrupan las relaciones, pero no muestra más información acerca de la estructura de la taxonomía.

Para el segundo sistema se propuso un modelo para la clasificación de relaciones taxonómicas basado en árboles de decisión. A pesar de que se entrenaron clasificadores con los datos existentes para cada dominio, se necesitan nuevos datos para realizar más pruebas con el fin de medir la efectividad de esta propuesta en términos de precisión, exhaustividad y F_1 .

En el tercer sistema, para la clasificación de relaciones de hiperonimia, se realiza el entrenamiento de una red neuronal usando la diferencia de vectores de palabras como única característica. Posteriormente, se realizan predicciones con este modelo respecto a un gold estándar, el cual se define como las anotaciones hechas por expertos en los dominios de estas ontologías, con el fin de medir la fidelidad de las predicciones obtenidas. El desempeño de este modelo es medio, ya que ronda el 60 % para las predicciones de la ontología de IA y tiene aproximadamente 42 % para la ontología de SCORM. Esto puede deberse a que los embeddings utilizados no fueron entrenados con corpus de dominio, sino mediante datos de *Common Crawl*¹. Es importante destacar que el conjunto de datos con el que fue entrenado la red, tampoco es de dominio sino que se trata de un conjunto de relaciones de hiperonimia muy generales; es decir, el conjunto de entrenamiento no es representativo respecto al conjunto de datos. Asimismo, Yildirim y Yıldız [72] menciona que el defecto más importante de usar la diferencia de los vectores de palabras es que se decide de manera pobre las instancias de un hiperónimo, si el conjunto de entrenamiento no incluye estas instancias. También debe considerarse el desbalance de clases.

¹<https://commoncrawl.org/>

Desde una perspectiva general, el sistema propuesto en este trabajo puede ser útil para añadir instancias a una ontología de dominio, pues la mayoría de las relaciones obtenidas son instancias. Asimismo, esta propuesta es conveniente para extraer relaciones taxonómicas en corpus considerables, ya que permite aprovechar los recursos de cómputo disponibles mediante multiprocesamiento. Debe considerarse que mediante la primer fase del sistema de extracción automática de relaciones taxonómicas, se pueden obtener nuevos datos que permitan continuar el entrenamiento del clasificador de relaciones de hiperonimia con el fin de volverlo más robusto.

Como trabajo a futuro, se están estudiando técnicas que permitan encontrar una mayor cantidad de relaciones taxonómicas en corpus de dominio como el análisis formal de conceptos (FCA) o bien análisis de frases. Para la representación de la jerarquía propuesta, se deben considerar otros métodos para llevar a cabo esta tarea; en particular, se desean expresar las relaciones obtenidas mediante RDF. Con respecto al clasificador de relaciones taxonómicas, se necesitan reunir nuevos datos para probar su rendimiento y se contempla la posibilidad de usar técnicas de aprendizaje profundo para conservar la semántica del texto y obtener una mejor representación del mismo; se tiene especial interés en BERT [14] pues permite generar embeddings informados del contexto, por lo que captura mejor la información contenida en el texto. En cuanto al clasificador de relaciones de hiperonimia, deben buscarse mejores representaciones vectoriales para obtener las características utilizadas para el entrenamiento del clasificador o bien continuar el entrenamiento de embeddings pre-entrenados usando corpus de dominio adecuados. Desde otra perspectiva, puede entrenarse otro modelo con el conjunto de datos léxico, con el fin de evitar la memorización léxica, ya que puede conducir a puntuaciones artificialmente altas.

Bibliografía

- [1] *Reflections on Language*. New York: Pantheon, 1975.
- [2] Charu C. Aggarwal. *Cluster Analysis*, págs. 153–204. Springer International Publishing, Cham, 2015. ISBN 978-3-319-14142-8. doi:10.1007/978-3-319-14142-8_6. URL https://doi.org/10.1007/978-3-319-14142-8_6.
- [3] Charu C. Aggarwal. *An Introduction to Data Mining*, págs. 1–26. Springer International Publishing, Cham, 2015. ISBN 978-3-319-14142-8. doi:10.1007/978-3-319-14142-8_1. URL https://doi.org/10.1007/978-3-319-14142-8_1.
- [4] Charu C. Aggarwal y ChengXiang Zhai. *An Introduction to Text Mining*, págs. 1–10. Springer US, Boston, MA, 2012. ISBN 978-1-4614-3223-4. doi:10.1007/978-1-4614-3223-4_1. URL https://doi.org/10.1007/978-1-4614-3223-4_1.
- [5] Steven Bird, Ewan Klein, y Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, 2009.
- [6] P. Buitelaar y P. Cimiano. *Ontology Learning and Population: Bridging the Gap Between Text and Knowledge*. Frontiers in artificial intelligence and applications. Ontology learning and population. IOS Press, 2008. ISBN 9781586038182. URL https://books.google.com.mx/books?id=wcBSjk2a2_sC.
- [7] Paul Buitelaar, Philipp Cimiano, y Bernardo Magnini. *Ontology learning from text: methods, evaluation and applications*, tomo 123. IOS press, 2005.

-
- [8] Francesco Camastra y Alessandro Vinciarelli. *Machine learning for audio, image and video analysis: theory and applications*. Springer, 2015.
 - [9] Hugo Raziel Lasserre Chavez y Mireya Tovar. Proposal for automatic extraction of taxonomic relations in domain corpus. *Research in Computing Science*, 133:29–39, 2017.
 - [10] Noam Chomsky. *Syntactic Structures*. The Hague: Mouton, 1957.
 - [11] Philipp Cimiano, Aleksander Pivk, Lars Schmidt-Thieme, y Steffen Staab. Learning taxonomic relations from heterogeneous sources of evidence. 2005.
 - [12] Guillaume Cleuziou y Jose Moreno. Qassit at semeval-2016 task 13: On the integration of semantic vectors in pretopological spaces for lexical taxonomy acquisition. págs. 1315–1319. 2016. doi:10.18653/v1/S16-1205.
 - [13] Robert Dale, Hermann Moisl, y Harold Somers. *Handbook of natural language processing*. CRC Press, 2000.
 - [14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, y Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
 - [15] Beniamino Di Martino y Pasquale Cantiello. Automatic ontology extraction with text clustering. tomo 237, págs. 215–220. 2009. doi:10.1007/978-3-642-03214-1_22.
 - [16] Kristina Doing-Harris, Yarden Livnat, y Stephane Meystre. Automated concept and relationship extraction for the semi-automated ontology management (seam) system. *Journal of Biomedical Semantics*, 6, 2015. doi:10.1186/s13326-015-0011-7.
 - [17] Rodrigo Fernandes de Mello y Moacir Antonelli Ponti. *Machine learning: a practical approach on the statistical learning theory*. Springer, 2018.
 - [18] Juan Diego Gómez Fierros. *Poblado automático de Ontologías Espaciales a Partir de Texto no Estructurado*. Maestría, Centro Nacional de Investigación y Desarrollo Tecnológico, Departamento de Ciencias Computacionales, 2012.

- [19] Alexander Gelbukh. Procesamiento de lenguaje natural y sus aplicaciones. *Komputer Sapiens*, "1":6–11, 2010.
- [20] A. Géron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2017. ISBN 9781491962268. URL <https://books.google.com.mx/books?id=khpYDgAAQBAJ>.
- [21] Emiliano Giovannetti, Simone Marchi, y Simonetta Montemagni. Combining statistical techniques and lexico-syntactic patterns for semantic relations extraction from text. 2008.
- [22] Nicola Guarino y Christopher Welty. Ontological analysis of taxonomic relationships. En *International Conference on Conceptual Modeling*, págs. 210–224. Springer, 2000.
- [23] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. En *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*. 1992. URL <https://www.aclweb.org/anthology/C92-2082>.
- [24] Iván Neftalí Ríos Hernández. El lenguaje: herramienta de reconstrucción del pensamiento. En *Razón y Palabra*". 2010.
- [25] Myriam Beatriz Hernández y José M. Gómez. Aplicaciones de procesamiento de lenguaje natural. *Revista Politécnica*, 32, 2013. URL https://revistapolitecnica.epn.edu.ec/ojs2/index.php/revista_politecnica2/article/view/32.
- [26] Matthew Honnibal y Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing, 2017. To appear.
- [27] Taeho Jo. *Introduction*, págs. 3–17. Springer International Publishing, Cham, 2019. ISBN 978-3-319-91815-0. doi:10.1007/978-3-319-91815-0_1. URL https://doi.org/10.1007/978-3-319-91815-0_1.

-
- [28] Yang Li y Tao Yang. *Word Embedding for Understanding Natural Language: A Survey*, págs. 83–104. Springer International Publishing, Cham, 2018. ISBN 978-3-319-53817-4. doi:10.1007/978-3-319-53817-4_4. URL https://doi.org/10.1007/978-3-319-53817-4_4.
- [29] Elizabeth D. Liddy. *Encyclopedia of library and information science*. 2001.
- [30] Ling Liu y M Tamer Özsu. *Encyclopedia of database systems*, tomo 6. Springer New York, NY, USA:, 2009.
- [31] Sébastien Ferré y Dickson Lukose Madalina Croitoru. Conceptual structures: From information to intelligence. *Lect. Note Comput. Sci*, 9:26–30, 2010.
- [32] Alexander Maedche, Steffen Staab, y Johanna Völker. *Ontology Learning*, págs. 245–267. 2009. doi:10.1007/978-3-540-92673-3_11.
- [33] Nesma Mahmoud, Heba Elbeh, y Hatem Abd elkader. Ontology learning based on word embeddings for text big data extraction. págs. 183–188, 2018. doi:10.1109/ICENCO.2018.8636154.
- [34] Diana Man. Ontologies in computer science. *Didactica Mathematica*, 31:43–46, 2013.
- [35] Alejandra Mathey. La comunicaci3n el lenguaje y el habla. s.f.
- [36] Diana Maynard, Adam Funk, Wim Peters, y Regent Court. Using lexico-syntactic ontology design patterns for ontology creation and population. *CEUR Workshop Proceedings*, 516:39–52, 2009.
- [37] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhersch, y Armand Joulin. Advances in pre-training distributed word representations. 2017.
- [38] E. Méndez y José A. Moreiro González. Lenguaje natural e indización automatizada. *Ciencias de la Información*, 30(3):11–24, 1999.
- [39] Claire Nédellec, Adeline Nazarenko, y Robert Bossy. *Information Extraction*, págs. 663–685. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-92673-3. doi:10.1007/978-3-540-92673-3_30. URL https://doi.org/10.1007/978-3-540-92673-3_30.

-
- [40] Ángel Cano. Comunicación y lenguaje. 2006. URL <https://previa.uclm.es/profesorado/agcano/comunicacion.htm>. [En línea].
- [41] Ángel Cano. sky-definition of language and grammar. 2015. URL <http://www.linguisticsplanet.com/chomsky-definition-of-language-and-grammar>. [En línea].
- [42] Andreia Dal Ponte Novelli y José Maria Parente de Oliveira. Simple method for ontology automatic extraction from documents. *International Journal of Advanced Computer Science and Applications*, 3(12), 2012. doi:10.14569/IJACSA.2012.031206. URL <http://dx.doi.org/10.14569/IJACSA.2012.031206>.
- [43] Mariano Fernández-López y Asunción Gómez-Pérez Oscar Corcho. Ontological engineering: principles, methods, tools and languages. En *Ontologies for software engineering and software technology*, págs. 1–48. Springer, 2006.
- [44] Joel Pocostales. NUIG-UNLP at SemEval-2016 task 13: A simple word embedding-based approach for taxonomy extraction. En *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, págs. 1298–1302. Association for Computational Linguistics, San Diego, California, 2016. doi:10.18653/v1/S16-1202. URL <https://www.aclweb.org/anthology/S16-1202>.
- [45] Daniel Francisco Polanco Fernández. Evaluación y mejora de un sistema automático de análisis sintagmático. 2000.
- [46] Longhua Qian, Guodong Zhou, Fang Kong, Qiaoming Zhu, y Peide Qian. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. En *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1, COLING '08*, págs. 697–704. Association for Computational Linguistics, USA, 2008. ISBN 9781905593446.
- [47] Joe Raad y Christophe Cruz. A survey on ontology evaluation methods. En *Proceedings of the International Conference on Knowledge Engineering and Ontology Development, part of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. 2015.

-
- [48] Karthik Ramasubramanian y Abhishek Singh. *Machine Learning Theory and Practice*, págs. 253–481. Apress, Berkeley, CA, 2019. ISBN 978-1-4842-4215-5. doi:10.1007/978-1-4842-4215-5_6. URL https://doi.org/10.1007/978-1-4842-4215-5_6.
- [49] Marta Sabou y Miriam Fernández. *Ontology (Network) Evaluation*, págs. 193–212. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-24794-1. doi:10.1007/978-3-642-24794-1_9. URL https://doi.org/10.1007/978-3-642-24794-1_9.
- [50] Saeed Sarencheh y Andrea Schiffauerova. Automatic algorithm for extracting an ontology for a specific domain name. págs. 49–56. 2017. doi:10.5220/0006500400490056.
- [51] Rajdeep Sarkar, John P. McCrae, y Paul Buitelaar. A supervised approach to taxonomy extraction using word embeddings. En *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*. European Languages Resources Association (ELRA), Miyazaki, Japan, 2018. URL <https://www.aclweb.org/anthology/L18-1324>.
- [52] Yatian Shen y Xuanjing Huang. Attention-based convolutional neural network for semantic relation extraction. En *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, págs. 2526–2536. The COLING 2016 Organizing Committee, Osaka, Japan, 2016. URL <https://www.aclweb.org/anthology/C16-1238>.
- [53] Vered Shwartz, Yoav Goldberg, y Ido Dagan. Improving hypernymy detection with an integrated path-based and distributional method. En *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, págs. 2389–2398. Association for Computational Linguistics, Berlin, Germany, 2016. doi:10.18653/v1/P16-1226. URL <https://www.aclweb.org/anthology/P16-1226>.
- [54] Ayush Singhal, Michael Simmons, y Zhiyong Lu. Text mining for precision medicine: automating disease-mutation relationship extraction from biomedical

- literature. *Journal of the American Medical Informatics Association*, 23(4):766–772, 2016. ISSN 1067-5027. doi:10.1093/jamia/ocw041. URL <https://doi.org/10.1093/jamia/ocw041>.
- [55] María Josefa Somodevilla García, Darnes Vilariño Ayala, y Ivo Pineda. An Overview of Ontology Learning Tasks. *Computación y Sistemas*, 22:137 – 146, 2018. ISSN 1405-5546. URL http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-55462018000100137&nrm=iso.
- [56] Bruno López Takeyas. Inteligencia artificial. 2005. URL [http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas_alumnos/PLN/PLN\(2005-II\).pdf](http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas_alumnos/PLN/PLN(2005-II).pdf).
- [57] Mireya Tovar. *Evaluación Automática de Ontologías de Dominio Restringido*. Tesis doctoral, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, Morelos, México, s.f.
- [58] Mireya Tovar, Gerardo Flores, José Reyes, y Meliza Contreras. *Validation of Semantic Relation of Synonymy in Domain Ontologies Using Lexico-Syntactic Patterns and Acronyms*, págs. 199–208. 2018. ISBN 978-3-319-92197-6. doi:10.1007/978-3-319-92198-3_20.
- [59] Mireya Tovar, David Pinto, Azucena Rendón, y Juan Gabriel Sernal. An approach based in lsa for evaluation of ontological relations on domain corpora. págs. 225–233. 2017. ISBN 978-3-319-59225-1. doi:10.1007/978-3-319-59226-8_22.
- [60] Mireya Tovar, David Pinto, Azucena Rendón, y Juan Gabriel Sernal. A metric for the evaluation of restricted domain ontologies. *Computación y Sistemas*, 22:147 – 162, 2018. doi:10.13053/cys-22-1-2792.
- [61] Mireya Tovar, David Pinto, Azucena Rendón, Juan Gabriel Sernal, y Darnes Vilariño. Identification of ontological relations in domain corpus using formal concept analysis. *Engineering Letters*, 23:72–76, 2015.

- [62] Mireya Tovar, David Pinto, Azucena Rendón, Juan Gabriel Sernal, y Darnes Vilariño. Patterns used to identify relations in corpus using formal concept analysis. págs. 236–245. 2015. ISBN 978-3-319-19263-5. doi:10.1007/978-3-319-19264-2_23.
- [63] Mireya Tovar, David Pinto, Azucena Rendón, Juan Gabriel Sernal, Darnes Vilariño, y Beatriz Beltrán. Use of lexico-syntactic patterns for the evaluation of taxonomic relations. tomo 8495, págs. 331–340. 2014. doi:10.1007/978-3-319-07491-7_34.
- [64] Guido Van Rossum et al. Python programming language. En *USENIX annual technical conference*, tomo 41, pág. 36. 2007.
- [65] Mireya Tovar Vidal, Emmanuel Santos Rodríguez, y José A. Reyes-Ortiz. Machine learning for the identification and classification of key phrases from clinical documents in spanish. En Kohei Arai, Rahul Bhatia, y Supriya Kapoor, eds., *Proceedings of the Future Technologies Conference (FTC) 2019*, págs. 164–174. Springer International Publishing, Cham, 2020. ISBN 978-3-030-32520-6.
- [66] Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojciech M Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, et al. Alphastar: Mastering the real-time strategy game starcraft ii. *DeepMind blog*, pág. 2, 2019.
- [67] Denny Vrandečić. *Ontology Evaluation*, págs. 293–313. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-92673-3. doi:10.1007/978-3-540-92673-3_13. URL https://doi.org/10.1007/978-3-540-92673-3_13.
- [68] Kun Xu, Yansong Feng, Songfang Huang, y Dongyan Zhao. Semantic relation classification via convolutional neural networks with simple negative sampling. 2015.
- [69] Mari Carmen Suárez-Figueroa y Asunción Gómez-Pérez. First attempt towards a standard glossary of ontology engineering terminology. En *Proceedings of the*

- 8th International Conference on Terminology and Knowledge Engineering*, págs. 1–16. 2008.
- [70] Augusto Cortez Vázquez Hugo Vega Huerta y Jaime Pariona Quispe. Procesamiento de lenguaje natural. *Revista de investigación de Sistemas e Informática*, 2009.
- [71] Steffen Staab y Rudi Studer. *Handbook on ontologies*. Springer Science & Business Media, 2010.
- [72] Savaş Yıldırım y Tuğba Yıldız. Learning turkish hypernymy using word embeddings. *International Journal of Computational Intelligence Systems*, 11(1):371–383, 2018.
- [73] Yong Yin, Ikou Kaku, Jiafu Tang, y JianMing Zhu. *Decision Analysis and Cluster Analysis*, págs. 1–8. Springer London, London, 2011. ISBN 978-1-84996-338-1. doi:10.1007/978-1-84996-338-1_1. URL https://doi.org/10.1007/978-1-84996-338-1_1.
- [74] Amal Zouaq, Dragan Gasevic, y Marek Hatala. Linguistic patterns for information extraction in ontocmaps. En *Proceedings of the 3rd International Conference on Ontology Patterns-Volume 929*, págs. 61–72. CEUR-WS. org, 2012.