UNIVERSIDAD DE LOS ANDES
FACULTAD DE INGENIERÍA
DOCTORADO EN CIENCIAS APLICADAS
MÉRIDA – VENEZUELA

ESTRATEGIAS BASADAS EN LA TÉCNICA DE DATOS ENLAZADOS PARA LA GENERACIÓN DE CONOCIMIENTO EN AMBIENTES INTELIGENTES

MSc. Ricardo José Dos Santos Guillén

Tutor: Dr. Jose Aguilar
Universidad de Los Andes
Mérida – Venezuela

Julio 2025

Resumen

Este trabajo doctoral se adentra en la generación y explotación de conocimiento dentro de Ambientes Inteligentes, abordando la inherente falta de estructura semántica en la Web. Para ello, se propone un enfoque central basado en el paradigma de Datos Enlazados, que permite interconectar y mejorar la comprensión del significado de los datos en Internet, tanto para humanos como para máquinas, mediante el uso de estándares y enlaces entre recursos. La explotación inteligente de este conocimiento en un Ambiente Inteligente se logra integrando mecanismos avanzados como el Aprendizaje Automático para construir modelos predictivos o de clasificación, incluyendo técnicas como los Autocodificadores Variacionales para la generación de datos sintéticos y Redes Neuronales Convolucionales para la extracción de características. Además, se incorpora el Meta-Aprendizaje para permitir que los sistemas de Aprendizaje Automático "aprendan a aprender" de experiencias previas, adaptando técnicas y gestionando metadatos como Meta-Dataset, Meta-Características, Meta-Técnicas y Meta-Modelos. Crucialmente, la Lógica Dialéctica se emplea para resolver situaciones de contradicción o ambigüedad, gestionando estados como la vaguedad, declaraciones contingentes sobre el futuro, fallos de presuposición, discurso ficticio y razonamiento contrafáctico. La investigación presenta arquitecturas computacionales innovadoras: un Sistema de Recomendación Híbrido que combina lógica descriptiva y dialéctica con Datos Enlazados para gestionar información inconsistente y explotar la semántica web, una extensión del middleware MiSCi con una capa de Datos Enlazados para el enriquecimiento y la explotación semántica en tiempo real en ciudades inteligentes, y un Sistema de Generación Automática de Ontologías que crea y enriquece ontologías emergentes de forma autónoma utilizando Datos Enlazados. Finalmente, la arquitectura de Meta-Aprendizaje fue extendida para la creación automática de modelos de conocimiento apoyado en ciclos autónomos para tareas de análisis de datos, facilitando su adaptación rápida a nuevos escenarios en contextos como la automatización de cadenas de producción agroindustrial, mediante un ciclo autónomo que supervisa la ejecución de sus módulos a través de tareas de Observación, Análisis y Decisión.

Palabras Claves: Datos Enlazados, Meta-Aprendizaje, Lógica Dialéctica, Aprendizaje Automático, Ambientes Inteligentes

Índice General

Índice de Figuras

Índice de Tablas

1 Introducción

1.1 Planteamiento del Problema

Los nuevos avances en las tecnologías de la información y la evolución de la World Wide Web, han transformado a la actual Web en un gran repositorio de documentos de distintos tipos, como imágenes, texto, entre otros. La Web presenta un gran problema en su uso como repositorio de información, por la falta de una estructura semántica que permita interpretar el contenido de la mayoría de la información contenida en la Web. En ese sentido, uno de los primeros esfuerzos que se ha hecho para explotar el gran contenido de información en ella, es su modelado usando ontologías, desarrollándose toda un área dedicada a estos temas conocida como la *Web Semántica* [1, 2]. Una extensión reciente de dicha área es el paradigma llamado *Datos Enlazados* o Vinculados (en inglés, *Linked Data*), el cual permite vincular un conjunto de datos publicados en la Internet (en este caso, conceptos o cosas) usando los mismos mecanismos de las páginas web, como las URL (en inglés Uniform Resource Locator) [3]. Los *Datos Enlazados* es sinónimo de datos semánticamente interconectados, lo que permite la interoperabilidad entre ellos, a pesar de que sean heterogéneos y estén distribuidos eventualmente en diferentes repositorios. En [4] se refieren a los *Datos Enlazados* como “los datos publicados en la web, de manera tal que sea legible por las máquinas. Por otro lado, sus significados explícitamente están vinculadas a otros conjuntos de datos”.

En particular, nosotros estamos interesados en usar el paradigma de *Datos Enlazados* para manejar y explotar el conocimiento generado en un ***Am****biente* ***I****nteligente* (AmI), sabiendo que un AmI es un “paradigma en el cual las personas están potenciadas o fortalecidas por el uso de entornos digitales que son conscientes de su presencia y su contexto, que son sensibles, adaptativos, y responden a sus necesidades, hábitos, gestos y emociones” [5, 6]. A su vez, en [7] lo definen como “los mecanismos que gobiernan los componentes de un entorno, siendo sensible a las demandas del usuario, aprendiendo o conociendo sus preferencias, para poder reaccionar de forma personalizada y consciente del contexto”. Sin embargo, la explotación inteligente del conocimiento en un AmI, implica extraer, transformar, filtrar y relacionar datos e información desde diferentes fuentes. En ese sentido, poder usar el método de publicación e interconexión de datos como lo prevé el paradigma de *Datos Enlazados*, permite el enriquecimiento cognitivo de un AmI con diferentes fuentes de información.

Ahora bien, los *Datos Enlazados* requieren de otros mecanismos para la generación de conocimiento desde diversas fuentes de datos e información. En específico, requieren de técnicas provenientes de las áreas del Meta-Aprendizaje, Aprendizaje Automático y Lógica Dialéctica, entre otras, para que conjuntamente se genere conocimiento para un AmI desde sus fuentes de información. Para automatizar las tareas de extracción y explotación de conocimiento, es de suma importancia las técnicas que provee el A*prendizaje Automático,* potenciado con el *Meta-Aprendizaje*. El *Aprendizaje Automático* se dedica a desarrollar algoritmos y sistemas que permiten crear modelos de conocimiento (de predicción, clasificación, diagnóstico, entre otros) a través de los datos o experiencias [8], los

cuales luego puedan ser usados para responder a las distintas necesidades que tenga el AmI. *El Meta-Aprendizaje* permite aprender a aprender, permitiendo a los diferentes procesos del Aprendizaje Automático a adecuar sus técnicas de aprendizaje basado en el conocimiento previo y en los requerimientos de los problemas a resolver [9]. Por último, la *Lógica Dialéctica* posee la capacidad de resolver situaciones de contradicción o ambigüedad, así mismo, también sirve para la toma de decisiones en contextos donde las presuposiciones fallan (por ejemplo, en tareas de diagnóstico o detección de fallas), con contingencias sobre el futuro (clave para el manejo de datos históricos, que indican que algo fue verdad y falso en el pasado), manejar situaciones contrafácticas que describen algo que podría haber ocurrido de la forma "Si A no hubiera ocurrido, C no habría ocurrido" (eso ayuda a los sistemas a aprender de los errores para hacer los correctivos en un futuro), o con un discurso ficticio (tomar decisiones bajo supuestos imaginarios) [10].

En particular, se hace necesario crear una Arquitectura Computacional que posea las capacidades de integrar los mecanismos para la generación de conocimiento con los *Datos Enlazados*, entendiendo a una Arquitectura Computacional como “la organización fundamental de un sistema definido por sus componentes, sus relaciones entre sí y con el medio ambiente, y los principios que guían su diseño y evolución” [11]. Así, esta tesis se enfoca en definir estrategias basadas en *Datos Enlazados*, en el contexto específico de los AmI, integrando mecanismos de Aprendizaje Automático, Meta-Aprendizaje y Lógica Dialéctica, para responder a las siguientes incógnitas:

- ¿Cómo se puede explotar el método de publicación de datos estructurados de los *Datos Enlazados*, para generar conocimiento útil en un AmI?
- ¿Cómo se integran los conceptos/paradigmas de Aprendizaje Automático, Meta-Aprendizaje y Lógica Dialéctica con *Datos Enlazados*, para la extracción autonómica y explotación del conocimiento en un AmI?
- ¿Cómo las herramientas y técnicas existentes alrededor de los conceptos/paradigmas de Aprendizaje Automático (por ejemplo, Tensorflow, Scikit-Learn, Keras), Lógica Dialéctica (por ejemplo, JGXYZ-RM3 con los razonadores Vampire y Eprover) y Datos Enlazados (por ejemplo, DBpedia, OpenLink Virtuoso), se pueden integrar para la gestión del conocimiento en un AmI?
- ¿Cómo se garantiza la interoperabilidad, la integración, la escalabilidad y la flexibilidad, de un entorno computacional de generación de conocimiento basado en *Datos Enlazados* para un AmI?

1.2 Objetivos

1.2.1 Objetivo General

Definir una arquitectura computacional que permita la generación de conocimiento usando la técnica de *Datos Enlazados* para Ambientes Inteligentes

1.2.2 Objetivos Específicos

- Estudiar aspectos teóricos y prácticos, y en general, el estado de arte, relacionados con los *Datos Enlazados*, Inteligencia Ambiental, Lógica Dialéctica, Aprendizaje Automático y Meta-Aprendizaje.
- Caracterizar los mecanismos de extracción y procesamiento de información en un AmI, desde la perspectiva de los *Datos Enlazados*.
- Especificar una arquitectura computacional que integre los diferentes mecanismos de Lógica Dialéctica, Aprendizaje Automático y Meta-Aprendizaje, requeridos por las estrategias basadas en *Datos Enlazados,* para la generación de conocimiento en un AmI.
- Desarrollar un conjunto de servicios de generación de conocimiento para un AmI usando la arquitectura computacional basada en *Datos Enlazados*, integrada con Lógica Dialéctica, Aprendizaje Automático y Meta-Aprendizaje.
- Elaborar un prototipo de la arquitectura computacional especificada, y realizar casos de estudio en un AmI específico.

1.3 Antecedentes

La investigación y revisión de los antecedentes, se enfocaron en los siguientes ámbitos: el uso de los *Datos Enlazados* para la generación de conocimiento, el uso de los *Datos Enlazados* en AmI, y la integración de los *Datos Enlazados* con Lógica Dialéctica, Aprendizaje Automático y Meta-Aprendizaje. A continuación, presentamos los trabajos más resaltantes recientes en cada uno de esos ámbitos.

En relación con el uso de los *Datos Enlazados* para la generación de conocimiento, tenemos los siguientes trabajos. En la revisión sistemática realizada en [12], categorizan y analizan una amplia gama de herramientas basadas en Datos Enlazados, específicamente, en tareas de: i. Extracción de Conocimiento: Transforman texto no estructurado o semiestructurado en formatos estructurados mediante el reconocimiento y la vinculación de entidades (Named Entity Recognition and Linking) y el enriquecimiento semántico; ii. Visualización y Exploración de Grafos de Conocimiento (Knowledge Graphs, KGs): Permite a los usuarios comprender las relaciones intrincadas a través de paradigmas de interacción como vistas tabulares (tripletes), nodo-enlace (grafos) y composición visual de consultas (consultas SPARQL[1]); iii. Reducción de Complejidad: métodos que evitan la sobrecarga de información, empleando estrategias como la visualización navegacional (centrada en un objeto y su entorno inmediato), la visualización incremental (permitiendo a los usuarios controlar un espacio de trabajo para añadir o eliminar vistas de objetos de datos dinámicamente), y la visualización resumida

[1] SPARQL (Query Language for RDF): https://www.w3.org/TR/sparql11-query/

(generando resúmenes de grafos para proporcionar una visión general concisa de un conjunto de datos grande).

Otro trabajo interesante es presentado Pham-Hang y otros [13], el cual describe el desarrollo de un novedoso sistema para compartir información sobre seguridad basado en Datos Enlazados. El sistema mejora el intercambio y la reutilización de datos pasados usando tres módulos principales. El Módulo de Ontología, que formaliza el conocimiento de los accidentes utilizando la metodología Linked Open Terms (LOT) y lenguajes como OWL[2], RDF[3] y RDFS[4] para estructurar la información de manera consistente. El Módulo de Procesamiento RDF, que se encarga de la conversión automática de datos existentes y nuevos a formato RDF, empleando librerías como RDFLib[5] y KGLAB[6] para manejar datos como grafos y tripletas (sujeto, predicado, objeto), que luego se almacenan en un RDF store (almacén de triples o almacén RDF,). Finalmente, el Módulo de Consulta permite la recuperación de información utilizando el protocolo SPARQL, el estándar para consultar Datos Enlazados abiertos y conjuntos de datos RDF (triplestores). Además, usand KGLAB para visualizaciones basadas en grafos que muestran las relaciones entre los elementos de los datos. De manera similar en [14], Thalahth y otros generan RDFs enriquecidos con información proveniente de Wikidata[7]. Para ello, previamente concilian los conjuntos de datos (por ejemplo, nombres de ciudades y países) con URIs (Uniform Resource Identifier) de Wikidata.

En cuanto al uso de los *Datos Enlazados* en AmI, un primer trabajo es [15], donde se presenta una revisión sistemática de técnicas avanzadas de inteligencia vestible multimodal aplicadas al cuidado de la demencia. Este enfoque permite un monitoreo no intrusivo del contexto mediante el uso de dispositivos como lentes inteligentes, pulseras inteligentes, registradores tipo clip y teléfonos inteligentes. Una de las tecnologías claves es el empleo de repositorios de datos semánticos. Estos repositorios permiten la búsqueda y el procesamiento de Datos Enlazados en el contexto de vida saludables, para reusar los recursos disponibles en internet. La integración de los Datos Enlazados a través de repositorios semánticos y endpoints SPARQL (como CardioSHARE[8] y Bio2RDF[9]) permite acceder y consultar una vasta red de conocimiento distribuido y heterogéneo. Esta capacidad es fundamental para proporcionar servicios de salud personalizados e inteligentes a personas que viven con demencia.

En el trabajo [16] Favarato y otros presentan un estudio sobre el uso de los Datos Enlazados como mecanismos de gestión de los datos en un AmI, permitiendo combinar una vasta cantidad de información de diversas fuentes para obtener una comprensión integral de cómo la contaminación del aire y las condiciones de la vivienda contribuyen a las hospitalizaciones por infecciones del tracto

[2] OWL (Web Ontology Language): https://www.w3.org/OWL/
[3] RDF (Resource Description Framework): https://www.w3.org/RDF/
[4] RDFS (RDF Schema): https://www.w3.org/TR/rdf-schema/
[5] RDFLib (RDF Library): https://rdflib.readthedocs.io/
[6] KGLAB (Knowledge Graph Laboratory): https://derwen.ai/docs/kgl/
[7] https://www.wikidata.org/
[8] https://code.google.com/archive/p/cardioshare/
[9] https://bio2rdf.org/

respiratorio en niños pequeños. Para ello, se unifican los datos aprovechando el paradigma de los Datos Enlazados, dónde se mapea los datos Registros del Censo Nacional (nombres, códigos postales y fechas de nacimiento) con los identificadores del Servicio Nacional de Salud (registros de salud). Además, se mapea los registros postales del Servicio de Datos Demográficos Personales con el código postal de los datos de Certificados de Rendimiento Energético y de exposición a la contaminación del aire. Por otro lado, en [17] describen un enfoque para la integración de sistemas heterogéneos mediante el uso de puntos de soporte semántico basado en Datos Enlazados. Estos puntos son microservicios ligeros que permiten la codificación de conocimiento sobre la marcha, transformando datos de sistemas existentes a un formato semántico (como RDF) y viceversa. La meta era crear una capa semántica no persistente para facilitar la interoperabilidad, consulta y razonamiento para sistemas inteligentes, como los sistemas multiagente, incluso en entornos originalmente no semánticos.

Finalmente, en cuanto a la integración de los *Datos Enlazados* con Lógica Dialéctica, Aprendizaje Automático y Meta-Aprendizaje, tenemos los siguientes trabajos. En [18, 19, 20, 21] se detalla el framework DL-Learner, que implementa varios algoritmos de Aprendizaje Automático Supervisado para construir clasificadores, usando archivos en OWL y RDF como entrada. El objetivo de DL-Learner es proporcionar un framework con componentes reutilizables para resolver distintos problemas, usando una variedad de fuentes de conocimiento que juntas forman el conocimiento de base para una tarea dada. Dentro de ese conocimiento se pueden seleccionar casos positivos y negativos, para ser procesados por un algoritmo de Aprendizaje Automático para generar un clasificador. Por otro lado, se describe el problema de aprendizaje, y se especifica el algoritmo que se desea utilizar para resolverlo.

Por otro lado, en el trabajo [22] se explora el Meta-Aprendizaje con el fin de mejorar la eficiencia y la eficacia de los modelos de Aprendizaje Automático basados en el conocimiento y la experiencia previos. Ese trabajo propone una arquitectura de Meta-Aprendizaje compuesta por tres módulos. El primer módulo está representado por las Meta-Características, que ayudan a caracterizar los conjuntos de datos de acuerdo con el rendimiento de los algoritmos de Aprendizaje Automático en diferentes tareas. El segundo módulo está compuesto por el Meta-Aprendiz, que son algoritmos que aprenden de las Meta-Características y los datos sobre el rendimiento de tareas anteriores. Utilizan esta información para seleccionar o construir modelos de conocimiento adecuados y ajustar sus hiperparámetros para nuevas tareas. Finalmente, los Meta-Dataset son colecciones de metadatos, es decir, que son datos sobre los datos. Estos se construyen a partir de los metadatos resultantes de otras tareas de aprendizaje y proporcionan una rica fuente de información para alimentar al Meta-Aprendiz. De la misma manera, en [23] se aborda el desafío de construir modelos Aprendizaje Automático usando Meta-Aprendizaje para automatizar la selección de algoritmos y el ajuste de hiperparámetros. En este trabajo se recopila información tanto sobre las características de los conjuntos de datos (Meta-Características) como de los modelos generados (Meta-Modelos), y con esta información, el algoritmo Meta-Aprendizaje realiza recomendaciones de configuraciones para la generación de nuevos modelos.

Como se puede constatar, en la literatura reciente no hay trabajos previos que combinen los *Datos Enlazados* con Lógica Dialéctica, ni que integren los *Datos Enlazados* con Aprendizaje Automático y Meta-Aprendizaje. Los trabajos previos tocan aspectos específicos sobre los usos de los *Datos Enlazados* con Aprendizaje Automático, sin la integración con Meta-Aprendizaje o con la Lógica Dialéctica. Además, aunque las aplicaciones basadas en Datos Enlazados han demostrado un enorme potencial en el ámbito de las AmIs, la realidad es que su integración ha permanecido, en gran medida, en la fase de propuesta teórica más que en propuestas implementadas. En ese sentido, este trabajo busca proponer una arquitectura computacional que explote las ventajas de los *Datos Enlazados* para generar conocimiento en un AmI, que permita la integración de los mecanismos/herramientas de Lógica Dialéctica, Aprendizaje Automático y Meta-Aprendizaje, para enriquecer ese proceso.

1.4 Organización de la Tesis

Esta tesis se estructura en seis capítulos, cada uno profundizando en aspectos fundamentales para la generación de conocimiento en AmI mediante Datos Enlazados, y la integración de Aprendizaje Automático, Meta-Aprendizaje y Lógica Dialéctica. En el capítulo 1 se describe el planteamiento del problema y su importancia, los objetivos de la investigación y los antecedentes relacionados con los enfoques asociados a las áreas de los Datos Enlazados, Lógica Dialéctica, Aprendizaje Automático y Meta-Aprendizaje. En el capítulo 2 se detallan los aspectos teóricos relacionados con las distintas áreas cubiertas en la tesis. En el capítulo 3 se presentan las arquitecturas de gestión de conocimiento basadas en Datos Enlazados, específicamente, una ampliación del middleware MiSCi (Middleware for Smart Cities) con una capa de Datos Enlazados y una arquitectura para la generación automática y enriquecimiento de ontologías emergentes, ambas fundamentadas en la metodología MEDAWEDE (Metodología para el Desarrollo de Aplicaciones Web utilizando Datos Enlazados). En el capítulo 4 se presenta un Sistema de Recomendación Híbrido que integra lógica descriptiva/dialéctica con Datos Enlazados, detallando su arquitectura, funcionamiento y aplicaciones en contextos con información inconsistente o ambigua, como el diagnóstico médico y el análisis de competencias profesionales. En el capítulo 5 se describe una arquitectura de Meta-Aprendizaje para la generación de modelos de Aprendizaje Automático basada en Datos Enlazados, incluyendo una ampliación con un Meta-Algoritmo Autónomo que incorpora aprendizaje por transferencia y generación de datos sintéticos, así como módulos para la generación de características y datos artificiales, ilustrado con diversos casos de estudio. Por último, en el capítulo 6 se presentan las conclusiones del trabajo y los trabajos futuros.

2 Marco Teórico

En este capítulo se presenta una revisión general de los aspectos teóricos más importantes, que coadyuven a definir estrategias basadas en los *Datos Enlazados*, en el contexto específico de los *AmI*, los cuales requieren la integración de mecanismos de *Aprendizaje Automático, Meta-Aprendizaje* y *Lógica Dialéctica*, para la generación de conocimiento en un *AmI*. En ese sentido, todas esas áreas serán revisadas.

2.1 Datos Enlazados

Los *Datos Enlazados* describen una forma de publicar los datos en Internet para que se puedan interconectar entre ellos [24]. Particularmente, los *Datos Enlazados* es la manera que tiene la Web Semántica de enlazar un conjunto de datos que estén publicados en la Internet, para mejorar la comprensión de sus significados, tanto para los humanos como para las máquinas [3,4].

A.1.1 Principios de los Datos Enlazados

Tim Berners-Lee introduce los principios de los *Datos Enlazados*, los cuales son [3,4,25]: i) *Identidad:* Utilizar URIs para identificar los recursos en Internet (por ejemplo, páginas web, objetos abstractos, servicios, ficheros, etc.); ii) *Accesibilidad:* Usar URIs HTTP (Hypertext Transfer Protocol) para que las personas puedan buscar recursos; iii) *Estructura:* Utilizar estándares RDF para describir recursos, y SPARQL para realizar consultas; iv) *Navegación:* Incluir enlaces a otras URIs para descubrir más recursos.

El *primer principio* busca asignar un nombre único a las cosas o conceptos en la Internet, y para ello, usa el mecanismo de identificación única URIs para referirse a cualquier recurso. Un ejemplo que puede dilucidar la importancia de las URIs es la representación del concepto “País”. Si se quiere identificar al país “Venezuela” por su nombre, surge un gran dilema, ¿Qué nombre usar?, se podrían usar los siguientes nombres: Venezuela, República Bolivariana de Venezuela, VE, VEN, Bolivarian Republic of Venezuela, entre otros, es decir, se tendría problemas con los sinónimos, los sobrenombres y los idiomas. La solución a este problema consiste en usar el siguiente URI: http://dbpedia.org/resource/Venezuela, que identifica a Venezuela como país, sin importar idioma, diminutivo, etc.

El *segundo principio* hace hincapié en el uso de URIs basado en el protocolo HTTP, para permitir recuperar desde la Web toda la descripción del recurso identificado por el URI. En el siguiente ejemplo se muestra el interés de este principio: si se usa como URI el código ISO numérico asignado a Venezuela, que es 862, y se coloca en un navegador, no se obtiene una descripción del recurso. En

cambio, si se usa una URI con un HTTP como http://dbpedia.org/resource/Venezuela, se mostraría una página con la información sobre el recurso.

El *tercer principio* se basa en el formato y la calidad de la descripción de los recursos, para poder obtener información útil sobre dicho recurso. Para ello, dichas descripciones se deben materializar en forma de documentos Web. Los destinados a ser leídos por los seres humanos a menudo se representan como HTML, y los destinados al consumo de las máquinas se representan como datos RDF o XML (en inglés, e**X**tensible **M**arkup **L**anguage). Por ejemplo, si se solicita la descripción de la URI http://dbpedia.org/resource/Venezuela desde un navegador, automáticamente es redireccionado a la versión HTML ubicada en http://dbpedia.org/page/Venezuela. Ahora, si se solicita la misma URI indicando a través de una aplicación, automáticamente es redireccionado a http://dbpedia.org/data/Venezuela.xml. Lo anterior, lo podemos comprobar con la aplicación CURL de la siguiente manera:

Por defecto:
curl -I http://dbpedia.org/resource/Venezuela
Indicando en la cabecera que es para una aplicación:
curl -I -H "Accept: application/rdf+xml" http://dbpedia.org/resource/Venezuela

El *cuarto principio* fomenta la interconexión de recursos relacionados, la cual es necesaria para conectar los datos que se tienen de forma que no queden aislados. Como dichos recursos están publicados con URI con el protocolo HTTP, otras personas o aplicaciones pueden vincularlo a sus datos. Esta capacidad de seguir los vínculos permite a la gente navegar por la Web de datos, tal como pueden navegar por la Web de documentos. Por ejemplo, si se detalla el recurso http://dbpedia.org/resource/Venezuela, se observa que tienen muchos enlaces a otros recursos, se puede mencionar algunos enlaces como: dbo:currency, dbo:anthem, dbp:languages, entre otros. Al seguir el recurso con su URI dbr:Spanish_language enlazado a través de dbp:languages, se muestra toda la información sobre el idioma español, el cual a su vez tienen más enlaces, lo que indica que estamos en presencia de un grafo de recursos.

A.1.2 Modelado de los Datos Enlazados

Un punto muy importante en los Datos Enlazados es el modelado, porque especifica la manera de *representar el conocimiento*, así como también, los *lenguajes y los vocabularios u ontologías* necesarias para ese fin. En el caso de la *representación del conocimiento* para los Datos Enlazados, el esquema actualmente utilizado se fundamenta en las redes semánticas, ya que es una forma de representar el conocimiento por medio de conceptos y sus interrelaciones, bajo la forma de grafos [26, 27]. Los elementos básicos en todos los esquemas de redes son: i) Los nodos, que en este caso representan conceptos, unidos por arcos que representan las relaciones entre los conceptos. ii) Un conjunto de procedimientos de inferencia que operan sobre la red.

La representación mental de estas redes son las *ontologías*. El concepto de ontología en el ámbito tecnológico más ampliamente aceptada es la propuesta por Gruber (2008), “La ontología define un

conjunto de primitivas de representación con la que se puede modelar un dominio de conocimiento" [28], es decir, una ontología es un sistema de conceptos (o un vocabulario), usado como elemento básico (primitivo), para la construcción de sistemas basados en el conocimiento. En la ontología, una afirmación se representa como una tripleta que consta de tres elementos: un sujeto, un predicado y un objeto. El *sujeto* y el *objeto* representan a los dos conceptos o recursos a relacionar; el *predicado* representa la naturaleza de esta relación, formulada de manera direccional (del sujeto al objeto). Por ejemplo, en la tripleta *Venezuela es_un País*, *Venezuela* es el sujeto, *País* es el objeto y *es_un* es el predicado. En las tripletas RDF (RDF-Triple), el predicado es denominado "propiedad" [29]. Un objeto puede ser también un literal o valor de texto, lo cual permite definir una propiedad para un recurso.

En cuanto a los *lenguajes de modelado*, los actuales desarrollos en la representación del conocimiento están siendo influenciados por la Web Semántica, y han incorporado lenguajes y estándares de representación del conocimiento basados en XML, e incluyen a RDF, RDFSchema, y OWL [30,31]. El XML permite la definición de gramáticas y etiquetas para la información contenida en los documentos, pero tiene un problema importante, y es que aporta una estructura, pero no una semántica [32,31]. Por otro lado, el lenguaje RDF es más expresivo para el procesamiento semántico, ya que a través de los recursos, propiedades y sentencias (combinación de recursos y propiedades), permiten una representación explícita de la semántica de los datos. La W3C considera al lenguaje RDF como el estándar para describir recursos (cualquier concepto que tenga una URI) en la web [30]. A continuación, se muestra un ejemplo, en donde se observa la descomposición de la información descrita en una frase hasta obtener una representación de ese conocimiento en RDF.

> **Venezuela es un país que forma parte de América del Sur, y su idioma es el Español**

La frase escrita anteriormente se puede descomponer en tres tripletas, donde el sujeto es Venezuela; los predicados son **es_un**, **es_parte** y **tiene_idioma**; y los objetos son **País**, **América_del_Sur** y **Español**, dando como resultado lo siguiente:

> **Venezuela es_un País**
> **Venezuela es_parte América_del_Sur**
> **Venezuela tiene_idioma Español**

Sujeto y Propiedad se expresan con una URI, y el Objeto se expresa con URI si se relaciona a otro concepto, o como un Valor si define una propiedad de un recurso:

> **dbpedia:Venezuela rdf:type dbpedia-owl:Country .**
> **dbpedia:Venezuela dcterms:subject dbpedia-c:Países_de_América_del_Sur .**
> **dbpedia:Venezuela dbpedia-owl:spokenIn dbpedia:Idioma_español .**

Ahora, el aporte de RDF a la sintaxis todavía es muy superficial para representar el conocimiento, ya que solo proporciona mecanismos para expresar declaraciones simples sobre recursos, utilizando propiedades y valores. En RDFSchema se agrega la noción de clases y propiedades, tal que se pueden crear jerarquías de clases y propiedades. También, permite especificar el dominio y rango de una propiedad, es decir, indica los tipos de sujetos y objetos de una propiedad en la tripleta. A continuación,

se muestra un conjunto de tripletas usando las nuevas especificaciones del lenguaje RDFSchema, para enriquecer el conocimiento semántico de los recursos descrito en RDF:

```
dbpedia-owl:Country rdf:type owl:Class .
wikidata:Q1211934 rdf:subClassOf dbpedia-owl:Country .
wikidata:Q1211934 rdfs:label "Hispanos"@es .
dbpedia-owl:Language rdf:type owl:Class .
dbpedia-owl:spokenIn rdf:type rdf:Property .
dbpedia-owl:spokenIn rdfs:dominio dbpedia-owl:Country .
dbpedia-owl:spokenIn rdfs:range dbpedia-owl:Language .
```

En las tripletas mostradas se indica lo siguiente: con la propiedad **type** se define las clases **Country** y **Q1211934** (Hispanos según la propiedad **label**). También se indica que **spokenIn** es un tipo de propiedad; con la propiedad **subClassOf** se indica que **Q1211934** es una subclase de **Country**, es decir, se crea una jerarquía de clases; con ***dominio*** se indica los sujetos de tipo **Country** de la propiedad **spokenIn**; y con ***range*** se indica los objetos de tipo **Lenguaje** de la propiedad **spokenIn**. Gracias a estas tripletas, se puede inferir nuevos conocimientos, como los siguientes:

```
wikidata:Q1211934 rdf:type owl:Class .
dbpedia:Idioma_español rdf:type dbpedia-owl:Language .
dbpedia:Venezuela rdf:type dbpedia-owl:Country .
```

A pesar de las capacidades que ofrece RDFSchema, aún hace falta una variedad más amplia de restricciones, y la posibilidad de especificar las condiciones necesarias y suficientes para la construcción de clases complejas a partir de otras definiciones de clases y propiedades. El lenguaje OWL extiende a RDF y RDFSchema, y ofrece un conjunto mucho más amplio de capacidades como, por ejemplo: características, restricciones y anotaciones de las propiedades; intersección, axiomas y combinaciones booleanas de clases; igualdad; cardinalidad; control de versiones, entre otros [29,30]. De esta gran variedad de capacidades, solo se mostrará el uso de **disjointWith** y **equivalentClass** en las siguientes tripletas.

```
dbpedia-owl:Country owl:disjointWith dbpedia-owl:Language .
```

La propiedad **disjointWith** permite indicar una restricción, donde se especifica que los individuos de tipo **Country** no pueden ser de tipo **Language**, es decir, que sí a un país se le asigna el tipo lenguaje esto producirá un error de inconsistencia en la ontología.

```
dbpedia:Idioma_español rdf:subClassOf dbpedia-owl:Language .
dbpedia:Idioma_español rdf:type dbpedia-owl:Idioma_español .
wikidata:Q1211934 owl:equivalentClass dbpedia-owl:spokenIn some
dbpedia:Idioma_español .
```

La propiedad **equivalentClass** permite especificar la equivalencia de una clase y una combinación de clase y propiedades específicas. Para este caso, se indica que la clase **Q1211934** (Países hispanos) es equivalente a los países de habla (**spokenIn**) española (**Idioma_español**). Esto permite inferir que a los países que se le asigne el idioma español serán agrupados en la jerarquía de países hispanos.

Finalmente, en el proceso de presentar la evolución de los *lenguajes de modelado*, se ha usado un conjunto de *vocabularios y ontologías*, que son una parte fundamental para los Datos Enlazados, estos permiten identificar los tipos de objetos del mundo que nos rodea como, por ejemplo: personas, direcciones, entre otros. Además, nos permiten identificar las relaciones que existen entre esos objetos. A continuación, se muestran algunos de los más importantes: i. *DBpedia Ontology*[10]*:* se basa en OWL, y es la columna vertebral de DBpedia. Esta ha sido creada de forma manual basado en los infoboxes (hojas informativas para mostrar un resumen del tema de una página) utilizados en Wikipedia. La ontología actualmente cubre 685 clases, y se describen con 2.795 propiedades diferentes. Entre las clases que se describen están: Persona, Parques, Deportes, Especies Ciudad, País, entre muchas más. La información en los artículos de Wikipedia se mapea a través de esta ontología. Por ejemplo, la URI ***http://dbpedia.org/ontology/country (dbpedia-owl:Country)*** para representar al concepto **País**. Adicionalmente a esta ontología, DBpedia cuenta con una base de conocimientos que explota el paradigma de Datos Enlazados en la Web, por medio de URIs a millones de conceptos. Por ejemplo, la URI ***http://dbpedia.org/resource/Venezuela (dbpedia:Venezuela)*** representa el concepto **Venezuela**. Actualmente, ya varios proveedores de datos han comenzado a establecer vínculos RDF de sus conjuntos de datos a DBpedia, haciendo DBpedia una de las herramientas más importantes en la web de datos. ii. *Friend of a Friend (FOAF):* es un proyecto que proporciona un vocabulario RDF para expresar metadatos que describen a personas, sus actividades, y sus relaciones con otras personas y objetos. Por ejemplo, ***http://xmlns.com/foaf/0.1/name (foaf:name)*** representa la relación de un concepto con su etiqueta, quedando la tripleta así: **dbpedia:Venezuela** ***foaf:name*** **"Venezuela"@es**. iii. *Dublin Core Metadata Initiative (DCMI):* es una iniciativa para crear un vocabulario para describir recursos como: imágenes, páginas web, videos, libros, etc. Este vocabulario es capaz de proporcionar la información descriptiva básica sobre cualquier recurso, sin que importe el formato de origen, el área de especialización o el origen cultural. Por ejemplo, ***http://purl.org/dc/terms/subject (dcterms:subject)*** representa la relación de un concepto como tema de otro concepto: **dbpedia:Venezuela dcterms:subject dbpedia:Países_de_América_del_Sur.**

Para los Datos Enlazados es recomendable utilizar términos de vocabularios y ontologías bien conocidos, en lugar de crear equivalentes, y solo se deben definir nuevas ontologías o vocabularios cuando los actuales no ofrezcan los requerimientos deseados. Existen páginas dedicadas a indexar vocabularios y ontologías para facilitar su búsqueda, ejemplo de estas páginas son http://lov.okfn.org y http://stats.lod2.eu/vocabularies.

El aprovechamiento óptimo de los Datos Enlazados se logra mediante la integración de tecnologías clave que permiten un flujo de datos eficiente y significativo. Estas tecnologías incluyen *Formatos de Almacenamiento* que estructuran los datos, *Lenguajes de Consulta* que permiten la extracción de información precisa y *Herramientas de Publicación* que facilitan el acceso y la reutilización de los datos. Para obtener detalles adicionales sobre estas tecnologías, refiérase al Anexo 2.A.

[10] http://dbpedia.org/ontology/

2.2 Aprendizaje Automático

El Aprendizaje Automático o Machine Learning en inglés, define a los algoritmos que buscan extraer conocimiento desde las características resaltante de un problema, permitiendo la construcción de modelos de conocimientos [8]. Las características o variables, los modelos de conocimiento a definir, y las técnicas de aprendizaje automático, son los ingredientes principales en el Aprendizaje Automático (ver Fig. 2.1). Las características representan los atributos/variables observables del problema, el modelo es el conocimiento generado con la técnica de Aprendizaje Automático (descriptivo, predictivo, de optimización, etc.), y las técnicas son los mecanismos o estrategias de aprendizaje que se utilizan para generar los modelos de conocimiento.

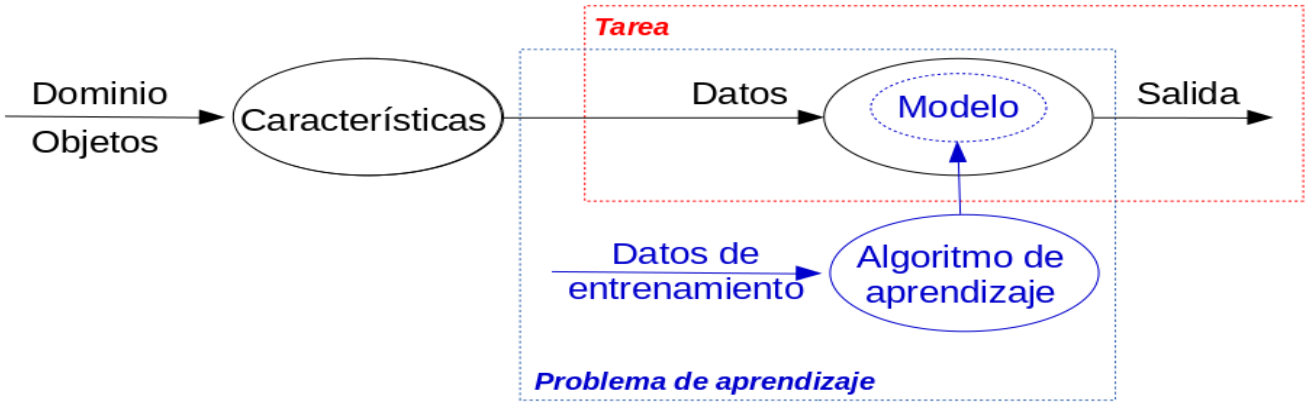

Figura 2.1: Aprendizajes Automáticos y sus elementos

2.2.1 *Tipos de aprendizaje*

El Aprendizaje Automático se divide en varios tipos de aprendizaje: supervisado, no supervisado, reforzado, entre otros. Particularmente, según la naturaleza del etiquetado de datos se definen dos tipo de aprendizaje (ver Fig. 2.2): supervisado y no supervisado [8, 33]. El *aprendizaje supervisado* se utiliza para estimar un mapeo desconocido de entrada y salida, a partir de muestras conocidas de entrada y salida, donde la salida está etiquetada. Este se divide en dos categorías de algoritmos, de regresión/predicción (el tipo de campo objetivo es numérico o continuo) y clasificación (el tipo de campo objetivo es categórico o discreto). El *aprendizaje no supervisado* se utiliza para encontrar relaciones de similitud, diferencia o asociación en los datos de entrada, y solo se dan muestras de entrada al sistema de aprendizaje. Este se divide en agrupamiento (datos que son similares entre sí.), anomalía (datos que se diferencian de las demás) y asociación (datos que se relacionan con otros datos).

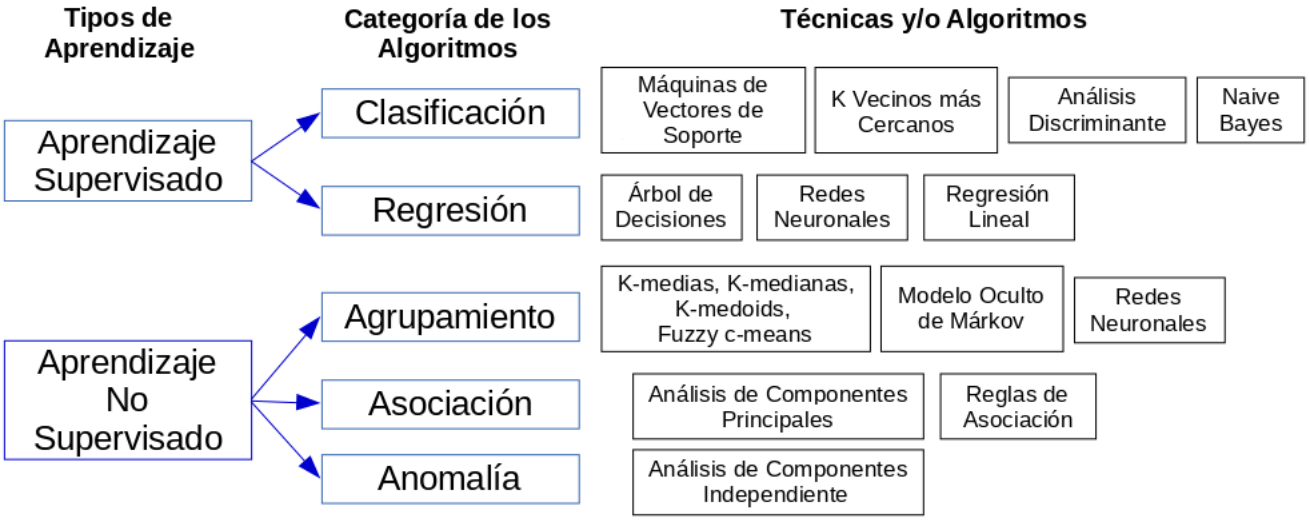

Figura 2.2: Los algoritmos de Aprendizajes Automáticos basados en datos

Por último, cada una de las técnicas y/o algoritmos de Aprendizaje Automático establece sus propias estrategias/mecanismos de aprendizaje para concebir sus modelos de conocimiento. Algunas de las técnicas o algoritmos más populares son los siguientes [34]:

- *Árbol de Decisiones:* Usa un árbol de decisión para modelar la relación entre las características del modelo y los potenciales resultados, cada árbol está formado por nodos, arcos y ramas. Cada nodo representa los atributos de un grupo de datos que se va a clasificar, y cada arco representa un valor que el nodo puede tomar [34]. Al moverse por las ramas del árbol se puede ver la relación entre los valores de los atributos para el conjunto de datos bajo estudio.
- *K-medias:* Su objetivo es partir un conjunto de n valores en k grupos distintos, en el que cada observación pertenece al grupo cuyo valor medio es más cercano. Todas las observaciones que poseen características similares se colocan en el mismo grupo, y la media de los valores en un grupo particular es el centro de ese grupo [34].
- *Redes Neuronales:* es un algoritmo que imita el funcionamiento del cerebro, y en particular de las neuronas, de sus interconexiones, y de cómo ciertos estímulos de entradas producen ciertas salidas o resultados. Una red neuronal multicapas funciona en tres grupos de capas [34]. La capa de entrada recibe la información. Las capas ocultas procesan la entrada. Por último, la capa de salida envía la respuesta calculada. En general, una neurona artificial suma sus entradas (como las dendritas), a partir de allí establece su estado de activación (como el soma), y envía su salida a sus neuronas vecinas (como el axón).

2.2.2 *Técnicas avanzadas de redes neuronales*

En el mundo de las técnicas avanzadas de redes neuronales, nuestros trabajos se centraron en dos arquitecturas de aprendizaje profundo que han revolucionado la generación de datos sintéticos y el campo de la visión por computadora: Autoencoders Variacionales (VAE, Variational AutoEncoders) y Redes Neuronales Convolucionales (CNN, Convolutional Neural Network). Específicamente, utilizamos VAE para la generación de datos sintéticos, aprovechando su capacidad para la reconstrucción de información, y CNN para la extracción automática de características relevantes.

- VAE: están diseñadas para aprender una representación latente comprimida de los datos de entrada. Los VAE introducen un elemento de probabilidad en su arquitectura, lo que les permite generar nuevos datos similares a los de entrenamiento, por lo tanto, son ideales para la reconstrucción de información faltante en imágenes y para la generación de datos sintéticos [35, 36]. Estas redes están compuestas por tres partes [37, 38] (ver Figura 2.3): El Encoder, representado en color verde, es la parte inicial del modelo que comprime los datos de entrada en un espacio latente de menor dimensión. El Espacio Latente, representado en color rojo, es la parte central del modelo y es una representación comprimida de los datos de entrada, donde cada punto representa una distribución de probabilidad. El Decoder, representado en color azul, es la parte final del modelo que reconstruye los datos originales a partir de la representación latente.

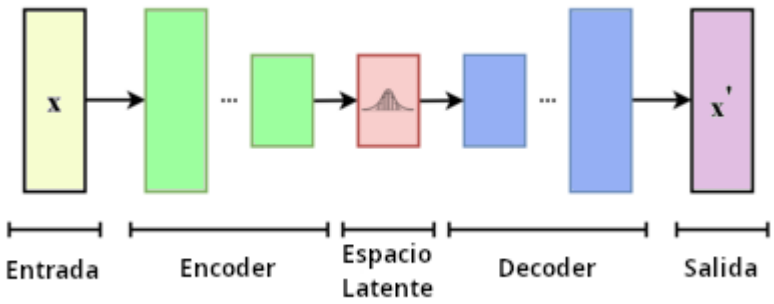

Figura 2.3: Arquitectura básica de un VAE

- CNN: están diseñadas para trabajar con datos de imágenes, y su arquitectura aprovecha la propiedad de la localidad de las imágenes (es decir, que los píxeles cercanos están más relacionados entre sí). Las CNN tienen la capacidad de aprender automáticamente las características más relevantes de las imágenes y de reconocer objetos independientemente de su posición y orientación en la imagen, por lo tanto, son ideales para extracción automática de características y para las tareas de clasificación y detección. Estas redes están compuestas por diferentes tipos de capas [39] (ver Figura 2.4): Las capas convolucionales, representadas en color amarillo, realizan la mayor parte de los cálculos, y es donde se extraen las características

esenciales de los datos de entrada. Esto se logra mediante la aplicación de múltiples filtros, también conocidos como kernels, que actúan como detectores de características. Cada filtro se desliza sobre la entrada, realizando operaciones de convolución para identificar patrones específicos y generar mapas de características que resaltan la presencia de dichas características. Las capas de pooling, representadas en color azul, reducen la dimensionalidad de las características extraídas y hacen que la representación sea más invariante a pequeñas traslaciones. La capa totalmente conectada, representada en color verde, realiza la tarea de clasificación a partir de las características obtenidas en las capas anteriores, generando una salida final.

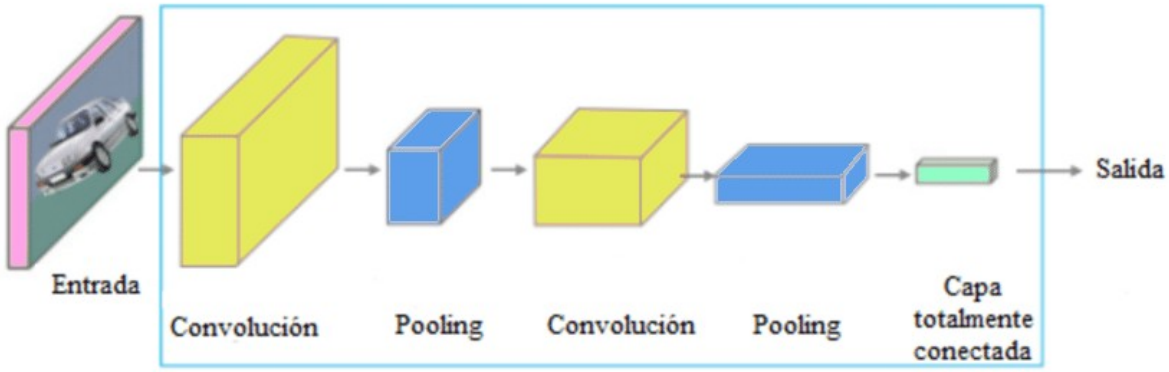

Figura 2.4: Arquitectura básica de una CNN

2.3 *Meta-Aprendizaje*

El Meta-Aprendizaje (en inglés *Meta-Learning*) permite a Aprendizaje Automático aprender a aprender, mediante tareas que adaptan a las técnicas de Aprendizaje Automático a nuevos entornos [40, 41, 9], es decir, aprende de las experiencias previas de forma sistemática y basada en datos. En general, el Meta-Aprendizaje adapta sus respuestas en función de las características inherentes a las tareas de Aprendizaje Automático que ha resuelto previamente [9, 42]. En específico, se necesita, por un lado, recopilar metadatos que describan las tareas de aprendizaje y los modelos aprendidos previamente. Esto comprende las configuraciones exactas del algoritmo utilizado para entrenar los modelos, incluidos los ajustes de hiperparámetros, las técnicas de aprendizaje, las evaluaciones del modelo resultante, entre otros. Para ello, se apoya en el conocimiento de las fuentes de dato (Meta-Dataset), conocimiento de las características de los datos (Meta-Características), conocimiento de las técnicas de aprendizaje (Meta-Técnicas) y conocimiento de los modelos de conocimiento (Meta-Modelos).

2.3.1 *Meta-Dataset*

Los Meta-Dataset consisten en información que caracteriza a los conjuntos de datos, describen el contenido, calidad, condiciones, historia, disponibilidad y otras características [9, 42]. Esta descripción facilita la localización, selección, recuperación y utilización de los conjuntos de datos [9].

2.3.2 *Meta-Características*

Las Meta-Características (Meta-Feature) son atributos que describen características de los datos, ofreciendo información sobre cómo se construyó los datos, su calidad, su complejidad y otras propiedades generales. Normalmente estos Meta-Feature se obtienen a partir de características individuales, combinaciones de características o información específica del dominio [9, 42]. A continuación se presentan diferentes tipos de características:

- Estadísticas descriptivas: Media, mediana, desviación estándar, mínimo, máximo, rango, cuartiles, etc.
- Medidas de complejidad: Número de instancias, número de atributos, número de clases, proporción de valores faltantes, etc.
- Medidas de balance de clases: Proporción de instancias en cada clase.
- Medidas de correlación: Correlación de Pearson, coeficiente de Spearman, etc.
- Medidas de redundancia: Número de atributos redundantes.
- Medidas de ruido: Nivel de ruido en los datos, número de valores extremos, entre otros.
- Dominios específicos: Polaridad o emoción, intensidad narrativa, similitud entre los productos o usuarios, engagement, género musical, entre muchos más.

2.3.3 *Meta-Técnicas*

Las Meta-Técnicas (Meta-Technique) es información sobre las estrategias, parámetros y algoritmos que buscan optimizar y mejorar el proceso de aprendizaje de los modelos de Aprendizaje Automático [42]. Gestiona información como:

- Tipo de técnica: Cómo construir el modelo adecuado (clasificación, regresión, clustering, entre otros).
- Selección de hiperparámetros: Cómo elegir los mejores valores para los parámetros de un modelo (tasa de aprendizaje, número de capas, etc.).
- Diseño de arquitecturas: Cómo construir modelos más eficientes y efectivos.
- Optimización de algoritmos: Cómo acelerar el entrenamiento y mejorar la convergencia de los modelos.

- Evaluación de modelos: Cómo medir la calidad y generalización de los modelos (métricas).

2.3.4 *Meta-Modelos*

Los Meta-Modelos (Meta-Model) son, en esencia, repositorios de conocimiento que almacenan información detallada sobre experimentos de Aprendizaje Automático realizados previamente [9, 42]. Esta información no solo incluye las métricas de rendimiento y los valores de los hiperparámetros, sino que también captura las características intrínsecas de los datos, las técnicas de preprocesamiento empleadas, la arquitectura de los modelos y las condiciones experimentales en general. Es decir, refleja las relaciones complejas entre las Meta-Características de una tarea y las configuraciones específicas de la técnica usada (Meta-Técnicas) para la construcción de un modelo [9]. En general, los Meta-Modelos actúan como una especie de "memoria colectiva" para el Aprendizaje Automático, permitiendo aprender de los errores y éxitos del pasado para construir modelos más eficientes y efectivos. A continuación se presenta la utilidad de los Meta-Modelos [9, 42]:

- Transferencia de conocimiento: Permiten reutilizar el conocimiento adquirido en tareas anteriores para acelerar el desarrollo de nuevos modelos y mejorar su rendimiento.
- Selección de modelos: Ayudan a seleccionar el modelo más adecuado para una nueva tarea, basándose en las características de los datos y los objetivos del proyecto.
- Optimización de hiperparámetros: Facilitan la búsqueda de los mejores valores de los hiperparámetros, evitando la exploración exhaustiva del espacio de búsqueda.
- Análisis de sensibilidad: Permiten evaluar la influencia de diferentes factores en el rendimiento de los modelos, como el tamaño del conjunto de datos o la elección de un algoritmo específico.
- Descubrimiento de patrones: Pueden revelar patrones y relaciones entre las características de los datos, las técnicas de aprendizaje y el rendimiento de los modelos, lo que puede conducir a nuevos conocimientos y avances en el campo del Aprendizaje Automático.

2.4 Lógica Dialéctica

La palabra *Dialéctica* tiene muchas definiciones, la más concisa indica que es una teoría y técnica retórica de dialogar y discutir para descubrir la verdad mediante la exposición y confrontación de razonamientos y argumentaciones contrarias entre sí. En otras palabras, siempre debe incluir de algún modo la contradicción [43]. En la Lógica Dialéctica, los axiomas dialécticos permiten que las contradicciones y ambivalencias sean válidas dentro de un modelo formal [44]. En este sentido, la Lógica Dialéctica y la lógica formal parecen seguir caminos opuestos, y en efecto, la Lógica Dialéctica sería el reino de la contradicción, mientras que la lógica formal sería el reino de la no-contradicción [43].

Ahora bien, en la lógica del razonamiento humano existen instancias o estados que son afectados por las contradicciones, es decir, que reducir la lógica a una lógica sin contradicción sería tanto como querer ocultar la realidad [43, 45]. La Lógica Dialéctica posee la capacidad de resolver situaciones en las que un razonamiento se encuentra con información inconsistente, ya que tiene a la mano información tanto para creer una cosa como para, al mismo tiempo, creer lo contrario. Es decir, está en un estado de contradicción o ambigüedad. Los posibles estados que gestiona la Lógica Dialéctica son los siguientes [10]:

- *Declaraciones contingentes sobre el futuro:* indica que algo fue verdadero y falso en el pasado, por lo que no se puede prever su futuro. Ej. **“Mañana habrá una guerra”** puede ser cierto o falso, ya que han ocurrido ambos casos en el pasado.
- *Falla de una presuposición:* suponer algo que no es realmente cierto. Ej. **“Es un niño”**, si en la presuposición se asume un posible valor, se puede pensar que es un niño, y allí está la falla, porque también podría ser una niña
- *Vaguedad:* falta de claridad, precisión o exactitud en los fenómenos del lenguaje natural. Ej. En la oración **“Él es calvo”**, no se puede negar que una persona con cero cabellos sea calva, como tampoco se puede negar que una persona con 1000 cabellos sea calva.
- *Discurso ficticio:* tomar decisiones según ciertos supuestos imaginarios (lógicas imaginarias no aristotélicas). Ej. **“Las vacas están volando”**, se puede decir que es falso porque nuestras creencias nos indica que las vacas no vuelan, pero podría ocurrir que las vacas están siendo transportadas en un avión, y la respuesta sería verdadera. Otro contexto donde sería verdadero es si se está hablando de un juego donde las leyes o creencias son distintas (mundo imaginario).
- *Razonamiento contrafáctico:* pensar lo que pudo ser y no fue. Ej. **“Si no hubiese salido, hubiera aprobado y ahora no tendría que estudiar para el recuperativo”.** Representa algo que no sucedió, pero que *podría* haber ocurrido. Allí subyace la incertidumbre, en eso que pudo haber ocurrido.

Finalmente, en la Universidad de Miami en EEUU se desarrolló un razonador dialéctico para el sistema TPTP, denominado JGRM3, basado en la ***Lógica Dialéctica RM3*** [4]. Esta lógica pertenece a la rama de las Lógicas Paraconsistente, o sea, permite que las inconsistencias y las contradicciones sean válidas. Sin embargo, la Lógica Dialéctica se distingue de otras lógicas paraconsistentes por su aceptación del Modus Ponendo Ponens en su forma clásica, lo que facilita la derivación de conclusiones a partir de implicaciones y antecedentes afirmados. Entendiéndose que el *Modus Ponendo Ponens* (modo que afirma afirmando), también llamado *Modus Ponens* (modo afirmativo), establece que si un término implica a otro, y el término es verdadero; entonces se puede inferir que el otro término es verdadero. Por ejemplo, **“Si está lloviendo, te espero dentro del teatro”**, y **“está lloviendo”**, por lo tanto, **“te espero dentro del teatro**.

3 Arquitecturas de Gestión de Conocimiento basado en Datos Enlazados

En este capítulo se presentan dos arquitecturas para la gestión de conocimiento basado en los Datos Enlazados, estás ideas surgen de la revisión de la literatura sobre los Datos Enlazados en [46]. Además, ambas arquitecturas son especificadas según MEDAWEDE [24].La estructura del capítulo es la siguiente: La sección 3.1 se basa en la sección 3 del artículo presentado en el Anexo 3.A, y presenta una ampliación de las capacidades del middleware MiSCi [47, 48, 49], al agregar una nueva capa denominada Datos Enlazados, para identificar, describir, conectar, relacionar y explotar los distintos datos generados por los sensores, usuarios y las aplicaciones de la ciudad inteligente. La sección 3.1.2 ilustra un caso de estudio del MiSCi, basándose en la sección 4 del artículo presentado en el Anexo 3.A. La sección 3.2 se basa en la Sección 3 del artículo presentado en el Anexo 3.B y la Sección "Materiales y métodos" del artículo presentado en el Anexo 3.C, y describe una arquitectura que permite crear y enriquecer ontologías emergentes de forma autónoma, usando como insumo el paradigma de Datos Enlazados. Finalmente, la sección 3.2.4 presenta un caso de estudio del generador ontológico, basándose en la sección 4 del artículo presentado en el Anexo 3.B.

3.1 Ampliación del MiSCi extendido con Datos Enlazados

Esta sección ofrece un resumen extenso del trabajo presentado en [50], y los detalles se encuentran en la sección 3 del Anexo 3.A. En esta investigación se amplía el trabajo propuesto en [47, 48, 49], al agregar una nueva capa denominada Datos Enlazados (Linked Data Layer – LDL), que aumenta las capacidades del middleware MiSCi (ver Figura 3.1). Los Datos Enlazados responden a varias necesidades en las ciudades inteligentes: la primera es para interpretar grandes cantidades de datos que provienen de distintas fuentes como: sensores, efectores, entre otros, donde muchos de estos datos son manejados en tiempo real. La segunda es para enriquecer los datos con información semántica, proveniente de MiSCi o de fuentes externas.

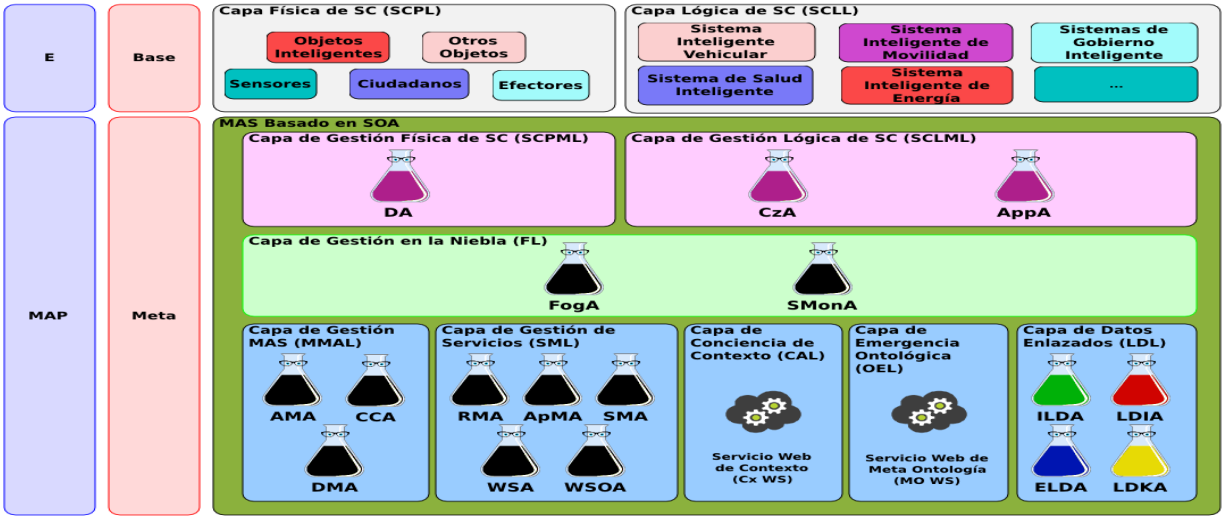

Figura 3.1: Extensión del middleware MiSCi con Datos Enlazados

En particular, explicaremos la capa desarrollada en ese trabajo, la capa LDL. Los distintos agentes de la capa LDL automatizan las etapas de la metodología MEDAWEDE, en el middleware MiSCi. MEDAWEDE es una metodología que permite desarrollar servicios basados en el paradigma de Datos Enlazados, y está compuesta por seis etapas divididas en dos grandes tareas. La primera tarea tiene la finalidad del enriquecimiento de los datos y su transformación a Datos Enlazados, y está integrada por las siguientes etapas [24]: *i) Especificación:* en esta fase se seleccionan las fuentes de datos; *ii) Modelado:* se centra en la creación del modelo que describe el conocimiento del área de estudio, para ello se utilizan vocabularios estándares, se reutilizan ontologías, e incluso, se diseñan ontologías propias, *iii) Generación:* se centra en la transformación de los datos al lenguaje RDF; *iv) Vinculación:* en esta fase se vinculan los datos con otros conjuntos de datos para aumentar su valor, visibilidad y calidad, *v) Publicación:* se pone a disposición los datos en repositorios de tripletas. La segunda tarea tiene como objetivo la explotación de los Datos Enlazados, y la integra la siguiente etapa: *vi) Explotación:* esta etapa permite el manejo e integración de distintas interfaces o aplicaciones, para consumir los recursos publicados de manera agradable y sencilla.

En la capa LDL se llevan a cabo dos procesos importantes, según MEDAWEDE: i) Enriquecimiento: Este proceso realiza las etapas de especificación, modelado y generación de los datos de MiSCi, por parte de los agentes ILDA (Internal Linked Data Agent) y ELDA (External Linked Data Agent). Además, se realizan las tareas de vinculación y publicación de los datos de MiSCi por parte del agente LDIA (Linked Data Integration Agent); ii) Explotación: Este proceso realiza la etapa de explotación de los datos de MiSCi, la cual es realizada por el agente LDKA (Linked Data Knowledge Agent). Ahora bien, estos agentes pueden ser activados simultáneamente desde distintos procesos, según las circunstancias que lo ameriten.

El proceso de enriquecimiento semántico implica recolectar datos internos y externos al MiSCi (ver Figura 3.2). La recolección de datos internos se activa cada vez que algún agente CzA, AppA o DA es actualizado, para lo cual se invoca al agente ILDA con los datos nuevos que pueden provenir simultáneamente de diferentes fuentes (ver paso 1 en Figura 3.2). Estos datos son preparados y enriquecidos simultáneamente con información del Servicio Web de Contexto (Context Web Services - Cx WS) y del Servicio Web de Meta Ontología (Meta Ontology Web Services – MO WS) (ver paso 2 y

3 en Figura 3.2). De la misma manera ocurre con la recolección de los datos externos, lo cual es realizado por el agente ELDA. Luego, el agente LDIA es activado con la información enriquecida generada por los agentes ILDA o ELDA, para vincular los datos previamente obtenidos con otros Datos Enlazados (ver paso 4 en Figura 3.2), para finalmente publicarlos como Datos Enlazados (ver paso 5 en Figura 3.2).

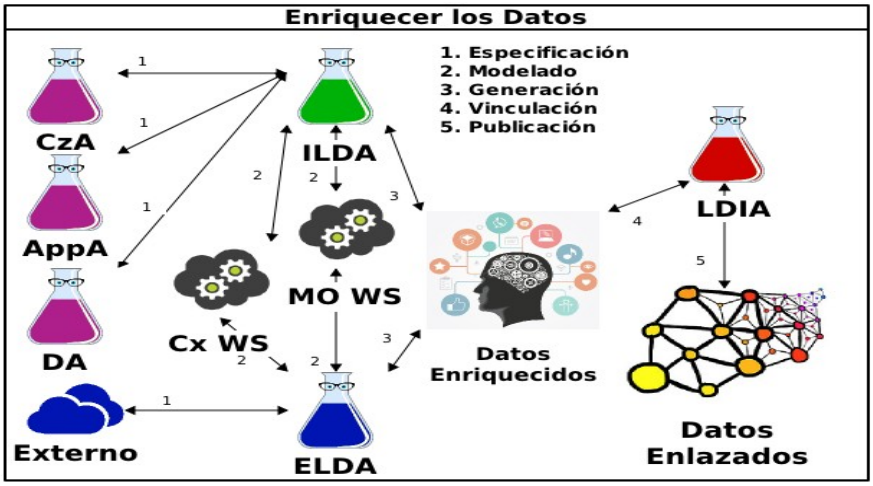

Figura 3.2: Proceso de enriquecimiento semántico de los datos

El proceso de explotación es activado por un agente del MiSCi cuando solicita información enlazada (ver Figura 3.3). Para este tipo de consulta se debe invocar al agente LDKA de MiSCi (ver paso 1 en Figura 3.3). Luego, el agente LDKA por medio de los distintos mecanismos inteligentes de explotación de conocimiento enlazado, explora simultáneamente todas las fuentes de Datos Enlazados, y retorna la información solicitada por el agente (ver paso 2 en Figura 3.3).

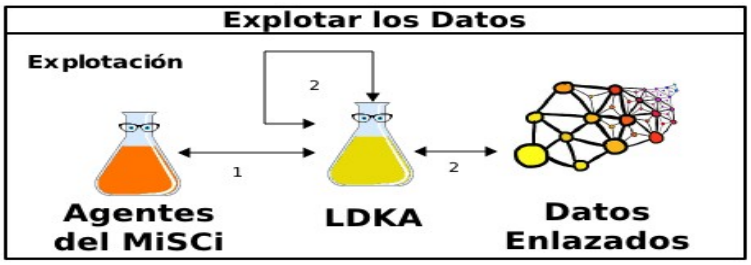

Figura 3.3: Proceso de explotación de los datos

Los agentes de la capa LDL definen cuatro ciclos autonómicos para la autogestión de los Datos Enlazados en MiSCi, basado en el concepto de ciclos autonómicos como servicios propuestos en [51, 52]. Cada ciclo autonómico establece la relación entre las tareas de los agentes, para la explotación de cada una de las formas de conocimiento que permite el agente LDKA. Algunas de esas tareas establecen las reglas a activar del sistema recomendador según el contexto, o identifican el modelo o el conjunto de datos pertinente a una situación dada, entre otras cosas.

3.1.1 *Especificación de los agentes de Datos Enlazados*

La capa de Datos Enlazados está conformada por 4 tipos de agentes. Para la especificación de cada agente se usa MASINA [53], y en específico, los modelos de agentes y de tareas.

3.1.1.1. Agente de Datos Enlazados Internos (Internal Linked Data Agent - ILDA)

Son los agentes que brindan la capacidad de extracción, curación, agregación y modelado de las fuentes de información provenientes de agentes internos, como CzA, AppA y DA, para finalmente ser transformadas a formatos adecuados para el enlazado de datos. Específicamente, su objetivo es enriquecer los datos del agente solicitante con información del contexto de MiSCi. El diagrama de actividad (ver Figura 3.4) muestra el servicio de este agente, y está compuesto por dos sub-servicios; el primero extrae los datos de las fuentes internas, y el segundo enriquece estos datos con información del contexto y de las ontologías.

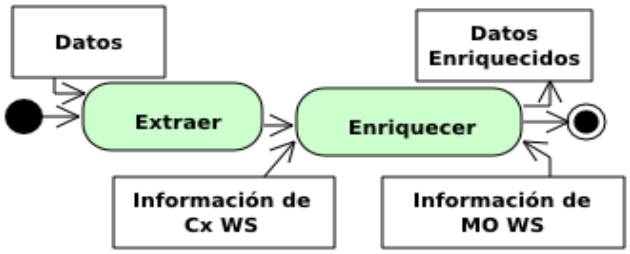

Figura 3.4: Diagrama de actividad de ILDA.

Modelo de Agente de ILDA. Este modelo indica el tipo de agente, el rol que cumple y la descripción de su funcionalidad:

- Tipo: agente de servicio.
- Roles: ofrecer servicio de extracción y enriquecimiento de los datos generados por el MiSCi.
- Descripción: procesa solicitudes para extraer y enriquecer datos de MiSCi con información del contexto (Cx WS) y de las ontologías (MO WS), a petición de los agentes CzA, AppA, DA

Modelo de tareas de ILDA. El servicio “Enriquecer Datos” del agente ILDA presenta las siguientes tareas:

- T1. Recibir la solicitud del agente solicitante
- T2. Extraer datos del agente solicitante
- T3. Modelar los datos del agente solicitante
- T4. Generar los datos como RDF

3.1.1.2. Agente de Datos Enlazados Externos (External Linked Data Agent - ELDA)

Son los agentes que brindan la capacidad de extracción, curación, agregación y modelado de las fuentes de información provenientes del exterior de MiSCi (Redes Sociales, Páginas Web, entre otros), para finalmente ser transformadas a formatos adecuados para el enlazado de datos, que permitan enriquecer semánticamente a la capa CAL del MiSCi con información del exterior. Es decir, ofrece el servicio de

extracción y enriquecimiento de las fuentes externas al MiSCi. En la Figura 3.5 se observa su diagrama de actividad.

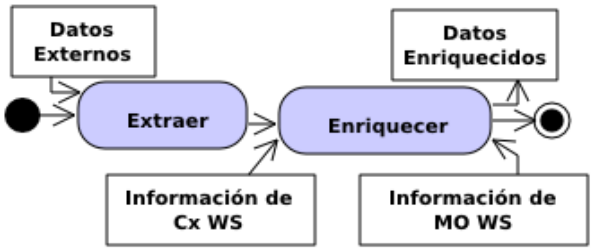

Figura 3.5: Diagrama de actividad del caso de uso de ELDA.

Modelo de Agente de ELDA:

- Tipo: agente de servicio.
- Roles: ofrecer servicio de extracción y enriquecimiento de los datos de fuentes externas al MiSCi.
- Descripción: procesa solicitudes para extraer y enriquecer datos provenientes del exterior (Redes Sociales, Páginas Web, entre otros) con información del contexto (Cx WS) y de las ontologías (MO WS).

Modelo de tareas de ELDA. Las tareas del servicio "Enriquecer Datos" de este agente son los siguientes:

- T1. Recibir la solicitud
- T2. Extraer datos de la fuente externa
- T3. Modelar los datos de la fuente externa
- T4. Generar los datos como RDF

3.1.1.3. Agente de Integración de Datos Enlazados (Linked Data Integration Agent - LDIA)

Es el agente que brinda la capacidad de vincular y publicar la información interna y externa generada por ILDA y/o ELDA. Este agente vincula los datos, para luego ser publicados como Datos Enlazados. En la Figura 3.6 se observa su diagrama de actividad.

Figura 3.6: Diagrama de actividad del caso de uso de LDIA.

Modelo de Agente de LDIA.

- Tipo: agente de servicio.

- Roles: ofrecer servicio de vinculación y publicación de datos enriquecidos.
- Descripción: procesa solicitudes de vinculación de datos enriquecidos con los distintos conjuntos de datos obtenidos en anteriores invocaciones, que luego son publicados como Datos Enlazados.

Modelo de tareas de LDIA. Las tareas del servicio "Vincular y Publicar los Datos Enriquecidos" de este agente son:

- T1. Recibir la solicitud con los datos enriquecidos
- T2. Vincular los datos enriquecidos
- T3. Publicar los datos enriquecidos

3.1.1.4. Agente de Conocimiento de Datos Enlazados (Linked Data Knowledge Agent - LDKA)

Son los agentes que ofrecen mecanismos para explotar el conocimiento vinculado a los Datos Enlazados, permitiendo capacidades de: análisis semántico, manejo de ambigüedad, etiquetado semántico, búsqueda exploratoria, visualización, filtrado, entre otros. Los servicios que presta LDKA son: i) Recomendar Información usando inferencia híbrida de lógica descriptiva/dialéctica para retornar información según la necesidad particular de un agente; ii) Generar Modelos de Aprendizaje Automático que permite retornar modelos de conocimiento basado en distintas técnicas de Aprendizaje Automático como árboles de decisiones, redes bayesianas, crónicas, redes neuronales como las de aprendizaje profundo, etc.; iii) Generar Datos para entrenamiento de modelos de conocimiento, muestreos, etc.; iv) Aprender Ontologías usa los Datos Enlazados para poblar nuevas ontologías. El diagrama de actividad (ver Figura 3.7) muestra los detalles de los servicios prestados por LDKA.

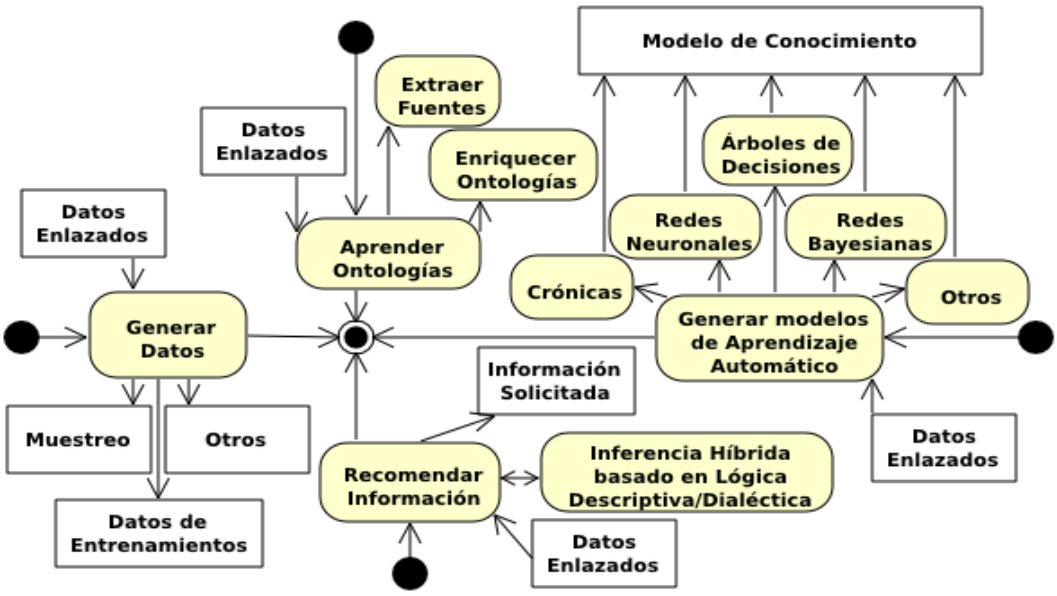

Figura 3.7: Diagrama de actividad del caso de uso de LDKA.

Los cuatros servicios para explotar el conocimiento vinculado a los Datos Enlazados, permiten mejorar trabajos previos vinculados a la integración de datos, construcción de recomendadores o modelos de máquinas de aprendizaje más robustos, entre otros.

Modelo de Agente de LDKA. El modelo de agente de LDKA se describe de la misma manera que los anteriores agentes.

- Tipo: agente de servicio
- Roles: ofrecer servicio de explotación del conocimiento, como Recomendar información, Generar modelos de Aprendizaje Automático, Generar datos, o Aprender Ontologías.
- Descripción: explora los Datos Enlazados y realiza las transformaciones del conocimiento según el tipo de solicitud (Recomendar información, Generar modelos de Aprendizaje Automático, Generar datos o Aprender Ontologías).

Modelo de tareas de LDKA. En la Tabla 3.1 se definen las tareas del agente LDKA, por cada servicio prestado.

Tabla 3.1: Servicios y tareas de LDKA.

LDKA-S1. Recomendar información
T1. Recibir la solicitud T2. Inferencia híbrida basado en lógica descriptiva/dialéctica T3. Retornar información solicitada
LDKA-S2. Generar modelos de Aprendizaje Automático
T1. Recibir la solicitud T2. Generar modelo de conocimiento basado en técnicas tales como crónicas, redes neuronales, árboles de decisión, redes bayesianas u otros. T3. Retornar modelo de conocimiento
LDKA-S3. Generar datos de entrenamiento
T1. Recibir la solicitud T2. Generar los datos de entrenamiento T3. Retornar los datos
LDKA-S4. Aprender Ontologías
T1. Recibir la solicitud T2. Extraer fuentes de datos T3. Enriquecer Ontologías

3.1.2 *Experimentación*

Esta sección muestra un resumen extenso del escenario 2 “Explotar Datos” del caso de estudio presentado en [50], cuyos detalles se encuentran en la sección 4 del Anexo 3.A. Los datos de entrada necesarios para este escenario son recogidos previamente, o en paralelo, por el proceso de

enriquecimiento (ver escenario 1 en la sección 4 del Anexo 3.A). Estos datos de entrada son publicados como Datos Enlazados por el agente LDIA.

Este escenario tiene como objetivo mostrar cómo se usan los Datos enlazados por el agente LDKA en la capa LDL del MiSCi. Además, se muestra el uso de los servicios de LDKA, que requieren de otros servicios ofrecidos por el mismo. Por ejemplo, en la Figura 3.8 se observa el diagrama de actividad del agente de conocimiento (Knowledge Agent - KA) del MiSCi, para el servicio de Generar Conocimiento, que invoca a otros dos servicios del agente LDKA.

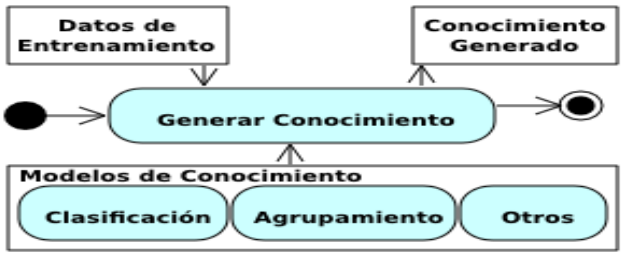

Figura 3.8: Diagrama de actividad del KA.

A continuación, se describe este escenario (ver Figura 3.9):

1. El agente CzA detecta problemas en los signos vitales del ciudadano, y genera la señal de alarma al HSS (Sistema Inteligente de Salud), a través del AppA que caracteriza a ese sistema.
2. El agente AppA (HSS) solicita al servicio Recomendar Información del agente LDKA, buscar los ciudadanos en los alrededores con capacidades de practicar los primeros auxilios, que permita brindarle atención médica inmediata al paciente.
3. LDKA retorna la información solicitada por AppA (HSS).
4. Luego, AppA (HSS) envía la notificación a los ciudadanos a través de CzA.
5. En los distintos procesos de actualización de conocimiento van emergiendo ontologías, en consecuencia el componente MO WS va solicitando el servicio de Aprender Ontología de LDKA, para extraer información y enriquecer dichas ontologías emergentes.
6. LDKA retorna las ontologías emergentes enriquecidas.
7. El AppA (HSS) solicita al servicio Recomendar Información del agente LDKA, recomendar los servicios de ambulancias, para tratar al paciente y trasladarlo al centro médico más cercano.
8. LDKA retorna la información solicitada por AppA (HSS).
9. El agente AppA (HSS) también solicita al servicio Generar Conocimiento (Figura 3.8) del agente KA, los posibles diagnósticos y sugerencias de tratamiento.
10. El agente KA necesita un modelo de conocimiento para resolver el problema, por lo que activa el servicio Generar Modelos de Aprendizaje Automático de LDKA, para obtener el modelo de conocimiento sobre dicho problema.
11. LDKA genera y retorna el modelo de conocimiento a KA.
12. También KA necesita datos de entrenamiento para afinar el modelo de conocimiento, para eso solicita al servicio Generar Datos de LDKA.
13. LDKA genera y retorna los datos de entrenamiento para el modelo de conocimiento de KA.
14. Finalmente, el agente del MiSCi retorna los posibles diagnósticos y sugerencias de tratamiento a AppA (HSS).

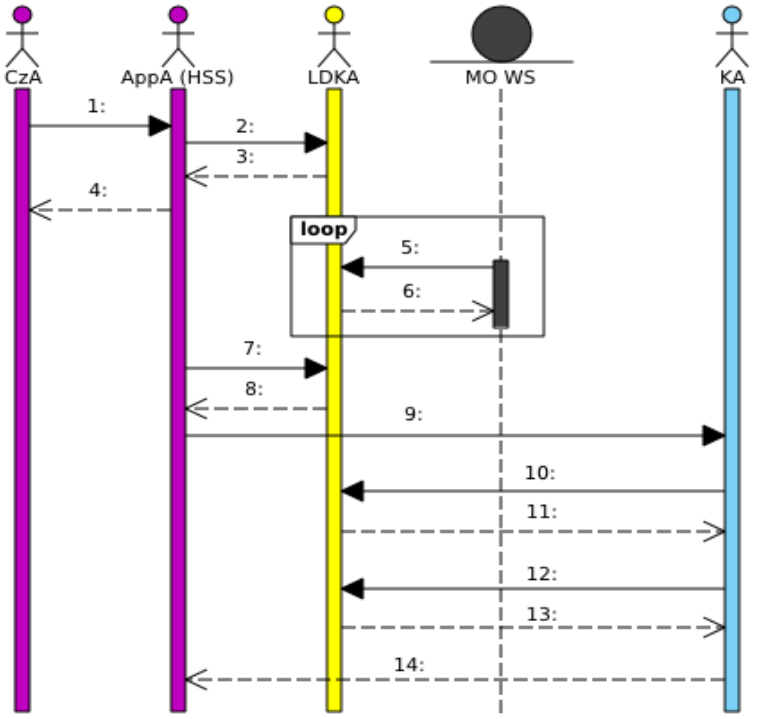

Figura 3.9: Diagrama de secuencia para explotar datos.

En este escenario, se le solicita a LDKA explotar los Datos Enlazados para generar conocimiento y responder a las distintas necesidades presentes en el MiSCi. LDKA responde con información contextualizada y adaptada a cada necesidad.

3.2 Generación Automático de Ontologías basado en Datos Enlazados

Esta sección presenta un resumen extenso de los trabajos presentados en [54, 55], cuyos detalles se encuentran en la sección 3 del artículo en el Anexo 3.B y la sección “Materiales y métodos” del artículo en el Anexo 3.C, dónde se describe la arquitectura general AOGS (Automated Ontology Generator System), que permite crear y enriquecer ontologías emergentes de forma autónoma, usando como insumo el paradigma de Datos Enlazados. Se compone de tres capas (ver Figura 3.10):

- *Knowledge Base Manager:* se basa en las etapas (i) y (ii) del MEDAWEDE. En nuestro caso, se encarga de gestionar y almacenar el conocimiento de AOGS.
- *Knowledge Generator Manager:* se basa en las etapas (iii), (iv) y (v) de MEDAWEDE. En nuestro caso, controla el procesamiento del conocimiento de la capa anterior, y genera ontologías extendidas con LD.
- *Web Services Manager:* se basa en la etapa (vi) de MEDAWEDE. En nuestro caso, se encarga de recibir las peticiones web realizadas por los clientes a AOGS, por ejemplo, servicio de generación de ontologías o servicio de gestión de fuentes de conocimiento.

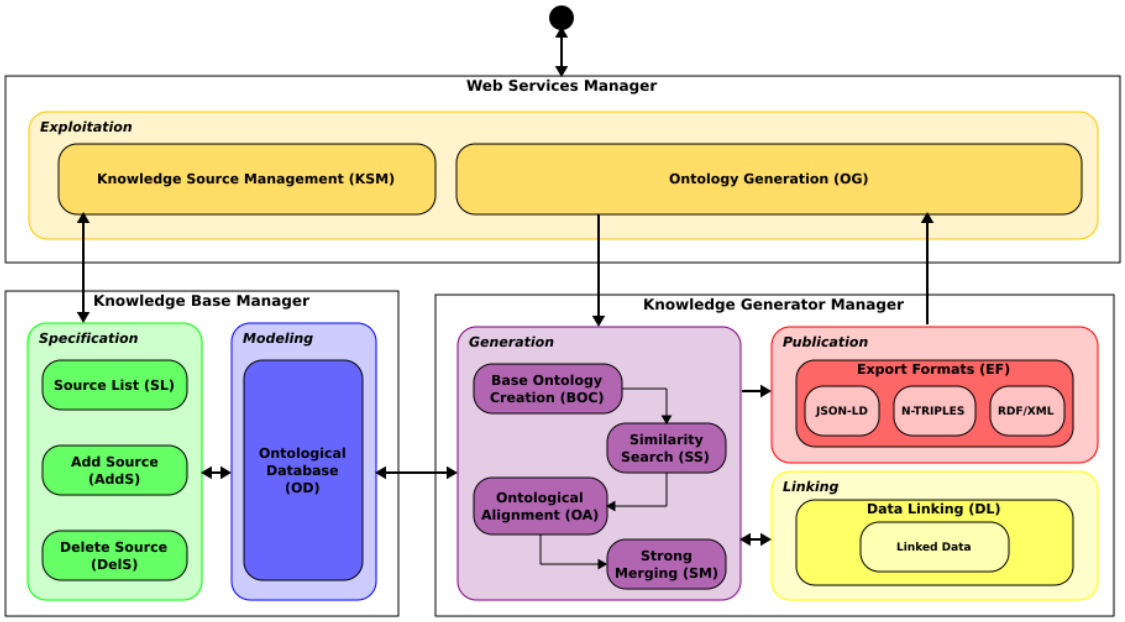

Figura 3.10: Diagrama de componentes de nuestra arquitectura basada en MEDAWEDE.

3.2.1 *Componentes de la capa Knowledge Base Manager*

Los componentes descritos en esta sección permiten gestionar la base de conocimientos del sistema.

1) Ontological Database (OD): Este componente se encarga de almacenar y poner a disposición todo el conocimiento que posee el sistema con el fin de generar ontologías para contextos específicos. La OD almacena las clases, propiedades y relaciones de las diferentes ontologías. Este componente es activado por el resto de componentes que necesitan gestionar la información del sistema.
2) Source List (SL): Este componente muestra una lista de las ontologías que están disponibles en la base de conocimiento del sistema. Este componente es activado por los servicios de Knowledge Source Management (KSM).
3) Add Source (AddS): Se encarga de ampliar la base de conocimiento disponible en el sistema para generar una ontología en un contexto específico. Este componente es activado por los servicios KSM junto con la ontología a añadir al sistema.
4) Delete Source (DelS): Se encarga de reducir la base de conocimiento disponible en el sistema para generar una ontología en un contexto específico. Este componente es activado por los servicios KSM junto con el id de la ontología a eliminar del sistema.

3.2.2 *Componentes de la capa Knowledge Generator Manager*

En esta sección se describen los componentes que permiten generar la ontología del contexto solicitada por el cliente.

1) Base Ontology Creation (BOC): Este componente crea una ontología base con los conceptos definidos como Términos de Búsqueda. Los Términos de Búsqueda son proporcionados por el cliente para obtener una ontología de un dominio específico. Estos conceptos serán los nodos raíz de la ontología a generar. En los siguientes componentes, estos conceptos base se relacionarán con los nuevos conceptos seleccionados y a partir del enriquecimiento con Datos Enlazados.
2) Similarity Search (SS): Este componente encuentra los posibles conceptos ontológicos a añadir a la ontología base. Para ello, busca los sinónimos de los Términos de Búsqueda, y compara los Términos de Búsqueda y sus sinónimos con los conceptos almacenados en el OD, extrayendo las coincidencias. El resultado es una lista de los conceptos que coinciden con los términos de búsqueda y sus sinónimos.
3) Ontological Alignment (OA): Este componente se encarga de la alineación de los conceptos ontológicos obtenidos en SS, ponderando las relaciones que existen entre los conceptos encontrados con los Términos de Búsqueda y sus sinónimos. La ponderación sigue las siguientes reglas: i. Si un Término de Búsqueda coincide perfectamente con el concepto encontrado, su puntuación es máxima (1). ii. Si un término de búsqueda está parcialmente inmerso en el concepto encontrado, es decir, se presenta como sufijo o prefijo, su puntuación es la mitad de la coincidencia perfecta (0,5). iii. Si un Término de Búsqueda está completamente inmerso como subcadena del concepto encontrado, su puntuación será la cuarta parte de la coincidencia perfecta (0,25). A continuación, castiga con 0 un concepto sin relación semántica con los términos de búsqueda. Por último, selecciona los conceptos que cumplen el umbral de aceptación proporcionado por el solicitante de la ontología. Este umbral define la severidad o permisividad del filtrado de los conceptos seleccionados. El resultado de este componente es una lista de conceptos que se integrarán en la ontología base.
4) Strong Merging (SM): Integra los conceptos seleccionados de las ontologías gestionadas por OD. En este proceso, copia la lista de conceptos seleccionados dentro de la ontología base, considerando las jerarquías conceptuales (nodos padre e hijo, propiedades y relaciones) presentes en cada ontología fuente. El resultado es la ontología base poblada con el conocimiento correspondiente al dominio solicitado.
5) Data Linking (DLi): Enriquece la ontología generada con Datos Enlazados, utilizando el servicio DBpedia Spotlight (https://www.dbpedia-spotlight.org/), que proporciona una solución basada en Datos Enlazados para relacionar palabras clave con identificadores de recursos relacionados en el grafo de conocimiento de Dbpedia. Por último, los recursos encontrados se vinculan a cada concepto de la ontología. Por ejemplo, al buscar «COVID», el servicio devuelve http://dbpedia.org/resource/COVID-19. A continuación, esta respuesta se vincula al concepto COVID de la ontología base.
6) Export Formats (EF): Se encarga de transformar la ontología enriquecida con Datos Enlazados al formato requerido por la interfaz. Entre los formatos manejados actualmente se encuentran JSON-LD, RDF/XML y N-TRIPLES.

3.2.3 *Componentes de la capa Web Services Manager*

A continuación se ofrece una visión general de los componentes que prestan los distintos servicios del sistema.

1) Knowledge Source Management (KSM): Este componente se encarga de ofrecer servicios para ampliar o reducir la base de conocimiento gestionada por el sistema.
2) Ontology Generation (OG): Este componente se encarga de activar el proceso de creación de ontologías para un contexto específico. El resultado de este componente es la creación de una ontología explotando el conocimiento ontológico disponible en el sistema y Datos Enlazados. El servicio ofrecido recibe los parámetros de búsqueda como Términos de Búsqueda, formato de generación de la ontología, entre otros.

3.2.4 *Comportamiento de AOGS*

Nuestro sistema presenta dos comportamientos principales: 1) Gestión del conocimiento y 2) Generación del conocimiento. En esta sección se detalla cada uno de ellos.

1) Gestión del Conocimiento
Objetivo: listar, añadir o borrar las ontologías que se utilizan como fuente de conocimiento gestionado por el sistema.

Descripción general: El proceso mostrado en la Tabla 3.2, comienza cuando el servicio web KSM recibe una petición de gestión de fuente de conocimiento con sus parámetros requeridos (Paso 1).

Tabla 3.2: Macro-algoritmo de la gestión del conocimiento.

Entrada: Tipo de solicitud y sus parámetros
Proceso: **1.** KSM procesa la solicitud. **2.** KSM comprueba los parámetros de la solicitud. **3.** El KSM activa el componente correspondiente al tipo de solicitud. **3.1.** Si el tipo es listar, KSM invoca SL. **3.1.1.** SL solicita el listado de ontologías a OD. **3.2.** Si el tipo es añadir, KSM invoca a AddS con la ontología a añadir. **3.2.1.** AddS solicita a OD que añada la ontología. **3.3.** Si el tipo es eliminar, KSM invoca a DelS con el ID de la ontología que se va a eliminar. **3.3.1.** DelS solicita al OD que borre el ID.
Salida: Solicitud procesada

En el paso 2, KSM verifica los parámetros en función del tipo de solicitud. En el caso de listar, no requiere parámetros adicionales. En el caso de añadir, el parámetro recibido es una ontología que debe añadirse al sistema. En el caso de eliminar, el parámetro recibido es un ID de la ontología que se desea eliminar del sistema. A continuación, KSM activa el componente correspondiente al tipo de solicitud (Paso 3). Si se trata de listar las ontologías que posee el sistema, KSM invoca a SL (Paso 3.1).

A continuación, SL solicita a OD la lista de ontologías (3.1.1). Si hay que añadir una ontología al sistema, KSM invoca a AddS (paso 3.2). A continuación, AddS solicita a OD que añada la ontología al sistema (3.2.1). Si se trata de eliminar una ontología del sistema, KSM invoca a DelS (paso 3.3). A continuación, DelS solicita a OD que elimine la ontología del sistema (3.3.1). Este proceso permite mantener actualizados los conocimientos gestionados por el AOGS.

2) Generación del conocimiento
Objetivo: generar una ontología con Datos Enlazados y el conocimiento ontológico del sistema, utilizando parámetros de búsqueda como términos de búsqueda, umbral de aceptación, entre otros.

Descripción general: La Tabla 3.3 presenta el proceso de Generación de Conocimiento, que comienza cuando el servicio web OG recibe la solicitud de creación de una ontología con los parámetros de generación (Paso 1). A continuación, BOC crea la ontología base y añade los Términos de Búsqueda como nodos raíz (Paso 2). En el paso 3, genera una lista de sinónimos de los términos de búsqueda. A continuación, SS compara los términos de búsqueda y sus sinónimos con los conceptos almacenados en OD (paso 4).

Tabla 3.3: Macro-algoritmo de generación de conocimiento.

Entrada: Términos de Búsqueda, umbral de aceptación y formato de la ontología
Proceso: **1.** La OG tramita la solicitud. **2.** BOC crea la ontología base. **3.** SS busca sinónimos de los Términos de Búsqueda. **4.** SS compara los parámetros de búsqueda con el conocimiento en OD. **5.** SS genera la lista de coincidencias con Términos de Búsqueda y sinónimos. **6.** OA pondera los conceptos obtenidos en la lista de coincidencias. **7.** OA filtra los conceptos que cumplen el umbral de aceptación suministrado. **8.** OA genera la lista de conceptos seleccionados. **9.** SM integra los conceptos seleccionados. **10.** DLi enriquece la ontología con Datos Enlazados. **11.** EF transforma la ontología al formato requerido
Salida: Ontología generada

A continuación, SS genera las listas de coincidencias con los términos de búsqueda y sus sinónimos (paso 5). En el paso 6, OA pondera las relaciones existentes entre las listas de coincidencias con los términos de búsqueda y sus sinónimos (véase la sección 3.2.2.3). Con las ponderaciones, OA filtra los conceptos que cumplen el umbral de aceptación suministrado (Paso 7) y genera la lista de conceptos seleccionados (Paso 8). Por último, SM integra los conceptos seleccionados en la ontología (paso 9). Una vez construida la ontología, DLi procede a buscar y vincular cada concepto de la ontología con el conocimiento disponible en la web utilizando el paradigma de Datos Enlazados (Paso 10). Como último paso, EF transforma la ontología al formato requerido (Paso 11). El resultado de estos procesos es una ontología enriquecida con Datos Enlazados que explota el conocimiento disponible en el sistema.

3.2.5 *Experimentación*

Esta sección presenta un resumen extenso del caso de estudio del generador ontológico [54], enfocado en el dominio del COVID-19, basado en la sección 4 del artículo en Anexo 3.B. Es importante destacar que esta sección del anexo también aborda su aplicación en el dominio de la gestión energética.

En este caso de estudio se toma como fuente de conocimiento las ontologías del dominio COVID-19, que es una enfermedad infecciosa causada por el virus SARS-CoV-2 [24]. La Tabla IV del Anexo 3.B muestra las ontologías vinculadas a COVID-19 que se seleccionaron del repositorio de ontologías https://bioportal.bioontology.org/ontologies.

AOGS activa su primer proceso, que es la Gestión del Conocimiento, este proceso gestiona la carga de ontologías al sistema como fuentes de conocimiento para el proceso de Generación de Conocimiento. Para ello, ejecuta el servicio web KSM con sus parámetros, proceso que se repite con cada ontología que se añade al sistema. Posteriormente, el KSM recibe, procesa y ejecuta los módulos correspondientes al tipo de petición recibida (ver pasos 1, 2 y 3 de la Tabla 3.2). En este caso, activa el módulo AddS (ver paso 3.2 en la Tabla 3.2), que se encarga de añadir la ontología en OD del sistema (ver paso 3.2.1 en la Tabla 3.2).

Luego, AOGS activa al proceso de Generación de Conocimiento, este proceso genera la ontología usando como insumo los Datos Enlazados del sistema. Para ello, se ejecuta el servicio web OG pasando como parámetros los Términos de Búsqueda, el formato de la ontología y el umbral de aceptación. Por ejemplo, la Figura 3.11 indica que la ontología se generará en formato RDF/XML, con un umbral de aceptación del 70%, y el área de interés de la nueva ontología es “COVID” y “test”.

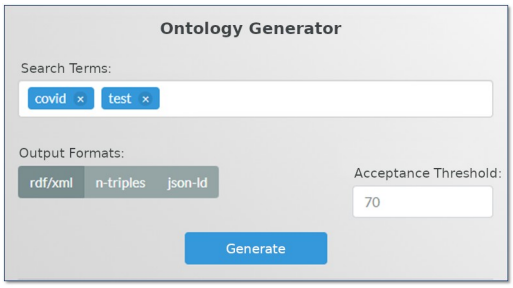

Figura 3.11: Interfaz de generación.

El componente SS, utilizando los términos de búsqueda, obtiene sus sinónimos (véase el paso 3 de la Tabla 3.3). Por ejemplo, para “COVID” se encontró “Coronavirus”, mientras que para “Test” se encontraron varios sinónimos como “trial”, “exam”, “quiz”, entre otros. A continuación, realiza consultas para obtener los conceptos ontológicos de las ontologías añadidas al sistema (véanse los pasos 4 y 5 de la Tabla 3.3). Algunos de los posibles resultados son “Untested for COVID-19”, “COVID-19 Diagnosis”, “Tested for 2019-nCov (Wuhan) infection”. Sin embargo, no todos los conceptos acabarán poblando la ontología, ya que muchos de ellos no están semánticamente relacionados con el dominio consultado. Por ejemplo, “intestine cancer”, “assay screened entity” y “testis”, por lo que será necesario filtrar estos conceptos.

La figura 3.12 muestra algunas ponderaciones entre las listas de coincidencias (azul), y los términos de búsqueda (verde oscuro) y sus sinónimos (verde claro) (véase el paso 6 de la tabla 3.3). Por ejemplo, la comparación de “COVID” de la lista de coincidencias con “COVID” del término de búsqueda es perfecta; por tanto, su puntuación será máxima (1).

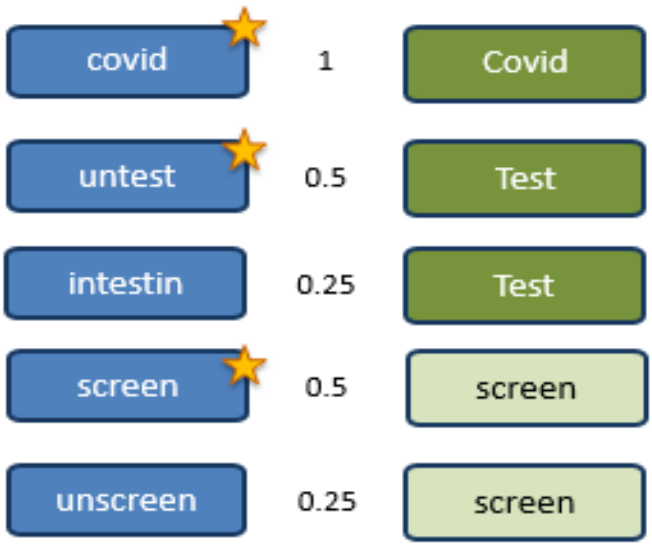

Figura 3.12: Ponderaciones entre los listados de coincidencias (azul) y los términos de búsqueda (verde oscuro) y sus sinónimos (verde claro).

En el caso del término de búsqueda “Test”, está parcialmente inmerso dentro del token “untest”, con un sufijo añadido (en otros casos, puede presentarse como prefijo); por lo tanto, su puntuación es la mitad de la coincidencia perfecta (0,5). En el tercer caso, el término buscado está completamente inmerso como una subcadena del concepto presente en el listado (“Test” en “intestin”). En este caso, la ponderación es una cuarta parte de la coincidencia perfecta (0,25) para castigar los posibles conceptos sin relación semántica con la búsqueda. Así se hace para el resto de conceptos.

En resumen, un concepto cuya etiqueta sea “Tested for COVID-19” puntuará más alto que otro cuya etiqueta sea “Study for COVID propagation”, ya que en el primer caso se hace referencia a “Test” y “COVID” al mismo tiempo, mientras que en el segundo sólo se hace referencia a “COVID”. Tras la ponderación, se eligen los conceptos que cumplen el umbral de aceptación definido (véase el paso 7 de la Tabla 3.3). Para este caso, 70% es el umbral, y el resultado es una lista con los conceptos seleccionados (véase la etapa 8 de la Tabla 3.3).

Con la ontología base integrada con los conceptos seleccionados (véase el paso 9 de la Tabla 3.3), se utiliza el paradigma de Datos Enlazados para buscar conceptos que puedan ser equivalentes a los conceptos de la ontología generada (véase el paso 10 de la Tabla 3.3), con el fin de crear nuevos conceptos en la ontología con información externa. La Figura 3.13 muestra el vínculo entre el concepto COVID de la ontología y el concepto COVID de DBpedia.

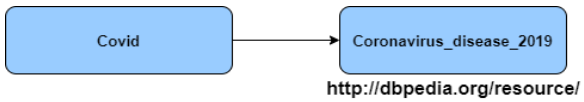

Figura 3.13: Vinculación de conceptos ontológicos con fuentes externas de Datos Enlazados.

Por último, la nueva ontología se transforma al formato requerido (véase el paso 11 de la Tabla 3.3), en este caso, a RDF/XML (véase la Figura 3.14).

Figura 3.14: Ontología generada y publicada en formato RDF/XML.

4 Recomendador Híbrido Basado en Lógica Descriptiva/Dialéctica y Datos Enlazados

En este capítulo se presenta un Sistema de Recomendación Híbrido (Hybrid Recommender System, HRS) que combina la lógica descriptiva/dialéctica con Datos Enlazados [56]. Este HRS nace de las ideas propuestas por Dos Santos et al. [46] y responde a la necesidad de resolver situaciones con información inconsistente, es decir, estados de contradicción o ambigüedad, y de explotar la información semántica proveniente de la web estructurada como los Datos Enlazados. La estructura del capítulo es la siguiente: La sección 4.1, que se basa en la sección 4.1 del Anexo 4.A, presenta el diseño arquitectónico nuestro HRS y detalla cada componente. La sección 4.2, en consonancia con la Sección 4.2 del Anexo 4.A, describe los algoritmos que implementan los componentes del HRS. La sección 4.3 ilustra un caso de estudio del HRS, basándose en la sección 5 del Anexo 4.A. Finalmente, la sección 4.4 explora el uso de esta arquitectura, incluyendo un análisis de uno de los fenómenos dialécticos en las competencias (sección 4.4.1, basada en la sección Knowledge Model del Anexo 4.B), una presentación de otros modelos dialécticos (sección 4.4.2, basada en la sección 3 del Anexo 4.C) y un análisis general del potencial de la Lógica Dialéctica (sección 4.4.3).

4.1 Arquitectura del HRS

Esta sección ofrece un resumen extenso del trabajo presentado en [56], y los detalles se encuentran en la sección 4.1 del Anexo 4.A. En dicho trabajo, se construye un HRS basado en las características de un Sistema de Recomendador Inteligente, las cuales se describen a continuación [57]: (i) Fuentes de Conocimiento: son las fuentes que proveen información sobre usuarios, contexto, recursos, entre otros. En nuestro sistema, estará conformada principalmente por fuentes de Datos Enlazados, las cuales proveen datos con información semántica; (ii) Adquisición de Conocimiento: se encarga de la extracción y procesamiento de datos. En nuestro caso, generará consultas en SPARQL que permitan identificar, filtrar y extraer la información disponible en las fuentes de Datos Enlazados para enriquecer el modelo de conocimiento; (iii) Modelado de Conocimiento: se especifica el paradigma de representación del conocimiento; para nuestro caso, se representa como Datos Enlazados y axiomas basados en lógica descriptiva/dialéctica; (iv) Razonamiento y verificación: se implementan los mecanismos de razonamiento, ofreciendo la capacidad de explotar el conocimiento e inferir recomendaciones. En nuestro sistema, se utilizará un mecanismo híbrido, por un lado, se utiliza un razonador de lógica descriptiva que permite explotar las fuentes de Datos Enlazados para enriquecer el modelo de conocimiento, y por otro lado, se utiliza un razonador de Lógica Dialéctica que verifica el modelo de conocimiento y evalúa las recomendaciones a través de los diferentes eventos dialécticos.

La Figura 4.1 muestra los componentes de nuestro HRS, distribuidos en dos grandes grupos: el primer grupo está compuesto por los componentes que permiten razonar para extraer información o inferir recomendaciones, incluso en presencia de inconsistencias o ambigüedades en el Problema o Consulta (PoQ), o en los datos extraídos de los Datos Enlazados, llamados *Reasoning Engines*. El segundo grupo, llamado *Manager*, está compuesto por los componentes encargados de gestionar todos los procesos necesarios para llegar a una recomendación. Estos determinan cuándo y qué se debe explotar

de los Datos Enlazados, ya sea para enriquecer las ontologías o vocabularios del modelo de razonamiento, o para enriquecer semánticamente los datos o recomendaciones encontradas.

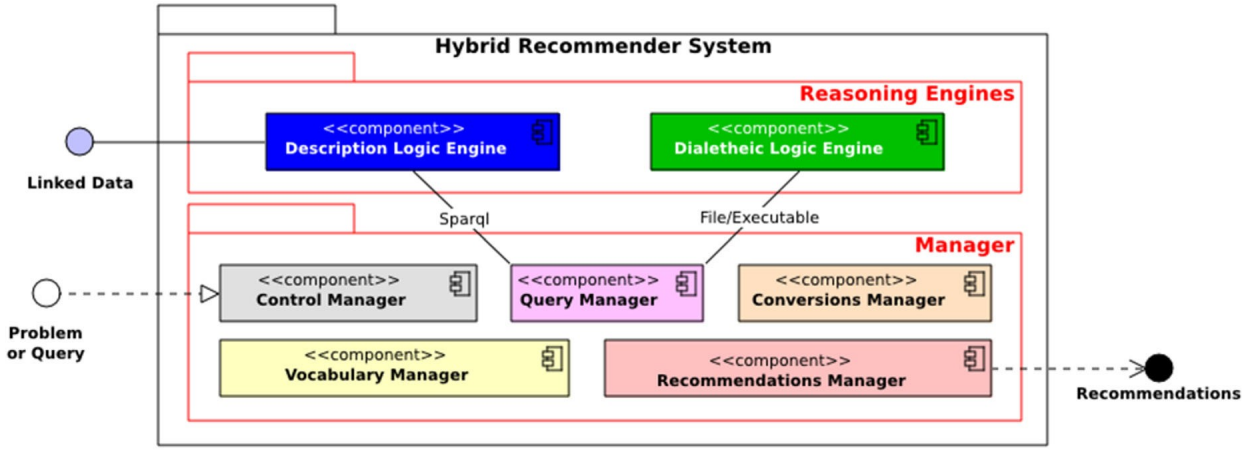

Figura 4.1: Diagrama de componentes de nuestro HRS.

4.1.1 *Componentes del Grupo Reasoning Engines*

Description Logic Engine (DeLE): Este motor es intrínseco a los Datos Enlazados, ya que tanto la estructura semántica de los datos como los mecanismos de consulta (lectura, creación, actualización o borrado de tripletes) se basan en lógica descriptiva. Además, el motor permite explotar diferentes fuentes de datos, gracias a la técnica de Datos Enlazados que interconecta los datos a través de puntos de acceso distribuidos o endpoint locales (para perfiles personales y contexto), o endpoint públicos (endpoint de Dbpedia, endpoint de Wikidata, entre muchos otros). Este motor recibe una consulta basada en tripletes y devuelve los datos encontrados como variables.

Dialetheic Logic Engine (DiLE): Este motor responde mediante consultas construidas como conjeturas sobre los modelos descritos en lógica de primer orden, donde se detallan axiomas y sus hechos, pudiendo detectar y razonar en estados de ambigüedad o inconsistencia, gracias a la capacidad que ofrece la Lógica Dialéctica. Este motor recibe una consulta como conjetura y modelo, y retorna lo inferido de la conjetura sobre el modelo.

4.1.2 *Componentes del Grupo Manager*

Vocabulary Manager (VM): se encarga de identificar y seleccionar los vocabularios y ontologías que son necesarios para procesar las peticiones recibidas por el recomendador. El gestor busca hacer coincidir los términos de la petición con las clases y propiedades del conocimiento que posee. Si este objetivo no se logra, entonces se apoya en Query Manager para extraer nuevo conocimiento de las fuentes de Datos Enlazados, que pueda enriquecer las ontologías o vocabularios del modelo de razonamiento. Este componente toma los axiomas presentes en la entrada del PoQ y extrae los predicados, ya que esos predicados son los conceptos que se necesitan identificar. Luego, esos

predicados se comparan con la base de conocimiento, para determinar si se conocen los URIs que identifican y se relacionan con los conceptos. Si no es así, se invoca el Query Manager para generar consultas para los URIs que representan estos conceptos desconocidos (más en la Tabla 4.3).

Query Manager (QM): se encarga de preparar y generar las consultas necesarias para enriquecer o recomendar la información.

- Por parte del DeLE, se generan consultas basadas en tripletas que explotan los datos contenidos en las fuentes de Datos Enlazados, apoyándose en la Base de Conocimiento (Knowledge Base, KB) de los vocabularios y ontologías manejados por el recomendador y las fuentes de Datos Enlazados. En concreto, la consulta se construye a través de tres plantillas, según el tipo de petición: (i) "*Concept as URI"* permite buscar la URI que representa un concepto. (ii) "*Property related to a Concept as URI"* permite buscar la URI que representa una propiedad que está relacionada con una URI de un concepto conocido.; y (iii) "*Knowledge Extraction"* permite extraer todo el conocimiento disponible de una URI.
- Por parte del DiLE, se generan consultas basadas en conjeturas descritas en los axiomas del PoQ, que permiten probar e inferir las recomendaciones, incluso si hay inconsistencia o ambigüedad en el PoQ o en los datos. En concreto, se analiza cada axioma del PoQ y se extrae el predicado que sigue a cada símbolo de implicación (=>), ya que formará parte de las conjeturas que el HRS utilizará para probar el modelo y los datos.

Conversion Manager (ConM): Se encarga de preparar y generar las transformaciones de datos para permitir el intercambio de información entre los diferentes razonadores que tiene el recomendador y la información encontrada, necesarias para que los resultados de un razonador puedan ser utilizados por el otro. Por ejemplo, para utilizar los datos generados por el DeLE con DiLE, se requiere transformar los datos, ya que el DeLE devuelve una tabla inferida con los valores encontrados por cada variable solicitada, y el DiLE necesita un modelo con los datos descritos como hechos en Lógica de Primer Orden. Estas transformaciones se crean a partir del modelo de razonamiento, utilizando la base de conocimiento que alcanzó VM y la consulta de extracción de conocimiento que genera QM. En concreto, se realizan dos tipos de conversiones, el primer tipo convierte los datos de una variable o concepto extraído de los Datos Enlazados. El segundo tipo convierte los datos de dos variables o conceptos relacionados que se extrajeron de los Datos Enlazados.

Recommendation Manager (RM): Se encarga de fusionar y filtrar la información obtenida por los razonadores, permitiendo recoger y entregar el conocimiento alcanzado por todos los mecanismos que componen el recomendador. En concreto, RM solicita a ConM las conjeturas que permitirán validar el modelo de razonamiento con los elementos extraídos a través de DeLE. A continuación, DiLE filtra las recomendaciones; para ello, se detectan los eventos dialécticos utilizando las diferentes conjeturas. Ahora, con las recomendaciones alcanzadas, se procede a evaluar el grado de cercanía de cada recomendación al perfil del usuario. Esta evaluación consiste en sumar las características del perfil de usuario que coinciden con la recomendación. Finalmente, las recomendaciones se ordenan de mayor a menor, según el resultado de cada evaluación (más detalles en la Tabla 4.5).

Control Manager (CM): es el responsable de todas las decisiones del HRS, determinando cuándo y cómo deben invocarse los gestores y razonadores en función de sus capacidades. Este gestor utiliza el meta-razonamiento para tomar estas decisiones (más detalles en la Tabla 4.2). El meta-razonamiento busca cumplir los siguientes objetivos: (i) verificar que el PoQ esté correctamente definido, probándolo con DiLE (ver Tabla 4.2); (ii) identificar los conceptos presentes en el PoQ, buscándolos con DeLE

(ver Tabla 4.3); (iii) extraer el conocimiento asociado a los conceptos identificados en el PoQ, extrayéndolos con DeLE (ver Tabla 4.4); (iv) verificar y filtrar las recomendaciones, usando DiLE (ver Tabla 4.5); y (v) extraer el contenido relacionado con las recomendaciones, usando DeLE (ver Tabla 4.6).

4.2 Funcionamiento del HRS

Esta sección es un resumen extenso del trabajo presentado en la sección 4.2 del Anexo 4.A, dónde se especifica el funcionamiento del HRS construido. En general, los pasos seguidos por el HRS pueden verse en el Macro-Algoritmo de la Tabla 4.1. El proceso comienza cuando un usuario solicita una recomendación, proporcionando la PoQ que desea resolver. Lo primero que hace CM es ejecutar la entrada con DiLE, determinando si los axiomas están correctamente descritos (Paso 1). Si no son correctos, el proceso se detiene (Paso 1.1). En caso de que sea correcta, entonces el CM activa una serie de procesos que permiten incorporar conocimiento de los Datos Enlazados y encontrar posibles recomendaciones (Paso 2). En el paso 2.1, se determina si existen ontologías y/o vocabularios que permitan identificar los términos utilizados por el PoQ. En el paso 2.2, se extrae de los Datos Enlazados la información relativa a los términos identificados en el paso anterior. Esto permite el enriquecimiento con conocimiento de las posibles recomendaciones que se alcanzarán. En el paso 2.3, se verifican las inconsistencias y/o ambigüedades presentes en los datos recogidos para el PoQ, y se obtienen las recomendaciones encontradas. En el paso 2.4, estas recomendaciones se enriquecen con el conocimiento relacionado buscado a través de las fuentes de Datos Enlazados. Por último, se entregan las recomendaciones enriquecidas.

Tabla 4.1: Macro-algoritmo de nuestro HRS.

Entrada: PoQ
Proceso: **1.** El CM verifica la PoQ con el DiLE. **1.1.** Si no es correcto el PoQ, se detiene el proceso de recomendación. **2.** CM activa cada proceso. **2.1. Proceso:** Identificación de URI utilizando Datos Enlazados (véase la Tabla 4.2). **2.2. Proceso:** Extracción de Conocimiento utilizando Datos Enlazados (véase la Tabla 4.3). **2.3. Proceso:** Verificación y Filtrado de Recomendaciones utilizando Datos Enlazados (Tabla 4.4). **2.4. Proceso:** Extracción de Contenidos Relacionados a las Recomendaciones utilizando Datos Enlazados (Tabla 4.5).
Salida: Recomendaciones enriquecidas

A continuación se describen con más detalle los distintos procesos:

4.2.1 *Identificación de URI utilizando Datos Enlazados*

Objetivo: identificar los términos o conceptos utilizados en el PoQ, buscando URIs de una ontología y/o vocabulario que los represente.

Descripción general: El proceso mostrado en la Tabla 4.2, comienza cuando el CM recibe la PoQ y activa la VM para reconocer las características de dicha PoQ (Paso 1). En el paso 2, la VM verifica si los Términos o Conceptos (ToC) presentes en el PoQ están relacionados con un vocabulario u ontología conocida por el HRS en su KB. Esta relación se identifica a través de un URI, que permite extraer nuevo conocimiento de las fuentes de Datos Enlazados. Por ejemplo, una enfermedad se representa mediante el URI http://dbpedia.org/ontology/disease. Para ello, el paso 2.1 extrae cada ToC en PoQ (véase el componente VM). A continuación, se comprueba cada ToC extraído de PoQ (paso 2.2). Si HRS no conoce la ToC en su KB, VM debe encontrar un URI que represente la ToC (paso 2.2.1). En este caso, VM activa QM (Paso 2.2.1.1), de modo que QM prepara consultas para encontrar un URI a través de DeLE que represente el ToC (Paso 2.2.1.2). Ejemplos de este proceso se describen en el componente QM, concretamente, en los tipos de consulta *Concept as URI* y *Property related to a Concept as URI*. En el paso 2.2.1.3, VM ejecuta DeLE con las consultas recibidas por QM, lo que le permite encontrar un URI que represente ToC. Entonces, ToC, con su URI, es añadido a KB como un concepto identificado por HRS (Paso 2.2.1.4). Todos estos URIs identificados en este proceso permitirán la extracción de conocimiento para el siguiente proceso.

Tabla 4.2: Macro-algoritmo de identificación de URIs mediante Datos Enlazados.

Entrada: PoQ
Proceso: **1.** El CM activa VM. **2.** VM comprueba si ToC está en KB: **2.1.** VM extrae los ToC presentes en el PoQ. **2.2.** Para cada ToC del PoQ: **2.2.1.** Si ToC no está Identificado, requiere actualización de KB: **2.2.1.1.** VM activa QM. **2.2.1.2.** QM prepara las consultas para extraer las URI. **2.2.1.3.** VM invoca DeLE con las consultas generadas. **2.2.1.4.** Se añade la ToC a la KB.
Salida: KB actualizados

4.2.2 *Extracción de Conocimiento utilizando Datos Enlazados*

Objetivo: extraer la información que debe recomendarse utilizando las URIs como punto de correspondencia entre las fuentes de Datos Enlazados y ToC.

Descripción general: La Tabla 4.3 presenta el proceso de Extracción de Conocimiento, que comienza cuando el CM ha identificado cada ToC con sus respectivas URIs, y necesita extraer información de las fuentes de Datos Enlazados para encontrar posibles recomendaciones (Paso 1). El CM lleva a cabo esta tarea activando QM (Paso 1.1), de forma que QM prepara las consultas para extraer información de los Datos Enlazados asociados a las PoQ utilizando KB (Paso 1.2). Se realiza de la siguiente forma: utilizando los URIs conocidos del proceso anterior y la tripleta para el tipo de consulta “Knowledge Extraction” (ver Componente QM), ahora bien, en los casos en que dos conceptos están relacionados, sustituye ?URI_CONCEPT y ?URI_PROPERTY por los URIs respectivos. En los casos en que un concepto no está relacionado con otros, solo se sustituye ?URI_CONCEPT. Teniendo las consultas, el

CM invoca el DeLE y recoge información de fuentes, como la preferencia del usuario, el contexto y otros conocimientos asociados al PoQ (Pasos 1.3 y 1.4).

Tabla 4.3: Macro-algoritmo de extracción de conocimiento mediante Datos Enlazados.

Entrada: PoQ
Proceso: **1.** CM necesita extraer información para el PoQ utilizando KB. **1.1.** CM activa QM. **1.2.** QM prepara las consultas para extraer la información asociada al PoQ utilizando la KB. **1.3.** CM invoca DeLE con consultas a la fuente local (conocimiento del contexto y preferencias del usuario). **1.4.** CM invoca DeLE con consultas a otras fuentes (conocimiento general).
Salida: Conocimientos Extraídos

4.2.3 *Verificación y Filtrado de Recomendaciones utilizando Datos Enlazados*

Objetivo: verificar las inconsistencias y/o ambigüedades presentes en los datos recogidos y en el modelo, y generar recomendaciones.

Descripción general: El proceso comienza cuando se ha identificado la ToC y se han extraído todos los datos utilizando el paradigma de Datos Enlazados (Tabla 4.4). Entonces, el CM activa el RM para generar las recomendaciones (Paso 1). El RM lleva a cabo una serie de procesos para conseguir las recomendaciones. El primero es convertir los datos recogidos por DeLE en el proceso anterior a la estructura recibida por DiLE. Para ello, se activa la tarea ConM (Paso 2). En el Paso 2.1, ConM realiza esta tarea mediante dos tipos de conversiones, que se describen en el componente ConM. La segunda es activar QM (Paso 3) para generar las consultas basadas en el modelo que recibe DiLE. Para ello, QM, en el Paso 3.1, debe entregar las consultas basadas en conjeturas según los axiomas presentes en PoQ. Este proceso se describe en el componente QM. Por último, RM ejecuta DiLE con PoQ, los datos transformados por ConM y las distintas conjeturas entregadas por QM (Paso 4). DiLE comprueba los datos y las conjeturas (paso 4.1). Con los resultados proporcionados por DiLE, las recomendaciones se filtran (paso 4.2) y se clasifican (paso 4.3), como se describe en el componente RM.

Tabla 4.4: Macro-algoritmo de verificación y filtrado de recomendaciones mediante Datos Enlazados.

Entrada: PoQ y los Conocimientos Extraídos
Proceso: **1.** CM activa RM para buscar recomendaciones. **2.** RM activa ConM. **2.1.** ConM convierte el Conocimiento Extraído según la necesidad del DiLE. **3.** RM activa QM. **3.1.** QM genera las consultas según el PoQ utilizando KB. **4.** RM busca recomendaciones ejecutando DiLE. **4.1.** RM verifica los datos. **4.2.** RM filtra los resultados. **4.3.** RM ordena los resultados.
Salida: Recomendaciones

4.2.4 *Extracción de Contenidos Relacionados a las Recomendaciones utilizando Datos Enlazados*

Objetivo: enriquecer las recomendaciones con contenidos relacionados extraídos a partir de los Datos Enlazados.

Descripción general: La tabla 4.5 muestra el proceso de enriquecimiento de las recomendaciones. En el paso 1, el CM activa a RM para que busque información relacionada con las recomendaciones alcanzadas en el proceso anterior (Paso 1). Para alcanzar este objetivo, el RM activa a QM (Paso 2), que se encargará de generar las consultas necesarias para extraer de las fuentes de Datos Enlazados, los contenidos relacionados con las recomendaciones (Paso 2.1). Se realiza utilizando las URIs de las recomendaciones alcanzadas del proceso anterior, y la tripleta para el tipo de consulta "Knowledge Extraction" (ver componente QM), sustituyendo ?URI_CONCEPT por la URI de cada recomendación alcanzada. Cuando se utiliza este URI, se evita la ambigüedad con respecto a las recomendaciones alcanzadas en procesos anteriores. A continuación, RM invoca a DeLE utilizando las consultas generadas por QM (Paso 3). Por último, el contenido relacionado extraído de las fuentes de Datos Enlazados se vincula a las recomendaciones alcanzadas en el proceso anterior, y se entrega como resultado final (Paso 4).

Tabla 4.5: Macro-algoritmo de extracción de contenidos relacionados con las recomendaciones mediante Datos Enlazados.

Entrada: Recomendaciones
Proceso: **1.** CM activa RM para enriquecer las recomendaciones. **2.** RM activa QM. **2.1.** QM genera consultas de acuerdo con las recomendaciones. **3.** RM activa DeLE con las consultas. **4.** RM fusiona y devuelve las recomendaciones enriquecidas.
Salida: Recomendaciones Enriquecidas

4.3 Experimentación

Esta sección presenta un caso práctico, que es un resumen extenso del trabajo presentado en la sección 5 del Anexo 4.A, en el que se detalla el comportamiento de nuestro sistema. Además, se muestra cómo se explotarían los recursos de los Datos Enlazados dentro del recomendador. La Figura 4.2 muestra las fuentes de entrada para nuestro HRS en este caso práctico. Además, se lleva a cabo la siguiente suposición: los doctores y/o los sensores corporales solo detectan los diferentes síntomas que puede presentar un usuario, y dicha información se almacena en un repositorio local de Datos Enlazados, que se implementa con OpenLink Virtuoso (Open-Source Edition). Por otro lado, existen datos asociados a tipos de enfermedades conocidas, así como sus síntomas, tratamientos y causas, entre otros. Esta

información se extrae del repositorio de Datos Enlazados externo, llamado Live-DBpedia, que es una fuente con datos actualizados, ya que recupera inmediatamente todos los cambios de Wikipedia.

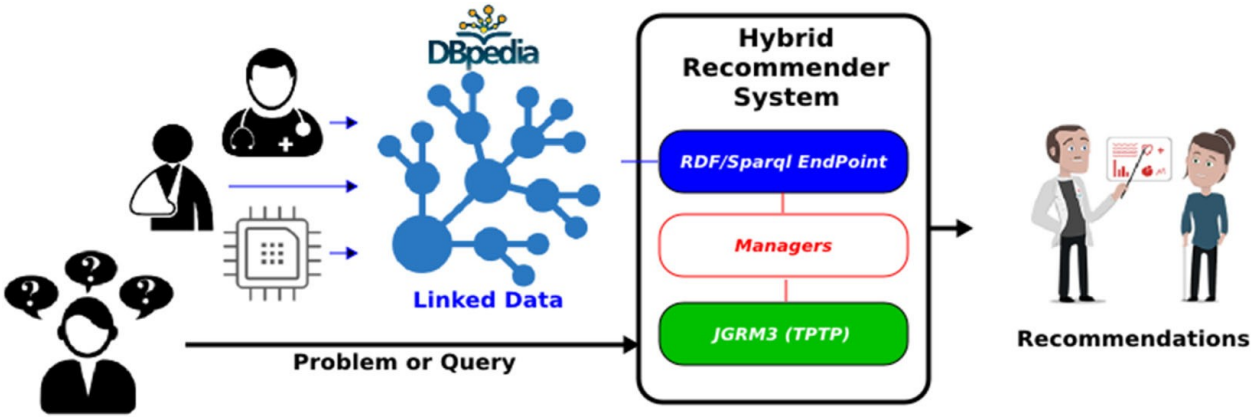

Figura 4.2: Caso de estudio del HRS.

Además, también necesita PoQ como entrada, que es el problema o consulta a resolver. Esta entrada será recibida por el CM, que se encarga de controlar todo el proceso llevado a cabo por el HRS. En este caso, busca una PoQ que represente una situación de Lógica Dialéctica denominada *Discurso Ficticio*. Para ello, busca determinar si una persona está enferma, de acuerdo con ciertos supuestos. En concreto, nuestro sistema debe gestionar los síntomas que pueden o no asociarse a una enfermedad, los conflictos entre los síntomas proporcionados por cada doctor o sensor, la falta de información, la inconsistencia de los datos, etc. Así, nuestro sistema permite gestionar las diferentes ambigüedades entre las opiniones de los doctores y/o la información captada por los sensores. De esta forma, se compone de cuatro axiomas que describen el problema, y dos conjeturas que representan las preguntas a resolver (ver Figura 3 del Anexo 4.A del artículo [56]):

1. Enfermo de D porque U presenta alguno de los síntomas S, según el Doctor Dr (ver axioma diseaseDoctor). Este axioma determina si alguno de los S síntomas es detectado por el Dr (symptomDoctor) en el usuario U. A continuación, se comprueba si el síntoma detectado está asociado a la enfermedad D (isSymtom).
2. Enfermo de D según la opinión del Doctor Dr, porque U presenta alguno de los síntomas S (ver axioma sickDoctor). Este axioma determina si existe una enfermedad D en el usuario U, según la opinión del doctor Dr (diseaseDoctor). Este axioma se basa en el axioma 1, porque el doctor Dr detecta cualquiera de los S síntomas.
3. Enfermo con D porque U presenta alguno de los síntomas S, según los doctores Dr1 y Dr2 (ver axioma diseaseDoctors). Este axioma también se basa en el axioma 1. Realiza una doble comprobación de las opiniones de los doctores sobre los síntomas, de forma que determina si la enfermedad D está presente en el usuario U basándose en estas opiniones de los doctores Dr1 y Dr2.
4. Enfermo según la opinión de los doctores Dr1 y Dr2, porque U presenta alguno de los síntomas S (ver axioma sickDoctors). Este axioma se basa en los axiomas 2 y 3, y determina si existe una enfermedad D en el usuario U, según las opiniones de los doctores Dr1 y Dr2 (diseaseDoctors).
5. Conjetura i: sickDoctors. El User_A está enfermo según la opinión del doctor_A y del doctor_B.

6. Conjetura ii: diseaseDoctors. El User_A está enfermo de D según la opinión del doctor_A y del doctor_B. D pertenece al conjunto de enfermedades disponibles en la base de conocimientos.

Por último, el HRS con las entradas definidas está listo para iniciar la extracción del conocimiento necesario para razonar y hacer sus recomendaciones.

Estos procesos se muestran a continuación

4.3.1 *Identificación de URIs mediante Datos Enlazados*

Este proceso muestra cómo la HRS identifica las ToC presentes en la entrada PoQ, y sus relaciones. Para ello, intenta asociar cada ToC a una URI que la represente (véase el macro-algoritmo de la Tabla 4.3). En este caso, se asume que la base de conocimiento del HRS conoce el ToC del usuario y del doctor, y las relaciones entre ellos, usando solo el vocabulario FOAF (ver Tabla 4.6).

Tabla 4.6: Base de conocimientos HRS sobre los ToC.

ToC		**URI**
user		foaf:Person
doctor		foaf:Person
opinionDoctor		foaf:Document
publications		foaf:publications
maker		foaf:maker
Subject	**Property**	**Object**
user	publications	opinionDoctor
opinionDoctor	maker	Doctor

Nota: foaf: http://xmlns.com/foaf/0.1/.

El proceso comienza cuando VM es activada por CM para identificar la ToC (véase el paso 1 en la Tabla 4.2). A continuación, VM obtiene los predicados disease, isSymptom y symptomDoctor, y con ellos extrae los ToC disease, symptom y doctor (paso 2.1 en la Tabla 4.2). Si no identifica los ToC disease y symptoms, y la relación entre ellos (paso 2.2.1 en la Tabla 4.2). VM resuelve este problema con la invocación a QM para que prepare consultas para extraer información de los Datos Enlazados a través del DeLE (paso 2.2.1.1 en la Tabla 4.2). De esta forma, para cada ToC no identificada, se busca una ontología que la describa. La Figura 4.3 muestra una consulta definida por QM (paso 2.2.1.2 en la Tabla 4.2), un proceso que se describió previamente en el componente QM, para extraer el URI. Esta consulta busca una ontología para la enfermedad ToC que esté contenida en la fuente de Datos Enlazados Live-DBpedia, obteniendo como resultado el URI para representar enfermedades: “http://dbpedia.org/ontology/disease” (paso 2.2.1.3 en la Tabla 4.2).

```
sparql = SPARQLWrapper2("http://live.dbpedia.org/sparql")
sparql.setQuery("""
SELECT DISTINCT ?URI
WHERE {
   [] a ?URI
   FILTER regex(?URI,'ontology', "i" ) .
   FILTER regex(?URI,'disease', "i" ) .
}
""")
```

Figura 4.3: Consulta para encontrar una ontología para los términos o conceptos: disease.

Por último, VM utiliza técnicas de Datos Enlazados y la información contenida en las fuentes de los Datos Enlazados (paso 2.2.1.4 en la Tabla 4.2), para identificar y relacionar los términos de la entrada PoQ (véase la Tabla 4.7).

Tabla 4.7: Nuevos términos o conceptos identificados y relacionados con sus URIs mediante Datos Enlazados.

ToC		**URI**
disease		dbo:disease
symptom		dbo:symptom
isSymptom		dbo:symptom
Subject	**Property**	**Object**
disease	isSymptom	symptom
opinionDoctor	isSymptom	symptom

Nota: dbo: http://dbpedia.org/ontology/.

Los procesos descritos en esta sección están automatizados en Python. El primer proceso (componente VM) extrae los predicados de los axiomas descritos en PoQ. Esos predicados se transforman en ToC, que son los términos que se buscarán en la fuente los Datos Enlazados. El segundo proceso utiliza las plantillas descritas en QM como “Concept as URI” o “Property related to a Concept as URI”, donde las palabras CONCEPT o PROPERTY se sustituyen por el ToC que necesita identificar su URI. A continuación, la consulta de búsqueda se ejecuta en el punto final de la fuente LOD (por ejemplo, http://live.dbpedia.org/sparql). Este proceso se repite con todos los ToC que necesitan ser identificados, y si alguno de estos ToC no está asociado a un URI, entonces el sistema detiene su ejecución e indica el problema.

4.3.2 *Extracción de conocimientos mediante Datos Enlazados*

En este caso, extrae información de las fuentes de Datos Enlazados para enriquecer el conocimiento sobre el ToC identificado a partir de la entrada PoQ (ver macro-algoritmo en la Tabla 4.3). Este proceso comienza cuando CM activa QM para generar dos consultas (pasos 1.1 y 1.2 en la Tabla 4.3), estas

consultas se generan siguiendo el proceso descrito en el componente QM para extraer el conocimiento. Por un lado, genera una consulta que extrae de la fuente local de Datos Enlazados los síntomas detectados en el user_A por doctores o sensores (Figura 4.4).

```
sparql = SPARQLWrapper2("http://localhost/sparql")
sparql.setQuery("""
PREFIX local: <http://localhost/>
SELECT ?symptom ?doctor
WHERE {
   local:user_A foaf:publications ?opinionDoctor .
   ?opinionDoctor foaf:maker ?doctor .
   ?opinionDoctor dbo:symptom ?symptom .
}
""")
```

Figura 4.4: Consulta para extraer los síntomas detectados en el user_A.

Por otro lado, genera una consulta que extrae una lista de enfermedades con sus síntomas, a partir de la fuente Live-Dbpedia de los Datos Enlazados (véase la Figura 4.5).

```
sparql = SPARQLWrapper2("http://live.dbpedia.org/sparql")
sparql.setQuery("""
SELECT ?disease ?symptom
WHERE{
      ?disease dbo:symptom ?symptom .
}
order by ?disease
""")
```

Figura 4.5: Consulta que genera la lista de enfermedades y sus síntomas.

Por último, CM ejecuta las consultas (pasos 1.3 y 1.4 de la Tabla 4.3). La Tabla 4.8 muestra una pequeña parte de los datos extraídos sobre las enfermedades, con sus síntomas.

Tabla 4.8: Enfermedades con sus síntomas extraídos de Datos Enlazados.

Enfermedad	**Síntoma**
dbpedia_Volvulus	dbpedia_Bloating
dbpedia_Volvulus	dbpedia_Constipation
dbpedia_Zika_fever	dbpedia_Conjunctivitis
dbpedia_Zika_fever	dbpedia_Fever

Este proceso se automatiza en Python, para lo cual utiliza la plantilla descrita en QM “Knowledge Extraction”, donde sustituye la palabra URI_PROPERTY por la URI del ToC que necesita extraer el conocimiento. A continuación, procede a ejecutar la consulta de búsqueda en el endpoint de la fuente

LOD (por ejemplo, http://live.dbpedia.org/sparql). Este proceso se repite con todos los ToC que necesitan extraer conocimiento.

4.3.3 *Verificación y filtrado de recomendaciones mediante Datos Enlazados*

Este proceso verifica las incoherencias y/o ambigüedades de los datos extraídos, y los filtra basándose en los eventos dialécticos encontrados por DiLE (véase el macroalgoritmo del Cuadro 4.4). Sin embargo, para lograr este objetivo, es necesario activar ConM y QM. ConM (paso 2.1 en la Tabla 4.4) transforma los datos extraídos por DeLE (véase la Tabla 4.8) en especificaciones DiLE, basadas en axiomas y hechos. Este proceso de transformación se ha explicado en el componente ConM. La Figura 4.6 muestra parte del resultado de la conversión de las enfermedades y sus síntomas, donde el axioma isSymptom_type representa la relación entre enfermedad y síntoma, y el resto son los hechos que representan los datos extraídos.

```
fof(isSymptom_type,axiom,(
    ! [D] :
      ( disease(D)
     => ? [S] :
          ( symptom(S)
          & isSymptom(D,S) ) ) )).

fof(isSymptom1, axiom, isSymptom(dbpedia_17Hydroxysteroid_dehydrogenase_III_deficiency , dbpedia_Hypothyroidism) ).
                       :                                :                                :
                       :                                :                                :
fof(isSymptom1333, axiom, isSymptom(dbpedia_Volvulus , dbpedia_Bloating) ).
fof(isSymptom1334, axiom, isSymptom(dbpedia_Volvulus , dbpedia_Constipation) ).
fof(isSymptom1335, axiom, isSymptom(dbpedia_Volvulus , dbpedia_Bloody_stool) ).
fof(isSymptom1336, axiom, isSymptom(dbpedia_Volvulus , dbpedia_Abdominal_pain) ).
                       :                                :                                :
                       :                                :                                :
fof(isSymptom1356, axiom, isSymptom(dbpedia_Zika_fever , dbpedia_Conjunctivitis) ).
fof(isSymptom1357, axiom, isSymptom(dbpedia_Zika_fever , dbpedia_Fever) ).
fof(isSymptom1358, axiom, isSymptom(dbpedia_Zika_fever , dbpedia_Maculopapular_rash) ).
fof(isSymptom1359, axiom, isSymptom(dbpedia_Zika_fever , dbpedia_Arthralgia) ).
```

Figura 4.6: Información sobre enfermedades y sus síntomas convertida para el motor de Lógica Dialéctica.

La Figura 4.7 muestra las conjeturas extraídas del PoQ por el QM (paso 3.1 en la Tabla 4.4). Con estas conjeturas, DiLE podrá razonar en el siguiente paso con el modelo y sus datos, permitiendo filtrar las recomendaciones.

```
%%%%%% Conjecture
fof(i,conjecture,( sickDoctors(user_A, doctor_A, doctor_B) )).
fof(ii,conjecture,( diseaseDoctors(user_A, D, doctor_A, doctor_B) )).
```

Figura 4.7: Conjeturas para verificar y filtrar los datos.

Ahora, el RM puede invocar al DiLE para verificar y filtrar las recomendaciones (paso 4 de la Tabla 4.4). En el caso de la conjetura I (véase la figura 4.8), DiLE determina que el user_A está enfermo basándose en la opinión del doctor_A y del doctor_B. En el caso de la conjetura II (ver Figura 4.8), DiLE comprueba cada enfermedad (variando el valor de D en la conjetura II) que tiene en la base de conocimiento, para verificar y filtrar las enfermedades a recomendar. La Figura 4.8 muestra la

verificación de dos enfermedades (dbpedia_Zika_fever y dbpedia_Volvulus) con la conjetura ii. Para dbpedia_Zika_fever DiLE determina que es una enfermedad a recomendar (YES), ya que en ambas los doctores determinan que existen síntomas asociados a la enfermedad. Para dbpedia_Volvulus DiLE determina que no es una enfermedad para recomendar (NO), ya que solo el doctor_B opinó que tiene síntomas asociados a esa enfermedad.

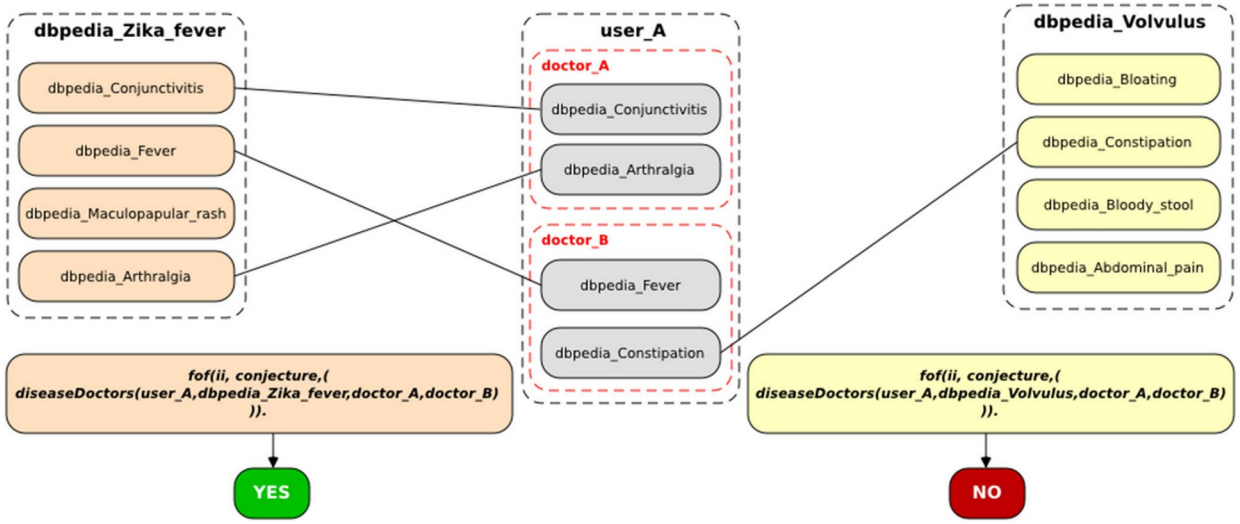

Figura 4.8: Comparación del resultado de dos conjeturas con respecto a los datos.

Por último, la Tabla 4.9 muestra el resultado obtenido, después de que DiLE filtrara y clasificara las enfermedades recomendadas, es decir, los casos que dieron YES con la conjetura II.

Tabla 4.9: Recomendación gracias al motor lógico dialéctico.

Recomendación	**Ranking**
dbpedia_Zika_fever	3
dbpedia_Chikungunya	2
dbpedia_Measles	2
dbpedia_Rheumatic_fever	2
dbpedia_Trichinosis	2

Los procesos descritos en esta sección están automatizados en Python. Son los procesos más complejos y constituyen el núcleo del HRS. El primer proceso (componente ConM) transforma las propiedades enriquecidas a axioma; en este caso particular, la propiedad isSymptom (ver PoQ en el caso de estudio), que tiene asociados dos conceptos: disease y symptom (conceptos que se obtuvieron usando la URI_PROPERTY en la sección anterior), quedando de la siguiente forma: isSymptom(disease, symptom). El segundo proceso ejecuta el razonador JGRM3 con la información contenida en PoQ y las propiedades enriquecidas transformadas en axiomas. Finalmente, los ítems recomendados por el razonador se almacenan en una lista y se ordenan por su ranking.

4.3.4 *Extracción de contenidos relacionados con las recomendaciones mediante Datos Enlazados*

En este caso, se extrae nueva información de las fuentes de Datos Enlazados relacionada con el URI que representa cada enfermedad alcanzada como recomendación (véase el macro-algoritmo en la Tabla 4.5). Para ello, RM activa QM para generar una consulta que permita extraer dicha información (paso 2 en la Tabla 4.5). Esta consulta se genera siguiendo el proceso descrito en el componente QM para extraer conocimiento. La Figura 4.9 muestra la consulta que busca y extrae todo el contenido relacionado con el URI que identifica a la Fiebre Zika, como síntomas, tratamientos, causa, entre otros (paso 2.1 en la Tabla 4.5).

```
sparql = SPARQLWrapper2("http://live.dbpedia.org/sparql")
sparql.setQuery("""
SELECT ?property ?object
WHERE {
   dbr:Zika_fever ?property ?object .
}
""")
```

Figura 4.9: Consulta para extraer datos asociados a Zika_fever.

RM recoge toda la información extraída con la consulta anterior de Fiebre Zika (pasos 3 y 4 de la Tabla 4.5), y la asocia al URI de dicha enfermedad (ver Tabla 4.10). Este proceso de extracción y asociación se repite con cada una de las enfermedades recomendadas por el HRS.

Tabla 4.10: Extracción de Datos Enlazados sobre la Fiebre Zika.

Enfermedad	Propiedad	Valor
dbpedia_Zika_fever	Duration	Less than a week
	Deaths	None during the initial infection
	Prevention	Decreasing mosquito bites, condoms
	Diagnosis	Testing blood, urine, or saliva for viral RNA or blood for antibodies
	Complications	During pregnancy can cause microcephaly, Guillain-Barré syndrome
	Symptom	Conjunctivitis
		Fever
		Maculopapular_rash
		Arthralgia
	Differential diagnosis	Leptospirosis
		Measles
		Malaria
		Chikungunya
		Dengue
	Treatment	Supportive_care

Por último, el RM entrega al usuario toda la información recopilada.

Además, este proceso se automatiza en Python a partir de la plantilla descrita en QM «Knowledge Extraction», para lo cual sustituye la palabra URI_CONCEPT por la URI del elemento recomendado por el sistema. A continuación, procede a ejecutar la consulta de búsqueda en el endpoint de la fuente LOD (por ejemplo, http://live.dbpedia.org/sparql). Este proceso se repite con todos los elementos recomendados.

4.3.5 *Análisis y Validación del Experimento*

En el experimento, se muestra cómo el HRS orquesta a los razonadores y gestores para extraer y procesar el conocimiento obtenido de los Datos Enlazados, y responder así a las necesidades de recomendación considerando los estados de ambigüedad o inconsistencia. En concreto, se describe cómo se identifican los conceptos asociados a PoQ (ver Sección 4.3.1), que luego se utilizan para extraer información de las fuentes de Datos Enlazados (ver Sección 4.3.2). Teniendo todo el conocimiento necesario para razonar, se procesa con el razonador de Datos Enlazados que identifica los casos ambiguos, y llega a las recomendaciones basadas en las preferencias de cada usuario (ver Sección 4.3.3). Por último, se extrae de los Datos Enlazados el contenido relacionado con cada recomendación alcanzada (ver Sección 4.3.4). En cada proceso se detalla el comportamiento de los algoritmos ejecutados por el HRS, para llegar a las recomendaciones.

En [56] se amplía toda la validación de este experimento, dónde se analiza el comportamiento de nuestro HRS y se evalúan las recomendaciones conseguidas utilizando la información extraída de los

Datos Enlazados. En general, se presenta el análisis del proceso en tres pasos: 1. Descripción general de los datos extraídos de las fuentes de Datos Enlazados con DeLE. 2. Recomendaciones alcanzadas con los eventos dialécticos detectados por DiLE. 3. Evaluación de las recomendaciones alcanzadas mediante el mecanismo de razonamiento híbrido. Además, se presenta un análisis comparativo tanto cualitativo como cuantitativo. Dónde todos los recomendadores utilizan la lógica descriptiva como mecanismo intrínseco para la explotación de los Datos Enlazados, y con respecto a la resolución de las ambigüedades y/o incoherencias, solo nuestro trabajo tiene esta capacidad. En nuestro enfoque, utilizamos un motor de Lógica Dialéctica basado en la lógica RM3, que permite razonar en los estados de ambigüedades o inconsistencias. Otro detalle está relacionado con la extracción y enriquecimiento, todos los recomendadores consideran el uso de Datos Enlazados como fuente de conocimiento, por su variedad y semántica, ya sea para caracterizar los elementos a recomendar y/o los perfiles de usuario. Sin embargo, solo un trabajo y nuestra propuesta aprovechan el paradigma de los Datos Enlazados para ofrecer información complementaria extraída de las mismas fuentes de Datos Enlazados, permitiendo ampliar la información que se presenta a los usuarios.

4.4 Aplicaciones

El uso de este tipo de arquitectura se extiende a muchos contextos como: *i. Recomendación de Productos*: manejar situaciones donde los consumidores tienen preferencias contradictorias o cuando hay información conflictiva sobre los productos. *ii. Recomendación de Viajes:* cuando hay información contradictoria sobre los destinos o servicios turísticos. *iii. Recomendación Educativa:* manejar situaciones donde los estudiantes tienen diferentes estilos de aprendizaje o cuando hay información contradictoria sobre los recursos educativos. *iv. Otros:* para cualquier contexto donde la información sea compleja, contradictoria o ambigua, y donde se necesite un razonamiento sofisticado para llegar a recomendaciones precisas y útiles. En este caso particular, nos centramos en la aplicación de la Lógica Dialéctica para analizar la ambigüedad en las competencias profesionales presentes en los textos digitales, como páginas web y redes sociales [58, 59] (Anexo 4.B y 4.C). La dificulta de este problema radica en comprender el significado real de una competencia en los perfiles profesionales digitales, ya que la interpretación puede variar según el conocimiento y la percepción del editor. Esto da lugar a inconsistencias y ambigüedades en la descripción de las competencias, lo que dificulta la identificación precisa de los conocimientos y habilidades necesarias para el diseño de programas de estudio universitarios y otros procesos de gestión de competencias.

4.4.1 *Fenómenos dialécticos en las competencias (Knowledge Model)*

Esta sección presenta un resumen extenso del trabajo presentado en la sección Knowledge Model del Anexo 4.B, dónde se muestran diferentes modelos dialécticos que contiene hipótesis que corresponden a los cinco fenómenos dialécticos [58]: vaguedad, declaraciones contingentes sobre el futuro, discurso ficticio, fallo de una presuposición y razonamiento contrafáctico. Aplicamos las descripciones de cada axioma sobre los términos de competencia, conocimiento y habilidad, pertenecientes a documentos de una colección de perfiles analizados por expertos [60]. Estos términos pertenecientes a la población ontológica del modelo Competencias Ontológicas, siguiendo un método desarrollado en [61], con el

apoyo de bases de conocimiento de definiciones de conocimiento y habilidades: DISCO II (para conocimiento), BLOOM (para habilidad) [61].

En las secciones siguientes, para cada fenómeno dialéctico, analizamos primero los axiomas utilizando ejemplos de términos. Luego, presentamos la descripción en RM3, estructurada en tres componentes: axiomas, que corresponden a las reglas dialécticas que los definen; hechos, que son las entradas al modelo a partir de las instancias extraídas de los perfiles digitales académicos o profesionales; y conjeturas, que se activan durante el razonamiento para realizar la interpretación de los perfiles digitales académicos o profesionales [10].

4.4.1.1Vaguedad

La vaguedad corresponde a una falta de claridad, precisión o exactitud en el lenguaje natural. Los patrones lingüísticos de las frases nominales y verbales que identifican las competencias de habilidades y conocimientos pueden ser los mismos (homónimos). La Tabla 1 muestra tres ejemplos de la ambivalencia de estos patrones, que se consideran frases nominales de la forma NC-SP-NC y NC-SP-NC-AQ, que representan el componente de conocimiento. Sin embargo, estos términos pueden interpretarse como una habilidad (Java expert y Hardware knowledge) o una competencia (Software development) [61]. De este modo, la estructura lingüística de las frases nominales es ambivalente, según la interpretación que el redactor haga de conocimiento y habilidad.

Tabla 4.11: Patrones lingüísticos de vaguedad.

Término	Patrón Lingüístico	Interpretación según el patrón	Interpretación del editor
Hardware knowledge	NC-SP-NC	Conocimiento	Habilidad
Java expert	NC-SP-NC	Conocimiento	Habilidad
Software development	NC-SP-NC	Conocimiento	Competencia

En particular, proponemos el siguiente axioma para los problemas de vaguedad explicados en la Tabla 4.11:

Si (el término T tiene un patrón P como conocimiento) y (P se interpreta como Habilidad (C1) o Competencia (C2)), entonces (T tiene un patrón ambivalente).

La Tabla 4.12 muestra el axioma en formato RM3 (Lógica Dialéctica). Como se puede observar, el axioma "*hasAmbivalentPattern*" establece la relación entre los patrones lingüísticos de los términos, dependiendo de si T (término) tiene un patrón P que representa conocimiento, pero que, cuando se interpreta es diferente (como habilidad (C1) o competencia (C2)), por lo que existe una ambivalencia. Así, aunque el patrón lingüístico del término indica una frase nominal que corresponde a conocimiento, el término se interpreta como habilidad o competencia.

Tabla 4.12: Axiomas de vaguedad.

Problema:	Si (el término T tiene un patrón P como conocimiento) y (P se interpreta como Habilidad (C1) o Competencia (C2)), entonces (T tiene un patrón ambivalente).
Axiomas	fof(hasAmbivalentPattern,axiom,(! [T,P,C1,C2] : ((hasPattern(T, P) & isInterpretedAs(T, C1) & isInterpretedAs(P, C2) & isDifferent(C1,P) & isDifferent(C2,P)) => hasAmbivalentPattern(T, P)))).
Hechos	fof(hasPattern1, axiom, hasPattern(hardware_knowledge, nc_sp_nc)). fof(isInterpretedAs1, axiom, isInterpretedAs (hardware_knowledge, competence)). fof(isInterpretedAs2, axiom, isInterpretedAs(hardware_knowledge, skill)). fof(isPattern1, axiom, isPattern(nc_sp_nc, knowledge)). fof(isDifferent1, axiom, isDifferent(knowledge, competence)). fof(isDifferent2, axiom, isDifferent(knowledge, skill)).
Conjeturas	If "SZS status Theorem for FOF" term has ambivalent pattern fof(conjeture1,conjecture, (hasAmbivalentPattern(hardware_knowledge, nc_sp_nc))).

El modelo parte de la propuesta de hechos como *fof(hasPattern1, axiom, hasPattern (hardware_knowledge,, nc_sp_nc))*, sobre la que los axiomas realizan las interpretaciones, desde axiomas básicos como *fof(isInterpretedAs2, axiom, isInterpretedAs(hardware_knowledge, skill))*, hasta llegar a la conjetura, que es un axioma que interpreta los hechos a partir del axioma básico *fof(conjeture1,conjecture,(hasAmbivalentPattern(hardware_knowledge,nc_sp_nc))*.

4.4.1.2Declaraciones Contingentes sobre el Futuro

Las declaraciones analizan acontecimientos futuros, acciones, etc. Este fenómeno se produce en frases verbales que generalmente describen competencias y habilidades. En este caso, la frase está formada por varios verbos que, considerando sus sinónimos, se encuentran en diferentes niveles de habilidad y procesos cognitivos, que no establecen qué habilidad desarrollará la competencia en breve. Por ejemplo, según el tesauro Bloom descrito en [61], para la competencia de la Tabla 4.13, “Design and manage systems”, la palabra “design” pertenece al nivel cognitivo 3 y la palabra “manage” al nivel cognitivo 5, ambas dentro de procesos cognitivos diferentes (inferior y superior, respectivamente). Por lo tanto, si finalmente se necesita esta competencia, resulta ambiguo establecer los mecanismos de enseñanza para conseguirla. Incluso en el proceso de evaluación del aprendizaje no está claro a qué nivel y proceso cognitivo debe considerarse la competencia.

Tabla 4.13: Casos de declaraciones contingentes sobre el futuro en términos de perfiles debido a la contradicción de los niveles cognitivos.

Frase verbal	Nivel cognitivo 1	Nivel cognitivo 2	Proceso cognitivo 1	Proceso cognitivo 2
Design and manage systems	Design: 3	Manage: 5	Inferior	Superior
Operate and maintain computer centers	Operate: 3	Maintain: 6	Inferior	Superior

Para formalizar esta contradicción, proponemos los siguientes axiomas para los problemas de enunciados contingentes de los ejemplos de la Tabla 4.13.

- Problema 1: Si (el término Th es sinónimo del término del tesauro Tb) y (Th y Tb pertenecen a niveles cognitivos diferentes Nc1 y Nc2), entonces (Th pertenece a varios niveles cognitivos).
- Problema 2: Si (el término Th1 es sinónimo del término Th2) y (Th1 y Th2 pertenecen a niveles cognitivos diferentes Nc1 y Nc2), entonces (Th1 y Th2 tienen varios niveles cognitivos).
- Problema 3: Si (el término T es sinónimo de los términos Th1 y Th2) y (Th1 y Th2 pertenecen a niveles cognitivos diferentes Nc1 y Nc2), entonces (T tiene varios niveles cognitivos).

En la Tabla 4.14, presentamos los modelos dialécticos de los axiomas partiendo del hecho de establecer que sus niveles cognitivos son diferentes usando *fof(isDifferent2, axiom, isDifferent (synthesis, application))*. A continuación, se establece la relación de sinonimia entre los términos con *fof(isSynonymous2, axiom, isSynonymous (design, plan))*, y de pertenencia de cada término a un nivel cognitivo con *fof(belongsCognitiveLevel1, axiom, belongsCognitiveLevel (design, synthesis))*. De este modo, la base de conocimiento para la interpretación se construye para la conjetura *fof(conjecture, conjecture,(termsBelongSeveralCognitiveLevels (design, plan)))*, que tiene un valor de verdadero porque “design” y “plan” son sinónimos y pertenecen a diferentes niveles cognitivos (“synthesis” y “application”, respectivamente).

Tabla 4.14: Axiomas de declaraciones contingentes sobre el futuro.

Problema:	1. Si Th es sinónimo del término del tesauro Tb y Th, y Tb tiene diferentes niveles cognitivos Nc1 y Nc2, entonces Th pertenece a varios niveles cognitivos. 2. Si el verbo relacionado Th1 es sinónimo del verbo correspondiente Th2 y Th1, y Th2 pertenecen a niveles cognitivos diferentes Nc1 y Nc2, entonces Th1 y Th2 tienen varios niveles cognitivos. 3. Si T es sinónimo de los verbos relacionados Th1 y Th2, y Th1 y Th2 pertenecen a diferentes niveles cognitivos Nc1 y Nc2, entonces T tiene varios niveles cognitivos.
Axiomas	fof(termBelongsSeveralCognitiveLevels, axiom,(! [Th,Tb,Nc1,Nc2] : ((isSynonymous(Th,Tb) & belongsCognitiveLevel(Th,Nc1)& belongsCognitiveLevel(Tb,Nc2) & isDifferent(Nc1,Nc2)) => termBelongsSeveralCognitiveLevels(Th)))). fof(termsRelatedVerbBelongsSeveralCognitiveLevels,axiom,(! [Th1,Th2,Nc1,Nc2] : ((belongsCognitiveLevel (Th1,Nc1)& belongsCognitiveLevel(Th2,Nc2) & isDifferent(Nc1,Nc2)) => termsRelatedVerbBelongsSeveralCognitiveLevels(Th1, Th2)))). fof(termsBelongsSeveralCognitiveLevels, axiom,(! [Th1,Th2] : ((termsRelatedVerbBelongSeveralCognitiveLevels(Th1, Th2) | termBelongsSeveralCognitiveLevels (Th1) | termBelongsSeveralCognitiveLevels (Th2)) => termsBelongSeveralCognitiveLevels (Th1,Th2)))).
Hechos	fof(isDifferent1, axiom, isDifferent(synthesis, knowledge)). fof(isDifferent2, axiom, isDifferent(synthesis, aplication)). fof(isDifferent3, axiom, isDifferent(aplication, synthesis)). fof(isSynonymous1, axiom, isSynonymous(design, sketch)). fof(isSynonymous2, axiom, isSynonymous(design, plan)). fof(isSynonymous3, axiom, isSynonymous(plan, sketch)). fof(belongsCognitiveLevel1, axiom, belongsCognitiveLevel(design, synthesis)). fof(belongsCognitiveLevel2, axiom, belongsCognitiveLevel(sketch,synthesis)). fof(belongsCognitiveLevel22, axiom, belongsCognitiveLevel(sketch,knowledge)). fof(belongsCognitiveLevel3, axiom, belongsCognitiveLevel(plan, aplication)).
Conjeturas	If "SZS status Theorem for FOF" terms belongs several cognitive levels fof(conjetura,conjecture, (termsBelongSeveralCognitiveLevels(design, plan))). fof(conjetura,conjecture, (termBelongsSeveralCognitiveLevels (design))).

4.4.1.3Discurso Ficticio

Según las creencias de las personas, los enunciados implican la toma de decisiones relacionadas con determinados supuestos reales o imaginarios. En el caso de las competencias y sus componentes de conocimientos y habilidades, es habitual que el editor de perfiles coloque estos tres componentes en secciones de un documento, como la descripción, el campo ocupacional, y no precisamente como

competencias, conocimientos o habilidades. La tabla 4.15 muestra algunos casos fundados en [61], donde el editor de perfiles colocó la competencia “Plan and manage computer projects” como antecedente. Un caso similar se refiere al tema de conocimiento “Industrial process control”, establecido en la sección de competencias. En consecuencia, depende mucho de la interpretación y los conocimientos del redactor reconocer una competencia o sus componentes de conocimientos y habilidades, lo que puede generar una ficción en la redacción del perfil académico o profesional.

Tabla 4.15: Casos de discurso ficticio en términos de perfiles por su significado y ubicación.

Término	**Componente**	**Sección de documentos**
Industrial process control	Conocimiento	Competencias
Development of computer applications	Conocimiento	Perfil de carrera
Plan and manage computer projects	Conocimiento	Antecedente

En particular, proponemos el siguiente axioma para este problema, de acuerdo con los ejemplos del cuadro 4.15. En este caso, la lógica de la descripción no consigue representar la contradicción de los hechos; por ejemplo, el término “industrial process control” es un componente del conocimiento, pero se sitúa como una competencia.

Si (el término T se encuentra en la sección del documento C1) y (T es un componente C2) y (C1 es diferente de C2), entonces (T es una frase ficticia).

En la Tabla 4.16, presentamos el axioma “*isFictitiousPhrase*” partiendo del hecho de que el término es un componente de “knowledge” (conocimiento) con *fof(isComponent1, axiom, isComponent(industrial_process_control, knowledge))*, que esté ubicado en la sección “competencies” (competencia) del documento con *fof(isLocated1, axiom, isLocated (industrial_process_control, competencies))*, siendo diferentes ”knowledge“ y ”competencies” con *fof(isDifferent2, axiom, isDifferent(knowledge, competencies))*. Con base en los hechos, la conjetura *fof(conjecture, conjecture, (isFictitiousPhrase(industrial_process_control, competencies)))* tiene un valor de verdadero, porque al mismo tiempo “industrial_process_control” es un componente de “knowledge” y se identifica como una “competence”.

Tabla 4.16: Axiomas de los términos ficticios.

Problema:	Si el término T se encuentra en la sección del documento C1 y T es un componente C2, y C1 es diferente de C2, entonces T es una frase ficticia.
Axiomas	fof(isFictitiousPhrase, axiom,(! [T,P,C1,C2] : ((isLocated(T, C1) & isComponent(T, C2) & isDifferent(C1,C2)) => isFictitiousPhrase(T)))).
Hechos	fof(isComponent1, axiom, isComponent(industrial_processes_control, knowledge)). fof(isComponent2, axiom, isComponent(industrial_process_control, competencies)). fof(isLocated1, axiom, isLocated(industrial_process_control, competencies)). fof(isDifferent1, axiom, isDifferent(competencies, knowledge)). fof(isDifferent2, axiom, isDifferent(knowledge, competencies)).
Conjeturas	If "SZS status Theorem for FOF" is Fictitious Phrase fof(conjeture, conjecture, (isFictitiousPhrase(industrial_process_control, competencies))).

4.4.1.4Falla de una Presuposición

La afirmación implica la presuposición de algo que en realidad no es cierto, aplicada a las competencias cuando el término se utiliza mal en una sección del perfil, de tal manera que el término presuposición es erróneo. Según la interpretación del redactor, es de un tipo, pero es de otro. Por ejemplo, en la Tabla 4.17, el término “Develop computer applications” se presupone como un “Perfil de carrera”, cuando en realidad se interpreta como una “Habilidad” [61]. Del mismo modo, “Hardware control” se supone un “Antecedente”, siendo un “Conocimiento”, y así para los demás casos. Del mismo modo, para cada término, la suposición del editor de perfiles es errónea en cuanto a la interpretación del experto.

Tabla 4.17: Casos de fallo de presuposición debido a la contradicción de la interpretación de los expertos de los términos de los perfiles.

Término	Presuposición	Interpretación del experto (patrón)
Develop computer applications	Perfil de carrera	Habilidad
Develop computer programs	Competencias	Habilidad
Plan and manage computer projects	Antecedente	Habilidad
Hardware Control	Antecedente	Conocimiento
Java knowledge	Experiencia	Habilidad

Para formalizar esta contradicción, proponemos el siguiente axioma para este problema, según los ejemplos de la Tabla 4.17.

Si (el término T se encuentra en la sección del documento C1) y (T tiene un patrón C2) y (C1 es diferente de C2), entonces (T es un fallo de presuposición).

En la Tabla 4. 18, presentamos el axioma en Lógica Dialéctica "*isPresuppositionFailure*" partiendo del hecho de que el término tiene un patrón de conocimiento "nc_aq" (*fof(hasPattern1, axiom, hasPattern(java_knowledge, nc_aq))*), que se encuentra en la sección "experiencia" del documento (*fof(isLocatedIn 1, axiom, isLocatedIn(java_knowledge, experience))*), siendo diferentes "conocimiento" y "experiencia" (*fof(isDifferent1, axiom, isDifferent(experience, knowledge))*). A partir de los hechos, la conjetura *fof(conjeture, conjeture, (isPresuppositionFailure (java_knowledge)))* tiene un valor de verdadero porque al mismo tiempo "java_knowledge" tiene un patrón de "conocimiento" que se identifica como una "experiencia".

Tabla 4.18: Axiomas de fallo de presuposición.

Problema:	Si el término T se encuentra en la sección del documento C1, T tiene un patrón C2, y C1 es diferente de C2, entonces T es un fallo de presuposición.
Axiomas	fof(isPresuppositionFailure, axiom,(! [T,P,C1,C2] : ((hasPattern(T, P) & isLocatedIn(T, C1) & isPattern(P, C2) & isDifferent(C1,C2)) => isPresuppositionFailure (T)))).
Hechos	fof(hasPattern 1, axiom, hasPattern (java_knowledge, nc_aq)). fof(isLocatedIn 1, axiom, isLocatedIn(java_knowledge, experience)). fof(isPattern 1, axiom, isPattern (nc_aq, knowledge)). fof(isDifferent 1, axiom, isDifferent(experience, knowledge)).
Conjeturas	If "SZS status Theorem for FOF" term is Presupposition Failure fof(conjetura,conjecture, (isPresuppositionFailure (java_knowledge))).

4.4.1.5Razonamiento Contrafáctico

Considerar el significado de los enunciados causales puede explicarse en términos de condicionales contrafácticos de la forma «Si no hubiera ocurrido A, entonces no habría ocurrido C». En el contexto de las competencias, el razonamiento contrafáctico se aplica en las hipótesis realizadas al alinear los términos de las competencias con los términos de los tesauros según medidas de similitud léxica, estableciendo umbrales para determinar las similitudes. Propondremos la siguiente hipótesis: "Un término y un tema de un tesauro de competencias pertenecen al mismo dominio de conocimiento cuando la medida de similitud entre ellos supera el límite de 0,45" [61]. Como se muestra en la Tabla 4.19, para los tres casos propuestos, dos pertenecen al mismo dominio porque la medida de similitud supera el límite de 0,45. Pero, si cambiamos el valor límite a 0,51, vemos que solo el caso "Software" frente a "Programming" cumple la hipótesis. En general, el valor umbral es subjetivo, lo que provoca errores y ambivalencias a la hora de interpretar la pertenencia de un término a un dominio de conocimiento.

Tabla 4.19: Casos de razonamiento contrafáctico debido a la pertenencia de un término a un dominio según una medida de similitud.

Término	Tópico	Dominio	Similitud
Software	Software debugging	Programming	0.53
	Software installation	IT installation and configuration	0.50
	Software Application Development	Software development	0.41

De acuerdo con los ejemplos de la Tabla 4.19, proponemos el siguiente axioma para este problema,

Si (el término T tiene una medida de similitud Ms con un tema Tr mayor que el umbral Us), entonces (pertenece al tema raíz del tesauro TD).

La tabla 4.20 muestra los axiomas dialécticos para esta contradicción, empezando por los hechos fof(relationMeasure1, axiom, relationMeasure (software, programming, ms0_48, td1)) y fof(relationMeasure3, axiom, relationMeasure (software, software_debugging, ms0_52, td12)), que define que el término “software” tiene una medida de similitud de 0,48 con “programming” y de 0,52 con “software_debugging”. Otro hecho es que las medidas de similitud de 0,48 y 0,52 son más significativas que el umbral (0,45), y también que los hechos “td1” y “td12” son diferentes. De este modo, la base de conocimientos para la interpretación se construye según el axioma fof(conjecture, conjecture, (termBelongsTopic (software, td12))), que es el axioma base para la conjetura fof(conjecture, conjecture, (termBelongsSeveralTopics (software))). Considerando el término “software”, el resultado es verdadero porque “software” pertenece a los temas “programación” y “depuración de software” pertenece al tema raíz del tesauro TD.

Tabla 4.20: Axiomas de razonamiento contrafáctico.

Problema:	Si el término T tiene una medida de similitud Ms con un tema Tr mayor que el umbral Us, entonces pertenece al tema raíz del tesauro TD.
Axiomas	fof(termBelongsTopic,axiom,(! [T,Tr,Ms,Us,TD] : ((relationMeasure(T, Tr, Ms, TD) & isGreaterThan(Ms, Us)) => termBelongsTopic(T, TD)))). fof(termBelongsSeveralTopics,axiom,(! [T,TD1,TD2] : ((termBelongsTopic (T, TD1) & termBelongsTopic (T, TD2) & isDifferent(TD1, TD2)) => termBelongsSeveralTopics (T)))).
Hechos	fof(relationMeasure1, axiom, relationMeasure (software, programming, ms0_48, td1)). fof(relationMeasure2, axiom, relationMeasure (software, software_installation, ms0_30, td11)). fof(relationMeasure 3, axiom, relationMeasure (software, software_debugging, ms0_52, td12)). fof(threshold, axiom, threshold = ms0_45). fof(isGreaterThan1, axiom, isGreaterThan(ms0_48 , threshold)). fof(isGreaterThan2, axiom, isGreaterThan (ms0_52 , threshold)). fof(isDifferent 1, axiom, isDifferent (td1 , td12)).
Conjeturas	If "SZS status Theorem for FOF term Belongs Topic fof(conjetura,conjecture, (termBelongsTopic (software, td1))). fof(conjetura,conjecture, (termBelongsTopic (software, td12))). If "SZS status Theorem for FOF" term Belongs Several Topics fof(conjetura,conjecture, (termBelongsSeveralTopics(software))).

4.4.2 *Otros Modelos de Conocimiento*

Esta sección es un resumen extenso del trabajo presentado en la sección 3 del Anexo 4.C, dónde se presentan dos modelo de conocimiento basado en axiomas dialécticos centrados en el razonamiento contrafáctico y fallo de la presuposición [59]. Para ello, se analizan los casos de ambigüedad dialéctica aplicados a términos de conocimiento y habilidad usando los dos tesauros presentados anteriormente: DISCO II (para conocimiento) y BLOOM (para habilidad) [61], con el fin de identificar la ambigüedad semántica presente para el alineamiento entre términos de competencias y los tópicos en los tesauros.

En el primer caso, abordamos la contradicción que existe en cuanto a la pertenencia de un término de conocimiento a un tópico de un tesauro, tomando a DISCO II como tesauro de referencia [60]. Los axiomas se definen en torno al siguiente problema:

Si el término T tiene una medida de similitud Ms con un tópico Tr mayor al umbral Us, entonces pertenece al tópico raíz del tesauro TD.

En la Tabla 4.21 se observan los 4 axiomas que lo describen, los cuales están relacionados entre sí, de tal forma que para que se cumpla un axioma, deben cumplirse los axiomas relacionados. Por ejemplo, el axioma “terminoPerteneceTopicos” requiere del cumplimiento de los axiomas

“términoPerteneceTopico”, “terminoPerteneceVariosTopicos” y “terminoPerteneceAlgunTopico”. Con estas relaciones, se describe que un término T pertenece a varios tópicos del tesauro si la medida de similitud es mayor que el umbral establecido.

Tabla 4.21: Axiomas caso 1.

Problema:	Si el término T tiene una medida de similitud Ms contra un tópico Tr mayor al umbral Us entonces pertenece al tópico raíz del tesauro.
Axiomas	fof(terminoPerteneceTopico,axiom,(! [T,Tr,Ms,Us] : ((medidaRelacion(T, Tr, Ms) & esMayor(Ms, Us)) => terminoPerteneceTopico(T, Tr)))). fof(terminoPerteneceAlgunTopico,axiom,(! [T,Tr] : ((terminoPerteneceTopico(T , Tr)) => terminoPerteneceAlgunTopico(T)))). fof(terminoPerteneceTopicos,axiom,(! [T,Tr1,Tr2] : ((terminoPerteneceTopico(T , Tr1) & terminoPerteneceTopico(T , Tr2)) => terminoPerteneceTopicos(T,Tr1,Tr2)))). fof(terminoPerteneceVariosTopicos,axiom,(! [T,Tr1,Tr2] : ((terminoPerteneceTopicos(T,Tr1,Tr2)) => terminoPerteneceVariosTopicos(T)))).
Hechos	fof(medidaRelacion1, axiom, medidaRelacion(software, programacion, s0_48)). fof(medidaRelacion2, axiom, medidaRelacion(software, depuracion_de_software, s0_52)). fof(medidaRelacion3, axiom, medidaRelacion(software, instalacion_de_software, s0_30)). fof(umbral, axiom, umbral = s0_45). fof(esMayor1, axiom, esMayor(s0_48 , umbral)). fof(esMayor2, axiom, esMayor(s0_52 , umbral)). fof(esMayor3, axiom, esMayor(s0_30, umbral)).
Conjeturas	Si “SZS status Theorem for FOF” término pertenece al tópico fof(conjetura1,conjecture, (terminoPerteneceTopico(software,programacion))). Si “SZS status Theorem for FOF” término pertenece a algún tópico fof(conjetura2,conjecture, (terminoPerteneceAlgunTopico(software))). Si “SZS status Theorem for FOF” término pertenece a los dos tópicos fof(conjetura3,conjecture, (terminoPerteneceTopicos(software,programacion,depuracion_de_software))). Si “SZS status Theorem for FOF” término pertenece a varios tópicos fof(conjetura4,conjecture, (terminoPerteneceVariosTopicos(software))).

Para el segundo caso, consideramos la ambigüedad que existe entre términos de habilidades cuando pertenecen a dos niveles cognitivos distintos, esto se da debido los sinónimos que tiene un término, y a los niveles cognitivos que pertenecen estos sinónimos. El tesauro con el cual realizamos este análisis es con el tesauro BLOOM que se explica en [61], el cual presenta estas contradicciones. Este ejemplo lo podemos ver en detalles en la sección 3.2 del Anexo 4.C.

4.4.3 *Análisis General*

La Lógica Dialéctica ofrece una herramienta poderosa para analizar la ambigüedad inherente a las descripciones de competencias profesionales en textos digitales, ya que la lógica tradicional, con su enfoque binario de Verdadero o Falso, resulta insuficiente para modelar las múltiples interpretaciones válidas que pueden surgir del lenguaje natural en este contexto. Para superar esta limitación, definir modelos basados en axiomas dialécticos permiten identificar y analizar contradicciones y ambivalencias en la descripción de competencias, abarcando fenómenos como la vaguedad, el fallo de presuposición, el razonamiento contrafáctico, el discurso ficticio y las declaraciones contingentes sobre el futuro. Para evaluar la efectividad de los modelos presentados, se utilizaron métricas como la Completitud, la Robustez y la Entropía que pueden ser revisados en [58, 59]. Los modelos propuestos buscan mejorar la precisión en la identificación de conocimientos y habilidades, con aplicaciones en la educación, la gestión de recursos humanos y el desarrollo de sistemas de aprendizaje inteligentes.

5 Arquitectura de Meta-Aprendizaje para Modelos de Aprendizaje Automático basado en Datos Enlazados

En este capítulo se presenta la construcción de una arquitectura de Meta-Aprendizaje para la generación de modelos de Aprendizaje Automático basado en el paradigma de los Datos Enlazados. Esta arquitectura lleva a cabo las diferentes tareas de los expertos en datos o científicos para la generación de modelos de Aprendizaje Automático, quienes se encargan de tareas como la extracción de información de las fuentes de datos, el procesamiento de datos, la selección de los algoritmos de Aprendizaje Automático, el ajuste de los hiperparámetros de los algoritmos de Aprendizaje Automático, entre otras. La estructura del capítulo es la siguiente: La sección 5.1, en consonancia con la sección II del artículo presentado en el Anexo 5.A, describe el diseño de la arquitectura de Meta-Aprendizaje basado en Datos Enlazados para la generación automática de modelos de conocimientos. La sección 5.2, se basa en la Sección 4 del artículo presentado en el Anexo 5.B, y presenta la ampliación de la arquitectura presentada en la sección 5.1., introduciendo un nuevo nivel de sofisticación en la generación de modelos de conocimiento y la integración de nuevas capacidades. Todo esto, gracias a su Meta-Algoritmo autónomo que permite automatizar la construcción de modelos de Aprendizaje Automático, invocando los diferentes módulos especializados como los de aprendizaje por transferencia (Transferencia de Modelos, de Parámetros y de Datos) y de Generación de Datos Sintéticos. La sección 5.2.1 describe la arquitectura de generación de características usando modelos de Aprendizaje Automáticos, basándose en la sección III del artículo presentado en el Anexo 5.C. La sección 5.2.2 presenta la arquitectura de generación de datos artificiales usando Datos Enlazados, basándose en la sección II del artículo presentado en el Anexo 5.D y en la sección III del artículo presentado en el Anexo 5.E. La sección 3 presenta varios casos de estudio. Concretamente, la subsección 5.3.1 ilustra un caso de estudio de la arquitectura de Meta-Aprendizaje, basado en la sección 5 del artículo del Anexo 5.B. La subsección 5.3.2 presenta un caso de estudio sobre la generación de características, fundamentado en la sección IV del artículo del Anexo 5.C. A su vez, la subsección 5.3.3 ilustra un caso de estudio centrado en la generación de datos artificiales, tomando como base la sección III del artículo del Anexo 5.D. Finalmente, la sección 5.4 explora el uso de este tipo de arquitectura en el contexto de las cadenas de producción agroindustrial, basada en la sección 3 del artículo presentado en el Anexo 5.F.

5.1 Arquitectura

Esta sección presenta un resumen extenso del trabajo presentado en [9], cuyos detalles se encuentran en la sección II del artículo presentado en el Anexo 5.A. En dicho trabajo, se propone una arquitectura conceptual que sigue las tres fases propuestas por la metodología MIDANO [53, 62], que son: en la fase 1, se identifican las fuentes para la extracción de conocimiento. En la fase 2, se preparan los datos, es decir, se procesan los datos disponibles en las fuentes de conocimiento mediante tareas de ingeniería de características, entre otras. Por último, en la fase 3, se implementan diferentes tareas para generar los modelos de conocimiento requeridos, como la configuración de técnicas de Aprendizaje Automático y la construcción e integración de modelos de Aprendizaje Automático. Esta arquitectura se compone de las siguientes capas (ver Figura 5.1):

- **KSL:** Esta capa almacena y gestiona la información sobre los elementos necesarios en los procesos de Aprendizaje Automático, como el conjunto de datos a utilizar, las características (conocimiento extraído del conjunto de datos) y los modelos de conocimiento a construir (hiperparámetros, técnicas de aprendizaje, métodos de validación, entre otros). Se compone principalmente de fuentes de Datos Enlazados, que proporcionan datos con información semántica. Esta capa se compone del módulo llamado Linked Data Module (LDM).
- **MKL:** Esta capa gestiona todo el conocimiento relacionado con los procesos, tareas y estrategias para la utilización de técnicas de Aprendizaje Automático para construir modelos de conocimiento. En concreto, describe todos los elementos que componen el Aprendizaje Automático, planifica y organiza los procesos a ejecutar, analiza y evalúa las estrategias utilizadas en los diferentes procesos y/o tareas, y descubre nuevo conocimiento basado en la experimentación, entre otros. Esta capa está compuesta por los módulos de Meta-Learning (MLM), Meta-Feature (MFM), Meta-DataSet (MDSM) y Meta-Model (MMM).
- **KML:** Esta capa ejecuta y registra todos los procesos de Aprendizaje Automático. En concreto, se procesan y enriquecen los conjuntos de datos; se aplica la ingeniería de características; se seleccionan los algoritmos de entrenamiento, los cuales se utilizan para construir modelos de conocimiento, que se evalúan posteriormente. Esta capa se compone de los módulos de Feature Engineering (FEM), Tuning (TM), Model Building (MBM) y Model Integration (MIntM).

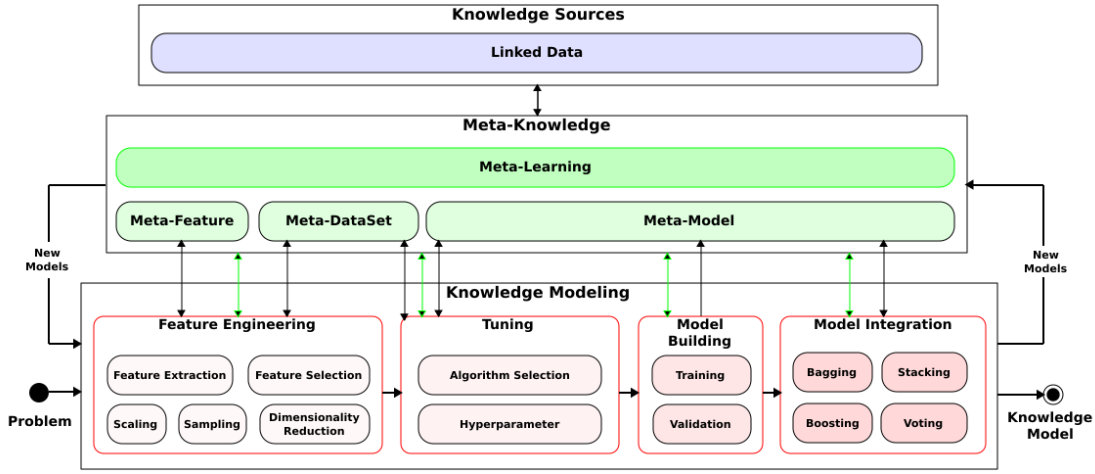

Figura 5.1: Arquitectura conceptual del Meta-Aprendizaje.

5.1.1 *Módulos de la arquitectura*

En esta sección se ofrece una descripción general de los módulos que componen las capas de la arquitectura de Meta-Aprendizaje. A continuación, se presenta el módulo KSL:

- **LDM:** Este módulo utiliza el paradigma de Datos Enlazados para gestionar la información generada por el MKL y el KML, ofreciendo mecanismos de consulta (leer, crear, actualizar o borrar triplas) a todos los módulos de la arquitectura.

Se describen a continuación los módulos de MKL:

- **MLM:** Este módulo es el encargado de tomar todas las decisiones generales de la arquitectura, es decir, es el responsable de invocar al resto de módulos especializados en cada tarea. También es responsable de recibir y caracterizar el problema a resolver, y de identificar la fuente de datos que se utilizarán para resolverlo.
- **MFM:** Este módulo se encarga de gestionar el conocimiento sobre las propiedades generales de las características de los conjuntos de datos. En particular, este módulo especifica la información que debe generarse a partir de los conjuntos de datos, como las medidas estadísticas estándar, la correlación entre los datos, entre muchas otras.
- **MDSM:** Este módulo se encarga de gestionar el conocimiento específico sobre los datos que se utilizan para generar un modelo Aprendizaje Automático. Para ello, mantiene un registro que describe los conjuntos de datos y sus metadatos, como el número de instancias, atributos, clases, autor, fecha de creación, fecha de modificación, etc. MDSM utiliza estándares (ontologías y vocabularios) para representar y describir estos datos. Además, también se encarga de especificar los procesos necesarios para limpiar y transformar un conjunto de datos.
- **MMM:** Este módulo se encarga de gestionar el conocimiento sobre el entrenamiento y validación de los modelos de Aprendizaje Automático. Para ello, guarda todas las características de los modelos de Aprendizaje Automático con sus reglas y procesos para su creación. Además, registra todas las validaciones y pruebas realizadas sobre estos modelos, con el objetivo de comparar sus calidades.

A continuación, se detallan los módulos de KML:

- **FEM:** Este módulo se encarga de preparar el conjunto de datos de entrada adecuado para el modelo de Aprendizaje Automático, utilizando como base de conocimiento la información que tiene LDM.
- **TM:** Este módulo se encarga de seleccionar el algoritmo de Aprendizaje Automático y preparar su configuración, utilizando el conocimiento proporcionado por el MLM sobre experiencias en ejecuciones anteriores, lo que le permite identificar los algoritmos adecuados para resolver el problema.
- **MBM:** Este módulo se encarga del entrenamiento y validación de los modelos de Aprendizaje Automático, utilizando como base de conocimiento la información proporcionada por TM sobre la configuración del algoritmo a utilizar.
- **MIntM:** Este módulo es activado opcionalmente por MLM, cuando se deben integrar los modelos de Aprendizaje Automático previamente creados, utilizando técnicas como Bagging, Boosting, Stacking, entre otras. En concreto, este módulo permite utilizar los modelos básicos previamente creados como bloques de construcción para diseñar modelos de Aprendizaje Automático más complejos mediante su combinación. La razón puede ser que estos modelos básicos no funcionen tan bien por sí solos, ya sea porque tienen un sesgo alto (por ejemplo, modelos de bajo grado de libertad) o porque tienen demasiada varianza para ser robustos (por ejemplo, modelos de alto grado de libertad).

5.2 Ampliación de la Arquitectura

Esta sección presenta un resumen extenso del trabajo presentado en [63], y la sección 4 del artículo presentado en el Anexo 5.B contiene los detalles completos. Esta investigación amplía

significativamente el trabajo presentado en la sección anterior, al optimizar la generación de modelos de conocimiento usando un ciclo autónomo de tareas, pero además, integra nuevas capacidades. A su vez, en la sección 5.2.1 se presentan los detalles sobre la generación de características usando modelos de Aprendizaje Automáticos, y en la sección 5.2.2 se presentan los detalles sobre la generación de datos artificiales usando Datos Enlazados. Estas ampliaciones se enumeran a continuación (véase la Figura 5.2):

- El corazón de esta innovación radica en la implementación de un Meta-Algoritmo autónomo que permite automatizar la construcción de modelos de Aprendizaje Automático, invocando los diferentes módulos especializados de forma estratégica, seleccionando las herramientas y técnicas más apropiadas (Nuevo módulo de Meta-Technique) para resolver las tareas específicas a mano. Además, incorporando de forma inteligente los nuevos módulos de aprendizaje por transferencia como Transferencia de Modelos, Transferencia de Parámetros y Transferencia de Datos, así como y Generación de Datos Sintéticos.
- A su vez, se ha reorganizado y ampliado KML, lo que permite agrupar características similares en el proceso de Aprendizaje Automático, facilitando futuras mejoras y ampliaciones de cada grupo de características. Esta capa se ha redefinido en tres nuevos módulos especializados:
  I. **Dataset Engineering (DEM):** Ingeniería de conjuntos de datos (DEM): Responsable de la preparación y optimización de los conjuntos de datos garantizando que son adecuados para el análisis y la modelización. Incluye nuevos submódulos, como Dataset Acquisition, Dataset Preparation, Data Transfer y Generate Synthetic Data.
  II. **Feature Engineering (FEM):** Dedicado a la creación y selección de características relevantes, que permitirán a los modelos capturar las relaciones subyacentes en los datos. Se añaden los submódulos AutoFeature y Feature Generation. Además, se integran nuevas técnicas a los submódulos Feature Extraction y Feature Selection.
  III. **Model Engineering (MEM):** Centrado en el diseño, entrenamiento y evaluación de modelos ML, seleccionando la arquitectura y los hiperparámetros más adecuados para cada tarea. Incluye nuevos submódulos como Model Transfer y Parameter Transfer.

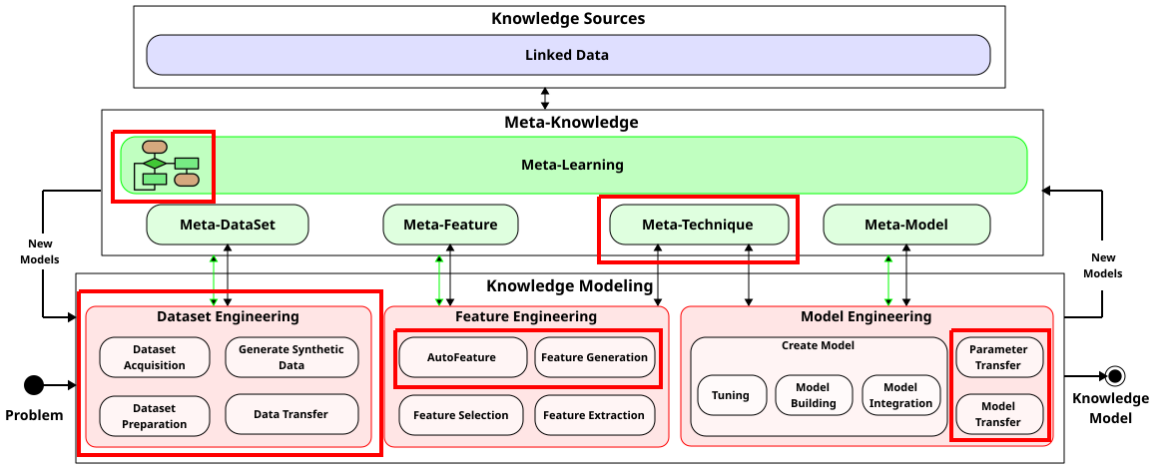

Figura 5.2: Arquitectura de Meta-Aprendizaje ampliada (nuevos componentes resaltados en recuadros rojos).

A. Meta-Algoritmo Autónomo

En la capa MKL, el módulo MLM gestiona la activación de los diferentes módulos de la arquitectura mediante un Meta-Algoritmo Autónomo. Este algoritmo guía cada decisión y proceso en función de las necesidades específicas de cada tarea y del conocimiento que la arquitectura ha adquirido de tareas anteriores. La Figura 5.3 presenta el diagrama de flujo que ilustra los principales procesos y decisiones. El Meta-Algoritmo Autónomo detecta cuando MLM recibe el problema a resolver. Analiza el problema y lo caracteriza, e invoca al módulo de *Dataset Acquisition*, solicitando los posibles datasets a utilizar para el problema caracterizado. Una vez que el conjunto de datos está disponible, procede a activar el proceso de *Dataset Preparation* para normalizar el conjunto de datos. En este punto, el Meta-Algoritmo Autónomo comprueba si existe un modelo previamente creado con las mismas características y conjunto de datos de entrada (Decision 1). Si el modelo existe, procede a realizar una transferencia de modelo usando el módulo *Model Transfer,* y entrega el modelo con su meta-información. En caso contrario, se inicia el proceso de creación de un nuevo modelo. Para comenzar con la creación de un nuevo modelo, se comprueba si existe un modelo con características y conjunto de datos similares (Decision 2). Si existe, se toma el modelo con mejores resultados y se realiza una transferencia de parámetros usando el módulo *Parameter Transfer*. En ambos casos, el proceso continúa con la tercera decisión (Decision 3), en la que se comprueba si el conjunto de datos es lo suficientemente grande como para crear el nuevo modelo. En este caso, el conjunto de datos se almacena con su *Meta-Dataset*. En caso contrario, se comprueba si existe un conjunto de datos similar en la arquitectura (Decision 4). Si existe, el conjunto de datos se utiliza para realizar una transferencia de datos usando el módulo *Data Transfer*. En caso contrario, el conjunto de datos original se utiliza para generar datos sintéticos (*Generate Synthetic Data*). En ambos casos, el conjunto de datos obtenido se almacena con su Meta-Datos. A continuación, el Meta-Algoritmo Autónomo procede a activar el módulo de *Feature Engineering* para seleccionar, extraer o generar las características relevantes para el problema específico. Por último, se activa *Create Model* para ejecutar los procesos *Tunning, Model Building* y *Model Integration*. Al finalizar, el modelo generado se almacena con todas sus características (usando *Create Meta-Model*), y se entrega con su meta-información. Para la creación de nuevos modelos sin transferencia previa de conocimientos, se utiliza la información proporcionada por el módulo *Meta-Technique*. Este módulo se dedica a gestionar de forma inteligente el conocimiento asociado a las técnicas de aprendizaje disponibles en la arquitectura para diferentes tareas. Así, facilita una comprensión más profunda y una gestión más eficaz de las técnicas de aprendizaje, incluyendo valores por defecto para sus hiperparámetros, métricas asociadas, entre otros.

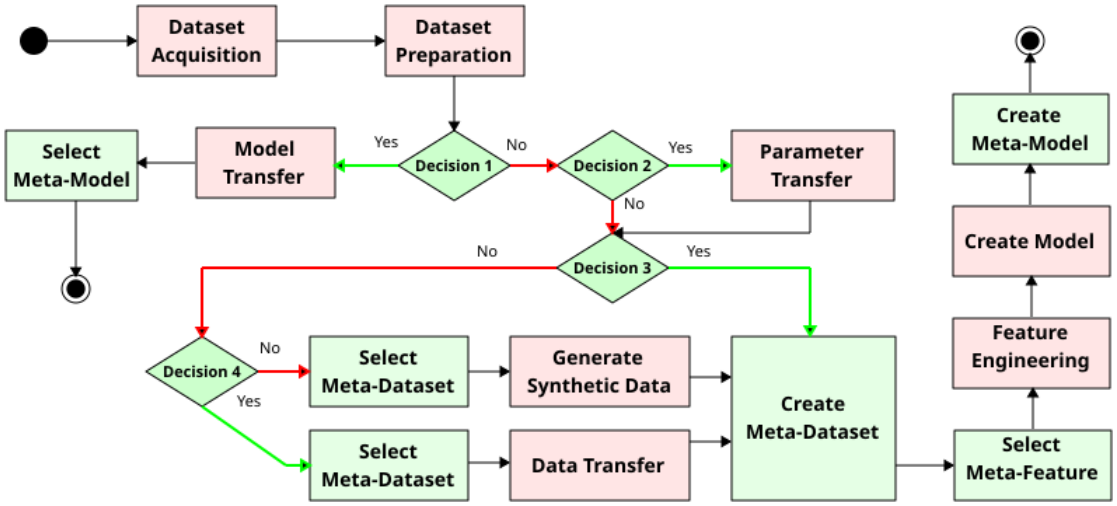

Figura 5.3: Meta-algoritmo autónomo para MLM, mostrando la invocación a los procesos KML (Rojo) y MKL (Verde).

B. Modificación de KML

Para mejorar la modularidad y la gestión de características, KML se ha estructurado en tres módulos especializados: DEM, FEM y MEM. Estos módulos ofrecen varias ventajas, como una mejor organización de las funcionalidades relacionadas en los procesos de KML, la simplificación de las tareas de mantenimiento y la posibilidad de realizar mejoras y ampliaciones específicas.

El ***nuevo MDE*** agrupa los procesos asociados a la gestión de conjuntos de datos, ocupándose de su adquisición, preparación, optimización y generación para garantizar su idoneidad en el análisis y construcción de modelos de conocimiento. El Meta-Algoritmo Autónomo activa estos procesos (submódulos) con la información del MDSM, según el requerimiento deseado. Entre los submódulos disponibles se encuentran los siguientes:

- **Dataset Acquisition [38, 37]:** El objetivo de este módulo es encontrar muestras de datos para el contexto dado, o recopilar el conjunto de datos proporcionado por el usuario. Cuando se requiere buscar muestras de datos, se obtienen utilizando mecanismos de búsqueda basados en ML, aprovechando Fuentes de Datos Abiertas (Open Data Sources, ODS) o endpoints, como los proporcionados oficialmente en países como Estados Unidos (https://www.data.gov/), España (https://datos.gob.es/) y Europa (https://data.europa.eu), entre muchos otros.
- **Dataset Preparation [38, 37]:** El objetivo de este módulo es transformar el conjunto de datos de la muestra en una representación óptima para los modelos que se van a construir. En este proceso, los atributos con datos textuales o numéricos que representan un conjunto finito específico de categorías o clases se procesan como Datos Categóricos, y los atributos con datos numéricos y alta varianza se normalizan.
- **Generate Synthetic Data [38, 37, 64, 65]:** El objetivo de este módulo es generar datos sintéticos a partir de la muestra de datos optimizada. En este proceso, se construye y entrena un modelo de conocimiento que extrae y aprende automáticamente las características de la muestra de datos. Con este modelo, se generan los datos sintéticos necesarios (más detalles en la sección 5.2.2).

- **Data Transfer [66, 67, 68]:** El objetivo de este módulo es transferir datos de un dominio de origen al dominio de destino. Este enfoque implica medir la similitud entre un dominio de origen y un dominio de destino, y seleccionar un dominio de origen similar que tenga muchos más datos de entrenamiento que el dominio de destino.

FEM concentra los procesos de generación y selección de características relevantes a partir de los datos, lo que permite a los modelos identificar las relaciones subyacentes en los datos. El Meta-Algoritmo Autónomo activa estos procesos (submódulos) utilizando información de MFM y MTM. Dentro del FEM, AutoFeature y Feature Generation son nuevos submódulos, pero además, se han ampliado los submódulos Feature Selection y Feature Extraction.

- **AutoFeature [69]:** Este módulo aplica automáticamente diferentes técnicas de ingeniería de características, como la generación de características basada en redes CNN, donde los datos se transforman en imágenes. A continuación, las imágenes se utilizan como entrada para un modelo CNN que genera las características (más detalles en la sección 5.2.1).
- **Feature Generation [37, 70]:** Este módulo aplica técnicas de generación de características como i. Característica de Interacción, ii. Característica Polinomial, iii. Característica Trigonométrica, iv. Creación de Clusters. y iv. Combinación de Niveles Raros.
- **Feature Extraction [37, 71]:** Este módulo aplica técnicas de extracción de características como i. Cálculo de característica basada en la media, ii. Cálculo de característica basada en la mediana y iii. Cálculo de característica basada en cuartiles.
- **Feature Selection [37, 72]:** Este módulo aplica técnicas de selección de características como i. Importancia de la Característica por Permutación. ii. Eliminación de Multicolinealidad. iii. Filtrado por Baja Varianza. Y iv. Selección de Características usando metaheurísticas como los Algoritmos Genéticos.

MEM agrupa los procesos (submódulos) asociados al diseño, entrenamiento y evaluación de modelos de Aprendizaje Automático, permitiendo al Meta-Algoritmo Autónomo, con información de MTM y MMM, determinar la arquitectura e hiperparámetros óptimos para cada tarea. Adicionalmente, ofrece procesos para la reutilización de la información de los modelos previamente creados. Los procesos disponibles se detallan a continuación:

- **Create Model [9]:** Este módulo se encarga de entrenar y validar los modelos de Aprendizaje Automático, utilizando como base de conocimiento la información proporcionada por MLM sobre la configuración del algoritmo a utilizar y su respectivo conjunto de datos.
- **Model Transfer [66, 38, 64]:** El objetivo de este módulo es transferir el mejor modelo construido del dominio de origen al dominio de destino. Para ello, se selecciona el mejor modelo entrenado del dominio de origen cuando las características del modelo de origen son muy similares a las características del modelo de destino. Este modelo será el que se transfiera. En segundo lugar, se ajustan las variables del conjunto de datos de destino para que coincidan con las variables del conjunto de datos de origen, lo que permitirá utilizar correctamente el modelo que se va a transferir. Por último, se entrega el conjunto de datos de destino ajustado con el modelo de origen transferido.
- **Parameter Transfer [66, 38, 64]:** El objetivo de este módulo es transferir los parámetros del mejor modelo construido en el dominio de origen al dominio de destino. Para ello, en primer lugar, se selecciona el mejor modelo entrenado en el dominio de origen. En segundo lugar, se transfieren los parámetros de este modelo al modelo del dominio de destino para mejorarlo. Los dominios de origen son aquellos modelos de Aprendizaje Automático con mayor rendimiento y cuyos conjuntos de datos sobre los que se ha entrenado tienen una mayor similitud estadística

con el dominio de destino. El modelo más similar y mejor es el que se utiliza para transferir todos sus parámetros.

5.2.1 *Generación de Características*

Este apartado resume el trabajo presentado en [69], cuyos detalles se encuentran en la sección III del artículo presentado en el Anexo 5.C. A continuación, se describe la arquitectura propuesta, denominada EAFECNN (Explainability Analysis FE-CNN), detallando cada uno de sus módulos. Esta arquitectura surge de la necesidad de aprovechar la capacidad de las CNN para la generación de características en problemas con datos tabulares. Para ello, se implementan dos mecanismos que automatizan la ingeniería de características (véase la Figura 5.4): en primer lugar, la transformación de datos tabulares a imágenes (*Multidimensional Transformation*), que introduce la generación implícita de características al cambiar la representación de la información; y en segundo lugar, el uso de la capacidad inherente de las CNN para generar automáticamente características relevantes en cada capa (*CNN Feature Generator*). Además, se incorporan técnicas de análisis de explicabilidad para comprender mejor el funcionamiento interno del modelo y ofrecer una mayor transparencia en las decisiones tomadas (*CNN Explainability Analysis*). Esta combinación permite una mayor confianza y transparencia en la generación de características en tareas de Aprendizaje Automático. Así, proponemos una arquitectura compuesta por los siguientes módulos (véase la Figura 5.4): Multidimensional Transformation (MT), CNN Feature Generator (CNN-FG), y CNN Explainability Analysis (CNN-EA). A continuación se describe detalladamente cada módulo.

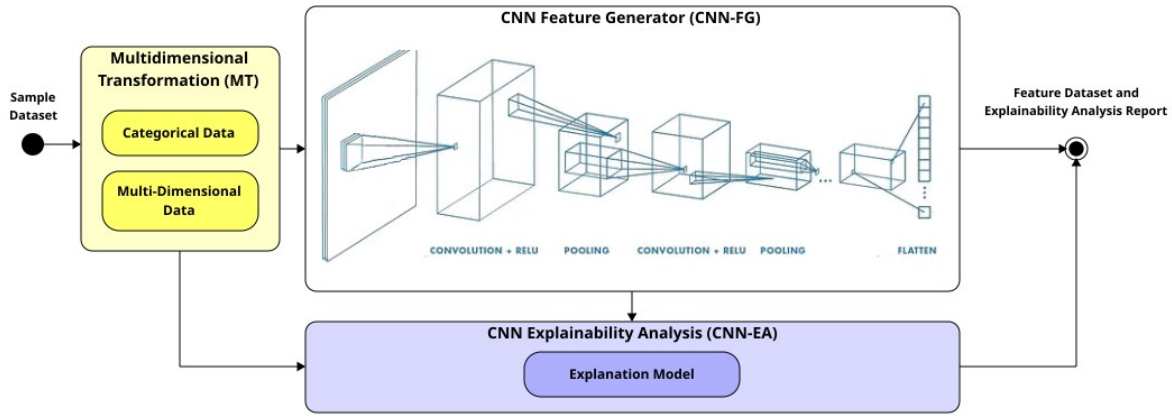

Figura 5.4: Arquitectura EAFECNN.

A. Módulo MT

El principal objetivo de este módulo es preparar la muestra de datos de forma óptima para su uso en modelos basados en CNN. Esto implica transformar los datos tabulares en una representación multidimensional, concretamente en imágenes sintéticas [73]. Existen varios métodos para realizar esta transformación que, en general, se basan en asignar los datos a posiciones o escalas de color específicas dentro de los píxeles de la imagen. De este modo, las características de los datos se representan

visualmente en la imagen sintética. TINTOlib (https://tintolib.readthedocs.io/) es una biblioteca que proporciona diversos métodos para realizar esta tarea de transformación, como:

- **TINTO:** Este algoritmo convierte datos en imágenes mediante la representación de píxeles característicos, aplicando métodos de reducción bidimensional como PCA y T-SNE.
- **SuperTML:** Este algoritmo asigna cada característica a una región única dentro de la imagen. El valor de la característica se representa como texto sobre fondo negro, utilizando un tamaño de fuente que puede ser fijo o variable en función de la importancia de la característica.
- **IGTD:** Este algoritmo asigna cada característica a una posición de píxel específica en la imagen. La intensidad del píxel se utiliza para representar el valor de la característica correspondiente en la muestra.
- **REFINED:** Este algoritmo tiene en cuenta las similitudes entre las características para generar un mapa de características conciso en forma de imagen bidimensional minimizando los valores de distancia por pares siguiendo un enfoque de escalado multidimensional métrico bayesiano.
- **BarGraph:** Este algoritmo genera un gráfico de barras blancas sobre fondo negro para representar cada una de las muestras. Cada barra representa una característica normalizada específica presente en el conjunto.
- **DistanceMatrix:** Este algoritmo calcula la matriz de distancias entre las características y luego aplica una normalización para establecer una escala blanco/negro entre 0 y 255.
- **Combination:** Este algoritmo combina BarGraph y DistanceMatrix, utilizando una capa de color diferente de la imagen para cada algoritmo.

La tabla 5.1 muestra el macro-algoritmo MT. Este proceso comienza con la preparación del conjunto de datos para garantizar que sean compatibles con el proceso de transformación (paso 1), ya que solo admite conjuntos de datos numéricos. Se trata de convertir los valores textuales en valores numéricos discretos que representen las distintas categorías presentes en los datos (paso 1.1). Por último, en la etapa 2, se aplica el método de transformación seleccionado al conjunto de datos preparado. El proceso de transformación consiste en convertir cada fila del dataset en una imagen. Las imágenes se agrupan en carpetas según las distintas clases objetivas. El resultado final es un conjunto de datos multidimensional que será utilizado por el modelo de Aprendizaje Automático basado en CNN.

Tabla 5.1: Macro-algoritmo del módulo MT que transforma el conjunto de datos para CNN.

Entrada: Dataset de muestra y Método de Transformación
Procedimiento: **1**. Prepara el Dataset para la Transformación **1.1.** Se categorizan los atributos textuales **2.** Transforma el Dataset de muestra con el Método de Transformación
Salida: Dataset Multidimensional

B. Módulo CNN-FG

El objetivo de este módulo es realizar el proceso de generación de características utilizando modelos basados en CNN. Estos modelos tienen la capacidad de automatizar este proceso, utilizando las capas iniciales e intermedias del modelo CNN, ya que es en estas capas donde se realiza este proceso. En concreto, se utiliza un modelo ResNet-50 pre-entrenado (ver Figura 5.5), aprovechando las capas de las etapas 1 a 5, encargadas de extraer características básicas y de bajo nivel de las imágenes, como bordes, colores y texturas; y descartando las capas siguientes (recuadro rojo), encargadas de aprender

características más específicas y de alto nivel. Al utilizar un modelo preentrenado como ResNet-50, se aprovecha el conocimiento adquirido a partir de grandes datasets, lo que reduce significativamente el tiempo y los recursos necesarios para entrenar el modelo en una tarea específica, además de regularizar y evitar el sobreajuste. Además, el modelo preentrenado ya ha aprendido representaciones genéricas de imágenes a partir de un enorme dataset, lo que lo hace adecuado para esta tarea de generación de características.

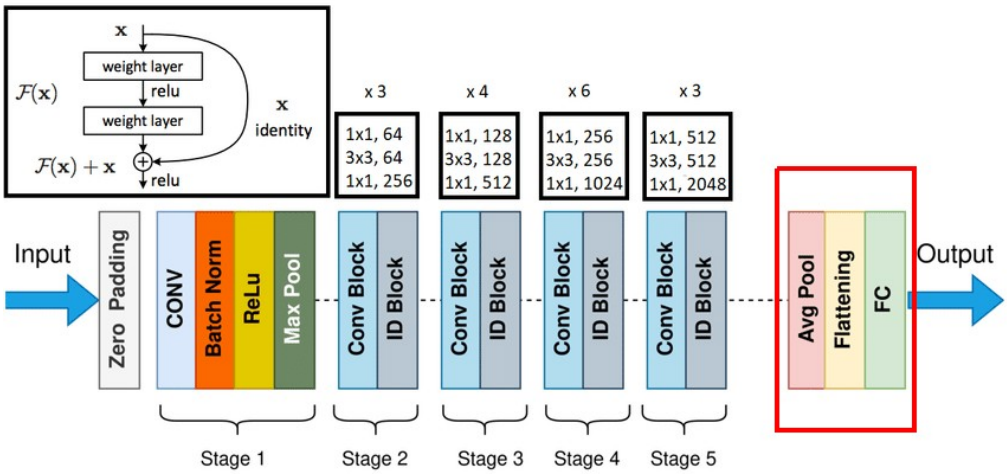

Figura 5.5: Arquitectura RESNET-50.

La tabla 5.2 muestra el macro-algoritmo CNN-FG. Este proceso comienza cargando el modelo ResNet-50 que ha sido previamente entrenado utilizando un conjunto masivo de datos de imágenes (Paso 1). En el Paso 2, se procede a eliminar las capas de clasificación, que son las capas finales del modelo, buscando aprovechar el conocimiento adquirido por las capas intermedias del modelo, expertas en extraer información visual relevante. A continuación, el modelo modificado se utiliza para procesar cada elemento del Dataset Multi-Dimensional generado por el módulo anterior, generando una representación vectorial de las características genéricas presentes en la imagen (Paso 3). Por último, las características generadas se extraen del modelo modificado y se almacenan en el Dataset de Características (Paso 4).

Tabla 5.2: Macro-algoritmo del módulo CNN-FG para generar las características.

Entrada: Dataset Multi-Dimensional
Procedimiento: **1.** Carga el modelo ResNet-50 preentrenado. **2.** Elimina las capas finales del modelo cargado. **3.** Procesa el Dataset Multi-Dimensional a través del modelo modificado. **4.** Guarda las características generadas en el Dataset Optimizado.
Salida: Dataset de características

C. Módulo CNN-EA

El objetivo de este módulo es revelar el funcionamiento interno del modelo CNN proporcionando una representación visual o textual que detalle el razonamiento que subyace a las decisiones tomadas por el modelo para cada imagen del conjunto de datos. Esta transparencia en el proceso de toma de decisiones permite a los usuarios comprender mejor las capacidades y limitaciones del modelo, identificar posibles sesgos o errores, y generar confianza en sus resultados. Existen varias técnicas de explicabilidad para llevar a cabo esta tarea, por ejemplo:

- **GRADCAM (Gradient-weighted Class Activation Mapping) [74]:** Genera un mapa de calor que resalta las regiones de la imagen que más contribuyeron a la predicción del modelo calculando el gradiente de la puntuación de clasificación con respecto a las activaciones de las características convolucionales.
- **SCORECAM (SCoring by Output RE-CAM) [75]:** Genera un mapa de calor que muestra las ROIs (Regiones de Interés) y sus puntuaciones. Las ROIs son regiones de interés donde más influyen en la predicción del modelo. Para asignarles una puntuación, se calcula el gradiente de la función de pérdida del modelo con respecto a la activación de cada ROI.
- **LAYERCAM (Layer-wise Attention Chain-based Attention Mapping) [76]:** Genera un mapa de activación visualizando qué partes de las entradas son más importantes para la predicción del modelo, utilizando una cadena de atención para propagar la atención desde la última capa de la CNN a las capas anteriores.
- **GUIDEDBP (Guided Upward Input Deep Back Propagation) [77]:** Genera un mapa de calor, que muestra las regiones que tuvieron mayor impacto en la clasificación. Se basa en la idea de modificar el proceso de retropropagación para identificar las regiones de entrada que más contribuyen a la activación de una neurona específica.

La Tabla 5.3 muestra el macro-algoritmo CNN-EA, este proceso comienza seleccionando la técnica de explicabilidad deseada (Paso 1). En el Paso 2, la técnica de explicabilidad seleccionada se aplica a cada imagen del Dataset Multi-Dimensional, y se genera un informe de las representaciones visuales que explican cómo el modelo CNN toma decisiones para cada imagen.

Tabla 5.3: Macro-algoritmo del módulo CNN-EA para analizar el modelo.

Entrada: Dataset Multi-Dimensional y Modelo CNN
Procedimiento: **1.** Selecciona la técnica de explicabilidad **2.** Genera un Informe de Análisis de Explicabilidad según la técnica seleccionada
Salida: Informe del Análisis de Explicabilidad

5.2.2 *Generación de Datos Artificiales*

Esta sección presenta un resumen extenso de los trabajos presentados en [38, 37], y los detalles se encuentran en la sección II del artículo presentado en el Anexo 5.D y en la sección III del artículo presentado en el Anexo 5.E. La generación de datos artificiales implica un conjunto de procesos complejos que permiten identificar, extraer, transformar y aprender las características relevantes del conjunto de datos a generar. La arquitectura SDGS (Synthetic Data Generation System) logra esto mediante la combinación del paradigma de Datos Enlazados para identificar y extraer datos de Internet, y la técnica VAE para transformar y aprender un modelo con estos datos, de forma que este modelo

pueda utilizarse posteriormente para generar datos sintéticos. Esta arquitectura se compone de los siguientes módulos (véase la Figure 5.6) [38]:

- DataSet Acquisition (DSA): El objetivo de este módulo es encontrar muestras de datos a través de mecanismos de búsqueda basados en Datos Enlazados, aprovechando ODS o endpoints.
- Data Preparation (DP): El objetivo de este módulo es optimizar la muestra de datos. Normaliza atributos numéricos con alta varianza y procesa datos textuales o numéricos que representan un conjunto finito específico de categorías o clases.
- Synthetic Data Generation (SDG): El objetivo de este módulo es generar datos sintéticos entrenando un modelo de conocimiento basado en VAE, que extrae y aprende automáticamente las características de la muestra de datos optimizada para un contexto determinado.

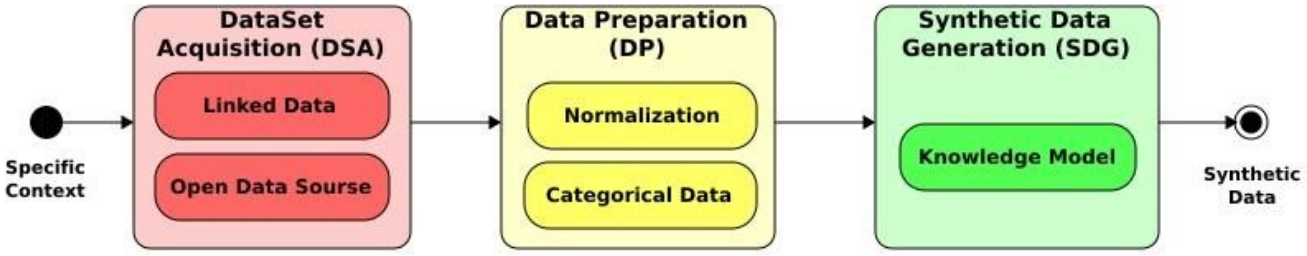

Figura 5.6: Arquitectura de generación sintética de datos.

Luego, en [37] se amplió el enfoque propuesto (ver Figura 5.7). En primer lugar, añadiendo dos procesos al módulo DSA, un primer proceso centrado en el uso de múltiples fuentes de datos y un segundo proceso centrado en la fusión de múltiples conjuntos de datos. Además, se añade un nuevo módulo, denominado Feature Engineering (FE), para analizar las características de los conjuntos de datos de muestra que utilizará el SDG, permitiendo la fusión de características, la extracción de características y la selección de características. Por último, el SDG se implementa utilizando la técnica VAE como generador de datos.

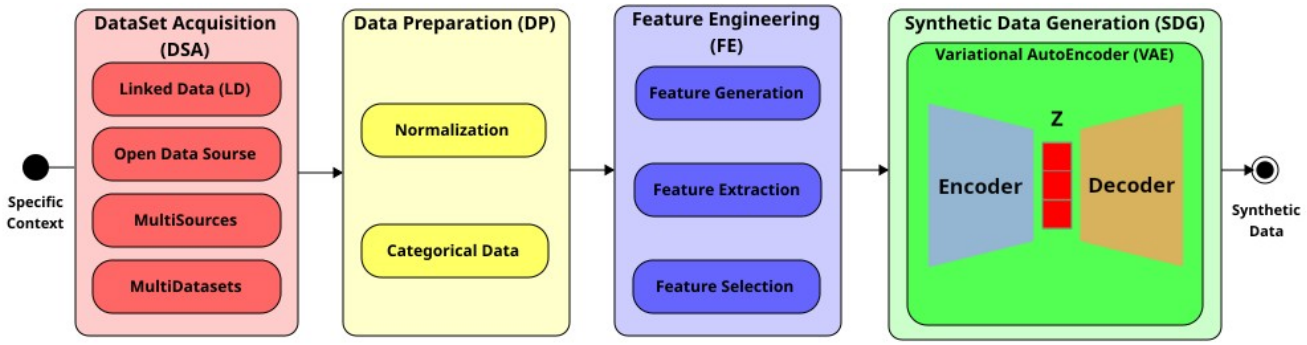

Figura 5.7: Ampliación del SDGS.

A. Módulo DSA

El objetivo de este módulo es encontrar muestras de datos para un contexto dado utilizando el paradigma de Datos Enlazados. La Tabla 5.4 muestra el macro-algoritmo del módulo DSA. Este

módulo comienza analizando el contexto de las muestras de datos requeridas, obteniendo las palabras clave para buscar las muestras de datos (Paso 1). A continuación, se invoca el proceso *Multi-sources* para buscar los datasets utilizando las palabras clave obtenidas en el paso anterior (Paso 2). Por último, si se decide fusionar los datasets obtenidos con otro dataset de un contexto asociado (Paso 3), se invoca el proceso de *Multi-datasets* (Paso 3.1).

Tabla 5.4: Macro-algoritmo del módulo DSA.

Entrada: Contexto específico
Procedimiento: 1. Se analiza el contexto de los datos requeridos para obtener las palabras clave de búsqueda. 2. Se invoca el proceso Multi-sources para encontrar una Muestra de Datos utilizando las palabras clave de búsqueda 3. Si se fusiona la Muestra de Datos: 3.1 Invocar el proceso Multi-dataset de datos para fusionar la Muestra de Datos con el dataset del contexto asociado.
Salida: Nuevo dataset

A.1. Proceso Multi-sources en DSA

El objetivo de este proceso es encontrar muestras de datos de diferentes fuentes de datasets. En concreto, las muestras de datos se obtienen utilizando mecanismos de búsqueda basados en Datos Enlazados, aprovechando el ODS, buscando en cada fuente de dataset registrada en el SDGS y seleccionando los datasets de muestra que mejor se ajusten al contexto específico requerido. Estas fuentes de datasets son proporcionadas oficialmente por países/regiones como Europa (https://data.europa.eu), España (https://datos.gob.es/), Canadá (https://open.canada.ca), Estados Unidos (https://www.data.gov/), y otros. Los metadatos publicados por estas fuentes siguen el estándar CKAN (https://ckan.org/), que permite realizar consultas utilizando el lenguaje SPARQL. La Tabla 5.5 muestra el macro-algoritmo del Proceso Multi-sources en DSA. Este proceso comienza preparando las consultas de búsqueda para cada fuente de dataset (Paso 1). A continuación, se ejecuta cada consulta en cada fuente de dataset, y se obtiene la lista de posibles datasets (Paso 2). Por último, la lista se ordena según el grado de coincidencia con el contexto requerido (Paso 3).

Tabla 5.5: Macro-algoritmo del proceso multisources para buscar muestras de datos.

Entrada: Palabras clave de búsqueda
Procedimiento: 1. Prepara las consultas de búsqueda con las palabras clave de búsqueda para cada fuente de dataset basándose en el paradigma de Datos Enlazados. 2. La búsqueda se ejecuta para cada fuente de dataset y se añade a la lista de posibles muestras de datos. 3. Se clasifican y seleccionan las muestras de datos que mejor se ajustan a la búsqueda.
Salida: Muestra de datos

A.2. Proceso Multi-datasets en DSA

El objetivo de este proceso es construir datos de muestra que combinen información de diferentes datasets. En concreto, teniendo un dataset del contexto principal requerido, se busca otro dataset perteneciente a un contexto asociado al contexto principal. A continuación, se buscan relaciones entre las características de los datasets; las coincidencias se utilizan como pivotes para la fusión de ambos datasets. La tabla 5.6 muestra el macroalgoritmo del proceso Multi-datasets en DSA. Este proceso comienza invocando el proceso Multi-sources para encontrar una Muestra de Datos del Contexto Asociado (Paso 1). En el Paso 2, busca las similitudes de ambas Muestras de Datos; esta similitud se basa en el nombre y el tipo de cada característica de la Muestra de Datos Principal y de la Muestra de Datos del Contexto Asociado. Por último, fusiona la Muestra de Datos Principal y la Muestra de Datos de Contexto Asociada utilizando las similitudes como pivote (Paso 3), generando una Muestra de Datos Fusionada.

Tabla 5.6: Macro-algoritmo del proceso multidatasets para fusionar muestras de datos.

Entrada: Muestra de datos principales, palabras clave de búsqueda contextual asociadas
Procedimiento: 1. Se invoca el proceso Multi-sources para encontrar una Muestra de Datos utilizando las Palabras Clave de Búsqueda de Contexto Asociadas. 2. Busca posibles similitudes entre las características de la Muestra de Datos Principal y la Muestra de Datos de Contexto Asociada. 3. Fusione ambas Muestras de Datos usando las similitudes como pivote.
Salida: Muestra de datos fusionada

B. Módulo DP

El objetivo de este módulo es transformar el dataset de la muestra en una representación óptima para el modelo VAE, sabiendo que este tipo de modelo funciona óptimamente con datos que oscilan entre [0 y 1] o [-1 y 1], ya sean datos binarios (digitales) o continuos (analógicos). En este proceso, se le atribuyen datos textuales o numéricos que representan un conjunto finito específico de categorías o clases que se procesan como Datos Categóricos, donde los datos numéricos y de alta varianza se normalizan. La Tabla 5.7 muestra el macro-algoritmo DP, que comienza analizando el dataset para determinar los procesos que serán necesarios para cada columna de la muestra de datos (Paso 1). Para las columnas con datos numéricos o con muchos valores diferentes, procede a normalizarlos (Paso 2). Para las columnas con datos textuales o numéricos que pueden representarse en categorías o clases, procede a categorizarlas (Paso 3).

Tabla 5.7: Macro-algoritmo del módulo DP para optimizar la muestra de datos.

Entrada: Dataset de muestra
Procedimiento: 1. Se analiza la muestra de datos. 2. Se normalizan los atributos con datos numéricos y alta varianza. 3. Se categorizan los atributos con datos textuales y numéricos con valores finitos.
Salida: Muestra de datos preprocesados

C. Módulo FE

El objetivo de este módulo es analizar las características del dataset de muestra. Concretamente, analiza la muestra de datos preprocesados generada por el módulo DP, obteniendo nueva información a partir de las características del dataset y seleccionando las características que ofrecen más información para el módulo SDG. La Tabla 5.8 muestra el macro-algoritmo del módulo FE. Este proceso comienza con el análisis del dataset (paso 1). Después, en el paso 1.1, la información se agrega aplicando técnicas de generación de características como i. Característica de Interacción, ii. Característica Polinómica, iii. Característica Trigonométrica, iv. Creación de Clusters, v. Combinación de Niveles Raros. En el paso 1.2 se añade información aplicando técnicas de extracción de características como i. Cálculo de característica basada en la media, ii. Cálculo de característica basada en la mediana y iii. Cálculo de característica basada en cuartiles. Por último, en el paso 1.3, selecciona las características que ofrecen más información, aplicando técnicas de Selección de Características como i. Importancia de la Característica por Permutación. ii. Eliminación de Multicolinealidad. iii. Filtrado por Baja Varianza. Y iv. Selección de Características usando metaheurísticas como los Algoritmos Genéticos.

Tabla 5.8: Macro-algoritmo de FE para mejorar la muestra de datos.

Entrada: Muestra de datos preprocesados
Procedimiento: 1. Analiza la Muestra de Datos Preprocesada: 1.1. Añade nueva información generada a partir de sus características 1.2. Añade nueva información extraída de sus características. 1.3. Seleccionar las características que aportan más información.
Salida: Muestra de datos mejorada

D. Módulo SDG

El objetivo de este módulo es generar los datos sintéticos a partir de la muestra de datos optimizada en el módulo anterior. En este proceso, se construye y entrena un modelo de conocimiento que extrae y aprende automáticamente las características de la muestra de datos utilizando VAE. La Tabla 5.9 muestra el macro-algoritmo SDG, el proceso comienza configurando y construyendo el modelo de conocimiento que aprenderá las características latentes en la muestra de datos (Paso 1). En el Paso 2 se procede a entrenar el modelo de conocimiento utilizando VAE y la muestra de datos. Por último, se genera el dataset sintético utilizando el modelo de conocimiento previamente creado y entrenado (Paso 3).

Tabla 5.9: Macro-algoritmo de SDG para la generación de datos sintéticos.

Entrada: Muestra de datos mejorada
Procedimiento: 1. Se construye el modelo de conocimiento con la configuración deseada. 2. Se entrena el modelo de conocimiento que representa la muestra de datos. 3. Se genera el dataset sintético con el modelo de conocimiento.
Salida: Dataset sintético

5.3 Casos de Estudio

El presente apartado detalla, en la sección 5.3.1, un caso de estudio demostrativo de la activación de los módulos de la arquitectura mediante el Meta-Algoritmo Autónomo. Posteriormente, las secciones 5.3.2 y 5.3.3 presentan casos de estudio enfocados en la generación de características y en la generación de datos, respectivamente.

5.3.1 *Caso 1: Meta-Algoritmo Autónomo*

Esta sección presenta un resumen del trabajo referenciado en [63], cuyos detalles se encuentran en la sección 5 del artículo presentado en el Anexo 5.B. El caso de estudio se centra en la información recogida en Café Galavis (Cúcuta, Colombia). Este caso de estudio muestra el proceso de activación de los módulos de la arquitectura, guiados por el Meta-Algoritmo Autónomo y basados en los requerimientos de Café Galavis, para generar modelos de conocimiento que resuelven diferentes problemas. En concreto, nuestra arquitectura ejecuta determinados grupos de pasos en función de los requerimientos y la información del entorno. La Figura 5.8 muestra estos pasos que se utilizarán en los experimentos: 1) Transferencia del Modelo, 2) Transferencia de Parámetros, 3) Creación del Modelo, 4) Transferencia de Datos y 5) Generación de Datos Sintéticos.

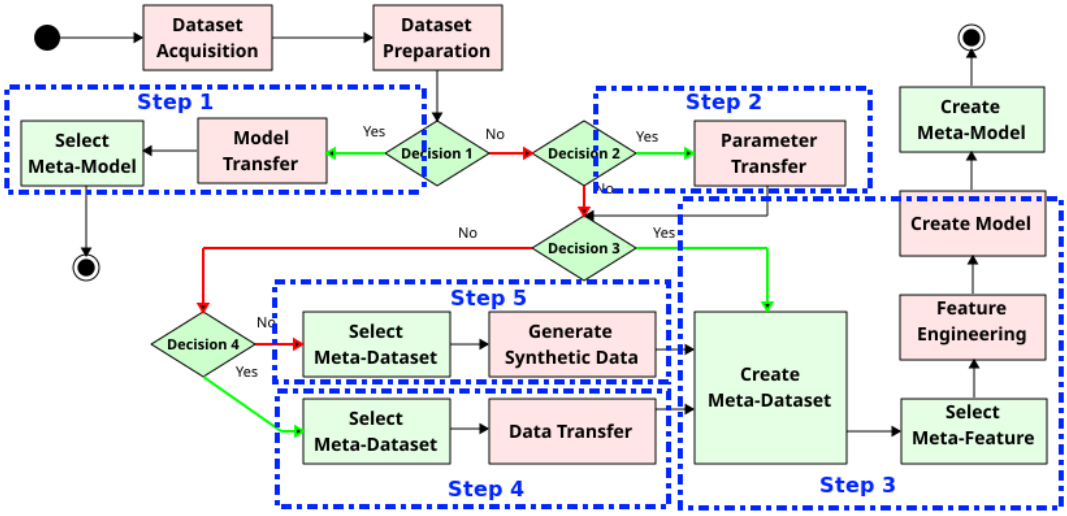

Figura 5.8: Grupos de pasos del meta-algoritmo autónomo.

En este caso de estudio se realizan 5 experimentos, los cuales se llevan a cabo de forma secuencial y cada uno de ellos se basa en los resultados de los anteriores, lo que permite una retroalimentación progresiva del sistema. Los experimentos muestran la dinámica de activación de las distintas partes del Meta-Algoritmo en la arquitectura. Para ello se utiliza la herramienta de trazabilidad implementada en la arquitectura, que facilita la visualización secuencial de las decisiones tomadas y los conjuntos de

pasos activados en cada instante de la ejecución. Ahora bien, en esta sección solo detallaremos el primer experimento, el resto se puede consultar en la sección 5 del artículo presentado en el Anexo 5.B.

El experimento 1 se dedica a evaluar la activación de los módulos de Generación de Datos Sintéticos (Paso 5) y Creación de Modelos (Paso 3) solicitando un modelo No Supervisado/Clustering a partir del archivo File1.csv (compuesto por 1.286 registros que incluye características como método de procesado, variedad de semilla, aroma, sabor, acidez, cuerpo, uniformidad, dulzor y humedad, y otros). Además, hay que considerar el estado inicial que la arquitectura mantiene para la gestión del proceso de Meta-Aprendizaje. Esta información se estructura en tablas que contienen el Meta-Modelo (modelos previamente entrenados), Meta-Dataset (características de los datasets) y Meta-Técnica (algoritmos de Aprendizaje Automático), los cuales se pueden consultar en la sección 5.1 del artículo presentado en el Anexo 5.B.

La Figura 5.9 presenta la trazabilidad de la ejecución del Meta-Algoritmo para resolver esta petición. El primer paso es Init (siempre llamado en todas las ejecuciones), que activa los procesos *Dataset Acquisition* y *Dataset Preparation*. El proceso *Datasets Acquisition* comprueba si se ha suministrado un dataset de entrada o si hay que buscarlo con Datos Enlazados. En este caso, se ha suministrado el dataset de entrada. A continuación, se activa el módulo *Datasets Preparation* para normalizar las variables. Entonces, como no hay modelos creados previamente para Unsupervised/Clustering en la tabla Meta-Model (Ver Tabla 3 en la sección 5.1 del Anexo 5.B), las decisiones 1 y 2 resultaron en “No” (ver Figuras 5.8 y 5.9). En la decisión 3, se rechazó la generación del modelo porque no se alcanzó el umbral mínimo de 2000 registros. Por último, en la decisión 4, el resultado es «No» debido a la ausencia de datasets muy similares en la Tabla de Meta-Datasets (Ver Tabla 2 en la sección 5.1 del Anexo 5.B). Por lo tanto, se decidió generar datos sintéticos activando el módulo *Generate Synthetic Data* para completar el dataset (Paso 5 en la Figura 5.8) y, posteriormente, realizar la activación de *Feature Engineering*, para finalmente, crear modelo usando el módulo *Create Model* (Paso 3 en la Figura 5.8). El módulo *Generate Synthetic Data* emplea un VAE para generar datos sintéticos a partir de conjuntos de datos dispersos. La VAE aprende una representación latente comprimida de los datos de entrada, capturando sus características más relevantes. A continuación, utiliza esta representación para generar nuevos datos que siguen una distribución similar a la de los datos originales. Para una comprensión más detallada del proceso, se recomienda revisar la sección 5.2.2.

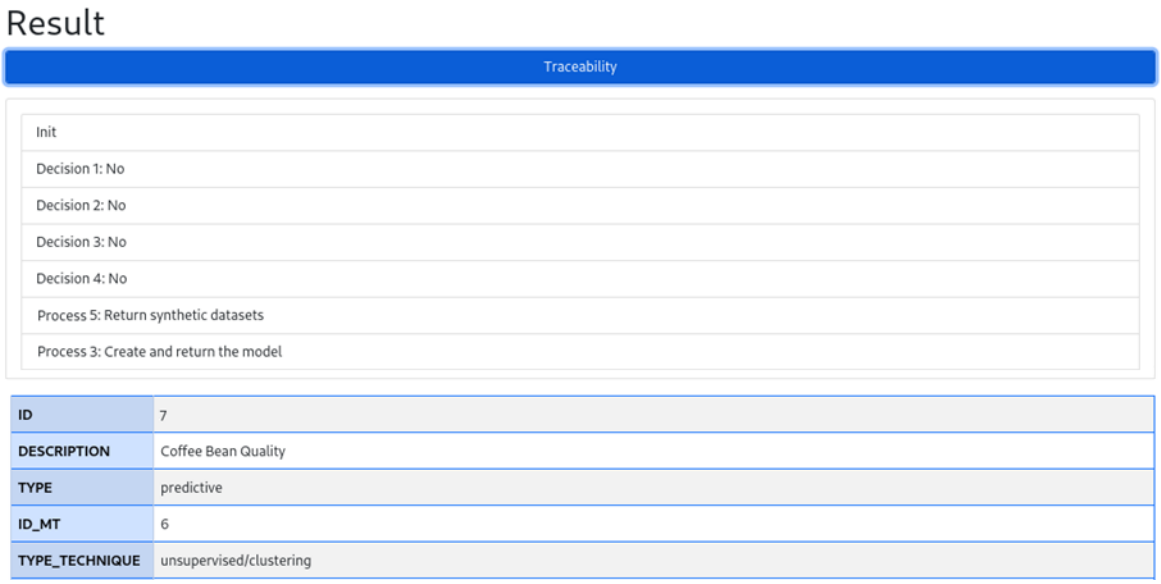

Figura 5.9: Trazabilidad de la búsqueda de un modelo para el experimento 1.

Además, como en este experimento no hubo transferencia de modelos ni de parámetros, el Meta-Algoritmo Autónomo buscó en la tabla de Meta-Técnicas (ver Tabla 1 en la sección 5.1 del artículo presentado en el Anexo 5.B) las posibles técnicas de Aprendizaje Automático para resolver el problema de Unsupervised/Clustering. Esto generó la creación de un modelo con cada una de las tres técnicas encontradas, que se añadieron a la Tabla de Meta-Modelos (ver Tabla 5.10).

Tabla 5.10: Modelos añadidos en la tabla de Meta-Modelos de la arquitectura.

Id_MM	Tipo	Id_MT	P1, P2, Pn	VD1 …VDn	Métrica	Id_MD
MM_05	No supervisado/ Agrupación	MT_05	K=2	Aroma, Sabor, Retrogusto, Acidez, Cuerpo, Equilibrio, Uniformidad.	silhouette index=0.62	File1
MM_06	No supervisado/ Agrupación	MT_07	K=4, Distance technique= Euclidean, distance calculation=max,.	Aroma, Sabor, Retrogusto, Acidez, Cuerpo, Equilibrio, Uniformidad.	silhouette index=0.57	File1
MM_07	**No supervisado/ Agrupación**	**MT_06**	**Epsilon (eps) = 5 and Minimum Points (minPts): 5.**	**Aroma, Sabor, Retrogusto, Acidez, Cuerpo, Equilibrio, Uniformidad.**	**DBCV = 0.93**	**File1**

Finalmente, el Meta-Algoritmo Autónomo responde con el mejor modelo alcanzado (ver Tabla 5.10). La Figura 5.10 muestra el mejor rendimiento (Métricas) alcanzado utilizando DBScan con 0,9315 (DBCV), muy próximo a uno. Este modelo clasificó la calidad del café en tres grupos: baja, media y alta.

Figura 5.10: Resultado del experimento 1 utilizando DBScan.

5.3.2 *Caso 2: Generación de Características*

Este escenario surge cuando la Arquitectura de Meta-Aprendizaje requiere la aplicación de ingeniería de características, específicamente, la generación de características a partir de los datasets. La sección IV del artículo [69] presentado en el Anexo 5.C realiza una descripción detallada. El contexto en el que se desarrolla este caso de estudio se centra en la generación de características de un dataset clásico utilizado en muchos artículos, que permite la clasificación multiclase a partir de 150 instancias con 4 características (SepalLength, SepalWidth, PetalLength y PetalWidth) y una variable «Species» con 3 clases (Setosa, Versicolor y Virginica), para clasificar las especies de la planta Iris.

Como este generador trabaja con un modelo CNN, el primer paso para optimizar el dataset es transformarlo en imágenes. Para ello, se utilizan los métodos disponibles en la librería TINTOlib de Python (ver sección 5.2.1.A), estos métodos en general asignan los datos a posiciones o escalas de color específicas dentro de los píxeles de la imagen generada. La figura 5.11 muestra las transformaciones a imágenes aplicadas a la primera instancia del conjunto de datos (SepalLength=5,1, SepalWidth=3,5, PetalLength=1,4, PetalWidth=0,2 y Species=1) utilizando todos los métodos disponibles en dicha librería. Las transformaciones se presentan de izquierda a derecha y de arriba abajo, en el orden siguiente de los métodos: TINTO, SuperTML, IGTD, REFINED, BarGraph, DistanceMatrix y Combination.

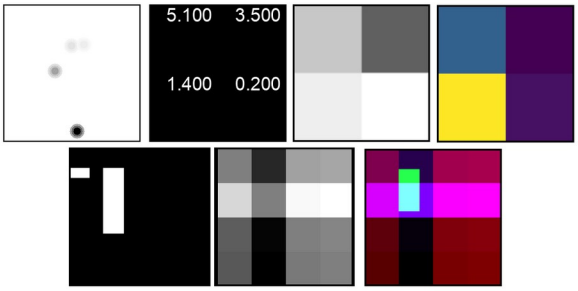

Figura 5.11: Transformaciones de la primera instancia del dataset mediante TINTOlib.

Luego, se carga el modelo preentrenado ResNet-50 de la biblioteca PyTorch (https://pytorch.org/vision/stable/models.html), se eliminan las capas de clasificación, y se añade una capa de aplanamiento para vectorizar la salida. Finalmente, utilizando el modelo modificado y el conjunto de datos multidimensional (imágenes) obtenido en el paso anterior, se procede a generar la representación vectorial de las características. Cabe destacar que este modelo genera 2048 características para cada imagen, debido a que la capa intermedia final del modelo tiene esa cantidad de neuronas de salida (ver figura 5.12). Estas características son el producto de las capas convolucionales. Las primeras capas convolucionales identifican elementos de bajo nivel, como bordes y texturas, y a medida que la información progresa, se detectan características cada vez más complejas y abstractas. Así, el resultado es una representación compacta y de alto nivel de la información relevante extraída de la entrada original.

	x0	x1	x2	x3	x4	...	x2044	x2045	x2046	x2047	Class
0	0.375542	0.475885	0.458746	0.437844	0.578186	...	0.430306	0.334730	0.421343	0.309855	1
1	0.366351	0.508071	0.468319	0.448030	0.568871	...	0.460284	0.323036	0.418428	0.322821	1
2	0.345320	0.470540	0.467325	0.417580	0.567802	...	0.443132	0.360291	0.418604	0.352692	1
3	0.353170	0.519243	0.485133	0.407714	0.569125	...	0.412885	0.329373	0.410133	0.312192	1
4	0.376448	0.467801	0.461052	0.436512	0.567671	...	0.428124	0.360698	0.403952	0.315482	1
145	0.351755	0.561540	0.450292	0.423473	0.536662	...	0.458759	0.315500	0.471180	0.343557	3
146	0.326104	0.535177	0.451778	0.430014	0.580830	...	0.451719	0.271245	0.475145	0.339071	3
147	0.333834	0.523492	0.429271	0.410115	0.585526	...	0.430730	0.311838	0.467074	0.333061	3
148	0.362585	0.542986	0.431546	0.346176	0.482921	...	0.432381	0.323366	0.442343	0.322393	3
149	0.357758	0.568827	0.453046	0.421222	0.599692	...	0.406421	0.334119	0.474076	0.282702	3

Figura 5.12: Características generadas por cada imagen pasada por el modelo CNN-FG.

5.3.3 *Caso 3: Generación de Datos Artificiales*

El presente escenario surge ante la necesidad de la Arquitectura de Meta-Aprendizaje de recurrir a la generación de datos artificiales. Esta necesidad se manifiesta cuando los datos disponibles son insuficientes para construir y entrenar eficazmente un modelo de Aprendizaje Automático. A continuación, se ofrece un resumen del trabajo descrito en [38], cuyos detalles se encuentran en la sección III del artículo presentado en el Anexo 5.D. El contexto en el que se desarrolla este caso de estudio se centra en la gestión energética de redes inteligentes. En este caso es necesario generar conjuntos de datos sintéticos para diferentes tareas con las características de la muestra de datos. Para ello, se utiliza un modelo VAE, con los siguientes parámetros: i) original_dim: número de neuronas de

entrada o dimensión de los datos de entrada, ii) intermediate_dim: número de neuronas en la capa oculta intermedia, tiene un valor por defecto: 256. iii) latent_dim: número de neuronas en el espacio latente, valor por defecto: 100. iv) batch_size: tamaño del lote, valor por defecto: 100, v) epochs: número de epochs, valor por defecto: 50, vi) epsilon: desviación estándar del tensor, valor por defecto: 0.5. Con esta información, se definen los valores de los parámetros para configurar el modelo de conocimiento que permitirá generar los datos sintéticos (ver Figura 2 en la sección III del artículo presentado en el Anexo 5.D).

Finalmente, se entrena el modelo y se genera el conjunto de datos con N registros utilizando el modelo entrenado. La Figura 5.13 muestra parcialmente el conjunto de datos generado utilizando el modelo de generación aprendido, concretamente, cuatro registros en los que cada columna es una variable diferente.

```
print(model.data_generate(4))
```

```
[[0.49952593 0.49189085 0.50488687 ... 0.48032427 0.5206242  0.5003404 ]
 [0.48942506 0.47716582 0.5042785  ... 0.48819405 0.5360801  0.49854395]
 [0.5010928  0.4956892  0.5165934  ... 0.47339827 0.5332263  0.49898785]
 [0.5035304  0.49954575 0.4980771  ... 0.46986866 0.5088126  0.50419647]]
```

Figura 5.13: Generación de datos sintéticos.

5.4 Entorno de Meta-Aprendizaje ACODAT

Esta sección presenta un resumen extenso del trabajo presentado en [78], cuyos detalles se encuentran en la sección 3 del artículo presentado en el Anexo 5.F. En este trabajo hace uso de la Arquitectura de Meta-Aprendizaje para la creación automática de los modelos de conocimiento para ACODAT (Autonomous Cycles of Data Analysis Tasks). El mecanismo de Meta-Aprendizaje ayuda a ACODAT a aprender a adaptarse rápidamente a nuevos escenarios, usando información como las fuentes de datos, pero además, las meta-características y los meta-modelos ya definidos.

5.4.1 *Sistema Arquitectónico ACODAT*

La figura 5.14 muestra la estructura general del marco propuesto. Esta arquitectura utiliza una base de conocimientos que incluye información sobre los modelos de conocimiento almacenados previamente en ella, así como los datasets y los hiperparámetros de las técnicas utilizadas para construirlos. Cuando es necesario construir un nuevo modelo para un nuevo dataset, el sistema compara la similitud del nuevo dataset con los datasets existentes en el marco para decidir el procedimiento a seguir. En concreto, las opciones son utilizar un modelo existente si el nuevo dataset es muy similar al utilizado para construir el modelo, utilizar solo los parámetros si son algo similares, y construir un nuevo modelo o generar datos sintéticos si no son muy similares. Así, esta arquitectura permite reutilizar conocimientos previos e integrar nuevos conocimientos. A continuación se describen brevemente los módulos de la arquitectura:

- **Management Module:** Este módulo gestiona la base de conocimientos, que consta de una tabla de metamodelos, una tabla de metadatos y una tabla de metatécnicas. Cada modelo de conocimiento del metamodelo está vinculado a un conjunto de datos (en los metadatos) y a la técnica de Aprendizaje Automático (en las Meta-Techniques) utilizada para crearlo. Además, el metamodelo almacena las métricas de calidad del modelo y otros datos relevantes. Los metadatos contienen detalles sobre los conjuntos de datos, como sus atributos y ubicación. Las metatécnicas almacenan información sobre las técnicas de Aprendizaje Automático, incluidos los valores óptimos de sus parámetros.
- **ACODAT Module:** Cuando se recibe un nuevo requisito para construir un modelo de conocimiento sobre un conjunto de datos, se activa este módulo. En primer lugar, extrae características del dataset entrante. A continuación, compara el nuevo dataset con los datasets anteriores que se han utilizado para construir modelos de conocimiento previos, para generar una clasificación de similitud.
- **Adaptation Module:** El módulo de adaptación toma decisiones basadas en el ranking de similitud. Basándose en este ranking, el módulo puede decidir varias cosas: realizar una transferencia de modelo (reutilizarlo) si son muy similares, realizar una transferencia de parámetros (reutilizar los parámetros de las técnicas) si son algo similares, realizar una transferencia de datos (si no son muy similares), o generar datos sintéticos (en estos dos últimos casos, si al nuevo conjunto de datos le faltan datos). En todos los casos, se construye un nuevo modelo con el nuevo conjunto de datos y se invoca el módulo de gestión.

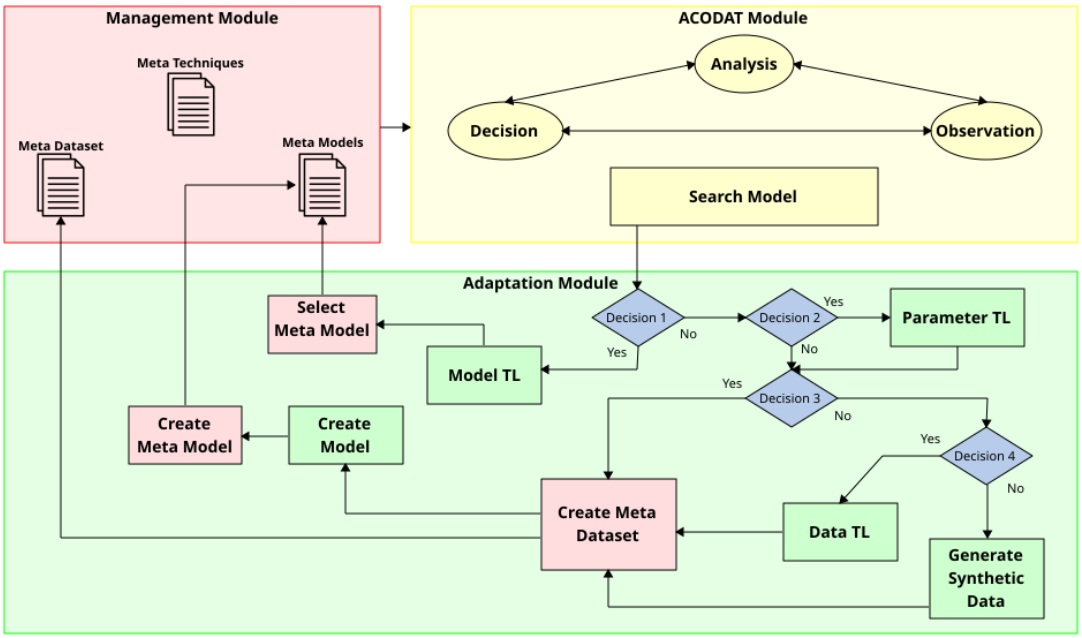

Figura 5.14: Nueva arquitectura MTL ACODAT.

El módulo ACODAT utiliza un ciclo autonomo para guiar el proceso de optimización de la creación de los modelos que se instancian, garantizando la selección de los recursos (modelos, técnicas o datos, según el tipo de transferencia que se vaya a realizar) para adaptar el ACODAT. En la sección siguiente, se detallará las tareas del módulo ACODAT y su impacto en la arquitectura general.

A. Tareas del módulo ACODAT

El módulo ACODAT se encarga de supervisar la ejecución de los demás módulos del framework, pero también, del ACODAT que se está adaptando. Cuando se activa este módulo, se ejecuta un macro-algoritmo que comienza con la tarea Observación, que supervisa y recopila datos e información del sistema o entorno que se está supervisando (información sobre los metadatos, las metatécnicas y los metamodelos), pero también sobre el ACODAT que se está supervisando (tareas que se están ejecutando, técnicas y conjuntos de datos utilizados, etc.). A continuación, la tarea Análisis lleva a cabo los procesos destinados a interpretar y comprender los datos recogidos para diagnosticar lo que está sucediendo en el contexto supervisado, especialmente, sobre las necesidades de adaptación a nivel de cada tarea del ACODAT supervisado. Por último, se activa la tarea Decisión, que en este caso implica acciones encaminadas a construir los modelos de aprendizaje para el ACODAT que se está instanciando. En este caso, invoca al módulo de adaptación para determinar qué tipo de transferencia (de modelos, datos, entre otros) realizar para cada tarea del ACODAT supervisado. Finalmente, este módulo, al ser un ciclo autónomo, observa el comportamiento del ACODAT que se está instanciando para optimizarlo (los resultados de los modelos de aprendizaje son revisados nuevamente en las tareas de Observación y Análisis, iniciando una nueva iteración del ciclo).

5.4.2 *Caso de estudio*

Esta sección presenta un resumen del trabajo presentado en [78], cuyos detalles se encuentran en la sección 4 del artículo presentado en el Anexo 5.F. La arquitectura ACODAT se implementó en la empresa Café Galavis. El ACODAT se diseñó para automatizar el proceso de producción de esta empresa y está compuesto por las siguientes tareas:

- *Tarea 1. Cantidad de insumos a transformar*: Analiza diversos aspectos como la evolución histórica y los usos de la materia prima para generar el producto (la semilla, en este caso). Requiere un modelo de diagnóstico para determinar la materia prima y la cantidad a transformar.
- *Tarea 2. Calidad de los insumos para el proceso*: Mediante esta tarea se establece la calidad de estos insumos, en función de factores como las prácticas culturales y los servicios de almacenamiento y transporte. En esta tarea se utiliza un modelo predictivo para determinar esta calidad.
- *Tarea 3. Método de procesamiento a utilizar*: La identificación de los factores relacionados con el método de procesamiento del café, como el secado natural, el lavado natural, el secado mecánico y la selección automática o manual, es la función principal de esta tarea. En esta tarea se utiliza para ello un modelo de clasificación.

Este módulo comienza activando sus tareas Observation y Analysis. La tarea Observation recoge datos e información de la arquitectura y del ACODAT supervisado. A continuación, la tarea Analysis analiza el ACODAT que se va a adaptar, especificando los tipos de tareas de análisis de datos que lo componen y las fuentes de datos que se van a utilizar para adaptar el ACODAT, entre otras cosas (véase la Tabla 5.11, con los resultados de esta tarea).

Tabla 5.11: Resumen de las tareas del ciclo autónomo supervisado.

Tarea	Tipo de Modelo	Tipo de Técnica	Dataset a usar
1. Calidad del grano de café	Diagnóstico	No supervisado	File1.csv: aroma, sabor, acidez, cuerpo, uniformidad y otras variables.
2. Disminución del grano en el proceso de tostado.	Predictivo	Supervisado	File2.csv: peso crudo, peso tostado, encogimiento, humedad ambiental, densidad, color y otras variables.
3. Método de procesamiento del café	Clasificación	Supervisado	File1.csv: método de transformación, variedad de semillas, aroma, sabor, acidez, cuerpo, uniformidad, dulzor y humedad, entre otras variables.

Luego, se ejecuta el módulo Adaptation, que es el encargado de construir los distintos modelos (ver sección 4 del Anexo 5.F). En la Tabla 5.12 se detallan los resultados y las métricas de los mejores modelos construidos para cada tarea (ver tabla 5.11). Finalmente, el módulo ACODAT verifica la calidad de los nuevos modelos como resultado del módulo Adaptation.

Tabla 5.12: Resultado de los mejores modelos construidos para cada tarea del ciclo autónomo supervisado.

Tarea	ID_Modelo	Algoritmo	Métrica	Dataset usado
1. Calidad del grano de café	6	DBScan	DBCV=0.9	File1.csv
2. Disminución del grano en el proceso de tostado.	8	Gradient Boosting Regressor	R2=0.9547, MAE=0.04, RMSE=0.19	File2.csv
3. Método de procesamiento del café	3	Random Forest Classifier	Accuracy=0.9217, Recall=0.9217, Precision=0.9264, F1=0.9224	File1.csv

6 Conclusiones y Trabajos Futuros

6.1 Conclusiones

La presente investigación ha abordado la generación de conocimiento en Ambientes Inteligentes mediante la integración de Datos Enlazados con mecanismos de Aprendizaje Automático, Meta-Aprendizaje y Lógica Dialéctica. Se partió de la necesidad de superar la falta de una estructura semántica en la Web y la complejidad de explotar inteligentemente el conocimiento en AmI, especialmente en presencia de información inconsistente o ambigua. La investigación se propuso definir una arquitectura computacional que integrara estas capacidades para la generación y explotación de conocimiento en AmI.

En primer lugar, se presentó una ampliación del middleware MiSCi con una capa de Datos Enlazados. Esta capa, guiada por la metodología MEDAWEDE, permite identificar, describir, conectar, relacionar y explotar grandes volúmenes de datos generados por sensores, usuarios y aplicaciones en una ciudad inteligente. Esta capa automatiza el enriquecimiento semántico y la explotación de los datos usando el paradigma de Datos Enlazados mediante cuatro agentes especializados. El Agente *ILDA* se encarga de la extracción, curación y modelado de la información generada por los propios agentes de MiSCi, enriqueciéndola con contexto y ontologías. Así mismo, el Agente *ELDA* realiza una función similar para datos provenientes de fuentes exteriores como redes sociales. El Agente LDIA se dedica a vincular la información enriquecida por ILDA o ELDA con otros conjuntos de datos, para luego publicarlos como Datos Enlazados. Finalmente, el Agente *LDKA* ofrece mecanismos para explotar el conocimiento vinculado a estos datos, proporcionando capacidades avanzadas como análisis semántico, manejo de ambigüedad, recomendación de información, generación de modelos de Aprendizaje Automático, y aprendizaje de ontologías. Además, se propuso una arquitectura para la generación automática y enriquecimiento de ontologías emergentes (AOGS). Esta arquitectura permite crear y poblar ontologías con conocimiento del dominio y vincularlas con fuentes externas de Datos Enlazados, facilitando la construcción de bases de conocimiento semánticamente ricas para contextos específicos.

En segundo lugar, se desarrolló un Sistema de Recomendación Híbrido (HRS) capaz de integrar lógica descriptiva/dialéctica con Datos Enlazados. Este sistema es capaz de resolver situaciones con información inconsistente o ambigua, lo que representa un avance significativo frente a los sistemas de recomendación tradicionales. Su arquitectura se compone de dos motores de razonamiento y cinco gestores de información. Dentro de los motores de razonamiento se encuentra DeLE y DiLE. DeLE explota diversas fuentes de Datos Enlazados mediante consultas basadas en tripletas. DiLE responde mediante consultas construidas como conjeturas sobre modelos de lógica de primer orden, detectando y razonando en estados de ambigüedad o inconsistencia. En cuanto a los gestores de información, VM identifica y selecciona los vocabularios y ontologías necesarios para procesar las peticiones,

apoyándose en el QM para extraer nuevo conocimiento si es preciso. QM prepara y genera las consultas para DeLE (basadas en tripletas) y para DiLE (basadas en conjeturas). ConM se encarga de transformar los datos para permitir el intercambio de información entre los razonadores. RM fusiona y filtra la información obtenida por los razonadores, validando y clasificando las recomendaciones. Finalmente, CM es el responsable de todas las decisiones del HRS, orquestando la invocación de los gestores y razonadores, utilizando meta-razonamiento para verificar consultas, identificar conceptos, extraer conocimiento y filtrar recomendaciones. Esta integración permite la extracción semántica, la verificación y el filtrado de recomendaciones en escenarios complejos como el diagnóstico médico o la evaluación de competencias profesionales. Asimismo, se demostró cómo la Lógica Dialéctica permite modelar y razonar sobre fenómenos como la vaguedad, declaraciones contingentes sobre el futuro, discurso ficticio, fallos de presuposición y razonamiento contrafáctico en el contexto de las competencias profesionales, lo que es crucial para comprender el significado real de las competencias en perfiles digitales.

Por último lugar, se diseñó una arquitectura de Meta-Aprendizaje para la generación de modelos de Aprendizaje Automático basada en Datos Enlazados. El aspecto más innovador es la implementación de un Meta-Algoritmo Autónomo que automatiza la construcción de modelos Aprendizaje Automático, seleccionando las técnicas y herramientas más apropiadas e integrando capacidades avanzadas. Esta arquitectura tiene cuatro módulos especializados que orquestan el proceso completo. El módulo de ingeniería de conjuntos de datos (DEM) se encarga de la preparación y optimización de los conjuntos de datos. El módulo de ingeniería de características (FEM) se dedica a la creación y selección de características relevantes. Existe un módulo específico para la generación de características con CNNs (EAFECNN), donde datos tabulares se transforman en imágenes que son insumo para las CNNs. Finalmente, el Módulo de ingeniería de modelos (MEM) se centra en el diseño, entrenamiento y evaluación de modelos de Aprendizaje Automático. Además, integra de forma inteligente mecanismos de aprendizaje por transferencia (de modelos, parámetros y datos) y la generación de datos sintéticos (utilizando VAE en la arquitectura SDGS). También, se ilustró su aplicación en la optimización de cadenas de producción agroindustrial, demostrando su capacidad para adaptarse rápidamente a nuevos escenarios y construir modelos de conocimiento eficientes.

Esta arquitectura fue extendida para la creación automática de modelos de conocimiento para ACODAT, facilitando su adaptación rápida a nuevos escenarios en contextos como la automatización de cadenas de producción agroindustrial. Esto se logra mediante la adición de un ciclo autónomo que supervisa la ejecución de sus módulos a través de tareas como la Observación (recopila datos del sistema y del ACODAT supervisado), el Análisis (interpreta los datos para diagnosticar necesidades de adaptación) y la Decisión (construye modelos de aprendizaje y determina el tipo de transferencia de conocimiento a realizar, como de modelos o datos).

La validación de las arquitecturas y mecanismos propuestos en esta tesis se ha llevado a cabo mediante diversos casos de estudio prácticos y simulados, demostrando su aplicabilidad y robustez en escenarios complejos de AmI. En el capítulo 3, la ampliación del middleware MiSCi con una capa de Datos Enlazados se ejemplificó en un escenario de ciudad inteligente, abordando la gestión de alarmas y

recomendaciones de servicios. Asimismo, la arquitectura AOGS fue validada en el dominio del COVID-19, evidenciando su capacidad para generar y enriquecer ontologías de forma autónoma. En el capítulo 4 se ilustró la utilidad del Sistema de Recomendación Híbrido en el diagnóstico médico y en el análisis de competencias profesionales, donde la Lógica Dialéctica resolvió inconsistencias y ambigüedades como la vaguedad y el discurso ficticio. Finalmente, en el capítulo 5 se demostró la funcionalidad de la arquitectura de Meta-Aprendizaje y su extensión con un Meta-Algoritmo Autónomo. Se hizo una prueba general de la extensión en experimentos con un dataset de la empresa Café Galavis, enfocándose en la automatización de su cadena de producción agroindustria. Por ejemplo, se usó el Meta-Algoritmo para la creación de modelos no supervisados, lo que permitió, por ejemplo, clasificar la calidad del café en grupos. Además, se hicieron experimentos con algunas partes específicas de la arquitectura de Meta-Aprendizaje. Por ejemplo, se evaluó el proceso de generación de datos sintéticos empleando un VAE para completar datasets dispersos, o la generación de características para clasificación multiclases. Para esto último, la arquitectura EAFECNN transformó datos tabulares en imágenes, que luego fueron procesadas por CNNs para generar características relevantes. Finalmente, se hicieron experimentos para la generación de datos artificiales (o sintéticos) para la gestión energética de redes inteligentes, mediante la arquitectura SDGS, que combina el paradigma de Datos Enlazados con la técnica VAE. Estos casos confirman la efectividad de la integración de Datos Enlazados, Aprendizaje Automático, Meta-Aprendizaje y Lógica Dialéctica para la generación y explotación de conocimiento en AmI, incluso frente a información inconsistente y ambigua.

En general, esta tesis ha logrado definir, especificar y validar arquitecturas computacionales que explotan el paradigma de Datos Enlazados como eje central para la generación, gestión y explotación inteligente del conocimiento en AmI, integrando de manera novedosa y robustas técnicas de Aprendizaje Automático, Meta-Aprendizaje y Lógica Dialéctica. Se han superado limitaciones existentes en la literatura al abordar la complejidad de la información, incluyendo la inconsistencia y la ambigüedad, al automatizar tareas que tradicionalmente requieren intervención humana.

A pesar de la demostrada aplicabilidad y robustez de las arquitecturas propuestas (MiSCi-LDL, AOGS, HRS y la arquitectura de Meta-Aprendizaje) en diversos casos de estudio prácticos y simulados en escenarios complejos de AmI, la implementación y validación exhaustiva en entornos reales de AmI de gran escala y en tiempo real, junto con la optimización de su rendimiento y escalabilidad, representan un desafío exigente por evaluar. Si bien el presente trabajo ha sentado bases sólidas para la gestión inteligente del conocimiento en presencia de grandes volúmenes de datos, la naturaleza inherentemente dinámica, heterogénea y en constante evolución de AmI reales, como ciudades inteligentes, hogares conectados o sistemas de salud integrados, demanda una investigación dedicada a probar y optimizar propuestas como estas en esos contextos. No se realizó una integración completa de todas estas herramientas en un entorno real, lo cual tiene sus propias complejidades. Esto incluye el posible aprovechamiento de paradigmas como la computación en la niebla o el borde, los cuales serán fundamentales para confirmar y potenciar la robustez y adaptabilidad de las soluciones propuestas a escenarios dinámicos, asegurando su eficiencia y viabilidad a largo plazo en el espectro completo de los AmI del futuro.

6.2 Trabajos Futuros

La investigación realizada abre múltiples vías para trabajos futuros, entre las que se destacan:

- **Extensión de la Lógica Dialéctica en otros contextos:** Explorar la aplicación del razonamiento dialéctico para abordar contradicciones y ambigüedades en otros dominios, más allá del diagnóstico médico y las competencias profesionales, como la gestión de crisis, ya que la naturaleza dinámica e impredecible de las crisis hace que la capacidad de la Lógica Dialéctica para manejar la incertidumbre y las inconsistencias sea una herramienta poderosa para la toma de decisiones.
- **Optimización del rendimiento de las arquitecturas:** Investigar métodos para optimizar el rendimiento y la escalabilidad de las arquitecturas propuestas (MiSCi-LDL, AOGS, HRS y la arquitectura de Meta-Aprendizaje) en entornos AmI de gran escala y en tiempo real, lo que podría implicar el uso de computación en la niebla o el borde.
- **Desarrollo de interfaces y herramientas de usuario:** Crear interfaces más intuitivas y herramientas de visualización para que los usuarios finales puedan interactuar con los sistemas de recomendación y los generadores de ontologías, facilitando la interpretación de las decisiones y el conocimiento generado.
- **Abordaje de la explicabilidad en IA híbrida:** Profundizar en los métodos de explicabilidad de los modelos de Aprendizaje Automático y las decisiones del HRS, especialmente en la intersección de lógicas descriptivas/dialécticas y técnicas de Aprendizaje Automático, para explicar de forma integral y coherente el razonamiento combinado y las decisiones que emergen de esta compleja interacción de paradigmas híbridos, facilitando que los usuarios comprendan la razón de las recomendaciones.
- **Casos de estudio en entornos reales:** Implementar y validar las arquitecturas propuestas en casos de estudio más complejos y realistas dentro de AmI, por ejemplo, en sistemas de gestión de tráficos o emergencias urbanas, sistemas de asistencia a personas discapacitadas o mayores, o sistemas de salud conectados, lo que permitirá evaluar su robustez y adaptabilidad a escenarios dinámicos.

7 Referencias Bibliográficas

[1] Dos Santos, R., Rangel, C., Rodríguez, T. (2018). Minería de la Web Semántica. En: Aguilar, J. (ed), Introducción a la Minería Semántica. Fondo Editorial Universidad Nacional Experimental del Táchira (FEUNET), San Cristóbal, Venezuela, pp. 101-116.

[2] Aguilar, J., Rodriguez, T. (2018). Generalidades de la Minería Semántica. En: Aguilar, J. (ed), Introducción a la Minería Semántica. Fondo Editorial Universidad Nacional Experimental del Táchira (FEUNET), San Cristóbal, Venezuela, pp. 13-24.

[3] Dos Santos, R., Aguilar, J. (2018). Enlazado de Datos. En: Aguilar, J. (ed), Introducción a la Minería Semántica. Fondo Editorial Universidad Nacional Experimental del Táchira (FEUNET), San Cristóbal, Venezuela, 177-216.

[4] Bizer, C., Heath, T., & Berners-Lee, T. (2009). Linked datathe story so far. Semantic services, interoperability and web applications: emerging concepts (2009), 205–227.

[5] Augusto, J. C., Nakashima, H., & Aghajan, H. (2010). Ambient intelligence and smart environments: A state of the art. Handbook of ambient intelligence and smart environments, 3-31. https://doi.org/10.1007/978-0-387-93808-0_1

[6] Cordero, J., Dos Santos, R. (2020). Reconocimiento de emociones desde un enfoque multimodal. En: Aguilar, J. (ed), Introducción a la Computación Afectiva. Fondo Editorial Universidad Nacional Experimental del Táchira (FEUNET), San Cristóbal, Venezuela, (pp 101-116).

Paz López, A. (2015). HI3: una aproximación integrada a la construcción de sistemas de inteligencia ambiental, Tesis Doctoral, Universidade da Coruña, (pp 15), 2015.

[8] Flach, P. (2012). Machine learning: the art and science of algorithms that make sense of data. Cambridge university press.

[9] Dos Santos, R., Aguilar, J., & Puerto, E. (2021, October). A meta-learning architecture based on linked data. In 2021 XLVII Latin American Computing Conference (CLEI) (pp. 1-10). IEEE. https://doi.org/10.1109/CLEI53233.2021.9640223

[10] Pelletier, F. J., Sutcliffe, G., & Hazen, A. P. “Automated Reasoning for the Dialetheic Logic RM3”. In 30th International Florida Artificial Intelligence Research Society Conference, FLAIRS 2017. AAAI Press. 2017.

[11] Architecture Working Group of the Software Engineering Committee. (2000). Recommended Practice for Architectural Description of Software Intensive Systems. IEEE Standards Department. https://doi.org/10.1109/IEEESTD.2000.91944

[12] Bernasconi, E., Ceriani, M., Di Pierro, D., Ferilli, S., & Redavid, D. (2023). Linked data interfaces: a survey. Information, 14(9), 483. https://doi.org/10.3390/info14090483

[13] Pedro, A., Pham-Hang, A. T., Nguyen, P. T., & Pham, H. C. (2022). Data-driven construction safety information sharing system based on linked data, ontologies, and knowledge graph technologies. International journal of environmental research and public health, 19(2), 794. https://doi.org/10.3390/ijerph19020794

[14] Thalhath, N., Nagamori, M., & Sakaguchi, T. (2023, March). Application Profile Driven Data Acquisition for Knowledge Graph and Linked Data Generation in Crowdsourced Data Journalism. In Proceedings of the International Conference on Dublin Core and Metadata Applications. Dublin Core Metadata Initiative. https://doi.org/10.23106/dcmi.953151686

[15] Yang, P., Bi, G., Qi, J., Wang, X., Yang, Y., & Xu, L. (2025). Multimodal wearable intelligence for dementia care in healthcare 4.0: A survey. Information Systems Frontiers, 27(1), 197-214. https://doi.org/10.1007/s10796-021-10163-3

[16] Favarato, G., Clemens, T., Cunningham, S., Dibben, C., Macfarlane, A., Milojevic, A., ... & Hardelid, P. (2021). Air Pollution, housing and respiratory tract Infections in Children: NatIonal birth Cohort study (PICNIC): study protocol. BMJ open, 11(5), e048038. https://doi.org/10.1136/bmjopen-2020-048038

[17] Spieldenner, D., Antakli, A., Spieldenner, T., & Sahota, H. Semantic Support Points for on the Fly Knowledge Encoding in Heterogenous Systems. https://doi.org/10.5220/0012191900003838

[18] Bühmann, L., Lehmann, J., Westphal, P., & Bin, S. (2018, April). Dl-learner structured machine learning on semantic web data. In Companion Proceedings of the International World Wide Web Conferences Steering Committee 2018 (pp. 467-471). https://doi.org/10.1145/3184558.3186235

[19] Bühmann, L., Lehmann, J., & Westphal, P. (2016). DL-Learner—A framework for inductive learning on the Semantic Web. Journal of Web Semantics: Science, Services and Agents on the World Wide Web, 39, 15-24. https://doi.org/10.1016/j.websem.2016.06.001

[20] Westphal, P., Grubenmann, T., Collarana, D., Bin, S., Bühmann, L., & Lehmann, J. (2022). Spatial concept learning and inference on geospatial polygon data. Knowledge-Based Systems, vol. 241, p. 108233. https://doi.org/10.1016/j.knosys.2022.108233

[21] Westphal, P., Vahdati, S., & Lehmann, J. (2022, February). A simulated annealing meta-heuristic for concept learning in description logics. In International Conference on Inductive Logic Programming. Cham: Springer International Publishing. p. 266-281. https://doi.org/10.1007/978-3-030-97454-1_19

[22] Koppisetti, V. S. K. (2024). Meta Learning: Harnessing AI to Optimize Machine Learning Models. ESP International Journal of Advancements in Science & Technology (ESP-IJAST) Volume, 2024, vol. 2, pp. 27-35. https://www.espjournals.org/IJAST/ijast-v2i2p106

[23] Sayed, E., Maher, M., Sedeek, O., Eldamaty, A., Kamel, A., & El Shawi, R. (2024). GizaML: A Collaborative Meta-learning Based Framework Using LLM For Automated Time-Series Forecasting. In EDBT. pp. 830-833. http://doi.org/10.48786/edbt.2024.81

[24] Rodríguez, T., Dos Santos, R., & Aguilar, J. (2017). Metodología para el desarrollo de aplicaciones Web utilizando datos enlazados. In Conferencia Nacional de Computación, Informática Y Sistemas (CoNCISa 2017) (Vol. 5, pp. 978-980).

[25] Ruth, L., Wood, D., Zaidman, M., & Hausenblas, M. (2014). Linked Data: Structured data on the Web. Simon and Schuster. Manning Publications Co.

[26] Tenesaca-Luna, G., Dos Santos, R., Moreno, K. (2018). Minería de Grafos. En: Aguilar, J. (ed), Introducción a la Minería Semántica. Fondo Editorial Universidad Nacional Experimental del Táchira (FEUNET), San Cristóbal, Venezuela, pp. 150-176.

[27] Markman, A. B. (2013). Knowledge representation. Psychology Press.

[28] Gruber, T. (2008). Ontology. Entry in the Encyclopedia of Database Systems.

[29] da Costa, T. I. A. (2013). Publishing Relational Data as Linked Data (Master's thesis, Universidade do Porto (Portugal)).

[30] Watson, M. (2010). Practical Semantic Web and Linked Data Applications. Mark Watson.

[31] Allemang, D., & Hendler, J. (2011). Semantic web for the working ontologist: effective modeling in RDFS and OWL. Elsevier.

[32] Bench-Capon, T. J. (2014). Knowledge representation: An approach to artificial intelligence (Vol. 32). Elsevier.

[33] El Naqa, I., & Murphy, M. J. (2015). What is machine learning?. In machine learning in radiation oncology. Springer, Cham. pp. 3-11, 2015.

[34] Dey, A. (2016). Machine learning algorithms: a review. International Journal of Computer Science and Information Technologies (IJCSIT), Vol, 7 no 3, pp. 1174-1179, 2016.

[35] Aref, S., J. Shortle, L. Sherry. (2024). Generating synthetic flight tracks for collision risk safety analysis: Variational autoencoders with a single seed track. In Proceedings of the Integrated Communications, Navigation, and Surveillance Conference, Herndon, VA.

[36] Hubert, N., Monnin, P., D'aquin, M., Monticolo, D., & Brun, A. (2024, May). PyGraft: Configurable Generation of Synthetic Schemas and Knowledge Graphs at Your Fingertips. In Semantic Web-21st International Conference, ESWC 2024. https://doi.org/10.5281/zenodo.10243209

[37] Dos Santos, R., & Aguilar, J. (2024). A synthetic data generation system based on the variational-autoencoder technique and the linked data paradigm. Progress in Artificial Intelligence, 13(2), 149-163. https://doi.org/10.1007/s13748-024-00328-x

[38] Dos Santos, R., Aguilar, J., & R-Moreno, M. D. (2022, October). A synthetic Data Generator for Smart Grids based on the Variational-Autoencoder Technique and Linked Data Paradigm. In 2022 XVLIII Latin American Computer Conference (CLEI) (pp. 1-7). IEEE. https://doi.org/10.1109/CLEI56649.2022.9959918

[39] Krichen, M. (2023). Convolutional neural networks: A survey. Computers, vol. 12, no 8, p. 151. https://doi.org/10.3390/computers12080151

[40] Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. Behavioral and brain sciences, 40. https://doi.org/10.1017/S0140525X16001837

[41] Yao, H., Wu, X., Tao, Z., Li, Y., Ding, B., Li, R., & Li, Z. (2020). Automated relational meta-learning. In International Conference on Learning Representations (ICLR 2020). https://doi.org/10.48550/arXiv.2001.00745

[42] Hutter, F., Kotthoff, L., & Vanschoren, J. (2019). Automated machine learning: methods, systems, challenges (p. 219). Springer Nature. https://doi.org/10.1007/978-3-030-05318-5

[43] Velarde, J. (1977). LA LOGICA DIALECTICA (1). Teorema: Revista internacional de filosofía, 7(2), 129-140.

[44] Pulcini, G., & Varzi, A. C. (2018). Paraconsistency in classical logic. Synthese, 195(12), 5485-5496.

[45] Kamide, N., & Wansing, H. (2015). Proof theory of N4-related paraconsistent logics (Vol. 54). London: College Publications.

[46] Dos Santos, R., Aguilar, J., & Rodríguez, T. (2019). Una Revisión de la Literatura sobre Datos Enlazados. Revista Ingeniería al Día, 5(1), 54-82.

[47] Aguilar, J., Jerez, M., Mendonca, M., & Sánchez, M. (2016). MiSCi: autonomic reflective middleware for smart cities. In Technologies and Innovation: Second International Conference, CITI 2016, Guayaquil, Ecuador, November 23-25, 2016, Proceedings 2 (pp. 241-253). Springer International Publishing. https://doi.org/10.1007/978-3-319-48024-4

[48] Aguilar, J., Sanchez, M. B., Jerez, M., Mendonca, M. (2019). An Extension of the MiSCi Middleware for Smart Cities Based on Fog Computing Smart Cities and Smart Spaces: Concepts, Methodologies, Tools, and Applications (pp. 778-798). Hershey. https://doi.org/10.4018/978-1-5225-7030-1.ch035

[49] Aguilar, J., Jerez, M., Mendonça, M., & Sánchez, M. (2020). Performance analysis of the ubiquitous and emergent properties of an autonomic reflective middleware for smart cities. Computing, 102(10), 2199-2228. https://doi.org/10.1007/s00607-020-00799-5

[50] Dos Santos, R., Aguilar, J., Rodríguez, T. (2020). Middleware MiSCi para Ciudades Inteligentes extendido con Datos Enlazados/MiSCi Middleware for Smart Cities extended with Linked Data. Dyna, 87(214), 229. https://doi.org/10.15446/dyna.v87n214.83226

[51] Vizcarrondo, J., Aguilar, J., Exposito, E., & Subias, A. (2017). MAPE-K as a service-oriented architecture. IEEE Latin America Transactions, 15(6), 1163-1175. https://doi.org/10.1109/TLA.2017.7932705

[52] Aguilar, J., Cordero, J., & Buendía, O. (2018). Specification of the autonomic cycles of learning analytic tasks for a smart classroom. Journal of Educational Computing Research, 56(6), 866-891. https://doi.org/10.1177/0735633117727698

[53] Aguilar, J., Cerrada, M., & Hidrobo, F. (2007). A methodology to specify multiagent systems. In Agent and Multi-Agent Systems: Technologies and Applications: First KES International Symposium, KES-AMSTA 2007, Wroclaw, Poland, May 31–June 1, 2007. Proceedings 1 (pp. 92-101). Springer Berlin Heidelberg. http://10.1007/978-3-540-72830-6_10

[54] Dos Santos, R., Puerto, E., & Aguilar, J. (2023, October). Automated Ontology Generator System Based on Linked Data. In 2023 XLIX Latin American Computer Conference (CLEI) (pp. 1-10). IEEE. https://doi.org/10.1109/CLEI60451.2023.10346110

[55] Dos Santos, R, Puerto, E. & Aguilar, J. (2022). Arquitectura para la Creación y Enriquecimiento Automático de Ontologías a partir de Datos Enlazados. Mundo FESC, 12(24), 190-200. https://doi.org/10.61799/2216-0388.1213

[56] Dos Santos, R. D., & Aguilar, J. (2023). A hybrid recommender system based on description/dialetheic logic and linked data. Expert Systems, 40(2), e13143. https://doi.org/10.1111/exsy.13143

[57] Aguilar, J., Valdiviezo-Díaz, P., & Riofrio, G. (2017). A general framework for intelligent recommender systems. Applied Computing and Informatics, 13(2), 147–160. https://doi.org/10.1016/j.aci.2016.08.002

[58] González-Eras, A., Dos Santos, R., & Aguilar, J. (2023). Evaluation of digital competence profiles using dialetheic logic. International Journal of Artificial Intelligence in Education, 33(1), 59-87. https://doi.org/10.1007/s40593-021-00286-8

[59] González-Eras, A., Dos Santos, R., & Aguilar, J. (2020). Análisis de las contradicciones en las competencias profesionales en los textos digitales usando lógica dialéctica. Revista Ibérica de Sistemas e Tecnologias de Informação, (E27), 150-163.

[60] González-Eras, A., & Aguilar, J . (2018). Esquema para la actualización de Ontologías de Competencias en base al Procesamiento del Lenguaje Natural y la Minería Semántica. Revista Ibérica de Sistemas e Tecnologias de Informação, (E17), 433-447.

[61] González-Eras, A., & Aguilar, J. (2019) Determination of professional competencies using an alignment algorithm of academic and professional profiles, based on competence thesauri and similarity measures, International Journal of Artificial Intelligence in Education. https://doi.org/10.1007/s40593-019-00185-z

[62] Pacheco, F., Rangel, C., Aguilar, J., Cerrada, M., & Altamiranda, J. (2014, September). Methodological framework for data processing based on the Data Science paradigm. In 2014 XL Latin american computing conference (CLEI). pp. 1-12. IEEE. https://doi.org/10.1109/CLEI.2014.6965184

[63] Dos Santos, R., Aguilar. (2025). An Autonomous Meta-Learning Architecture for Transfer Learning based on Linked Data.. EN REVISTA

[64] Jordon, J., Szpruch, L., Houssiau, F., Bottarelli, M., Cherubin, G., Maple, C., ... & Weller, A. (2022). Synthetic Data--what, why and how?. arXiv preprint arXiv:2205.03257. https://doi.org/10.48550/arXiv.2210.07574

[65] Raghunathan, T. E. (2021). Synthetic data. Annual review of statistics and its application, 8(1), 129-140. https://doi.org/10.1146/annurev-statistics-040720-031848

[66] Toscano-Miranda, R., Aguilar, J., Hoyos, W., Caro, M., Trebilcok, A., & Toro, M. (2024). Different transfer learning approaches for insect pest classification in cotton. Applied Soft Computing, vol. 153, pp. 111283. https://doi.org/10.1016/j.asoc.2024.111283

[67] Hosna, A., Merry, E., Gyalmo, J., Alom, Z., Aung, Z., & Azim, M. A. (2022). Transfer learning: a friendly introduction. Journal of Big Data, 9(1), 102. https://doi.org/10.1186/s40537-022-00652-w

[68] Agarwal, N., Sondhi, A., Chopra, K., & Singh, G. (2021). Transfer learning: Survey and classification. Smart Innovations in Communication and Computational Sciences: Proceedings of ICSICCS 2020, 145-155. https://doi.org/10.1007/978-981-15-5345-5_13

[69] Dos Santos, R., Aguilar. (2025). An Explainable Feature Generation Approach for Classification Models Using CNNs. EN REVISTA

[70] Zhang, T., Zhang, Z. A., Fan, Z., Luo, H., Liu, F., Liu, Q., ... & Jian, L. (2023, July). OpenFE: automated feature generation with expert-level performance. In International Conference on Machine Learning (pp. 41880-41901). PMLR. https://doi.org/10.48550/arXiv.2211.12507

[71] Sarhan, M., Layeghy, S., Moustafa, N., Gallagher, M., & Portmann, M. (2024). Feature extraction for machine learning-based intrusion detection in IoT networks. Digital Communications and Networks, 10(1), 205-216. https://doi.org/10.1016/j.dcan.2022.08.012

[72] Dhal, P., & Azad, C. (2022). A comprehensive survey on feature selection in the various fields of machine learning. Applied Intelligence, 52(4), 4543-4581. https://doi.org/10.1007/s10489-021-02550-9

[73] Castillo-Cara, M., Talla-Chumpitaz, R., García-Castro, R., & Orozco-Barbosa, L. (2023). TINTO: Converting Tidy Data into image for classification with 2-Dimensional Convolutional Neural Networks. SoftwareX, vol. 22, p. 101391. https://doi.org/10.1016/j.softx.2023.101391

[74] Selvaraju, R. R., Das, A., Vedantam, R., Cogswell, M., Parikh, D., & Batra, D. (2016). Grad-CAM: Gradient-weighted Class Activation Mapping. Arxiv. https://doi.org/10.48550/arXiv.1610.02391

[75] Wang, H., Wang, Z., Du, M., Yang, F., Zhang, Z., Ding, S., ... & Hu, X. (2020). Score-CAM: Score-weighted visual explanations for convolutional neural networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, p. 24-25. https://doi.org/10.48550/arXiv.1910.01279

[76] Jiang, P. T., Zhang, C. B., Hou, Q., Cheng, M. M., & Wei, Y. (2021). Layercam: Exploring hierarchical class activation maps for localization. IEEE Transactions on Image Processing, vol. 30, p. 5875-5888. https://doi.org/10.1109/TIP.2021.3089943

[77] Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., & Viegas, F. (2018, July). Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In International conference on machine learning. PMLR, p. 2668-2677. https://doi.org/10.48550/arXiv.1711.11279

[78] Fuentes, J., Dos Santos, R., Aguilar. Shi, Donghui. (2025). Meta-learning Architecture for ACODAT in the Context of Agro-Industrial Production Chains of MSMEs.. EN REVISTA

8 Anexos

8.1 Anexo 2.A: Tecnologías de los Datos Enlazados

Las tecnologías existentes son de suma importancia para implementar los Datos Enlazados, entre los elementos a considerar están los *formatos de almacenamiento*, los *lenguajes de consulta y las herramientas de publicación*.

Los *formatos de almacenamiento* son muy variados, para tener una idea aproximada de la variedad de formatos, si se carga la URI que representa a Venezuela en DBpedia ***"http://dbpedia.org/page/Venezuela"*** en el navegador, en la parte superior se observa un menú que permite seleccionar el formato en el que se desea obtener esta información. Entre los formatos de intercambio que se muestran están los siguientes: RDF (N-Triples, N3/Turtle y XML), ODATA (Atom, JSON), Microdata (JSON, HTML), Embebidos (JSON, Turtle), CXML, CSV y JSON-LD.

El formato más usado en los Datos Enlazados es **N3/Turtle** (usado en los ejemplos mostrados), por su sencilla lectura y porque su serialización RDF es similar al lenguaje de consulta SPARQL. **N-Triples**[1112131415161718] es un subconjunto de **Turtle**[19] y **N3**[20], fue diseñado para ser un formato más simple y fácil de analizar e interpretar por los programas, sin embargo, carece de algunos de los accesos directos proporcionados por RDF, haciéndolo tedioso para escribir a mano y leer grandes cantidades de datos.. El formato **RDF/XML**[21] es la primera notación usada y estandarizada por el W3C, basado en el formato de intercambio XML, su ventaja inicial radica en el soporte a los lenguajes de programación con el formato XML, pero presenta un grave problema para su lectura por los humanos, ya que su codificación es bastante compleja. Otro formato que ha tenido gran aceptación en los Datos Enlazados es el **JSON-LD (Javascript Object Notation for Linked Data)**[22], porque está basado en el formato de intercambio JSON, y uno de sus objetivos es ayudar y facilitar la transformación de datos en JSON a JSON-LD, ya que existen muchos servicios web que proveen información en JSON.

---

11 https://www.w3.org/TR/microdata/

12 https://es.wikipedia.org/wiki/SQL

13 https://www.ics.forth.gr/isl/RDF/RQL/

14 https://www.w3.org/Submission/RDQL/

15 http://jena.apache.org/documentation/fuseki2/index.html

16 http://jena.apache.org

17 http://rdf4j.org

18 https://virtuoso.openlinksw.com

19 http://live.dbpedia.org

20 Formato CONLL, donde NC: sustantivo, SP; preposición y AQ: adjetivo.

21 **Lógicas no aristotélicas:** Que no cumple los principios de la lógica aristotélica de identidad, no contradicción y del tercero excluido [69].

22

Las principales críticas hacia los formatos anteriormente comentados, radican en que gran cantidad de información ya está disponible en las páginas HTML, y la duplicación en un formato diferente es tanto una inversión inicial significativa como una molestia para su mantenimiento. Para solucionar estos problemas, se considera integrar etiquetas semánticas o anotaciones semánticas en las webs existentes, a través de pequeños cambios en la información y los hipervínculos, haciendo explícito el significado de la información a las aplicaciones de Datos Enlazados o buscadores. Entre estos tipos de formatos que proveen capacidades embebidas se cuenta con: Microformatos, RDFa y Microdatos.

Los **Microformatos**[23] se constituyen en la primera iniciativa de agregar información extra al código HTML, aprovechando que los intérpretes de este código ignoran cualquier etiqueta desconocida a sus especificaciones. Es allí donde se utilizan los atributos y propiedades, para permitir identificar eventos, información de contacto, relaciones sociales, direcciones, ubicaciones (coordenadas), etc. El **RDFa** [24] al igual que los microformatos, funciona agregando atributos a las etiquetas; sin embargo, este permite definir espacio de nombres de la misma manera que en RDF/XML. Gracias a esto, los escritores no están restringidos a solo los vocabularios oficiales, por lo que pueden definir sus propios vocabularios. Por último, los **Microdatos**[25], o como los llama Google, **Rich Snippets**, son una especificación de HTML5 que ayuda a las aplicaciones a entender el contenido que hay en una página. Esto se logra con las propiedades añadidas a las etiquetas HTML5, que ofrecen diferentes esquemas según el tipo de contenido del que se trate, y los buscadores lo usan para extraer información y facilitar la indexación de los datos, ya que permiten contextualizar la información contenida en las páginas web.. A continuación se presenta un ejemplo de Venezuela en Microformatos:

```
<span class="country">

<a class="url" href="http://dbpedia.org/resource/Venezuela">
        <span class="country-name" title="País">Venezuela</span>
</a>
</span>
```

Los *lenguajes de consultas* son fundamentales en el proceso para encontrar la información que se modeló, concentrándose en los datos de manera abstracta, ignorando el formato (N-Triples, N3, RDF/XML, etc.) en que fueron almacenadas originalmente las tripletas. Los almacenes o repositorios donde se guardan las tripletas, también se les denominan bases de datos semánticas. En las bases de datos relacionales se usa un lenguaje de consulta, como el SQL[26], para obtener información contenida en dichas bases de datos. De igual manera, las bases de datos semánticas necesitan de un estándar que permita encontrar los datos o tripletas. Existen muchos lenguajes de consulta RDF, como por ejemplo, RQL[27], RDQL[28], SPARQL, entre otros, y en la gran mayoría se modelan las consultas de manera

23 http://purl.org/dc/terms/ https://www.w3.org/TR/n-triples/

24 https://www.w3.org/TR/turtle/

25 https://www.w3.org/TeamSubmission/n3/

26 https://www.w3.org/TR/rdf-syntax-grammar/

27 https://www.w3.org/TR/json-ld/

28 http://microformats.org/

semejante al SQL. El lenguaje de consulta usado en los Datos Enlazados es el SPARQL, ya que sus consultas son muy sencillas y potentes, y permite a las organizaciones que se encargan de generar estos datos, una forma fácil de compartir y acceder a los datos, ya sea en repositorios locales como remotos. Un ejemplo de este tipo de consulta es mostrado a continuación:

```
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX dbpedia-c: <http://es-la.dbpedia.org/resource/Categoría:>
SELECT ?Personas ?Resumen
WHERE {
   ?URIPersonas dcterms:subject dbpedia:Científicos_de_Venezuela .
   ?URIPersonas dbpedia-owl:abstract ?Resumen .
   ?URIPersonas rdfs:label ?Personas .
}
```

En la palabra **SELECT** se escriben las variables que se desean obtener, también se podría usar el símbolo de **asterisco (*)** y se obtendrían todas las variables definidas en la tripleta que se está consultando. Luego se usa la palabra **WHERE**, para indicar de dónde se recogerá la información. En esta parte hay que pensar en tripletas. La palabra **PREFIX** es solo para abreviar las direcciones de cada URI usada.

Las *herramientas de publicación* son las que permiten almacenar y compartir los Datos Enlazados. Estas herramientas también se les denominan motores de consultas. Normalmente, estas herramientas implementan SPARQL endpoint, que es un servicio web que permite realizar consultas, por medio del protocolo SPARQL (capa que funciona sobre HTTP), en lenguaje SPARQL, sobre un grafo compuesto por tripletas RDF [25]. Entre los motores más comunes se encuentran los siguientes: D2RQ, Apache Jena Fuseki, RDF4J (Sesame), OpenLink Virtuoso, AllegroGraph, RDFStore, Ontotext GraphDB, entre otros. A continuación se detallan algunos de estos motores [16]:

- *Apache Jena Fuseki*[29]*:* es un servidor SPARQL implementado en Java, que puede funcionar como un servicio del sistema operativo, como una aplicación web (archivo WAR), o como un servidor independiente. Forma parte del conjunto de herramientas ofrecidas por ***Apache Jena***[30] para el desarrollo de aplicaciones para Web Semántica y Datos Enlazados. Entre el conjunto de herramientas ofrecidas por Apache Jena están: una API para extraer e insertar datos en los grafos RDF, y un soporte para ontologías especificadas en OWL, con diversos razonadores como Pellet, Racer y FaCT++.
- *Eclipse RDF4J*[31]*:* es el sucesor del anteriormente conocido proyecto ***OpenRDF Sesame***, que es un poderoso framework en Java para el procesamiento de datos RDF, que incluye la creación, análisis, almacenamiento, inferencia y consulta sobre dichos Datos Enlazados. Ofrece una API que permite conectar las distintas soluciones líderes en almacenamiento de RDF, y pone a disposición los datos con SPARQL endpoint, para así lograr crear aplicaciones que aprovechen el poder de los Datos Enlazados y web semántica.

---

29 https://www.w3.org/TR/rdfa-syntax/

30 https://www.w3.org/TR/microdata/

31 https://es.wikipedia.org/wiki/SQL

- *OpenLink Virtuoso*[32]*:* es una de las soluciones más completas y moderna para el acceso, integración y gestión datos. La arquitectura de OpenLink Virtuoso es un híbrido de Servidor de Aplicaciones Web y Sistema de Gestión de Datos, que permite la persistencia de diferentes tipos datos: Base de Datos Relacionales (Oracle, SQL Server, MySql, PostgreSql, DB2, Sybase, CA-Ingres, Informix, etc), RDF, XML, texto, documentos Web (con o sin microformatos embebidos), Datos Enlazados, datos provenientes de Web Services o Web APIs, entre otros. Todo lo anterior, lo logra gracias a su potente característica principal, el componente Sponger, que permite transformar datos no RDF a RDF en tiempo de ejecución. Sponger usa unos pequeños paquetes llamados cartridge, que indican como hacer el mapeo de grupos de datos no RDF a RDF, e indican todas las configuraciones de seguridad y acceso a las fuentes de donde se van a realizar las extracciones. En OpenLink Virtuoso ya vienen integrados una gran cantidad de cartridges, pero también ofrece al usuario la posibilidad de construirlos para su formato de datos específico.

32

8.2 Anexo 3.A: Middleware MiSCi para Ciudades Inteligentes extendido con Datos Enlazados

UNIVERSIDAD NACIONAL DE COLOMBIA
SEDE MEDELLÍN
FACULTAD DE MINAS

DYNA
http://dyna.medellin.unal.edu.co/

MiSCi Middleware for smart cities extended with linked data

Ricardo Dos Santos [a,c], Jose Aguilar-Castro [a,b,c] & Tatiana Rodríguez [a,c]

[a] CEMISID, Facultad de Ingeniería, Universidad de Los Andes, Mérida, Venezuela. ricardojdsg@gmail.com, aguilar@ula.ve, tatiana@ula.ve
[b] CIDITIC, Universidad EAFIT, Medellín, Colombia
[c] Tepuy R+D Group. Artificial Intelligence Software Development. Mérida, Venezuela.

Received: October 29th, 2019. Received in revised form: April 3rd, 2020. Accepted: April 21th, 2020.

Abstract
This article proposes an extension of the capabilities of the MiSCi middleware, by adding a new layer called Linked Data, to identify, describe, connect, relate and exploit the various user-generated data and applications of the smart city using the Linked Data paradigm. This new layer is composed of different agents that automate the specification, modeling, generation, linking, publication and exploitation of data based on MEDAWEDE. Such agents can enrich existing MiSCi ontologies, generate knowledge models required by MiSCi services, generate data to build knowledge models for MiSCi, and recommend information in contexts of uncertainty through hybrid inference based on descriptive/dialectical logic. In addition, this work specifies a case study, where the MiSCi's capabilities to handle different critical situations are shown, supported by the new data link layer.

Keywords: agents; linked data; ontology; semantic enrichment.

Middleware MiSCi para ciudades inteligentes extendido con datos enlazados

Resumen
Este artículo propone una ampliación de las capacidades del middleware MiSCi, al agregar una nueva capa denominada Datos Enlazados, para identificar, describir, conectar, relacionar y explotar los distintos datos generados por los usuarios y las aplicaciones de la ciudad inteligente usando el paradigma de datos enlazados. Esta nueva capa está compuestas por distintos agentes que permiten automatizar las etapas de especificación, modelado, generación, vinculación, publicación y explotación de los datos basados en MEDAWEDE. Dichos agentes pueden enriquecer ontologías existentes en MiSCi, generar modelos de conocimiento requeridos por los servicios de MiSCi, generar datos para construir modelos de conocimiento para MiSCi, y recomendar información en contextos de incertidumbre a través de una inferencia híbrida basada en lógica descriptiva/dialéctica. Además en este trabajo se especifica un caso de estudio, donde se muestran las capacidades del MiSCi para manejar distintas situaciones críticas, apoyado en la nueva capa de enlazado de datos.

Palabras clave: agentes; datos enlazados; ontología; enriquecimiento semántico.

1. Introduction

En [1] se propone MiSCi (***Middleware for Smart Cities***), un middleware reflexivo autonómico para Ciudades Inteligentes (*Smart City* - SC), el cual soporta la comunicación entre los diferentes elementos de una ciudad inteligente, y el despliegue de las aplicaciones de base de toda ciudad inteligente, para la gestión inteligente del sistema de transporte, energético, entre otros. MiSCi permite ver a los elementos de la ciudad inteligente como agentes, y a las tareas que realizan ellos, y en general, a las aplicaciones de gestión de la ciudad, como servicios. MiSCi es una arquitectura multicapas basado en los conceptos de Sistemas Multi-Agentes (MAS, por sus siglas en inglés), Conciencia de Contexto (*Context-Awareness*), Emergencia Ontológica y Computación en la Nube. Luego, en [2] es ampliada MiSCi al incluir el concepto de Computación en la Niebla, que permite la capacidad de procesar los datos y las aplicaciones

How to cite: Dos Santos-Guillén, R.J, Aguilar-Castro, J.L. and Rodríguez-de Paredes, T.J., Middleware MiSCi para Ciudades Inteligentes extendido con Datos Enlazados. DYNA, 87(213), pp. 229-238, July-September, 2020.

como servicios en los dispositivos al borde de la red, en lugar de hacerlo en la nube.

En las ciudades inteligentes, el gran reto tecnológico es saber recolectar, procesar y aprovechar el gran volumen de datos que la ciudad tiene, en tiempo real [3]. Además, todos estos datos recolectados necesitan ser enriquecidos con información semántica que ayuden al descubrimiento de conocimiento oculto, para mejorar los procesos de toma de decisiones. En este sentido, los datos enlazados resultan factibles para resolver dicho reto.

Los datos enlazados o vinculados (en inglés, *Linked Data*) describen una forma de publicar los datos estructurados en la Web para que se puedan interconectar entre ellos [4], es decir, permiten identificar, describir, conectar y relacionar los distintos recursos en la Web.

En esta investigación, se amplía el trabajo propuesto en [1,2], al agregar una nueva capa, denominada Datos Enlazados, que aumenta las capacidades del middleware MiSCi, para enriquecer los datos recogidos a través de los distintos mecanismos del middleware con información externa (Redes Sociales, Páginas Web, entre otros).

Algunas similares investigaciones sobre framework o middleware, basadas en el paradigma de datos enlazados, son los siguientes: En [5] se desarrolla un framework que busca aprovechar las tecnologías de los datos enlazados para la colaboración empresarial, enriqueciendo los procesos de negocio con información de sensores ubicados en las maquinarias. En ese trabajo se aborda el problema de fuentes de datos heterogéneos, y se crea un modelo de datos semántico del entorno empresarial como una ontología de dominio, luego se transforman los datos a RDF, y se publican como datos enlazados en un servidor. Finalmente, la aplicación cliente accede a los datos realizando operaciones de consulta y filtrado. En [6] se describe *Sense2Web platform*, una plataforma que permite publicar y acceder a los datos de la red de sensores semánticos, y los enlaza a recursos existentes en la Web. En esa investigación se especifican los sensores web con atributos espaciales, temporales, temáticos y específicos. Los atributos espaciales son la ubicación geográfica (longitud y latitud), y conceptos de alto nivel que son tomados de fuentes de datos enlazados (*GeoNames* y *LinkedGeoData*). Los atributos temporales son zona horaria, la marca de tiempo de las mediciones, el tiempo de disponibilidad de los nodos de sensores, entre otros. Los atributos temáticos son tipo de sensor, tipo de medición, atributos de operación y despliegue, enriquecidos por fuentes como *DBPedia*. Toda la información es representada en tripletas RDF, que luego son publicadas en repositorios de datos enlazados.

Otro trabajo es *IES CITIES platform* [7], que es una plataforma que explota los datos enlazados ofrecidos por los ayuntamientos, enriquecidos con datos dinámicos recopilados a través de redes de sensores en toda la ciudad, o de aplicaciones que funcionan en los teléfonos de los ciudadanos. Lo interesante de ese trabajo es la manera como se enriquecen los datos enlazados con las distintas fuentes de información, por parte de los ciudadanos, ya que cada uno de ellos va publicando fotos y textos de la zona donde está ubicado el dispositivo. También, *Linked Sensor Middleware* (LSM) [8] es un middleware conformado por capas, una de adquisición de datos desde los sensores, otra de gestión de datos enlazados, otra de acceso a los datos, y finalmente, otra para el consumo de datos por parte de las aplicaciones, para facilitar la integración de datos con otras fuentes de datos enlazados, enriqueciendo los flujos de datos desde los sensores con descripciones semánticas, para su explotación a través de las aplicaciones.

La principal diferencia de los trabajos descritos anteriormente con respecto a esta investigación, es que son trabajos enfocados a resolver un problema específico usando datos enlazados (por ejemplo, gestión de sensores ambientales, enriquecimiento de datos empresariales, etc.); en cambio, esta investigación aprovecha las ventajas que ofrece el middleware MiSCi, que permite la ubicuidad, la conciencia del contexto, la emergencia ontológica, la computación en la nube y en la neblina, entre otras cosas, en la gestión de las aplicaciones de una ciudad, y lo extiende con una capa de gestión de Datos Enlazados, para enriquecerlo, darle acceso y permitirle explotar una cantidad de conocimiento aún mayor. De esta manera, MiSCi puede darle soporte de una manera más eficiente, escalable y robusta, a las aplicaciones inteligentes, basadas en el contexto de una ciudad inteligente.

Este artículo está organizado de la siguiente manera: la sección 2 presenta el marco teórico alrededor de MiSCi y los Datos Enlazados; la sección 3 describe la ampliación de MiSCi con los Datos Enlazados; la sección 4 describe los casos de estudios donde se prueba la ampliación del MiSCi; la sección 5 presenta el análisis de los resultados y se compara con otras arquitecturas; y finalmente, la sección 6 presenta las conclusiones de esta investigación.

2. Marco teórico

2.1. MiSCi

MiSCi es un middleware reflexivo multicapas que se basa en el paradigma de MAS. Los distintos elementos de este middleware aportan características esenciales a una ciudad inteligente, permitiendo la ubicuidad, conciencia del contexto, emergencia ontológica, decisiones inteligentes, computación en la nube y en la neblina, entre otras cosas. MiSCi contiene 9 capas (ver [9-14] para más detalles):

Capa de Gestión MAS (MAS Management Layer - MMAL): Esta capa es una adaptación del estándar FIPA [9,10], y define las reglas que permiten a una sociedad de agentes coexistir. Sus agentes son [9]: AMA (*Agents Manager*), CCA (*Communication Control Agent*) y DMA (*Data Management Agent*).

Capa de Gestión de Servicios (Service Management Layer -SML): Posibilita la integración entre los paradigmas MAS y SOA (*Service-Oriented Architecture*). Esta capa consta de los agentes [11,12]: SMA (*Services Management Agent*), WSA (*Web Service Agent*), WSOA (*Web Service Oriented Agent*), RMA (*Resource Manager Agent*) y ApMA (*Application Manager Agent*).

Capa de Conciencia de Contexto (Context-Awareness Layer - CAL): El propósito de esta capa es ofrecer servicios contextuales, para la gestión de la información sobre el contexto [1]. Esta información se gestiona en un ciclo conformado por: el descubrimiento y modelización del contexto, el razonamiento basado en el contexto, y la distribución del mismo. Esta capa está basada en [13].

Capa de Emergencia Ontológica (Ontological Emergence Layer - OEL): Esta capa proporciona un conjunto de servicios para el manejo de ontologías. Estos servicios han sido propuestos en [14], los cuales permiten la actualización automática de ontologías, entre otras cosas.

Capa de Gestión Lógica de SC (SC Logical Management Layer - SCLML): Esta capa es donde se encuentran todas las aplicaciones (software, objetos virtuales, etc.) y las personas presentes en la Ciudad. Básicamente, cada elemento/persona se caracteriza por un agente (es una abstracción de él). Esta capa contiene agentes como CzA (*Citizen Agent*, caracteriza a cada ciudadano en la ciudad) y AppA (*Application Agent*, caracteriza aplicaciones necesarias para una ciudad inteligente, como el Sistema de Movilidad Inteligente, el Sistema de Salud Inteligente, entre otros).

Capa de Gestión Física de SC (SC Physical Management Layer - SCPML): En esta capa se caracterizan todos los elementos físicos del entorno, permitiendo la interacción entre agentes y dispositivos de MiSCi. Así, cada dispositivo físico se caracteriza por un agente DA (*Device Agent*, es una abstracción de él). Esta capa se comunica con el dispositivo físico real que está en la capa SCPL.

Capa Lógica de SC (Smart City Logical Layer - SCLL): es la capa donde se despliegan los principales sistemas de una ciudad inteligente, tales como: Sistema Inteligente Vehicular, responsable del control del tráfico; Sistema de Salud Inteligente, encargado de facilitar el acceso a los servicios de salud, entre otros. Los agentes de MiSCi pueden comunicarse con estos sistemas a través de los agentes de AppA.

Capa Física de SC (SC Physical Layer - SCPL): Es en esta capa donde se despliegan todos los componentes físicos de la ciudad inteligente, tales como: a) Sensores, b) Efectores, c) Objetos Inteligentes.

Capa de Gestión en la Niebla (Fog Layer - FL): Esta capa permite el paradigma de computación en la niebla en MiSCi. Los agentes de esta capa ayudan a decidir si los datos serán procesados localmente o en la nube, siendo FogA (*Fog Agent*) el responsable de esta tarea. FogA utiliza una meta-ontología proporcionada por la capa de OEL, información contextual proporcionada por la capa CAL, e información del sistema recopilada por SmonA (*System Monitor Agent*), sobre el nivel de ocupación en términos de procesamiento y comunicación (ancho de banda) de los agentes y servicios web locales, para decidir si los datos deben ser procesados localmente o en la nube (ver [24] para más detalles de esta capa).

2.2. Datos Enlazados

Los datos enlazados o vinculados describen una forma de publicar los datos en Internet para que se puedan interconectar entre ellos [4]. Particularmente, los datos enlazados es la manera que tiene la Web Semántica de enlazar un conjunto de datos que estén publicados en la Internet, para mejorar la comprensión de sus significados, tanto para los humanos como para las máquinas [15,16]. En específico, este concepto hace referencia a que los datos están enlazados mediante tecnologías de la Web Semántica, lo que permite que sean conectados y consultados datos desde diferentes fuentes, para agregar más semántica a los datos. Tim Berners-Lee introduce los principios de los datos enlazados, los cuales son [17,18]: i) ***Identidad:*** Utilizar URIs (*Uniform Resource Identifier*) para identificar los recursos; ii) ***Accesibilidad:*** Usar URIs HTTP (*Hypertext Transfer Protocol*) para que las personas puedan buscar recursos; iii) ***Estructura:*** Utilizar estándares RDF para describir recursos, y SPARQL (*Query Language for RDF*) para realizar consultas; iv) ***Navegación:*** Incluir enlaces a otras URIs para descubrir más recursos.

Por otro lado, la implementación de datos enlazados implica un proceso dinámico complejo con diferentes etapas [18]. En [4] se propone una metodología, llamada **M**etodología para el **D**esarrollo de Aplicaciones **W**eb utilizando **D**atos **E**nlazados (MEDAWEDE), que permite desarrollar servicios basados en datos enlazados, compuesta por seis etapas divididas en dos grandes tareas. *La primera tarea* tiene la finalidad del enriquecimiento de los datos y su transformación a datos enlazados, y está integrada por las siguientes etapas: i) ***Especificación:*** se centra en el análisis de fuentes de datos, en esta fase se selecciona el conjunto de datos; ii) ***Modelado:*** se centra en la creación del modelo que describe el conocimiento del área de estudio, para ello se utilizan vocabularios estándares, se reutilizan ontologías, e incluso, se diseñan ontologías propias, iii) ***Generación:*** se centra en la transformación de los datos al lenguaje RDF; iv) ***Vinculación:*** en esta fase se vinculan los datos con otros conjuntos de datos para aumentar su valor, visibilidad y calidad, v) ***Publicación:*** se pone a disposición los datos en repositorios de tripletas. *La segunda tarea* tiene como objetivo la explotación de los datos enlazados, y la integra la siguiente etapa: i) ***Explotación:*** esta etapa permite el manejo e integración de distintas interfaces o aplicaciones, para consumir los recursos publicados de manera agradable y sencilla.

3. Extensión de MiSCi con datos enlazados

3.1. Datos enlazados en MiSCi

Los datos enlazados responden a varias necesidades en las ciudades inteligentes: la primera es para interpretar grandes cantidades de datos que provienen de distintas fuentes como: sensores, efectores, entre otros, donde muchos de estos datos son manejados en tiempo real. La segunda es para enriquecer los datos con información semántica, proveniente de MiSCi o de fuentes externas. En esta investigación se amplía el trabajo propuesto en [1,2,25], al agregar una nueva capa denominada Datos Enlazados (*Linked Data Layer* - LDL), que aumenta las capacidades del middleware MiSCi, para

identificar, describir, conectar, relacionar y explotar los distintos datos generados por los usuarios y las aplicaciones de la ciudad inteligente usando el paradigma de datos enlazados (ver Fig. 1).

Los distintos agentes de la capa LDL automatizan las etapas de especificación, modelado, generación, vinculación, publicación y explotación de los datos que aparecen en la metodología MEDAWEDE, en el middleware MiSCi. En esta capa se llevan a cabo dos procesos, según MEDAWEDE: i) ***Enriquecimiento:*** Este proceso realiza las etapas de especificación, modelado y generación de los datos de MiSCi, por parte de los agentes ILDA (*Internal Linked Data Agent*) y ELDA (*External Linked Data Agent*). Además, se realizan las tareas de vinculación y publicación de los datos de MiSCi por parte del agente LDIA (*Linked Data Integration Agent*); ii) ***Explotación:*** Este proceso realiza la etapa de explotación de los datos de MiSCi, la cual es realizada por el agente LDKA (*Linked Data Knowledge Agent*). Ahora bien, estos agentes pueden ser activados simultáneamente desde distintos procesos, según las circunstancias que lo ameriten.

El proceso de enriquecimiento semántico implica recolectar datos internos y externos al MiSCi (ver Fig. 2). La recolección de datos internos se activa cada vez que algún agente CzA, AppA o DA es actualizado, para lo cual se invoca al agente ILDA con los datos nuevos que pueden provenir simultáneamente de diferentes fuentes (ver paso 1 en Fig. 2), Estos datos son preparados y enriquecidos simultáneamente con información del Servicio Web de Contexto (*Context Web Services* - Cx WS) y del Servicio Web de Meta Ontología (*Meta Ontology Web Services* - MO WS) (ver paso 2 y 3 en Fig. 2). De la misma manera ocurre con la recolección de los datos externos, lo cual es realizado por el agente ELDA. Luego, el agente LDIA es activado con la información enriquecida generada por los agentes ILDA o ELDA, para vincular los datos previamente obtenidos con otros datos enlazados (ver paso 4 en Fig. 2), para finalmente publicarlos como datos enlazados (ver paso 5 en Fig. 2).

El proceso de explotación es activado por un agente del MiSCi cuando solicita información enlazada (ver Fig. 3). Para este tipo de consulta se debe invocar al agente LDKA de MiSCi (ver paso 1 en Fig. 3). Luego, el agente LDKA (ver paso 2 en Fig. 3) por medio de los distintos mecanismos inteligentes de explotación de conocimiento enlazado, explora simultáneamente todas las fuentes de datos enlazados, y retorna la información solicitada por el agente.

Los agentes de la capa LDL definen cuatro ciclos autonómicos para la autogestión de los Datos Enlazados en MiSCi, basado en el concepto de ciclos autonómicos como servicios propuestos en [19,20]. Cada ciclo autonómico establece la relación entre las tareas de los agentes, para la explotación de cada una de las formas de conocimiento que permite el agente LDKA. Algunas de esas tareas establecen las reglas a activar del sistema recomendador según el contexto, o identifican el modelo o el conjunto de datos pertinente a una situación dada, entre otras cosas.

3.2. Especificación de los agentes de datos enlazados

La capa de datos enlazados está conformada por 4 tipos de agentes. Para la especificación de cada agente se usa MASINA [21], y en específico, los modelos de agentes y de tareas.

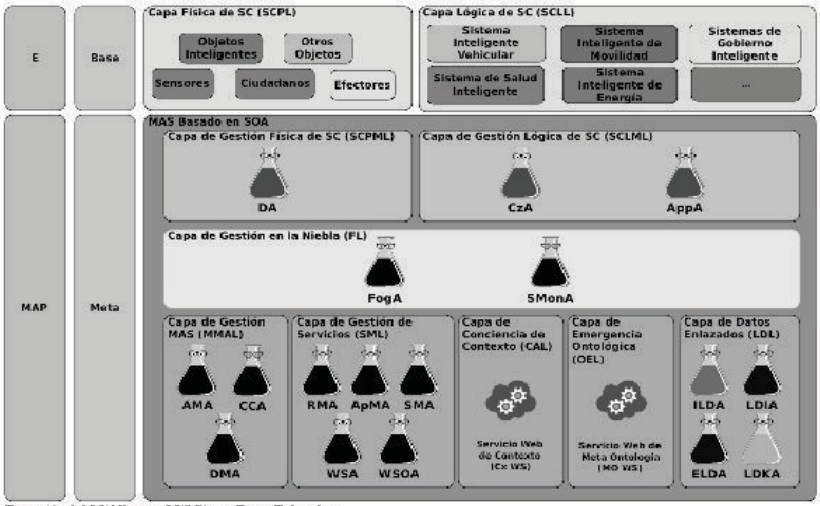

Figura 1. Extensión del Middleware MiSCi con Datos Enlazados.
Fuente: Los Autores

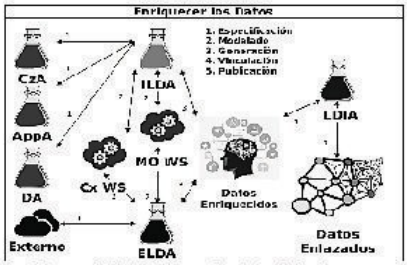

Figura 2. Proceso de Enriquecimiento Semántico de los Datos.
Fuente: Los Autores.

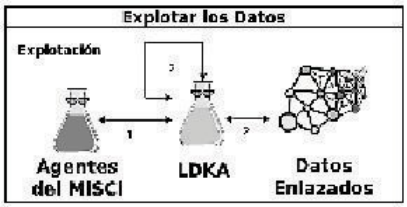

Figura 3. Proceso de Explotación de los Datos.
Fuente: Los Autores.

3.2.1. Agente de Datos Enlazados Internos (*Internal Linked Data Agent* - ILDA)

Son los agentes que brindan la capacidad de extracción, curación, agregación y modelado de las fuentes de información provenientes de agentes internos, como CzA, AppA y DA, para finalmente ser transformadas a formatos adecuados para el enlazado de datos. Específicamente, su objetivo es enriquecer los datos del agente solicitante con información del contexto de MiSCi. El diagrama de actividad (ver Fig. 4) muestra el servicio de este agente, y está compuesto por dos sub-servicios; el primero extrae los datos de las fuentes internas, y el segundo enriquece estos datos con información del contexto y de las ontologías.

Modelo de Agente de ILDA. Este modelo indica el tipo de agente, el rol que cumple y la descripción de su funcionalidad:

- **Tipo:** agente de servicio.
- **Roles:** ofrecer servicio de extracción y enriquecimiento de los datos generados por el MiSCi.
- **Descripción:** procesa solicitudes para extraer y enriquecer datos de MiSCi con información del contexto (Cx WS) y de las ontologías (MO WS), a petición de los agentes CzA, AppA, DA.

Modelo de tareas de ILDA. El servicio "Enriquecer Datos" del agente ILDA presenta las siguientes tareas:

- T1. Recibir la solicitud del agente solicitante
- T2. Extraer datos del agente solicitante
- T3. Modelar los datos del agente solicitante
- T4. Generar los datos como RDF

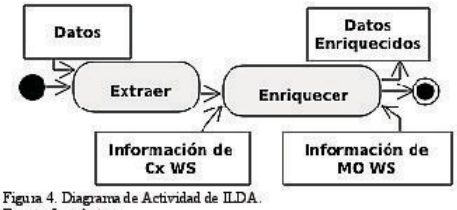

Figura 4. Diagrama de Actividad de ILDA.
Fuente: Los Autores.

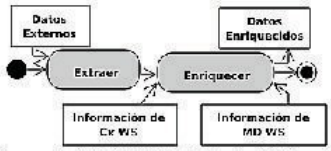

Figura 5. Diagrama de Actividad del Caso de Uso de ELDA.
Fuente: Los Autores.

3.2.2. Agente de Datos Enlazados Externos (*External Linked Data Agent* - ELDA)

Son los agentes que brindan la capacidad de extracción, curación, agregación y modelado de las fuentes de información provenientes del exterior de MiSCi (Redes Sociales, Páginas Web, entre otros), para finalmente ser transformadas a formatos adecuados para el enlazado de datos, que permitan enriquecer semánticamente a la capa CAL del MiSCi con información del exterior. Es decir, ofrece el servicio de extracción y enriquecimiento de las fuentes externas al MiSCi. En la Fig. 5 se observa su diagrama de actividad.

Modelo de Agente de ELDA:

- **Tipo:** agente de servicio.
- **Roles:** ofrecer servicio de extracción y enriquecimiento de los datos de fuentes externas al MiSCi.
- **Descripción:** procesa solicitudes para extraer y enriquecer datos provenientes del exterior (Redes Sociales, Páginas Web, entre otros) con información del contexto (Cx WS) y de las ontologías (MO WS).

Modelo de tareas de ELDA. Las tareas del servicio "Enriquecer Datos" de este agente son los siguientes:

- T1. Recibir la solicitud
- T2. Extraer datos de la fuente externa
- T3. Modelar los datos de la fuente externa
- T4. Generar los datos como RDF

3.2.3. Agente de Integración de Datos Enlazados (*Linked Data Integration Agent* - LDIA)

Es el agente que brinda la capacidad de vincular y publicar la información interna y externa generada por ILDA

y/o ELDA. Este agente vincula los datos, para luego ser publicados como datos enlazados. En la Fig. 6 se observa su diagrama de actividad.

Modelo de Agente de LDIA

- **Tipo:** agente de servicio.
- **Roles:** ofrecer servicio de vinculación y publicación de datos enriquecidos.
- **Descripción:** procesa solicitudes de vinculación de datos enriquecidos con los distintos conjuntos de datos obtenidos en anteriores invocaciones, que luego son publicados como datos enlazados.

Modelo de tareas de LDIA. Las tareas del servicio "Vincular y Publicar los Datos Enriquecidos" de este agente son:

- T1. Recibir la solicitud con los datos enriquecidos
- T2. Vincular los datos enriquecidos
- T3. Publicar los datos enriquecidos

3.2.4. Agente de Conocimiento de Datos Enlazados (*Linked Data Knowledge Agent* - LDKA)

Son los agentes que ofrecen mecanismos para explotar el conocimiento vinculado a los datos enlazados, permitiendo capacidades de: análisis semántico, manejo de ambigüedad, etiquetado semántico, búsqueda exploratoria, visualización, filtrado, entre otros. Los servicios que presta LDKA son: i) ***Recomendar Información*** usando inferencia híbrida de lógica descriptiva/dialéctica para retornar información según la necesidad particular de un agente; ii) ***Generar Modelos de Aprendizaje Automático*** permite retornar modelos de conocimiento basado en distintas técnicas de aprendizaje automático como árboles de decisiones, redes bayesianas, crónicas, redes neuronales como las de aprendizaje profundo, etc.; iii) ***Generar Datos*** para entrenamiento de modelos de conocimiento, muestreos, etc.; iv) ***Aprender Ontologías*** usa los datos enlazados para poblar nuevas ontologías. El diagrama de actividad (ver Fig. 7) muestra los detalles de los servicios prestados por LDKA.

Figura 6. Diagrama de Actividad del Caso de Uso de LDIA.
Fuente: Los Autores.

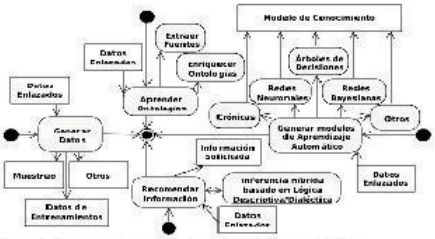

Figura 7. Diagrama de Actividad del Caso de Uso de LDKA.
Fuente: Los Autores.

Los cuatros servicios para explotar el conocimiento vinculado a los datos enlazados, permitirá mejorar trabajos previos vinculados a la integración de datos [22], construcción de recomendadores o modelos de máquinas de aprendizaje más robustos [23], entre otros.

Modelo de Agente de LDKA. El modelo de agente de LDKA se describe de la misma manera que los anteriores agentes.

- **Tipo:** agente de servicio
- **Roles:** ofrecer servicio de explotación del conocimiento, como Recomendar información, Generar modelos de Aprendizaje Automático, Generar datos, o Aprender Ontologías.
- **Descripción:** explora los datos enlazados y realiza las transformaciones del conocimiento según el tipo de solicitud (Recomendar información, Generar modelos de Aprendizaje Automático, Generar datos o Aprender Ontologías).

Modelo de tareas de LDKA. En la Tabla 1 se definen las tareas del agente LDKA, por cada servicio prestado.

4. Caso de estudio

En esta sección se presenta un caso de estudio, dividido en dos escenarios que ocurren en paralelo, para mostrar las capacidades del MiSCi usando la capa de enlazado de datos.

4.1. Escenario 1: enriquecer datos

Este escenario (ver Fig. 8) tiene como objetivo mostrar cómo se realiza el proceso de enriquecimiento de los datos del MiSCi. En este proceso es dónde se extraen y enriquecen los datos de las fuentes de información, que luego son publicados como datos enlazados.

1. Los agentes CzA, AppA y DA envían datos constantemente al Agente ILDA.
2. El agente ILDA recibe y procesa los datos de CzA, AppA y DA. Además, solicita información del contexto (Cx WS) y de las ontologías (MO WS).
3. El agente ILDA recupera los datos solicitados del contexto y de las ontologías, para aprovechar la información del entorno y de los servicios que posee el middleware.
4. Luego, ILDA prepara y enriquece los datos usando la información recuperada. Luego, genera los datos enriquecidos en formato de Tripletas RDF.
5. El agente ELDA procesa los datos de fuentes externas (Redes Sociales, Páginas Web, etc.). Además, solicita información del contexto y de las ontologías.
6. El agente ELDA recupera los datos solicitados del contexto y de las ontologías.
7. ELDA también prepara y enriquece los datos, pero mezclando las fuentes externas con la información solicita del contexto y de las ontologías.

Tabla 1.
Servicios y Tareas de LDKA.

LDKA-S1. Recomendar información
T1. Recibir la solicitud
T2. Inferencia híbrida basado en lógica descriptiva/dialéctica
T3. Retornar información solicitada
LDKA-S2. Generar modelos de Aprendizaje Automático
T1. Recibir la solicitud
T2. Generar modelo de conocimiento basado en técnicas tales como crónicas, redes neuronales, árboles de decisión, redes bayesianas u otros.
T3. Retornar modelo de conocimiento
LDKA-S3. Generar datos de entrenamiento
T1. Recibir la solicitud
T2. Generar los datos de entrenamiento
T3. Retornar los datos
LDKA-S4. Aprender Ontologías
T1. Recibir la solicitud
T2. Extraer fuentes de datos
T3. Enriquecer Ontologías

Fuente: Los Autores.

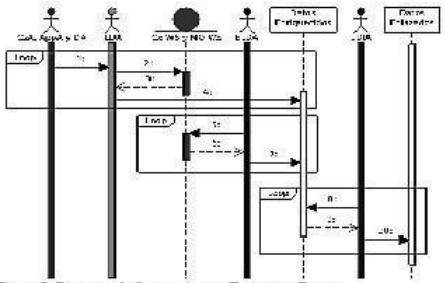

Figura 8. Diagrama de Secuencia para Enriquecer Datos.
Fuente: Los Autores.

8. El agente LDIA solicita los datos enriquecidos que se encuentre en espera de ser procesados.
9. Luego, LDIA enlaza los datos con otros datos previamente registrados/enlazados.
10. Finalmente, LDIA pone a disposición los nuevos conocimientos para ser consultados como Datos Enlazados.

En este primer escenario es donde los distintos agentes de la capa LDL enriquecen y transforman sus datos a datos enlazados. Por ejemplo, ILDA obtiene información interna como los signos vitales del ciudadano por medio de sensores (presión arterial, respiración, temperatura), o el historial clínico del ciudadano (enfermedades anteriores, enfermedades ancestrales, tratamientos actuales, hábitos, alergias, etc.). ELDA recoge datos externos, por ejemplo sobre los síntomas de las distintas enfermedades y sus tratamientos. Finalmente, LDIA vincula toda la información enriquecida y la transforma a datos enlazados.

4.2. Escenario 2: explotar datos

Este escenario tiene como objetivo mostrar cómo se explotan los datos enlazados por la capa LDL del MiSCi en la ciudad. Los datos enlazados necesarios para este escenario son recogidos previamente, o en paralelo, por el proceso de enriquecimiento mostrado en el escenario 1. En este escenario también se muestra el uso de servicios de LDKA, que requieren de otros servicios ofrecidos por el mismo. Por ejemplo, en la Fig. 9 se observa el diagrama de actividad del agente de conocimiento (*Knowledge Agent* - KA) del MiSCi, para el servicio de ***Generar Conocimiento***, que invoca a otros dos servicios del agente LDKA. A continuación se describe este escenario (ver Fig. 10):

1. El agente CzA detecta problemas en los signos vitales del ciudadano, y genera la señal de alarma al HSS (Sistema Inteligente de Salud), a través del AppA que caracteriza a ese sistema.
2. El agente AppA (HSS) solicita al servicio ***Recomendar Información*** del agente LDKA, buscar los ciudadanos en los alrededores con capacidades de practicar los primeros auxilios, que permita brindarle atención médica inmediata al paciente.
3. LDKA retorna la información solicitada por AppA (HSS).
4. Luego, AppA (HSS) envía la notificación a los ciudadanos a través de CzA.
5. En los distintos procesos de actualización de conocimiento van emergiendo ontologías, en consecuencia el componente MO WS va solicitando el servicio de ***Aprender Ontología*** de LDKA, para extraer información y enriquecer dichas ontologías emergentes.
6. LDKA retorna las ontologías emergentes enriquecidas.
7. El AppA (HSS) solicita al servicio ***Recomendar Información*** del agente LDKA, recomendar los servicios de ambulancias, para tratar al paciente y trasladarlo al centro médico más cercano.
8. LDKA retorna la información solicitada por AppA (HSS).
9. El agente AppA (HSS) también solicita al servicio ***Generar Conocimiento*** (Fig. 9) del agente KA, los posibles diagnósticos y sugerencias de tratamiento.
10. El agente KA necesita un modelo de conocimiento para resolver el problema, por lo que activa el servicio ***Generar Modelos de Aprendizaje Automático*** de LDKA, para obtener el modelo de conocimiento sobre dicho problema.
11. LDKA genera y retorna el modelo de conocimiento a KA.
12. También KA necesita datos de entrenamiento para afinar el modelo de conocimiento, para eso solicita al servicio ***Generar Datos*** de LDKA.
13. LDKA genera y retorna los datos de entrenamiento para el modelo de conocimiento de KA.
14. Finalmente, el agente del MiSCi retorna los posibles diagnósticos y sugerencias de tratamiento a AppA (HSS).

En este segundo escenario, se le solicita a LDKA explotar los datos enlazados para generar conocimiento para responder a las distintas necesidades presentes en el MiSCi. LDKA responde con información contextualizada y adaptada a cada necesidad.

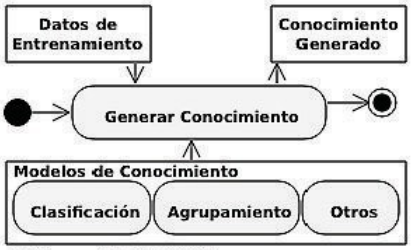

Figura 9. Diagrama de Actividad del KA.
Fuente: Los Autores.

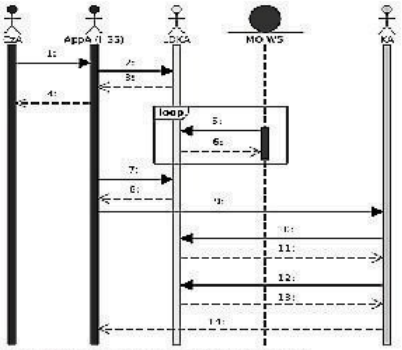

Figura 10. Diagrama de Secuencia para Explotar Datos.
Fuente: Los Autores.

5. Análisis de resultados y comparación

En esta sección, se compara el MiSCi ampliado con datos enlazados, con otras plataformas y/o middleware que poseen características similares. Cada una de las etapas de la metodología MEDAWEDE [4] es usada como criterio de comparación (ver Tabla 2), basados en las características o procesos considerados en cada trabajo para esas etapas. Por ejemplo, el criterio de la etapa de Especificación se subdivide en criterios basados en el origen de la información usada en cada trabajo.

En cuanto al criterio de especificación, todos usan fuentes comunes como sensores y datos enlazados. La diferencia es que [7] y el sistema propuesto en este trabajo poseen también información recolectada por las aplicaciones, e información de los perfiles de las personas o ciudadanos en distintas redes sociales, y/o de sus interacciones con el middleware ([7] no considera esto último). En relación con el criterio de modelado, todos los sistemas enriquecen sus fuentes internas con información extraída de las fuentes externas, ontología y vocabularios. Además, [5,6] y el sistema propuesto en este trabajo se enriquecen con información del contexto, pero este sistema va más allá, al enriquecerse con información de las características de los servicios que presta en general MiSCi. En cuanto al criterio de generación, todos usan las tripletas RDF, y en los criterios de vinculación y publicación, todos usan los datos enlazados, aunque este trabajo no se enfoca a resolver un problema específico usando los datos enlazados, como el resto. Este trabajo fusiona las ventajas de los datos enlazados con las provenientes de las otras capas del MiSCi, que permiten ubicuidad, conciencia del contexto, emergencia ontológica, entre otras cosas, para proveer un entorno con una cantidad de conocimiento aún mayor, a las aplicaciones inteligentes de una ciudad inteligente, que puedan explotarlos según sus necesidades.

Tabla 2.
Tabla Comparativa de Trabajos Similares

Sistemas / Criterios			[5]	[6]	[7]	[8]	Este Trabajo
Especificación	Fuentes Internas	Sensores	✓	✓	✓	✓	✓
		Aplicaciones	x	x	✓	x	✓
		Ciudadanos	x	x	✓	x	✓
	Fuentes Externas	Datos Enlazados	✓	✓	✓	✓	✓
		Páginas Web	x	x	x	x	✓
		Redes Sociales	x	x	x	x	✓
		Otros	x	x	x	x	✓
Modelado	Ontologías		✓	✓	✓	✓	✓
	Vocabularios		✓	✓	✓	✓	✓
	Servicios		x	x	x	x	✓
	Información del Contexto		✓	✓	x	x	✓
	Fuentes Externas		✓	✓	✓	✓	✓
Generación	Tripletas RDF		✓	✓	✓	✓	✓
Vinculación y publicación	Datos Enlazados		✓	✓	✓	✓	✓
Explotación	Búsquedas		✓	✓	✓	✓	✓
	Extracción		✓	✓	✓	✓	✓
	Filtrado		✓	✓	✓	✓	✓
	Inferencia		x	x	x	x	✓
	Enriquecimiento de Ontologías		x	x	x	x	✓
	Generación de Modelos de Conocimiento		x	x	x	x	✓
	Generación de Datos de Entrenamientos		x	x	x	x	✓
	Otros		x	x	x	x	✓

Fuente: Los Autores

En el criterio de explotación, que es donde más diferencias existe entre este trabajo y los anteriores, todos los sistemas realizan tareas comunes, como la búsqueda, extracción y filtrado, pero el sistema propuesto en este trabajo ofrece servicios nuevos, como recomendación híbrida basado en lógica descriptiva/dialéctica, enriquecimiento de ontologías, generación de modelos de conocimiento (por ejemplo, modelos de clasificación o predicción), y generación de datos de entrenamientos para modelos de conocimiento, entre otros. Así, este sistema ofrece un conjunto de servicios innovadores, que permiten explotar los datos enlazados. Además, el sistema propuesto en este trabajo permite aprovechar los datos ofrecidos por las distintas fuentes en una ciudad, ya sea interna (sensores, aplicaciones o ciudadanos) o externa (redes sociales, páginas web, etc.), para convertirlos en conocimiento. A su vez, en el caso de los servicios de enriquecimiento de ontologías y generación de datos de entrenamiento, permite filtrar los datos relevantes de la ciudad. Por otro lado, el servicio de recomendación basado en inferencia híbrida usando lógica descriptiva/dialéctica, es necesario para la toma de decisiones en momentos de ambigüedades, en contexto donde la presuposición falla (para el diagnóstico o detección de fallas), en contingencias sobre el futuro (que indican que algo fue verdad y falso en el pasado), y/o en el discurso ficticio donde se toman decisiones según ciertos supuestos.

El añadir una capa a MiSCi para explotar los datos enlazados, si bien introduce costos computacionales y de almacenamiento extras (ver sección 3), son largamente sobrepasados por los beneficios que provee la misma a las aplicaciones inteligentes conscientes del contexto, por el acceso a una gran cantidad de conocimiento.

6. Conclusiones

La extensión de MiSCi con datos enlazados, provee un middleware innovador para el manejo de datos enlazados en un ambiente inteligente, aprovechando los datos generados por las distintas fuentes, ya sea interna (sensores, aplicaciones o ciudadanos) o externa (redes sociales, páginas web, etc.), para integrarlos para ser explotados. Además, la integración permite aprovechar los datos externos para enriquecer los internos, lo que es una ventaja al momento de arrancar un sistema. Por otro lado, la capa de datos enlazados, al aprovechar la información del entorno y de los servicios del MiSCi, le permite adaptarse a distintas circunstancias, es decir, aprovecha la información del contexto del MiSCi para responder a problemas y necesidades particulares, como se muestra en los escenarios del caso de estudio. Por ejemplo, en dichos escenarios aprovecharía las ontologías taxonómicas que MiSCi posee, y sus datos enlazados, para enriquecer el contexto en cada escenario.

Asimismo, este middleware tiene la capacidad de transformar y explotar la información obtenida, usando distintos mecanismos inteligentes. Por ejemplo, puede enriquecer ontologías existentes en MiSCi, puede generar modelos de conocimiento requeridos por los servicios de MiSCi, generar datos para construir modelos de conocimiento para MiSCi, y recomendar información en contextos de incertidumbre a través de una inferencia híbrida basada en lógica descriptiva/dialéctica.

En los casos de estudio se muestra como el middleware MiSCi orquesta los agentes para trabajar con los datos enlazados, y así responder a las necesidades de la ciudad y sus ciudadanos. En el escenario 1, se describe cómo interactúan los agentes para extraer y enriquecer los datos desde las distintas fuentes, para que luego estos datos enriquecidos sean vinculados y publicados como datos enlazados. En el escenario 2, se observa cómo se aprovechan los distintos servicios ofrecidos por LDKA, para explotar el conocimiento obtenido de los datos enlazados. En particular, esta nueva capa es vital en los procesos de actualización de conocimiento, en los que van emergiendo ontologías para responder a situaciones imprevistas (emergentes) [1]. En específico, estas ontologías necesitan poblarse, y gracias al servicio de aprender ontología de LDKA se puede extraer conocimiento proveniente de los datos enlazados del MiSCi, que permita enriquecer dichas ontologías emergentes.

Así, los elementos diferenciadores de nuestra propuesta con respecto a las anteriores, es que explota diferentes fuentes internas y externas para el enlazado de datos, y ofrece un grupo de servicios de gestión del conocimiento basado en datos enlazados.

En trabajos futuros se desarrollarán prototipos de los distintos agentes y procesos involucrados en el enriquecimiento y explotación de los datos. En este trabajo, en la sección 3.2 se hizo una especificación general de los agentes de esta capa, particularmente de sus características generales y las tareas que brindan como servicios en el contexto de MiSCi. Ahora, bien, falta por definirle a cada uno sus mecanismos de razonamiento y aprendizaje, así como también, diseñar como representarán sus conocimientos. Además, en trabajos futuros se harán desarrollos para la extracción automática de datos desde las fuentes internas y externas a MiSCi, para la inferencia híbrida basada en lógica descriptiva/dialéctica, para la generación de modelos de conocimiento y generación de datos (entrenamiento, muestreo, entre otros), y finalmente, para el enriquecimiento ontológico. Para la implementación de los mismos, algunos de los problemas a resolver será cómo realizar la construcción automática de la base de conocimiento y su motor de inferencia para el sistema recomendador según el contexto; o cómo identificar el modelo de conocimiento o el conjunto de datos pertinente a una situación dada, para el contexto de aprendizaje automático, entre otras cosas.

Referencias

[1] Aguilar, J., Jerez, M., Mendonca, M. and Sánchez, M., MiSCi: autonomic reflective middleware for smart cities, in: Technologies and Innovation. Second International Conference, 2016, Guayaquil, Ecuador. Proceedings, Cham, Suiza, Springer International Publishing, , pp. 241-253. DOI: 10.1007/978-3-319-48024-4.

[2] Aguilar, J., Sanchez, M.B., Jerez, M. and Mendonca, M., An extension of the MiSCi Middleware for smart cities based on fog computing. Journal of Information Technology Research (JITR), 10(4), pp. 23-41,

2017. DOI: 10.4018/JITR.2017100102
[3] Enerlis, Ernst and Young, Ferrovial y Madrid Network. Libro Blanco Smart Cities, 1ra ed, España, Imprintia, [en línea]. 2012. Disponible en: http://www.innopro.es/pdfs/libro_blanco_smart_cities.pdf
[4] Rodríguez, T., Dos Santos, R. y Aguilar, J., Metodología para el desarrollo de aplicaciones Web utilizando datos enlazados, en: Conferencia Nacional de Computación, Informática y Sistemas, 2017, Ciudad Guayana, Venezuela. Memorias, Universidad Católica Andrés Bello, Ciudad Guayana, Venezuela, [online]. 2017, pp. 114-122. Disponible en: http://concisa.net.ve/memorias/CoNCISa2017/CoNCISa2017-p114-122.pdf
[5] Salmen, A., Münch, T., Buzin, S., Hladik, J., Altmann, W., Weber, C. and Graube, M., Comwantage: mobile enterprise collaboration reference framework and enablers for future internet information interoperability. The Future Internet Assembly, Springer, Berlin, Heidelberg, 7858, pp. 220-232, 2013. DOI: 10.1007/ 978-3-642-38082-2_19
[6] Barnaghi, P. and Presser, M., Publishing linked sensor data, in: International Conference on Semantic Sensor Networks, Shanghai, China., [online]. 2010, pp. 1-16. Available at: http://ceur-ws.org/Vol-668/paper2.pdf
[7] López-de-Ipiña, D., Vanhecke, S., Peña, O., De Nies, T. and Mannens, E., Citizen-centric linked data apps for smart cities, in: Urzaiz, G., Ochoa, S.F., Bravo, J., Chen, L.L., Oliveira, J. (eds), Ubiquitous computing and ambient intelligence. Context-Awareness and Context-Driven Interaction. Lecture Notes in Computer Science, 8276, pp. 70-77, Springer, Champp. 2013. DOI: 10.1007/978-3-319-03176-7_10
[8] Le-Phuoc, D., Nguyen-Mau, H.Q., Parreira, J.X. and Hauswirth, M., A middleware framework for scalable management of linked streams. Journal Web Semantics: Science, Services and Agents on the World Wide Web, 6, pp. 42-51, 2012. DOI: 10.1016/j.websem.2012.06.003
[9] Aguilar, J., Bolívar, A.R., Hidrobo, F. y Cerrada, M., Sistemas multiagentes y sus aplicaciones en automatización industrial, 2da ed., Talleres Gráficos, Universidad de Los Andes, Mérida, Venezuela, 2013.
[10] FIPA, IEEE, 2013, [en línea]. Disponible en: http://www.fipa.org
[11] Sánchez, M., Aguilar, J., Cordero, J. and Valdiviezo, P., Basic features of a reflective middleware for intelligent learning environment in the cloud (IECL), in: Asia-Pacific Conference on Computer Aided System Engineering, Quito, Ecuador, IEEE, 2015, pp. 1-6. DOI: 10.1109/APCASE.2015.8
[12] Aguilar, J., Sánchez, M., Cordero, J., Valdiviezo P., Baiba L. and Chamba, L., Learning analytics tasks as services in smart classrooms. Universal Access in the Information Society Journal, 17(4), pp. 693-709, 2018. DOI: 10.1007/s10209-017-0525-0
[13] Aguilar, J., Jerez, M. and Rodríguez, T., CAMeOnto: context awareness meta ontology modeling, Applied Computing and Informatics, 14(2), pp. 202-213, 2018. DOI:10.1016/j.aci.2017.08.001
[14] Mendonça, M., Aguilar, J. and Perozo, N., MiR-EO: reflective middleware for ontological emergency in intelligent environments, Latin American Journal of Computing, 3(2), pp. 25-39, 2016.
[15] Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J. and Sheets, D., Tabulator: exploring and analyzing linked data on the semantic web, in: Proceeding, International Semantic Web User Interaction Workshop, 2016, 2006, pp. 159-174.
[16] Dos Santos, R. y Aguilar, J., Enlazado de datos, en: Introducción a la Minería Semántica, 1ra Ed., Fondo Editorial Universidad Experimental del Táchira (FEUNET),, Venezuela, 2018, pp. 173-212.
[17] Berners-Lee, T., Linked Data, [en línea]. 2006, Disponible en: https://www.w3.org/DesignIssues/LinkedData.html
[18] Wood, D., Zaidman, M., Ruth, L. and Hausenblas, M., Linked Data: structured data on the Web, 1st Ed., Manning Publications Co., Greenwich, Connecticut, USA, 2014.
[19] Vizcarrondo, J., Aguilar, J., Exposito, E. and Subias, A., MAPE-K as a service-oriented Architecture. IEEE Latin America Transactions, 15(6), pp. 1163-1175. 2017. DOI: 10.1109/ TLA.2017.7932705
[20] Aguilar, J., Cordero, J. and Buendía, O., Specification of the autonomic cycles of learning analytic tasks for a smart classroom. Journal of Educational Computing Research, 56(6), pp. 866-891. 2018. DOI: 10.1177/0735633117727698
[21] Aguilar, J., Cerrada, M. and Hidrobo, F., A methodology to specify multiagent systems. In: Conference on KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications. Lecture Notes in Computer Science, Berlin, Heidelberg, Springer, 2007, pp. 92-101. DOI: http10.1007/978-3-540-72830-6_10
[22] Hidalgo, Y., Ortiz, E. and Febles, J.P., Method for integrating bibliographic data from OAI-PMH data providers. IEEE Latin America Transactions, 15(9), pp. 1695-1699, 2017. DOI: 10.1109/TLA.2017.8015075
[23] Rojas, G. and Garrido, I., Toward a rapid development of social network-based recommender systems. IEEE Latin America Transactions, 15(4), pp. 753-759, 2017. DOI:10.1109/TLA.2017.7896404
[24] Sanchez, J., Aguilar, J. and Exposito, E., Fog computing for the integration of agents and web services in an autonomic reflexive middleware, Service Oriented Computing and Applications, 12(3-4), pp. 333-347, 2018. DOI: 10.1007/s11761-018-0238-0
[25] Aguilar, J., Jerez, M., Mendonça, M., et al., Performance analysis of the ubiquitous and emergent properties of an autonomic reflective middleware for smart cities. Computing, 2020. DOI: 10.1007/s00607-020-00799-5

R.J. Dos Santos-Guillén, obtuvo el título en Ing. de Sistemas en 2006 y en MSc. en Computación en 2016, ambos en la Universidad de Los Andes-Venezuela. Actualmente es candidato a Dr. en Ciencias Aplicadas en Universidad de los Andes-Venezuela y profesor en la Universidad Politécnica Territorial del Estado Trujillo "Mario Briceño Iragorry", Trujillo-Venezuela. Es investigador en el Centro Estudios en Microelectrónica y Sistemas Distribuidos (CEMISID); y en Tepuy R+D Group, Artificial Intelligence Software Development. Su interés en investigación incluye inteligencia artificial, datos enlazados, big data, minería semántica, lógica dialéctica, analítica social, y ambientes inteligentes.
ORCID: 0001-8752-8550

J.L. Aguilar-Castro, obtuvo el título en Ing. de Sistemas en 1987 de la Universidad de Los Andes-Venezuela, MSc. en Ciencias de la Computación en 1991 en la Universidad Paul Sabatier-Francia, y Dr. en Ciencias de la Computación en 1995 de la Universidad René Descartes-Francia. Fue investigador postdoctoral en el Departamento de Ciencias de la Computación de la Universidad de Houston (1999-2000) y en el Laboratoire d'Analyse et d'Architecture des Systèmes, CNRS, Francia (2010-2011), e investigador prometeo en Ecuador (2014-2017). Actualmente es profesor titular del Departamento de Ciencias de la Computación de la Universidad de Los Andes, Mérida, Venezuela, e investigador invitado en la Universidad EAFIT, Medellín, Colombia. Es investigador en el CEMISID; y en Tepuy R+D Group. Además, es miembro de la Academia de Mérida y del Comité Técnico de Redes Neuronales del IEEE. Ha publicado más de 550 artículos y libros, en el campo de sistemas paralelos y distribuidos, inteligencia computacional, etc. Su interés en investigación incluye Inteligencia Artificial, Minería Semántica, Big Data, Sistema Emergente y Entornos Inteligentes.
ORCID: 0003-4194-6882

T.J. Rodríguez-de Paredes, obtuvo el título de Ing. de Sistemas en 1993, MSc. en Computación en 2000 y Dr. en Ciencias Aplicadas en 2016, todos en la Universidad de Los Andes-Venezuela. Actualmente es profesora en el postgrado en Computación, de la Universidad de los Andes, Mérida-Venezuela. Es Investigadora en el CEMISID; y en Tepuy R+D Group. Su interés en investigación incluye Datos Enlazados, Aprendizaje de Ontologías, Minería Semántica y Procesamiento de Lenguaje Natural
ORCID: 0002-0429-3141

8.3 Anexo 3.B: Automated Ontology Generator System based on Linked data

Automated Ontology Generator System based on Linked Data

Ricardo Dos Santos
CEMISID, Universidad de Los Andes, Mérida, Venezuela.
ricardojdsg@gmail.com

Eduard Puerto
GIA, Univ. Francisco de Paula Santander, Cúcuta, Colombia
eduardpuerto@ufps.edu.co

Jose Aguilar
CEMISID, Universidad de Los Andes, Venezuela
GIDITIC, Universidad EAFIT, Colombia
IMDEA Network Institute, Madrid, Spain
aguilar@ula.ve

Abstract - In this work, an Automated Ontology Generator System (AOGS) is developed to autonomously create and populated ontologies based on the linked data paradigm. AOGS generates context-specific ontologies and populates these with information extracted from various linked data sources and pre-existing ontologies. AOGS allows reducing the time and effort required to create an ontology, ensuring consistency and accuracy in the ontology, and that domain experts to focus on higher-level tasks such as ontology refinement and evaluation. MEDAWEDE methodology is used to develop AOGS, which is then applied to create ontologies for COVID-19 and Energy. The quality of these ontologies is evaluated using precision, recall and f-measure metrics, which yield values between 0.9 and 0.7. The ontologies are also validated using Competence Questions, Consistency, and Quality Validation methods, all of which show good results.

Keywords -Emerging Ontology, Linked Data, Alignment, Web Services.

I. INTRODUCTION

The integration of knowledge from scattered data sources is a problem for its timely and effective exploitation when making decisions. Thus, it is relevant the integration of information, e.g., the one that the sensors have collected in advance, or that is stored in external data sources such as Ontologies and Linked Data (LD).

LD is a way of publishing data on the Web, so that they can interconnect between them, allowing them to identify, describe, connect, and relate the different elements on the Web [1, 2, 3, 4]. It provides a huge interconnected and decentralized knowledge base, which is readable by people and machines, but requires services and applications that make intensive use of all this knowledge [4]. LD provides a rich source of semantic knowledge since it identifies and describes the web resources and the different contexts in which they are found.

On the other hand, for the automatic creation of ontologies, it is necessary to have the ability to merge ontologies [5, 6], since combining the knowledge distributed among several ontologies will allow having a broader ontology on a specific domain. It is also necessary to automate the extraction and linking in the process of creating an ontology. For this, we need to define a method that allows the extraction and linking of data from an internal and external source. This involves several key steps. First, we must collect relevant data from LD sources in a more efficient manner. Next, we must identify the concepts and relationships that are most relevant based on the collected data and existing ontologies. Then, we can extract and map the relevant data from LD to the ontology concepts and relationships. Finally, the new ontology can be populated with information from Linked Open Data (LOD). By automating this process, we can save time and effort while creating more accurate and comprehensive ontologies.

The objective of this work is to develop an AOGS that allows creating and populating ontologies using the LD paradigm. Similar works to our approach include Lod-Abog [7], which is based on Natural Language Processing (NLP); Onto6 [8], which is based on ontologies, and [9], which uses decision trees. Additionally, DeepCBRS [10] utilizes recurrent neural networks, while [11] is based on deep learning (DL), and [12] is based on NLP. They all try to build ontologies automatically using a different approach. Unlike these works, our AOGS model autonomously [16] explores and evaluates concepts from other ontologies using the LD paradigm. In general, the main contributions of AOGS are:

- An efficient and scalable ontology creation approach using mechanisms to build and populate ontologies. It is a time-consuming and labor-intensive task, and is especially valuable for organizations or researchers who need to create and manage large-scale ontologies.
- An improved data integration approach, by enabling the system to automatically identify and link related heterogeneous data sources. This leads to faster and more accurate data integration, which is critical for many applications, including data analytics and knowledge discovery.

This paper is divided into the following sections: in Section II, related works are descriptive. In Section III, the OAGES architecture is presented. In Section IV are instantiated two case studies using OAGES: one for COVID-19 and another for energy management. Section V discusses the results, presenting a brief discussion of limitations, and a comparative analysis with related work, and finally, section VI describes the conclusions.

II. RELATED WORK

In this section, we present related works about ontology

learning approaches with LD. Then, we make a comparative study, and at the end, some discussion.

A. Previous Works

In [13], the authors propose an ontology matching (OM) system, named BERTMap, which can support both unsupervised and semi-supervised settings. It first identifies mappings using a classifier based on fine-tuning the contextual embedding model BERT on text semantics corpora extracted from ontologies, then refines the mappings through extension, and repairs them by utilizing the ontology structure and logic. In [8], Straujums proposes the identification of significant concepts for a given domain based on a semi-informal meta-ontology. This approach, called ONTO6, is useful for the development of a unified understanding of the domain. The steps are the development of a meta-ontology instance appropriate for the domain to be digitalized; the development of an initial ontology from the meta-ontology instance; and the gradual breakdown of domain concepts that appear in the initial ontology. In [7] is proposed LOD-ABOG (Linked Open Data approach for Automatic Biomedical Ontology Generation). LOD-ABOG performs concept extraction using LOD and NLP operations; and applies relation extraction using the Breadth first Search (BSF) graph method and Freepal repository patterns. In [9], an enrichment of Ontology Instances Using LD is presented. The system enriches an ontology in two phases. First, it extracts new instances and relations from a reference ontology of the same or similar domain. Second, it validates the possible relationships between the original instances and the new ones using crawled data from the web search. Musto et al. [10] provided an ontology enrichment approach from text, in which the Web of LD, particularly DBpedia, is used. They explored DBpedia as background knowledge to discover implicit knowledge, from which new ontological relations, specifically object properties, were inferred. In [14], a methodology for extracting concepts for a target domain from large-scale LOD is proposed to support the construction of domain ontologies. It allows the creation of an initial model of the domain ontology, which defines entities by linking the LOD vocabulary with technical terms related to the target domain. The entities are then used as a starting point for obtaining upper-level concepts in the LOD.

On the other hand, Lamine et al. [11] deal with concepts' extraction using DL models and word embedding techniques to automatically extract relevant concepts from large amounts of textual data. Four phases were proposed: the preprocessing phase, the classification phase based on DL models, the terms filtering phase, and the semantic enrichment phase to semantically enrich the discovered concepts. While in [12], the process of enrichment is summarized in three main steps. In the first step, the utilization of NLP techniques to obtain tagged sentences. The second step cuts down on each extracted sentence to an SVO (Subject, Verb, and Object) sentence. Finally, in the third step is enriched an initial ontology manually by adding extracted terms in the generated SVO as new concepts or instances of concepts and new relations.

B. Qualitative Comparison

To provide a comparative study, we introduce some criteria:

- Source: structured, semi-structured, and unstructured input data; other ontologies; LOD.
- Methods: techniques used to treat ontologies such as NLP, DL, etc.
- Automation degree. The acquisition of knowledge may be performed automatically, semi-automatically, or with users or experts.
- Result: a new ontology, an enriched ontology, or part of an ontology (concepts, axiom, rules, etc.)

Table I show the comparison with works whose objective is to enrich/create an ontology.

TABLE I
COMPARISON WITH OTHER WORKS

	Source	Method	Result	Automation
[7]	(UMLS)[a], LOD	NPL, ontology	Ontology domain	Automatic
[8]	Textual Domain	Meta-ontology	relevant concept	Semi-automatic
[9]	Relational databases	decision trees	taxonomies	automatic
[10]	textual description	Recurrent Neural Networks	Items recommend	automatic
[11]	textual data	DL	relevant concepts	automatic
[12]	Textual domain	NPL	sentences; Subject, Verb, and Object	automatic
Our AOGS	Ontology and LD	Similarity and Rules	ontology enriched	automatic

[a] UMLS Base-Unified Medical Language System

While other works use only one source such as texts, ontologies, or LD, OAGS uses ontologies and LD. Regarding methods, our approach uses an algorithm to determine the similarity between concepts, and then relies on rules that determine the level of cohesion of the concept with respect to the position in the ontology hierarchy. The biggest advantage offered by our method is its automation, as it does not require human support to select the concepts that most closely relate to the client's need.

Additionally, the other works partially obtain semantic elements, such as taxonomies, concepts, and sentences, while OAGS goes further and integrates them. Finally, our system operates in an automatic manner in the whole process, like almost everyone.

III. GENERAL ARCHITECTURE OF AOGES

The architectural design of our proposal is based on the different stages of the MEDAWEDE methodology [2, 15]. This methodology guides and specifies the construction of services that allow the consumption of sources described as

LD. MEDAWEDE is composed of six stages [2]: *i. Specification:* it analyzes and selects the ontological data sources to be used. *ii. Modeling:* it defines the database of the area of study for the ontologies to be generated. *iii. Generation:* it executes the process of transformation, filtering, and integration of the data in the created ontology. *iv. Linking:* it associates the data of the created ontology with the knowledge coming from the linked datasets. *v. Publication:* it makes available the created ontology in different standard formats. *vi. Exploitation:* it defines interfaces to access the ontology.

A. *General Architecture of AOGS*

The general architecture of AOGS is shown in Fig. 1. It is composed of three layers: Knowledge Base Manager, Knowledge Generator Manager and Web services Manager.

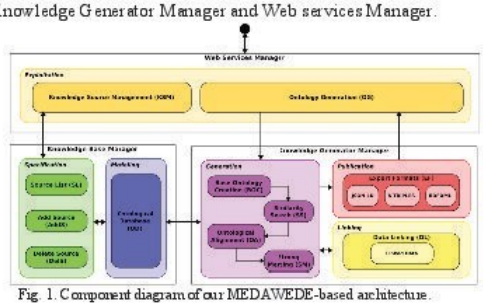

Fig. 1. Component diagram of our MEDAWEDE-based architecture.

- *Knowledge Base Manager:* it is based on stages (i) and (ii) of MEDAWEDE. In our case, it is in charge of managing and storing the knowledge of AOGS.
- *Knowledge Generator Manager:* it is based on stages (iii), (iv) and (v) of MEDAWEDE. In our case, it controls the processing of the knowledge of the previous layer, and generates extended ontologies with LD.
- *Web Services Manager:* it is based on the stage (vi) of MEDAWEDE. In our case, it is in charge of receiving the web requests made by the clients to AOGS, for example, ontology generation service or knowledge source management service.

B. *Components of the Knowledge Base Manager*

The components described in this section allow the management of the system's knowledge base.

1) Ontological Database (OD): This component is responsible for storing and making available all the knowledge that the system has in order to generate ontologies for specific contexts. The OD stores the classes, properties and relations of the different ontologies. This component is activated by the rest of the components that need to manage system information.

2) Source List (SL): This component shows a list of the ontologies that are available in the system's knowledge base. This component is activated by *Knowledge Source Management* (KSM) services.

3) Add Source (AddS): It is responsible for extending the knowledge base available in the system to generate an ontology in a specific context. This component is activated by the KSM services together with the ontology to be added to the system.

4) Delete Source (DelS): It is responsible for reducing the knowledge base available in the system to generate an ontology in a specific context. This component is activated by KSM services together with the id of the ontology to be deleted from the system.

C. *Components of the Knowledge Generator Manager*

This section describes the components that allow generating the ontology of the context requested by the client.

1) Base Ontology Creation (BOC): This component creates a base ontology with the concepts defined as *Search Terms*. The *Search Terms* are provided by the client to obtain an ontology of a specific domain. These concepts will be the root nodes of the ontology to be generated. In the following components, these base concepts will relate to the new concepts selected and from the enrichment with LD.

2) Similarity Search (SS): This component finds the possible ontological concepts to add to the base ontology. For this, it searches for the synonyms of the *Search Terms*, and compares the *Search Terms* and their synonyms with the concepts stored in the OD, extracting the matches. The result is a list of the concepts that matched the *Search Terms* and their synonyms.

3) Ontological Alignment (OA): This component is responsible for the alignment of the ontological concepts obtained in SS, weighting the relationships that exist between the concepts found with the *Search Terms* and their synonyms. The weighting follows the next rules: i. If a *Search Term* perfectly matches the found concept, its score is maximum (1). ii. If a *Search Term* is partially immersed within the found concept, i.e. it is presented as a suffix or prefix, its score is half of the perfect match (0.5). iii. If a *Search Term* is completely immersed as a substring of the found concept, its score will be a quarter of the perfect match (0.25). Then, it punishes with 0 a concept with no semantic relation to the search terms. Finally, it selects the concepts that meet the acceptance threshold provided by the ontology requester. This threshold defines how severe or permissive the filtering of the selected concepts is. The result of this component is a list of concepts that will be integrated into the base ontology.

4) Strong Merging (SM): It integrates the concepts selected from the ontologies managed by OD. In this process, it copies the list of concepts selected within the base ontology, considering the conceptual hierarchies (parent and child nodes, properties and relationships) present in each source ontology. The result is the base ontology populated with the

corresponding knowledge of the requested domain.

5) Data Linking (DLi): It enriches the generated ontology with LD, using the DBpedia Spotlight service (https://www.dbpedia-spotlight.org/), which provides an LD based solution to relate keywords with related resource identifiers in the Dbpedia knowledge graph. Finally, the found resources are linked to each concept in the ontology. For example, when searching for "COVID", the service returns http://dbpedia.org/resource/COVID-19. Then, this response is linked to the COVID concept of the base ontology.

6) Export Formats (EF): It is responsible for transforming the ontology enriched with LD to the format required by the interface. Among the formats currently handled are JSON-LD, RDF/XML and N-TRIPLES.

D. Components of the Web Services Manager

An overview of the components that provide the different services of the system is given below.

1) Knowledge Source Management (KSM): This component is responsible for offering services to expand or reduce the knowledge base managed by the system.
2) Ontology Generation (OG): This component is responsible for activating the ontology creation process for a specific context. The result of this component is the creation of an ontology exploiting the ontological knowledge available in the system and LD. The offered service receives the search parameters such as *Search Terms*, ontology generation format, among others.

E. Behavior of AOGS

Our system presents two main behaviors: 1) Knowledge Management and 2) Knowledge Generation. This section details each of them.

1) Knowledge Management
Objective: to list, add or delete the ontologies that are used as a source of knowledge managed by the system.
General description: The process shown in Table II, starts when the KSM web service receives a knowledge source management request with its required parameters (Step 1).

TABLE II
MACRO-ALGORITHM OF KNOWLEDGE MANAGEMENT

Input: Type of request and its parameters
Procedure:
1. KSM processes the request.
2. KSM verifies the parameters of the request.
3. KSM activates the component corresponding to the type of request.
3.1. If it is listed, KSM invokes SL.
3.1.1. SL requests the ontology listing from OD.
3.2. If it is added, KSM invokes AddS with the ontology to be added.
3.2.1. AddS requests OD to add the ontology.
3.3. If it is deleted, KSM invokes DelS with the ID of the ontology to be deleted.
3.3.1. DelS requests OD to delete ID.
Output: Request Processed

In Step 2, KSM verifies the parameters according to the type of request. In the case of listing, it does not require additional parameters. In the case of adding, the received parameter is an ontology to be added to the system. In the case of deleting, the received parameter is an ID of the ontology to be removed from the system. Then, KSM activates the component corresponding to the type of request (Step 3). If it is to list the ontologies owned by the system, then KSM invokes SL (Step 3.1).

Then, SL asks OD for the list of ontologies (3.1.1). If an ontology must be added to the system, KSM invokes AddS (Step 3.2). Then, AddS requests OD to add the ontology to the system (3.2.1). If the request is to remove an ontology to the system, then KSM invokes DelS (Step 3.3). Then, DelS requests OD to remove the ontology from the system (3.3.1).

This process allows keeping the knowledge managed by AOGS up to date.

2) Knowledge Generation
Objective: to generate an ontology with LD and the ontological knowledge of the system, using search parameters such as search terms, acceptance threshold, among others.
General description: Table III presents the Knowledge Generation process, which starts when the OG web service receives the request to create an ontology with the generation parameters (Step 1). Then, BOC creates the base ontology and adds the *Search Terms* as root nodes (Step 2). In Step 3, it generates a list of synonyms of the *Search Terms*. Then, SS compares the *Search Terms* and their synonyms with the concepts stored in OD (Step 4).

TABLE III
MACRO-ALGORITHM OF KNOWLEDGE GENERATION

Input: Search Terms, acceptance threshold and ontology formatting
Procedure:
1. OG processes the request.
2. BOC creates the base ontology.
3. SS searches for synonyms of Search Terms.
4. SS compares the search parameters with the knowledge in OD.
5. SS generates the list of matches with Search Terms and synonyms.
6. OA weights the concepts obtained in the list of matches.
7. OA filters the concepts that meet the supplied acceptance threshold.
8. OA generates the list of selected concepts.
9. SM integrates the selected concepts.
10. DLi enriches the ontology with Linked Data.
11. EF transforms the ontology to the required format
Output: Generated Ontology

Then, SS generates the lists of matches with the *Search Terms* and their synonyms (Step 5). In Step 6, OA weights the relationships that exist between the lists of matches with the *Search Terms* and their synonyms (see section III.B.3). With the weights, OA filters the concepts that meet the supplied acceptance threshold (Step 7) and generates the list of selected concepts (Step 8). Finally, SM integrates the selected concepts into the ontology (Step 9). With the ontology built, DLi proceeds to search and link each concept of the ontology with knowledge available on the web using the LD paradigm (Step

10). As a last step, EF transforms the ontology to the required format (Step 11). The result of these processes is an ontology enriched with LD exploiting the knowledge available in the system.

IV. CASE STUDY

This section presents two application domains for the ontology generator.

A. *Case Study 1: COVID-19*

A1. Description. In this test is taken as a source of knowledge the ontologies of the COVID-19 domain, which is an infectious disease caused by the SARS-CoV-2 virus [15]. Table IV shows the ontologies linked to COVID-19 that were selected from the ontology repository https://bioportal.bioontology.org/ontologies.

A.2. Knowledge Management. This process presents the loading of ontologies to the system as knowledge sources for the Knowledge Generation process. It executes the KSM web service with its parameters, a process that is repeated with each ontology to be added to the system. Afterward, KSM receives, processes, and executes the modules corresponding to the type of request received (see steps 1, 2 and 3 in Table II). For this case, it activates the AddS module (see step 3.2 in Table II), which is responsible for adding the ontology in OD of the system (see step 3.2.1 in Table II).

TABLE IV
ONTOLOGICIES ON COVID-19

N	Ontology	Description	URL
1	COVID-19 Ontology for Cases and Patient information (CODO).	It contains knowledge related to the patients of COVID-19 (52 concepts)	URL[1]
2	COVID-19 Surveillance Ontology (COVID19)	It contains knowledge related to the surveillance of COVID-19 in primary care (52 concepts)	URL[2]
3	The COVID-19 Infectious Disease Ontology (IDO-COVID-19)	It contains knowledge related to infectious diseases around COVID-19 (486 concepts)	URL[3]
4	COVID-19 Ontology (COVID-19)	It contains general knowledge of COVID (2286 concepts)	URL[4]
5	WHO COVID-19 Rapid Version CRF semantic data model (COVIDCRFRAPID)	It contains knowledge about standardized clinical data on COVID-19 (398 concepts)	URL[5(a)] URL[5(b)]

URL[1] = https://bioportal.bioontology.org/ontologies/CODO
URL[2] = https://bioportal.bioontology.org/ontologies/COVID19
URL[3]= https://bioportal.bioontology.org/ontologies/IDO-COVID-19
URL[4]= https://bioportal.bioontology.org/ontologies/COVID-19
URL[5(a)]= https://bioportal.bioontology.org/ontologies/COVIDCRFRAPID
URL[5(b)]= https://vodan-ontology.github.io/

A.3. Knowledge Generation. This process generates the ontology with LD. For this, the OG web service is executed by passing as parameters the *Search Terms*, the ontology format and the acceptance threshold. For example, Figure 2 indicates that the ontology will be generated in RDF/XML format with an acceptance threshold of 70% and the area of interest of the new ontology is "COVID" and "test".

Fig. 2. Generation interface.

The component SS, using the *Search Terms*, obtains their synonyms (see step 3 in Table III). For example, for "COVID" was found "Coronavirus", while for "Test" were found several synonyms such as "trial", "exam", "quiz", among others. Then, it carries out queries to obtain the ontological concepts of the ontologies added to the system (see steps 4 and 5 in Table III). Some of the possible results are "Untested for COVID-19", "COVID-19 Diagnosis", "Tested for 2019-nCov (Wuhan) infection". However, not all concepts will end up populating the ontology since many of them are not semantically related to the domain that was consulted. For instance, "intestine cancer", "assay screened entity" and "testis", therefore, a filter of these concepts will be necessary.

Fig. 3 shows some weightings between the match listings (blue), and the *Search Terms* (dark green) and their synonyms (light green) (see step 6 in Table III). For example, the comparison of "COVID" from the list of matches with "COVID" from the search term is perfect; therefore, its score will be maximum (1).

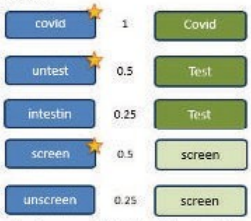

Fig. 3. Weightings between match listings (blue) and Search Terms (dark green) and their synonyms (light green).

In the case of the search term "Test", it is partially immersed inside the token "untest", with an added suffix (in other cases, it can be presented as a prefix); therefore, its score is half of the perfect match (0.5). In the third case, the search term is completely immersed as a substring of the concept present in the listing ("Test" in "intestin"). In this case, the weighting is one quarter of the perfect match (0.25) to punish possible concepts with no semantic relation to the search. This is how it is done for the rest of the concepts.

In summary, a concept whose label is "Tested for COVID-19" will score higher than one whose label is "Study for COVID propagation", since in the first case it refers to "Test" and "COVID" at the same time, while in the second case, only

"COVID" is referred to. After weighting, the concepts that meet the acceptance threshold defined are chosen (see step 7 in Table III). For this case, 70%, is the threshold, and the result is a list with the selected concepts (see step 8 in Table III).

With the base ontology integrated with the selected concepts (see step 9 in Table III), the LD paradigm is used to search for concepts that may be equivalent to the concepts of the generated ontology (see step 10 in Table III), in order to create new concepts in the ontology with external information. Fig. 4 shows the link between the COVID concept of the ontology and the COVID concept of DBpedia.

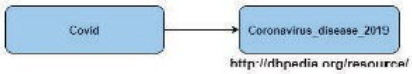

Fig. 4. Linking ontology concepts with external linked data sources.

Finally, the new ontology is transformed to the required format (see step 11 in Table III), in this case, to RDF/XML (see Fig. 5).

Fig. 5. Ontology generated and published in the RDF/XML format.

B. Case Study 2: Energy Management

B.1. Description. In this test, Energy Management has been taken as the knowledge domain, which is the set of processes and actions that allow optimizing energy consumption [17]. Table V shows the ontologies to be used:

TABLE V
ONTOLOGIES ON ENERGY MANAGEMENT

N	Ontology	Description	URL
1	ASPECT	This is a taxonomy of variables, parameters, and constants used in the wind energy community.	URL[1]
2	EXTRACT	Wind energy taxonomy of external conditions.	URL[2]
3	WEAR	Wind energy materials taxonomy.	URL[3]
4	WEAVE	Wind Energy Taxonomy of Activities.	URL[4]
5	DEM	Wind Energy Taxonomy of Models	URL[5]
6	REEGLE	Vocabulary used to describe clean energy actors, projects and technologies.	URL[6]
7	EEPSA	EEPSA (Energy Efficiency Prediction Semantic Assistant) Ontology aims to capture knowledge related to buildings, sensing and actuating devices, and their corresponding observations and actuations	URL[7]
8	SAREF4EE:	Is the EEbus/Energy@home extension of SAREF (Smart Applications REFerence) for the Smart Energy domain.	URL[8]
9	SAREF4SYST:	Is an extension of SAREF for the typology of systems and their interconnections (s4syst); defines Systems, Connections between systems, and Connection Points at which systems may be connected.	URL[9]
10	SEAS	SEAS (Smart Energy Aware Systems) project. It has the description of electricity measurements of a site using the Data Cube W3C vocabulary.	URL[10]
11	SEASBO	The SEAS Building ontology describes a taxonomy of buildings, building spaces, and rooms.	URL[11]
12	SEAS-SYS	This ontology is specialized in electric energy, power connections, etc.	URL[12]
13	TOPO	This ontology covers the topographical entities and administrative boundaries of a territory, such as railway lines, public and private transports, urban and rural roads, energy zones, municipality delimitations, etc.	URL[13]
14	IoT-O	IoT-O is a core domain of Internet of Things ontology. It is intended to model knowledge about IoT systems and applications.	URL[14]

URL[1]= https://bioportal.bioontology.org/ontologies/ASPECT/?p=summary
URL[2] = https://bioportal.bioontology.org/ontologies/EXTRACT
URL[3]= https://bioportal.bioontology.org/ontologies/WEAR/?p=summary
URL[4]= https://bioportal.bioontology.org/ontologies/WEAV
URL[5]= https://bioportal.bioontology.org/ontologies/IDEM/?p=summary
URL[6]= https://lov.linkeddata.es/dataset/lov/vocabs/reegle
URL[7]= https://lov.linkeddata.es/dataset/lov/vocabs/eepsa
URL[8]= https://lov.linkeddata.es/dataset/lov/vocabs/s4ener
URL[9]= https://lov.linkeddata.es/dataset/lov/vocabs/s4syst
URL[10]= https://lov.linkeddata.es/dataset/lov/vocabs/seas
URL[11] = https://lov.linkeddata.es/dataset/lov/vocabs/seasbo
URL[12] = https://lov.linkeddata.es/dataset/lov/vocabs/seas-sys
URL[13] = https://lov.linkeddata.es/dataset/lov/vocabs/topo
URL[14]=https://lov.linkeddata.es/dataset/lov/vocabs/ioto

B.2. Knowledge Management. It runs the KSM web service with each ontology to be added, this process is carried out by the Adds module (see step 3.2 in Table II).

B.3. Knowledge Generation. Run the OG web service using the *Search Terms* "Energy" and "Management" as parameters, the RDF/XML ontology format and an acceptance tolerance of 70%.

Then, it searches for synonyms of Energy and Management (see step 3 in Table III). For example, for Energy is found "power", "electricity", "voltage", "conductivity", among others. In the case of management, it found "administration", "operation", "control", "governance", among others.

Once the *Search Terms* and their synonyms are available, it proceeds to extract the ontological concepts from the ontologies in the system and weights each one of them (see steps 4, 5 and 6 in Table III). Then, it selects the concepts that exceed the acceptance threshold (see steps 7 and 8 in Table III). Table VI shows a small sample of the selected concepts, with their respective similarities to the *Search Terms*.

Now, our system proceeds to look for concepts using the LD paradigm that can be equivalent to the concepts of the generated ontology (see step 10 in Table III). For example, the concept "Management" is associated with "http://dbpedia.org/resource/Management", "Energy" with "http://dbpedia.org/resource/Energy", "Energy_management" with "http://dbpedia.org/resource /Energy_management","Total_electrical_energy_use_per_capita_(kWh/year)_(supporting_indicator)" with http://dbpedia.org /resource/Electric_energy_consumption, among others.

TABLE VI
SELECTED CONCEPTS AND THEIR SIMILARITY VALUE WITH RESPECT TO THE *SEARCH TERMS*

Concept	Energy	Management
Total_electrical_energy_use_per_capita_(kWh/year)_(supporting_indicator)	7.65	1.0
Average_number_of_electrical_interruptions_per_customer_per_year_(supporting_indicator)	6.02	1.0
Number_of_personal_automobiles_per_capita_(core_indicator)	2.28	2.11
Percentage_of_city_population_with_potable_water_supply_service_(core_indicator)	2.14	2.09

C. Ontology Evaluation Methods

This section presents three ontology evaluation methods [20, 21]:

- *Consistency Validation using Protégé:* to verify the consistency of the generated ontologies in terms of their hierarchical and axiomatic structure.
- *Application of Competence Questions:* to verify the amount of information that was obtained from the ontology sources and the concepts that were enriched by LD.
- *Quality Validation through Metrics:* to determine the precision, recall and F-measure of the alignment of the extracted concepts with respect to the *Search Terms* and of the concepts enriched with LD.

C.1. Consistency Validation using Protégé

The Protégé Pellet reasoner evaluates the consistency of the ontologies, verifying the problems of inconsistencies in the relationships of the classes, properties, and individuals present in the generated ontology by our system [22]. If any error occurs, then it stops the tests, and the reasoner is not activated. The reasoner checks the hierarchy of classes and properties, i.e., it checks the levels and dependencies between classes and properties. Furthermore, it checks the assertions of classes and properties for ambiguities between them.

C.2. Application of Competence Questions

Since there are no ontology inconsistency problems, the competency questions can be run for each ontology in Protégé. These questions are focused on the answers that a user wishes to verify when querying the ontology [23]. In this research, the competency questions check the number of concepts that are added to the ontology through the generator system. The competency questions are presented below:

- Q1: Concepts from ontological sources.
- Q2: Concepts from LD.
- Q3: Concepts linked with LD.
- Q4: Concepts associated with each *Search Term* from Ontologies.
- Q5: Concepts associated with each *Search Term* from LD.
- Q6: Concepts associated with each *Search Term* linked with LD.

C.3. Quality Validation through Metrics

The most common ontology quality validation metrics are based on the classical measures of Information Retrieval [24, 25], such as Precision, Recall, Accuracy, F-measure, among others. All these measures are based on the confusion matrix, which is divided into four groups: True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN). In this research, we will measure the precision, recall and F-measure of the generated ontologies by validating the quality of the aggregated concepts coming from ontology sources and LD. Precision metric is defined as the ratio between the number of correctly classified items and the total number of selected items [25], in other words, TP/(TP + FP). Recall is defined as TP/(TP + FN), and F-measure as (2 * Precision * Recall) / (Precision + Recall). On the other hand, to use this metric, it is necessary to have the correctly classified items. In our case, such items are manually selected and classified by an expert.

D. Ontology Evaluation

D.1. Consistency Validation using Protégé

Fig. 6 and Fig. 7 show the ontologies created in cases 1 and 2. In both figures, the Pellet reasoning of Protégé was activated to validate the consistency of the ontologies, and as shown in the terminal, it did not present consistency problems. It can also be observed in the figures, that the class hierarchies that compose each ontology are displayed with the option "Inferred". This indicates that the reasoner is active and without errors. Otherwise, "owl:Thing" would be displayed

without the class hierarchies that compose the ontology.

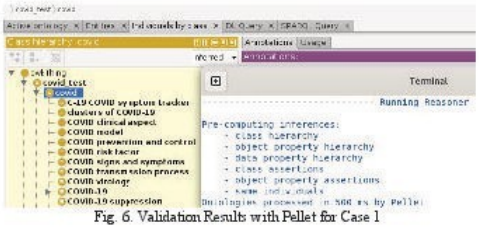

Fig. 6. Validation Results with Pellet for Case 1

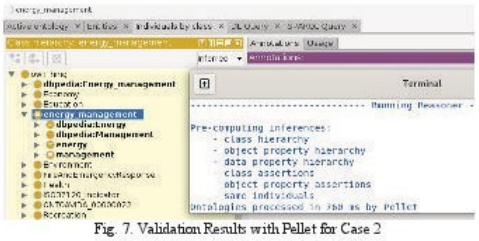

Fig. 7. Validation Results with Pellet for Case 2

Thus, in this case, Protégé does not detect any kind of inconsistency in the generated ontologies, and is capable of using them in inference processes that one wants to do with it and the built ontologies.

D.2. Application of Competence Questions

To perform this validation, the competency questions must be transformed to DL Query or SPARQL Query to be executed in Protégé. In our case, the Competency Questions will be transformed into SPARQL Query.

Table VII shows the results of the different queries executed for each ontology generated. For example, in case 1, Q1 shows that 105 concepts were obtained from the ontologies used as sources in the generating system. Q2 indicates that 80 concepts were obtained from the LD, this means that the resulting ontology integrated 80 concepts that were extracted from LD. Q3 indicates that these 80 concepts were linked to 52 concepts of the ontology. Regarding the same example, but associated to the search term COVID, Q4 indicates that 76 concepts were taken from the ontologies used as sources. Q5 indicates that 58 concepts were obtained from the LD and Q6 indicates that these 58 concepts were linked to 38 concepts of the ontology. This means that various LD concepts are associated with a concept of the ontology.

In case 2, the analysis is similar, only the number of concepts varies. For example, in Q1, 225 concepts were obtained from the ontologies that the system has in its knowledge base, and in Q2, 201 concepts were obtained from LD. In general, in case 2, it was possible to create an ontology with slightly more than twice as many concepts as in case 1.

TABLE VII
CONCEPTS OBTAINED IN EACH ONTOLOGY GENERATED ACCORDING TO THE COMPETENCY QUESTIONS

	CASE 1		CASE 2	
Q1	105		225	
Q2	80		201	
Q3	52		99	
Q4	**COVID**	76	**energy**	196
	test	56	**management**	184
Q5	**COVID**	58	**energy**	197
	test	54	**management**	180
Q6	**COVID**	38	**energy**	95
	test	27	**management**	87

D.3. Quality Validation through Metrics

Table VIII shows the results obtained with the Precision (P), Recall (R), and F-measure (F) metrics in the two generated ontologies. For this calculation, the precision, recall and F-measure are measured in two important processes within the generation., on the one hand, the concepts obtained through the ontologies stored in the knowledge base of the system, and on the other hand, the concepts obtained through LD. In addition, an expert is required to analyze each extracted concept to determine whether it matches the context requested by the client (see section IV.C.3).

TABLE VIII
RESULTS OF THE QUALITY VALIDATION OF THE GENERATED ONTOLOGIES

PROCESSES	CASE 1			CASE 2		
	P	R	F	P	R	F
Ontology source	0.90566	0.88889	0.89719	0.69524	0.82954	0.75647
LD	0.85519	0.90232	0.87812	0.74104	0.79413	0.76666
AVERAGE	**0,88042 5**	**0,89560 5**	**0,88765 5**	**0,71814**	**0,81183 5**	**0,76156 5**

In the first process, the concepts are obtained from ontology sources that the system has. Specifically, for case 1, a TP of 48 concepts (e.g. COVID-19, Tested for 2019-nCoV Wuhan infection, etc.) and FP of 5 concepts (e.g. data analysis operation, physical examination finding, among others) were obtained, which gives a Precision of 0.90566. Regarding Recall, an FN of 6 concepts (e.g., signs, cause, and mortality) was obtained, giving a Recall of 0.88889. Using Recall and Precision, an F-measure of 0.89719 was obtained. For case 2, a Precision of 0.69524 was achieved, since a TP of 73 (e.g. Energy Indicators, Energy consumption of public buildings per year kWh/m2, etc.) and FP of 32 (e.g. flow-head, logic operation, etc.) were obtained. With an FN of 15 concepts (e.g. Thermodynamics, Electricity generation, and Photovoltaic effect), a Recall of 0.82954, and an F-measure of 0.75647 were obtained.

In the second process, it enriches the concepts with information from the LD. In both cases are obtained very good results in the different metrics: Precision (case 1: 0.85519 and case 2: 0.74104), Recall (case 1: 0.90232 and case 2: 0.79413) and F-measure (case 1: 0.87812 and case 2: 0.76666). These precisions, recalls and F-measures were calculated by averaging the metrics obtained in the enrichment of each

concept of the ontologies, since each concept of the ontologies can be associated with several concepts arranged as LD.

Finally, the generated ontology is composed of two main processes: on the one hand, the concepts coming from the base ontologies of the system, and on the other hand, the concepts coming from LD. In other words, the metric of each generated ontology is the average of that metric in both processes. For case 1, it obtained a Precision of 0.880425, a Recall of 0.895605 and an F-measure of 0.887655. For case 2, it obtained a Precision of 0.71814, Recall of 0.811835 and F-measure of 0.761565. In both cases, the results were very good.

Something important to comment is that our approach is very easy to use in any application domain, it only depends on choosing good search terms to start the process. It is very important in domains where there are many data sources, such as the educational field [18], and in general, what the Internet of Things offers us [19].

V. ANALYSIS OF RESULTS

In this section, we analyze the behavior of our system in different contexts, discussing its advantages and limitations. In addition, we compare our approach with different works using the metrics that each system has.

A. Analysis of capacities and limitations of our approach

AOGS exhibits diverse practical applications, enabled by its automated ontology creation, use of base ontologies, and LD sources. For instance, in Natural Language Processing (NLP) Applications, AOGS can be used for sentiment analysis, chatbots, and text summarization, enabling precise and context-aware language processing. In the Biomedical Domain, AOGS creates up-to-date biomedical ontologies, benefiting medical research, drug discovery, and decision support systems. In Education and E-Learning, AOGS facilitates personalized learning experiences through content recommendation, and educational analytics on e-learning platforms. In the Information Retrieval, AOGS contributes to the Semantic Web, improving web content understanding and information retrieval for users. In Business and Industry, AOGS generates domain-specific ontologies, enhancing data integration and organizational knowledge management.

However, AOGS faces challenges that impact its effectiveness. For example, the quality of LD can affect system performance. Inaccurate or incomplete information from LD sources can introduce inconsistencies and affect ontology quality. Changes or unavailability of LD sources may lead to outdated or incomplete ontologies. Data privacy and security concerns arise from integrating sensitive information from LD sources into ontologies. Proactive measures, like data quality assessment and regular updates, and privacy safeguards, such as anonymization and data access controls, are essential to mitigate these threats and enhance the credibility and utility of AOGS-generated ontologies.

On the other hand, AOGS's dependence on LD sources may pose difficulties in integrating and comprehending ontologies in different languages, leading to potential inaccuracies or inconsistencies in the generated ontologies. Addressing the challenges associated with ontologies in languages other than English requires a combination of technical solutions, cross-cultural collaboration, and user-centric approaches. Moreover, the complexity of the generated ontologies could be a drawback, as large and intricate ontologies might be challenging to manage and interpret.

Furthermore, the evaluation of AOGS's performance is currently based on a limited set of evaluation metrics. Expanding the evaluation metrics and conducting more comprehensive testing would be beneficial in ensuring the reliability and effectiveness of AOGS.

B. Quantitative Comparison

In this section, we compare the results obtained with works in the literature. For this purpose, our approach uses the same metrics as other systems (see Table IX) presented in the papers [7], [10], and [12]. When comparing our work with respect to their metrics, it is observed that our F-Measure metric far exceeds the result of the rest of the works, with the exception of [12], where minimal differences are observed. For example, when compared with the metrics obtained in the generation of the ontology of case 1, our system gives better values in the Precision, Recall and F-Measure metrics, and when compared with our ontology generated in case 2, [12] gives better values in the Precision and F-Measure metrics.

TABLE IX
COMPARATIVE ANALYSIS WITH PREVIOUS WORK USING THE SAME METRICS AS OUR APPROACH

Work	Metrics		
	Precision	Recall	F-measure
[7]	x	x	0.58
[10]	x	x	0.65
[12]	0.83	0.75	0.78
Our AOGS's	case 1: 0.88 case 2: 0.71	case 1: 0.89 case 2: 0.81	case 1: 0.88 case 2: 0.76

VI. CONCLUSIONS

A system architecture was implemented to generate ontologies for a user-specified context. The architecture extracts concepts from two types of sources. Firstly, concepts are extracted from multiple base ontologies relevant to the required area. Secondly, concepts are extracted from LD sources to popular those obtained from base ontologies. Two macro-algorithms were developed to manage knowledge and generate new knowledge. Both algorithms operate within AOGS, which was customized into three layers: Knowledge Base Manager, Knowledge Generator Manager, and Web Services Manager.

AOGS has successfully generated acceptable ontologies in various domains, including energy and health, demonstrating progress in automating the creation and population of ontologies in these areas. Thus, AOGS has proven to be useful

in situations where a compact view is desired in a domain. In general, good precision, recall and F-measure were obtained in both cases.

While AOGS offers many benefits, including reducing the time and effort required to create an ontology and ensuring consistency and accuracy in the new ontology, there are also some limitations. These include its dependence on LD sources, the potential complexity of the generated ontologies, and the use of a limited set of evaluation metrics. Despite these limitations, AOGS represents an important step towards automating the process of ontology creation and could have many applications in areas.

In the context of future work, the focus is on enhancing and extending the capabilities of AOGS. One important aspect is the development and implementation of an ontological web application that allows external applications to consume and utilize the generated ontologies. This would increase the usability and accessibility of the ontologies produced by AOGS. Additionally, the creation of an automatic tool for indexing and comparing data sources is essential. This tool will aid in identifying similarities and correlations between different data repositories, facilitating the generation of ontologies that link information from diverse sources in a unified structure.

REFERENCES

[1] R. Dos Santos and J. Aguilar, "Enlazado de Datos," in *Introducción a la Minería Semántica* (J. Aguilar, Ed.) Fondo Editorial Universidad Nacional Experimental del Táchira, pp. 177-216, 2018, ISBN: 978-980-426-008-7.

[2] T. Rodriguez, R. Dos Santos, and J. Aguilar, "Metodología para el desarrollo de Aplicaciones Web utilizando Datos Enlazados," in Proc. *Conferencia Nacional de Computación, Informática y Sistemas*, pp. 114-122, I, 2017, SBN: 978-980-7683-03-6.

[3] T. Berners-Lee et al., "Tabulator: Exploring and analyzing Linked Data on the Semantic Web," in Proc. 3rd *International Semantic Web User Interaction Workshop*, 2006, pp. 159-174.

[4] R. Dos Santos, J. Aguilar, and E. Puerto, "A Meta-Learning Architecture based on Linked Data," in Proc. *XLVII Latin American Computing Conference* (CLEI), 2021.

[5] I. Osman, S. B. Yahia, and G. Diallo, "Ontology integration: approaches and challenging issues," *Information Fusion*, vol. 71, pp. 38-63, 2021.

[6] A. Smirnov and T. Levashova, "Knowledge fusion patterns: A survey," *Information Fusion*, vol. 52, pp. 31-40, 2019.

[7] M. Alobaidi, K. M. Malik, and S. Sabra, "Linked open data-based framework for automatic biomedical ontology generation," *BMC bioinformatics*, vol. 19, no. 1, 2018.

[8] S. Straujums, "Towards ONTO6 Framework for Concept Elicitation," *CEUR-WS.org/*, vol. 3158, 2021.

[9] S. Sbai, O. Chabik, M. R. C. Louhdi, H. Belja, E. M. Zemmouri, and B. Trousse, "Using decision trees to learn ontology taxonomies from relational databases," in Proc. *6th IEEE Congress on Information Science and Technology* (CiSt), pp. 54-58, 2020.

[10] C. Musto, T. Franza, G. Semeraro, M. de Gemmis, and P. Lops, "Deep content-based recommender systems exploiting recurrent neural networks and linked open data," in Pro. *26th Conference on User Modeling, Adaptation and Personalization*, pp. 239-244, 2018.

[11] S. Lamine, M. Dachraoui, and H. Baazaoui-Zghal, "Deep learning-based extraction of concepts: A comparative study and application on medical data," *Journal of Information & Knowledge Management*, vol. 22, no. 5, 2250072, 2022.

[12] N. Mellal, T. Guerram, and F. Boulahlassa, "An approach for automatic ontology enrichment from texts," *Informatica*, vol. 45, no. 1, 2021.

[13] Y. He, J. Chen, D. Antonyrajah, and I. Horrocks, "BERTMap: a BERT-based ontology alignment system," in Proc. *AAAI Conference on Artificial Intelligence*, vol. 36, no. 5, pp. 5684-5691, 2022.

[14] S. Kirrane and K. Kozaki, "Extracting domain-specific concepts from large-scale linked open data," in Proc. *10th International Joint Conference on Knowledge Graphs*, 2021, pp. 28-37.

[15] A. González-Eras, R. Dos Santos, J. Aguilar, and A. Lopez, "Ontological engineering for the definition of a COVID-19 pandemic ontology," *Informatics in Medicine Unlocked*, vol. 28, 100816, 2022.

[16] J. Vizcarrondo, J. Aguilar, E. Exposito and A. Subias, "ARMISCOM: Autonomic reflective middleware for management service composition," Proc. *Global Information Infrastructure and Networking Symposium (GIIS)*, 2012,

[17] J. Aguilar, A. Garcés-Jiménez, N. Gallego-Salvador, J. De Mesa, J. Gomez-Pulido and A. Garcia-Tejedor, "Autonomic Management Architecture for Multi-HVAC Systems in Smart Buildings," *IEEE Access*, vol. 7, pp. 123402-123415, 2019.

[18] M. Sánchez, J. Aguilar, J. Cordero, P. Valdiviezo-Diaz, L. Barba-Guamán, L. Chamba-Eras, "Cloud Computing in Smart Educational Environments: Application in Learning Analytics as Service". In: New Advances in Information Systems and Technologies (Rocha, Á., Correia, A., Adeli, H., Reis, L., Mendonça Teixeira, M., eds.), vol. 444. Springer, pp. 993–1002, 2016.

[19] L. Morales, C. Ouedraogo, J. Aguilar, C. Chassot, S. Medjiah, K. Dinia. "Experimental comparison of the diagnostic capabilities of classification and clustering algorithms for the QoS management in an autonomic IoT platform". *Service Oriented Computing and Applications*, vol. 13, pp. 199–219, 2019.

[20] J. Bandeira, I. Bittencourt, P. Espinheira, and S. Isotani, "A methodology for ontology evaluation," Cornell university, Tech. Rep., 2016. [Online]. Available: http://arxiv.org/abs/1612.03353

[21] J. Miao and W. Zhu, "Precision-recall curve (PRC) classification trees," *Evolutionary Intelligence*, vol. 15, no. 3, pp. 1545-1569, 2022.

[22] J. Ding, J. Zhang, Z. Zhan, X. Tang, and X. Wang, "A precision efficient method for collapsed building detection in post-earthquake UAV images based on the improved NMS algorithm and Faster R-CNN," *Remote Sens.*, vol. 14, no. 3, 663, 2022.

[23] M. Musen, "The Protégé project: A look back and a look forward," *AI Matters*, vol. 1, no. 4, pp. 4-12, Jun. 2015.

[24] S. Ibrikich, A. Oussous, O. Ibrikich, and M. Esghir, "A Review on recent research in information retrieval," *Procedia Computer Science*, vol. 201, pp. 777-782, 2022.

[25] R. Kumar and S. C. Sharma, "Hybrid optimization and ontology-based semantic model for efficient text-based information retrieval," *The Journal of Supercomputing*, vol. 79, no. 2, pp. 2251-2280, 2023.

8.4 Anexo 3.C: Arquitectura para la Creación y Enriquecimiento Automático de Ontologías a partir de Datos Enlazado

Mundo Fesc 12 (24) pp. 8-15, Julio-Diciembre 2022, ISSN 2216-0353, 2216-0388

Arquitectura para la Creación y Enriquecimiento Automático de Ontologías a partir de Datos Enlazados

Architecture for the Automatic Creation and Enrichment of Ontologies from Linked Data

[a].Ricardo José Dos Santos-Guillén, [b].Eduard Gilberto Puerto-Cuadros, [c]. Jose Lisandro Aguilar

a. PhD(c). en Ciencias Aplicadas, Centro de Estudios en Microelectrónica y Sistemas Distribuidos (CEMISID), ricardojdsg@gmail.com, Universidad de los Andes, Mérida, Venezuela.

b. PhD. en Ciencias Aplicadas, Facultad de Ingeniería, Departamento de Sistemas e Informática, Grupo GIA, eduardpuerto@ufps.edu.co Universidad Francisco de Paula Santander, Cúcuta -Colombia.

c. PhD. Ciencias de la Computación Centro de Estudios en Microelectrónica y Sistemas Distribuidos (CEMISID), aguilarjos@gmail.com, Universidad de los Andes, Mérida, Venezuela.

Recibido: Julio 1 de 2021 Aceptado: Noviembre 8 de 2021

Forma de citar: Ricardo José Dos Santos-Guillén, b E.G Puerto-Cuadros J.I Aguilar. "Arquitectura para la Creación y Enriquecimiento Automático de Ontologías a partir de Datos Enlazados.", *Mundo Fesc, vol. 12 (24), pp. 8-24, 2022*

Resumen

En los ambientes inteligentes (p.ej. ciudades inteligentes) se presentan problemas muy diversos que deben ser atendidos inmediatamente, por lo tanto, estos ambientes deben generar conocimiento que les permitan responder a esas necesidades particulares, aprovechando la información del contexto (ej.: datos históricos, medición de sensores, descripción del problema, entre otras). Como propuesta para la generación oportuna de estas bases de conocimiento, en este trabajo se desarrolla una arquitectura que permite crear y enriquecer ontologías emergentes de forma autónoma, usando como insumo el paradigma de Datos Enlazados, que son estructuras de datos que se vinculan unos con otros para servir tanto a usuarios humanos como a otros sistemas dentro de la web semántica. En este trabajo se especifican los servicios que ofrecen la capacidad de generar ontologías emergentes según el problema que se presente, buscando aprender los distintos axiomas (conceptos o propiedades) de las estructuras que posean las fuentes de Datos Enlazados empleada. Luego, estas ontologías creadas son enriquecidas/pobladas con la información que se extraiga desde las distintas fuentes de datos enlazados, para ser devueltas al sistema solicitante de forma automática. Para el desarrollo de este trabajo se usa la metodología MEDAWEDE, que permite especificar esta arquitectura, y guiar la construcción de los servicios para consumir las fuentes descritas como datos enlazados. De dicha metodología se utilizan sus seis etapas: i. Especificación: para analizar y seleccionar las fuentes de datos ontológicas a usar. ii. Modelado: para implementar la base de datos ontológica del área de estudio de las ontologías a generar. iii. Generación: para realizar el proceso de transformación, filtrado e integración de los datos en la ontología generada. iv. Vinculación: para asociar los datos de la ontología generada con el conocimiento proveniente de los conjuntos de datos enlazados. v. Publicación: para poner a disposición la ontología generada en los distintos formatos estándares. vi. Explotación: esta etapa se considera porque está dedicada a definir interfaces para acceder a la ontología. Finalmente, el artículo presenta un caso de estudio enfocado en mostrar el proceso de generación de una ontología emergente en el área de la pandemia del Covid-19, donde se aprovecha la información disponible en la Internet a través de ontologías preexistentes y fuentes de datos enlazados.

Palabras clave: Ontología Emergente, Arquitectura, Datos enlazados, Alineación, Servicio Web.

Autor para correspondencia:
*Correo electrónico: ricardojdsg@gmail.com

Abstract
In smart environments (e.g.: smart cities) there are very diverse problems that must be addressed immediately; therefore, these environments must generate knowledge that allows them to respond to those particular needs, taking advantage of information from the context (e.g.: historical data, sensor measurement, problem description, among others). As a proposal for the timely generation of these knowledge bases, in this work an architecture is developed for allows to create and enrich emerging ontologies autonomously, using as input the linked data paradigm, which are data structures that are linked to each other, to serve both human users and other systems within the semantic web. This work specifies the services that offer the ability to generate emerging ontologies according to the problem that arises, seeking to learn the different axioms (concepts or properties) of the structures that the linked data sources used have. Then, these created ontologies are enriched/populated with the information that is extracted from the different linked data sources, to be returned to the requesting system automatically. For the development of this work, the MEDAWEDE methodology is used, which allows specifying this architecture, and conducting the construction of services to consume the sources described as linked data. Its six stages are used from this methodology: i) Specification: to analyze and select the ontological data sources to use. ii) Modeling: to implement the knowledge model of the study area of the ontologies to be generated. iii) Generation: to carry out the process of transformation, filtering, and integration of the data in the generated ontology. iv) Linking: to associate the data from the generated ontology with the knowledge from the linked data sets. v) Publication: to make the generated ontology available in the different standard formats. The exploitation stage is considered because it is dedicated to defining interfaces to access the ontology. The article also presents a case study focused on showing the process of generating an emerging ontology in the area of the Covid-19 pandemic, where the information available on the Internet is used through pre-existing ontologies and linked data sources.

Keywords: Migration, immigration, social work, human rights, border.

Introducción

La integración del conocimiento a partir de fuentes dispersas de datos, es un problema que se tiene para el funcionamiento oportuno y eficaz de Ambientes Inteligentes (AmI) a la hora de tomar decisiones, dado que requieren de conocimiento que le permitan garantizar que la respuesta sea acorde al dominio que se les plantea. Para ello, estos AmI requieren de datos que generalmente son tomados desde sensores dispuestos dentro del ambiente. Sin embargo, dentro de estos AmI se requieren, a su vez, información histórica relevante; es decir, aquella que los sensores han recolectado con antelación o que está albergada en fuentes de datos externas, como son las Ontologías y los Datos Enlazados.

Según [1],[2],[3] una ontología es la representación del conocimiento sobre un dominio específico, donde un conjunto de objetos y relaciones interactúan de forma consistente y coherente. Es decir, un conjunto de axiomas lógicos diseñados para explicar el significado previsto de un vocabulario que pueden ser procesado por máquinas para usarse en algunos tipos de aplicaciones [4], [5], [6],[7]. Así mismo, los Datos Enlazados, según [8] describen una forma de publicar los datos estructurados para que se puedan interconectar entre ellos. Los autores, a su vez citando a [9] añaden que los Datos Enlazados describen un conjunto de prácticas para publicar, compartir y conectar piezas de datos, información y conocimiento en la Web Semántica, usando identificadores URIs y RDF para describir los recursos publicados.

Por otra parte, dado que estos AmI son un entorno donde coexisten las personas con las máquinas de una manera cómoda e imperceptible y estas se ponen a su

servicio para facilitarles la vida y hacerlas más agradable, en donde convergen computadores ubicuos incrustados en objetos cotidianos, comunicaciones inalámbricas entre ellos, interfaces de nueva generación, sensores biométricos, agentes inteligentes, sistemas de personalización, entre otras cosas, se requieren de Servicios Web dispuestos a responder las necesidades que se van generando en estos ambientes.

Los Servicios Web son aplicaciones auto contenidas y auto descriptivas que pueden ser alojadas, publicadas e invocadas a través de la web [10], [11]. Con esto, los autores definen un servicio web como un objeto de software que puede ser ejecutado a través de internet usando protocolos estandarizados, con el fin de cumplir tareas, funciones o ejecutar procesos completos de negocio. Por otro lado, sirviendo de complemento a la definición anterior, en [12] explican que los servicios web, y en general las arquitecturas orientadas a servicios, son tecnologías que emergieron como elección al querer implementar sistemas distribuidos a través de la modularidad. Gracias a esto, el negocio se divide en pequeñas funciones especializadas que se publican como servicios.

Actualmente, existen diversos trabajos orientados a responder a estos y otros problemas relacionados con los AmI [13]. El primero de estos trabajos [14] propone un nuevo método de enriquecimiento de ontologías a partir de extracción de conceptos. Este proceso semi automático puede exportar datos enlazados minando textos, sin importar su lenguaje, estilo o dominio basado en redes neuronales [15]. Por otro lado, otro trabajo [4] detalla el uso de datos abiertos para impulsar la innovación y el crecimiento económico de ciudades inteligentes; por lo que proyectan una arquitectura que sirve como mediador entre la explotación de datos abiertos y la generación de conocimiento. [16] destaca la importancia que tiene la metadata dentro del mundo de los datos enlazados al momento de publicar datos enlazados de manera completa y consistente. En otro orden de ideas, con una temática similar a la del proyecto actual, los autores de [17] muestran la importancia para manejar los datos enteros de una ciudad en forma de datos enlazados. Proponen una metodología para lograr la homogeneidad de los datos, utilizando técnicas de regresión como estrategia para depurar los datos incluyendo datos enlazados integrados. Así mismo, en [18] proponen crear y poblar ontologías a partir de textos no estructurados. Su metodología consiste en la utilización del contexto para mejorar la comprensión del texto de entrada basándose en la búsqueda y análisis de grandes conjuntos de datos enlazados. Finalmente, en [19] se propone un framework para la integración de ontologías, donde se identifican términos comunes para extraer datos similares de distintas fuentes, para relacionarlos dentro de una misma ontología.

Se plantea así la hipótesis que aprovechando los principios de desarrollo de la web 3.0 es posible hacer que este conocimiento sea de fácil acceso para los sistemas inteligentes nuevos y actuales. Para lo cual se genera la siguiente interrogante ¿Cómo desarrollar un servicio web que permita crear y enriquecer ontologías de forma autónoma, usando fuentes de datos enlazados para que sirva para la toma de decisiones en AmI? ¿De qué manera

generar ontologías de forma automática, a partir de datos enlazados en fuentes públicas?.

A continuación, el artículo en su Sección de Materiales y métodos se describe el diseño de la arquitectura computacional propuesta para la creación y enriquecimiento automático de ontologías a partir de datos enlazados siguiendo la metodología MEDAWEDE. En la sección de Resultados y discusión, se presenta un caso de entorno en el dominio del Covid-19, donde se presenta el uso de la arquitectura. Finalmente se dan las Conclusiones en torno a los resultados.

Materiales y métodos

El diseño arquitectónico de la propuesta se basa en la metodología MEDAWEDE [20], que permite especificar y guiar la construcción de los servicios para consumir las fuentes descritas como datos enlazados. La metodología tiene seis etapas: i. Especificación: para analizar y seleccionar las fuentes de datos ontológicas a usar. ii. Modelado: para implementar la base de datos ontológica del área de estudio de las ontologías a generar. iii. Generación: para realizar el proceso de transformación, filtrado e integración de los datos en la ontología generada. iv. Vinculación: para asociar los datos de la ontología generada con el conocimiento proveniente de los conjuntos de datos enlazados. v. Publicación: para poner a disposición la ontología generada en los distintos formatos estándares. vi. Explotación: esta etapa se considera porque está dedicada a definir interfaces para acceder a la ontología.

La arquitectura de la solución está constituida por tres capas (ver figura 1). La primera capa, denominada Servidor Web, se encarga de recibir las peticiones web que realicen los clientes. La segunda capa, Explorador de Recursos, consulta los diferentes repositorios de Datos Enlazados a partir del dominio solicitado por el cliente. La tercera capa, Generador Ontológico, procesa los datos de la capa anterior y genera una ontología extendida con Datos Enlazados según el conocimiento solicitado.

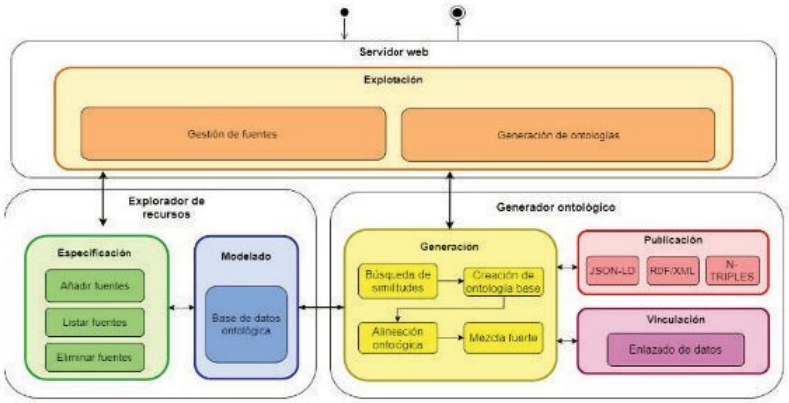

Figura 1 Diagrama de componentes de nuestra arquitectura basada en MEDAWEDE.

A continuación, se describe con más detalle cada uno de los componentes de las capas de la arquitectura.

Capa Servidor Web.
Esta capa está compuesta por las interfaces Gestión de Fuentes y Generación de Ontologías. La primera interfaz de Gestión de Fuentes tiene como objetivo ofrecer tres servicios: añadir, eliminar y listar las fuentes de datos. La segunda interfaz tiene por objetivo ofrecer el servicio que genera la ontología enriquecida con Datos Enlazados explotando el conocimiento dispuesto por la primera interfaz.

Capa Explorador de recursos.
La capa del Explorador de recursos está constituida por los componentes de Especificación y modelado. En el componente de Especificación se gestionan las ontologías como fuentes de conocimiento, donde se seleccionan y consultan los conjuntos de datos en los repositorios disponibles ya sea localmente o desde Internet. En el componente de Modelado se genera una Base de Datos Ontológica (BDO) a partir del conjunto de ontologías del componente anterior.

Capa Generador ontológico. Esta capa está conformada por tres componentes: El componente de Generación, de Vinculación y de Publicación (ver Figura 2). En Generación, se consultan conceptos ontológicos a partir del dominio solicitado, para poblar una ontología nueva que será el resultado de la alineación y mezcla del conjunto de ontologías de la BDO que correspondan a la consulta.

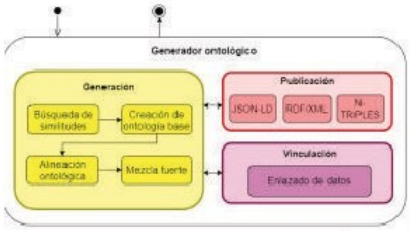

Figura 2. Componentes Capa Generador Ontológico.

A continuación, se describe con más detalle el proceso: Un primer paso consiste en la búsqueda de similitudes, donde se reciben los términos de búsqueda que se usan como criterio para la consulta a la BDO y se seleccionan los conceptos ontológicos que contienen estos términos.

El siguiente paso es la creación de la ontología base, que corresponde a una ontología nueva que contiene únicamente las clases de la búsqueda. Esta ontología albergará también los conceptos que sean seleccionados en los siguientes pasos.

A continuación, sigue el proceso de alineación ontológica. Durante este paso se realiza el preprocesado de los conceptos ontológicos obtenidos en la búsqueda, buscando ponderar su relación con cada uno de los términos de búsqueda. Estas ponderaciones ayudan al filtrado de los conceptos que no corresponden a la alineación ontológica, buscando asegurar la calidad de la ontología generada. El paso final durante el proceso de generación es la mezcla fuerte. Durante este se copian los conceptos seleccionados de las ontologías de la BDO y se integran dentro de la ontología base, obteniendo como resultado una ontología generada y poblada que corresponde al conocimiento del dominio solicitado en la búsqueda.

En Vinculación se enriquece la ontología generada en el componente anterior con Datos Enlazados. Para ello, se usa el servicio de búsqueda de DBpedia (https://lookup.dbpedia.org/api/search/) para enriquecer con información de DBpedia a cada concepto de la ontología. En Publicación se transforma la ontología enriquecida con Datos Enlazados al formato requerido por la interfaz. Entre los formatos que actualmente maneja son JSON-LD, RDF/XML y N-TRIPLES. A continuación, se desarrolla un caso de estudió a partir del cual se evalúa la arquitectura.

Resultados y Discusión

Para el desarrollo y prueba de la arquitectura propuesta (ver figura 1) se ha tomado como fuente o insumo de conocimiento el marco ontológico de sobre el dominio del Covid-19. Para esto se ha seleccionado un conjunto de ontologías vinculadas al Covid-19 del repositorio de ontologías https://bioportal.Bioontology.Org/ontologies (ver Tabla 1).

Tabla I. Fuentes de Conocimiento Ontológicas sobre Covid-19

N	Ontología	Descripción	URL
1	COviD-19 Ontology for Cases and Patient information (CODO)	Contiene conocimiento relacionado con el paciente y el Covid-19 especificado en 52 conceptos	https://bioportal.bioontology.org/ontologies/CODO
2	COVID-19 Surveillance Ontology (COVID19)	Contiene conocimiento relacionado con el control, vigilancia del Covid-19 especificado en 52 conceptos.	https://bioportal.bioontology.org/ontologies/COVID19
3	The COVID-19 Infectious Disease Ontology (IDO-COVID-19)	Contiene conocimiento relacionado con enfermedades infecciosas entorno al Covid-19 especificado en 486 Conceptos	https://bioportal.bioontology.org/ontologies/IDO-COVID-19
4	COVID-19 Ontology (COVID-19)	Contiene conocimiento general del Covid especificado en 2286 conceptos	https://bioportal.bioontology.org/ontologies/COVID-19
5	WHO COVID-19 Rapid Version CRF semantic data model (COVIDCRFRAPID)	Contiene conocimiento sobre datos clínicos estandarizados sobre Covid-19 especificado en 398	https://bioportal.bioontology.org/ontologies/COVID-CRFRAPID

Con estas fuentes de datos se establece la especificación y el modelado de la base de datos ontológica del Servicio Web. A continuación, se ha hecho la solicitud al servicio web que genere una ontología en formato RDF/XML acerca de los términos de búsqueda "Test" y "Covid".

Para llevar a cabo el paso de Generación, la aplicación ha debido buscar en diversas fuentes sinónimos de cada término de búsqueda. Por ejemplo, para "Test" ha encontrado sinónimos como "trial, exam, quiz" entre otros, mientras que para "Covid" ha encontrado "Coronavirus". Cabe aclarar en este punto que el caso de prueba se realizó en inglés por ser el idioma de las ontologías utilizadas. El sistema podrá ser expandido en un futuro para funcionar de la misma manera con otros idiomas.

Continuando con el proceso, se utiliza la librería de Python OlwReady 2 para realizar consultas sobre la base de datos ontológica, donde se obtienen

conceptos ontológicos de cualquiera de las ontologías integradas, donde sus etiquetas contengan cualquiera de los términos de búsqueda o sus sinónimos. Entre estos resultados destacan algunos como "Untested for Covid-19", "Covid-19 Diagnosis", "Tested for 2019-nCov (Wuhan) infection".

Sin embargo no todos estos conceptos ontológicos terminarán poblando la ontología, puesto que muchos de ellos no guardan relación semántica con el dominio que se consultó, como es el caso de resultados como "intestine cancer", "assay screened entity", "testis"; por lo tanto es necesario realizar un filtrado entre los conceptos. Su objetivo es ponderar cada concepto ontológico obtenido en la consulta de acuerdo a su relación con los términos de búsqueda. Para esto se preprocesan y comparan sus etiquetas simplificadas, otorgando mayor puntaje si el concepto coincide con varios términos de la búsqueda a la vez.

Contextualizando este paso al caso de prueba, un concepto cuya etiqueta sea "Tested for covid-19" tendrá una puntuación mayor a uno cuya etiqueta sea "Study for covid propagation", ya que en el primer caso se hace referencia a "Test" y a "Covid" a la vez, mientras que en el segundo sólo se hace referencia a "Covid". De la misma forma, un concepto etiquetado como "intestine cancer" obtendrá una puntuación muy baja ya que sólo se relaciona con "Test" como una subcadena dentro de uno de sus términos ("intestine")

Esto se aprecia mejor en la figura 3 donde se ilustran algunas comparaciones entre diversos tokens provenientes del preprocesado de las etiquetas de conceptos ontológicos distintos (Azul) con los conceptos de búsqueda y sus sinónimos (Verde oscuro) El primer caso, la comparación de "covid" como token en el concepto evaluado con "Covid" como término de búsqueda es una coincidencia perfecta, por tanto su puntuación será máxima (1). En el segundo caso se observa que el término "Test" está inmerso parcialmente dentro del token "untest" con un sufijo o prefijo añadido, por esto su puntuación es la mitad de la de las coincidencias perfectas (0.5). En el caso de que el término de búsqueda esté inmerso completamente como una subcadena del token del concepto, siendo el caso de "Test" e "intestin", la ponderación es de un cuarto de la coincidencia perfecta (0.25), buscando castigar posibles conceptos sin relación semántica con la búsqueda. En caso de los sinónimos de los términos de búsqueda (verde claro) se utiliza el mismo criterio, pero la puntuación obtenida se reducirá a la mitad.

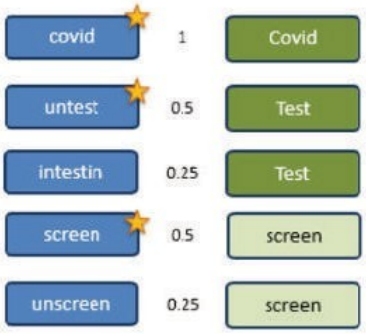

Figura 3. Criterios de ponderación al comparar conceptos ontológicos dentro de los filtros de búsqueda.

Después de la ponderación se escogen los conceptos que cumplan con rangos de puntajes, teniendo en cuenta el umbral de aceptación, que es un porcentaje escogido por quien solicita la ontología, que define qué tan severo o permisivo es

el filtro de los conceptos. Finalmente, habiendo concluido la alineación ontológica, sigue el subproceso de mezcla fuerte, donde todos aquellos conceptos que superen los filtros se integran dentro de la ontología base recién creada.

Concluido esto, puede procederse al paso de Vinculación de la metodología, donde se buscan conceptos en los Datos Enlazados que puedan ser equivalentes a los de la ontología generada, permitiendo vincularse de forma externa, mientras que el razonador interno de manera automática busca equivalencias y relaciones entre los conceptos de la propia ontología. En el caso del ejemplo, la búsqueda de recursos en dbpedia respecto a "Covid" ha permitido la vinculación entre el concepto covid de la ontología generada (covid_test#covid) con el concepto Coronavirus_disease_2019 de dbpedia (http://dbpedia.org/resource#Coronavirus_disease_2019)

Una vez la ontología se ha generado y vinculado, puede publicarse en el formato seleccionado, en este caso RDF/XML. La figura 4 muestra una representación gráfica (http://www.visualdataweb.de/webvowl/) de la ontología generada resultante del caso de prueba, haciendo énfasis en los conceptos raíz de la búsqueda ("covid", "test", "covid_test") y su relación con los conceptos obtenidos en la búsqueda.

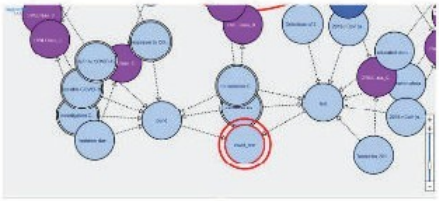

Figura 4. Fragmento de los Conceptos filtrados y vinculados en la ontología generada en el caso de prueba.

Conclusiones

El artículo presenta un servicio web enriquecedor de ontologías que se encargue de poblar ontologías con nuevos conceptos y relaciones, a partir de Datos Enlazados de forma automática. Un servicio que facilita la integración del conocimiento a partir de fuentes dispersas de datos enlazados, aprovechando los principios de desarrollo de la web 3.0 de fácil acceso para los sistemas inteligentes nuevos y actuales que lo soliciten. Se ofrece una solución la cual es desplegada en forma de servicio web, para ser utilizado de forma remota.

El algoritmo propuesto para la creación de ontologías permite que la respuesta sea acorde al dominio que se le plantea, a través de filtros en la extracción e integración de datos desde repositorios dispersos y otras ontologías existentes; todo con el fin de que el conocimiento generado sea aprovechado al máximo por los sistemas siguientes dentro del ciclo de vida de los datos.

Se ofrece una solución capaz de servir, de forma general, a arquitecturas no supervisadas, de sistemas inteligentes que puedan consumir este servicio web sin necesidad de participación humana. De ahí que la arquitectura propuesta tenga propiedades de autonomía, para garantizar una respuesta óptima, que no

se obtendría al existir intervención humana en el proceso de selección de datos o de integración del conocimiento.

Se realizó un caso de prueba de enriquecimiento en el dominio de COVID para consumir este servicio es el middleware MiSCi, propuesto por Dos Santos (2020).

Cabe notar que el sistema desarrollado no servirá como repositorio para albergar ontologías y/o conjuntos de datos enlazados. El sistema será alojado en un servidor de pruebas y su despliegue final no contempla alojamiento en producción ni nombre de dominio. El proyecto utiliza como datos fuentes de Datos Enlazados.
Siguiendo metódicamente la arquitectura diseñada, se especificó un caso de estudio enfocado al dominio del Covid-19, donde se obtuvo de manera automática una ontología enriquecida con Datos Enlazados para este dominio. La arquitectura mostró que se pueden obtener ontologías no supervisadas lo cual permite resolver situaciones emergentes de manera automática en los AmI, permitiendo la interoperabilidad y razonamiento entre los diferentes componentes del AmI.

Es de notar que en la alineación se obtuvo algunas clases disjuntas que no tienen relaciones fuertes con los demás conceptos, pero aun así obtuvieron la puntuación suficiente para superar los filtros o umbrales. En la revisión del enlazado de conceptos automático con DBpedia, por su parte, se presentaron inconsistencias semánticas. Estas debilidades serán abordadas en trabajos futuros.

Referencias

[1] T. R. Gruber, N. Guarino y R. Poli, "Formal ontology in conceptual analysis and knowledge representation," capítulo "Towards principles for the design of ontologies used for knowledge sharing" en Conceptual Analysis and Knowledge Representation, 1993.

[2] I. Osman, S. B. Yahia, and G. Diallo, "Ontology integration: approaches and challenging issues," Information Fusion, vol. 71, pp. 38-63, 2021.

[3] A. González-Eras, R. Dos Santos, J. Aguilar, and A. Lopez, "Ontological engineering for the definition of a COVID-19 pandemic ontology," Informatics in Medicine Unlocked, vol. 28, 100816, 2022.

[4] O. El Bacha, O. Jmad, Y. El Bouzekri El Idrissi y N. Hmina, "Exploiting Open Data to Improve the Business Intelligence & Business Discovery Experience". no 27 In Proceedings of the 2nd international Conference on Big Data, Cloud and Applications, (New York), pp 1–6, March 2017.

[5] A. Degbelo, "A snapshot of ontology evaluation criteria and strategies," in Proceedings of the 13th International Conference on Semantic Systems, septiembre de 2017, pp. 1-8.

[6] N. Guarino (Ed.), Formal ontology in information systems: Proceedings of the first international conference (FOIS'98), June 6-8, Trento, Italy, vol. 46, IOS press, 1998.

[7] N. Piedra, J. Chicaiza, E. Cadme y R. Guaya, "Una aproximación basada en Linked Data para la detección de

potenciales redes de colaboración científica a partir de la anotación semántica de producción científica: Piloto aplicado con producción científica de investigadores ecuatorianos," Maskana, vol. 5, 2014.

[8] T. Rodríguez, R. D. Santos y J. Aguilar, "Metodología para el desarrollo de aplicaciones Web utilizando datos enlazados," in Conferencia Nacional de Computación, Informática Y Sistemas (CoNCISa 2017), vol. 5, pp. 978-980, 2017.

[9] O. Hartig y A. Langegger, "A database perspective on consuming linked data on the web," Datenbank-Spektrum, vol. 10, pp. 57-66, octubre de 2010.

[10] D. Fensel y C. Bussler, "The web service modeling framework WSMF," Electronic Commerce Research and Applications, vol. 1, no. 2, pp. 113-137, 2002.

[11] F. H. Vera-Rivera, E. Puerto, H. Astudillo y C. M. Gaona, "Microservices backlog–A genetic programming technique for identification and evaluation of microservices from user stories," IEEE Access, vol. 9, pp. 117178-117203, 2021.

[12] B. Benatallah, F. Casati, and F. Toumani, "Analysis and Management of Web Service Protocols," in P. Atzeni, W. Chu, H. Lu, S. Zhou, and T. W. Ling (Eds.), Conceptual Modeling – ER 2004, vol. 3288, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg.

[13] M. Sánchez, J. Aguilar y E. Exposito, "Integración SOA-MAS en ambientes inteligentes," Dyna, vol. 85, no. 206, pp. 268-282, 2018.

[14] A. Alba, A. Coden, A. L. Gentile, D. Gruhl, P. Ristoski y S. Welch, "Multi-lingual concept extraction with linked data and human-in-the-loop", no 24 In *Proceedings of the Knowledge Capture Conference*, pp. 1-8, (New York), December 2017.

[15] J. M. V. Carrero y E. Puerto, "GeoMotor: Design with nature. Recognition of geometries using a convolutional neural-network approach (CNN)," in ICGG 2020-Proceedings of the 19th International Conference on Geometry and Graphics, Springer International Publishing, 2021, pp. 916-919.

[16] A. Dimou, T. De Nies, R. Verborgh, E. Mannens, P. Mechant y R. Van de Walle, "Automated metadata generation for Linked Data generation and publishing workflows," in Proceedings of LDOW2016, CEUR-WS.org, 2016, pp. 1-10.

[17] S. Bischof, C. Martin, A. Polleres y P. Schneider, "Collecting, integrating, enriching and republishing open city data as linked data," in The Semantic Web-ISWC 2015: 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part II, vol. 14, pp. 57-75, Springer International Publishing.

[18] M. Booshehri y P. Luksch, "Towards adding linked data to ontology learning layers," in Proceedings of the 16th International Conference on Information

Integration and Web-based Applications & Services, diciembre de 2014, pp. 401-409.

[19] L. Zhao y R. Ichise, "Ontology integration for linked data," Journal on Data Semantics, vol. 3, pp. 237-254, 2014.

[20] R. Dos Santos, J. Aguilar, and E. Puerto, "A Meta-Learning Architecture based on Linked Data," in Proc. XLVII Latin American Computing Conference (CLEI), 2021.

8.5 Anexo 4.A: A hybrid recommender system based on description/dialetheic logic and linked data

Received: 15 January 2022 | Revised: 30 May 2022 | Accepted: 5 September 2022
DOI: 10.1111/exsy.13143

ORIGINAL ARTICLE

Expert Systems WILEY

A hybrid recommender system based on description/dialetheic logic and linked data

Ricardo Dos Santos[1,2] | Jose Aguilar[1,2,3,4]

[1]CEMISID, Facultad de Ingeniería, Universidad de Los Andes, Mérida, Venezuela
[2]Tepuy R&D Group, Artificial Intelligence Software Development, Mérida, Venezuela
[3]GIDITIC, Universidad EAFIT, Medellín, Colombia
[4]Departamento de Automática, Universidad de Alcalá, Alcalá de Henare, Spain

Correspondence
Jose Aguilar, CEMISID, Facultad de Ingeniería, Universidad de Los Andes, Mérida, Venezuela.
Email: aguilar@ula.ve

Abstract

In this research, we developed a hybrid recommender system based on description/dialetheic logic, which provides an innovative architecture for the recommendation based on ambiguous reasoning. In addition, our approach allows enriching the knowledge during the reasoning using the linked data paradigm. Thus, the architecture allows the integration of linked data, in order to be exploited by the recommender. On the other hand, the reasoning mechanisms of dialetheic logic allow handling situations of contradiction or inconsistency, to determine the information to offer to users. According to the reviewed literature, our proposal is the first that implements a dialetheic engine in a Recommender System. In general, there are a lot of works about the utilization of linked open data (LOD) to improve the Recommender Systems, for example, to enrich the information to recommend or to solve the cold start problem. However, there are no proposals to deal with the problems of ambiguity, incoherence or incompleteness of the information in these environments extended with LOD. In this way, this paper proposes a hybrid reasoning engine that uses the linked data paradigm for the knowledge extraction, and dialetheic logic for the management of inconsistency/ambiguity information, in order to obtain recommendations. Particularly, the information extracted with Linked Data is processed with the Dialetheic Logic reasoner to solve ambiguous cases. Thus, each recommendation is enriched with related content extracted from the linked data sources, which has been disambiguated. The results are very promising since our hybrid reasoning mechanism allows obtaining more precise recommendations, considering the different classical states of ambiguity in dialetheic logic (contingent statements about the future, failure of a presupposition, vagueness, counterfactual reasoning), according to various metrics of quality used to evaluate the recommendations achieved.

KEYWORDS
dialetheic logic, hybrid recommender system, linked data

1 | INTRODUCTION

Linked data (LD) is a way of data publishing on the Web, so that they can interconnect between them, allowing them to identify, describe, connect and relate the different elements or concepts on the Web (Berners-Lee et al., 2006; Dos Santos & Aguilar, 2018; Dos Santos et al., 2021; Rodríguez et al., 2017). At present, the linked open data (LOD) clouds cover many domains, from structured information about life sciences (diseases, symptoms, among others) and media (movies, books, etc.), to geographic and government data (Ristoski, 2019). These clouds provide a

huge interconnected and decentralized knowledge base, which is readable by people and machines, but requires services and applications that make intensive use of all this knowledge (Dos Santos et al., 2021). Recommender systems (RSs) can benefit from all this information contained in the LOD cloud, by exploiting the non-trivial connections encoded in the cloud. In the context of RSs, LD provides a rich source of semantic knowledge, since it identifies and describes the resources and the different contexts in which they are found. Clearly, some of the information is quite trivial (such as the types of diseases), but many others are very fine-grained (such as the symptoms, causes or medication associated with a disease). In general, they can enrich the information by automatically injecting new and useful knowledge. Consequently, user preferences and tastes can be better modelled.

Nowadays, many types of RSs differ depending on the domain in which they are deployed or the strategies applied for their design (Monsalve-Pulido et al., 2020). One of these types of RSs is the intelligent recommender system (IRS) (Aguilar et al., 2017). An IRS has a set of techniques and algorithms that allow building knowledge models with the ability to foresee and predict the preferences of users, for which filters information and adjusts its results to their needs. Its main objective is to suggest or recommend information or resources (movies, music, books, web pages, news, etc.), based on the knowledge obtained from other people's experiences and/or their preferences (Dos Santos & Aguilar, 2018; Ricci et al., 2011). A fundamental part of the recommendations is the user's profile (tastes and preferences), which must contain the necessary information to be able to identify specific characteristics of a user. For the construction of such profiles, data must be collected, which may be explicit (obtained directly from the user) or implicit (obtained from their search histories, location of the equipment, similarity with other users, among others) (Dos Santos & Aguilar, 2018; Ricci et al., 2011). In addition, IRSs make use of reasoning mechanisms, which require knowledge bases that characterize and model all the knowledge of the environment. However, such knowledge of the environment is usually ambiguous, and in many cases, with inconsistent or incomplete information. An example is the information obtained from LOD, which comes from different sources, such as institutions or organizations. Classically, the knowledge bases in an IRS are described using description logic, but there are other types of logical languages used to represent knowledge, such as those based on temporal or dialetheic logic (DL), which can solve situations with inconsistent information (Aguilar, 2011; González-Eras et al., 2022). Particularly, DL allows handling situations of contradiction or ambiguity, and in particular, contexts where presupposition fails, there is vagueness in expressions in natural language, contingencies about the future based on something being true and false in the past, and/or fictitious discourses that generate assumptions (González-Eras et al., 2022; Pelletier et al., 2017). DL's ability to reason in states of ambiguity and inconsistency, makes it an ideal candidate to resolve these issues facing the IRSs.

The objective of this research is the development of a hybrid recommender system (HRS) capable of taking advantage of DL and LD, for more accurate recommendations. To fulfil this objective, the work defines a hybrid reasoning mechanism that mixes the benefits of Description/Dialetheic Logic to handle ambiguities, and a knowledge model that exploits LD to describe knowledge with semantic enrichment. Thus, the main contributions of this work are the definition of:

1. A HRS that uses the DL and LD approaches.
2. A hybrid reasoning mechanism that mixes the benefits of Description/Dialetheic Logic for the management of inconsistency/ambiguity information, like contingent statements about the future, failure of a presupposition, vagueness, and counterfactual reasoning.
3. A knowledge model that exploits LD for the knowledge extraction for semantic enrichment.

The HRS is tested in different contexts, and according to various metrics of quality used, it allows obtaining very precise recommendations for different ambiguity states.

This paper is structured as follows: Section 2 describes related works, Section 3 presents the theoretical framework around LD and DL; Section 4 describes the architecture of our HRS based on Description/Dialetheic Logic and LD; Section 5 describes the case studies where HRS architecture is tested; Section 6 presents the analysis of the results, and compares them with other architectures; and finally, Section 7 presents the conclusions and future works of this research.

2 | RELATED WORKS

In previous research about RSs with the ability to reason with LD, the following recent papers are presented: Musto, Basile, et al. (2017) present a graph-based recommendation Methodology Based on PageRank with Priors and LD. This methodology proposes the construction of a tripartite graph composed of three sets of nodes: (i) User nodes (U); (ii) Item nodes to be recommended (I) and; (iii) nodes with information extracted from the LD related to the Items (V). To generate the recommendations, the graph is customized to the preferences of the user who needs the recommendations. Then, the PageRank with Priors algorithm is executed, obtaining a score for each node based on the connectivity between them and their weights. Finally, the nodes I that were not directly connected to the user are ordered from highest to lowest, and these ordered nodes are delivered as the list of recommended items. Additionally, in Musto, Lops, et al. (2017) this work is extended, using classification techniques (Random Forests, Naïve Bayes and Logistic Regression) and three families of variables to represent the items, with the aim of predicting the most interesting elements for the user.

In the educational context, in Chicaiza et al. (2017) is detailed an Educational-Resources Recommender System with LD. In the paper, they use an Latent Semantic Indexing (LSI) algorithm to select resources according to a set of criteria about the interest of the user; and they incorporate different strategies, to semantically infer and derivative data, according to the interactions between the users and the system. Likewise, in Pereira et al. (2018) is presented a Social RS, which aims to identify the characteristics, interests and preferences in the profile of a user in educational contexts, extracting information from Facebook, and enriching it through different Semantic Web technologies. This recommender identifies the available data, through the profile of users on the social network and their interactions; and enriches them using a matching mechanism based on the information provided by the specific domain ontologies. In the same way, in De Angelis et al. (2017), the authors present other Social RS, which takes into account the social media activities carried out by the target user and his/her friends, and the information collected from the LD sources. Specifically, this recommender extracts social information from social networks (e.g. Facebook); and the user modelling is a directed, tagged and heterogeneous Social Graph, where each node is associated with a class (Person, Place, Location and Category), and the edges are label (KNOW, VISIT, LOCATE and HAVE CATEGORY).

On the other hand, Fogli et al. (2018) present an Itinerary RS, which takes advantage of the LOD to carry out a recommendation of personalized itineraries, taking into account the social context, and integrating multimedia and text contents. User knowledge is modelled in a directed graph, where each node represents a place with its popularity. Each arc represents a direct connection between two nodes, with information on the shortest route to go from one node to another and the travel time, taking into account the user's means of transport. Musto et al. in 2018 present an RS with LD, which aims to recommend an explanation to the user of why they might like some recommendation. Basile et al. (2019) define an RS based on a holographic embedding of knowledge graphs built from Wikidata, a free and open knowledge base. In this way, they introduce LOD into their RS, to enrich the representation of items by leveraging RDF statements and adopting graph-based methods to implement effective RSs. In the same way, Natarajan et al. (2020) propose an approach to the cold start issue with the collaborative filtering methods, which is due to the deficient information about new entities. The approach is a Recommender System with Linked Open Data (RS-LOD), which finds enough information about new entities for the cold start issue, and improves the matrix factorization model to handle data sparsity. This paper overcomes the data sparsity and the cold start problem in CF Recommender System. They use LOD for the cold start problem, and a Matrix Factorization model with LOD for the data sparsity problem. García-Sánchez et al. (2020) propose an ontology-based advertisement recommendation system that exploits the data produced by users on social networks. The ontology model represents both users' profiles and the content of advertisements by means of vectors generated using NLP techniques. In Van Rossum & Frasincar (2019), the authors investigate the incorporation of graph-based features into LOD path-based recommender systems. They propose two normalization procedures that adjust the user-item path counts by the degree of centrality of the nodes connecting them. Their experiments show that the linear normalization approaches yield a significant increase in recommendation accuracy. Finally, Oliveira et al. (2019) propose a similarity measure for linked data that personalizes the RDF graph by adding weights to the edges, based on previous user's choices. This approach minimizes the sparsity problem by ranking the best features for a particular user, and also, by solving the item cold-start problem.

Zitouni et al. (2017) present a Content-based RS based on a Semantic Vector Space Model and LD to recommend documents. This architecture is composed of two modules: (i) content-based pre-filtering: This module is used by new users, where the recommendation engine calculates the distance of a new user from the existing ones; and (ii) semantic content-based filtering: This module takes advantage of the user's feedback with respect to the current list of documents. For that, it is represented in a vector model the tastes and preferences of the user, and the textual attributes that represent or characterize each document. These two vectors are enriched with information extracted from different LD sources. Finally, the new list of recommendations is extracted according to the value of the similarity of a user with respect to each document. In Musto, Franza, et al. (2018) is specified a content-based RS based on the techniques of Deep Learning and LD. Specifically, it is an architecture that learns a joint representation of the elements to be recommended, by exploiting the unstructured information extracted from the textual description of the elements. Then, this representation is extended by introducing structured features extracted from the sources of LOD. In addition, Jiménez et al. (2019) extends a contextual ontology, called CAMeOnto Aguilar et al. (2018), with LD and DL paradigms, to enrich and manage ambiguities in context-aware applications. These micro-location applications, provide services depending on the location and need to model the context using ontologies.

In Selvan et al. (2019), Selvan et al. present a Fuzzy RS based on fuzzy ontologies and LD to recommend food and drugs to chronic patients (diabetics). Specifically, the architecture processes all the data acquired from different sources (Sensors and medical history, Twitter, Facebook, MedlinePlus, etc., each one in different formats, such as CSV, etc.) and transforms them into LD (RDF triples). Then, it generates the fuzzy ontologies of the patient and the sensor, in order to calculate the Patient's Health Condition. Finally, the architecture uses two ontologies for these recommendations. The pharmacological ontology assigns drugs according to the patient's health conditions, and the food ontology is used to determine an appropriate patient's metabolic rate and caloric intake. The work of Laenen & Moens (2020) proposes a deep learning-based recommender system for an outfit recommendation. This work compares different fusion methods for outfit recommendation, which combine product features extracted from visual and textual data in semantic, multimodal item representations. Najafabadi et al. (2019) define a graph-based structure to model the users' priorities and capture the association between users and items. Users' profiles are created based on their past and current interest. Then, their algorithm keeps the preferred items of the active user at the beginning of the recommendation list. In this way, these items come under top-n recommendations.

Mahdi and Hadi (2021) present the current state of LOD in recommendation systems, considering the problems facing, the importance of using LD, and the various recommendation techniques used, among other things. They remark that there are still many challenges. Also, in Yochum et al. (2020) is presented a systematic review of LOD in RSs in the tourism domain. They analyse problem formulations, data collections, proposed algorithms/systems, and experimental results. Finally, Rahayu et al. (2022) review ontology-based RSs, particularly, the ontologies used and their recommendation processes in open learning environments.

Sebbaq et al. (2020) improve the quality of MOOCs by assisting teachers and designers from the initiation phase of MOOCs, using a recommendation system Framework based on the knowledge about teachers and MOOCs. The framework integrates different techniques: LD to extract and integrate different sources, ontologies to model, among others. Ammar et al. (2021) implement a system to deliver tailored recommendations for improving self-care behaviours in diabetic adults, which uses both digital health data stored in patients' Personal health libraries and other sources of contextual knowledge. They use a social LD platform to design a fully decentralized and privacy-aware platform that supports interoperability and care integration. Chew et al. (2021) define a hybrid model-based recommender system, which is pre-trained with data to generate recommendations for a user using an ontology that helps to represent the semantic information and relationships to model the expressivity and linkage among the data. They define a matrix factorization model accuracy by utilizing the ontology to enrich the information of the user-item matrix by integrating the item-based and user-based collaborative filtering techniques. For that, they use a semantic similarity metric together with a rating pattern.

There is a significant number of works that have started to use LOD to improve the performance of RSs. Some to enrich the information to recommend, others to solve the cold start problem, among other tasks. However, SRs have to deal with the problem of information uncertainty, both at the level of user queries (query ambiguity), and of the relationships between information on the Internet. In general, the RSs described in the different proposals do not solve the problems of ambiguity, incoherence or incompleteness of the information that exists in these environments, with special emphasis on the data collected from LOD, which is a cloud of very varied knowledge sources, built by many entities. On the other hand, also it is observed that many works do not take advantage of LOD for the enrichment of the recommendations reached in their systems, which would allow a richer recommendation of knowledge for the final users. This work seeks to fill out these two gaps by proposing an HRS, for which mixes DL and LD in its mechanisms of representation/extraction of knowledge and reasoning.

Our research proposes an HRS based on the Description/Dialetheic Logic and LD, in order to define a hybrid reasoning mechanism that mixes the benefits of Description/Dialetheic Logic to handle ambiguities, and a knowledge model that exploits LD to describe knowledge with semantic enrichment. Particularly, the mechanisms of LD are used to offer additional information, and to enrich the recommendation with information coming from social networks, and web pages, among others. At the same time, the reasoning mechanisms of DL allow handling contradictory or inconsistent situations, both at the level of the knowledge represented and in the inference of what will be recommended. We can compare our HRS based on LD with respect to other RSs that possess similar characteristics, using as points of comparison the following criteria: (i) *Reasoning Technique*: the mechanisms used to recommend and filter the information. (ii) *Data Sources*: data sources used to enrich the input and by the reasoning models of the RS. (iii) *Vocabularies and Ontologies*: vocabularies and/or ontologies used to identify the different characteristics of the elements of the knowledge model. None of the RSs use the DL as a reasoning mechanism, which allows reasoning in cases of ambiguities or inconsistencies. In addition, our HRS allows the use of any LD source, as also the use of different vocabularies/ontologies according to the particular reasoning needs.

3 | DIALETHEIC LOGIC

The word *Dialetheic* has many definitions, the most concise indicates that it is a theory and technique of rhetoric of dialogue and discussion to discover the truth through the exposition and confrontation of the reasoning and arguments contrary to each other (Velarde, 1977). In DL, the dialetheic axioms allow contradictions and ambivalences to be valid within a formal model (Pulcini & Varzi, 2018). In this sense, DL and formal logic seem to follow opposite paths, and indeed, DL would be the realm of contradiction, while formal logic would be the realm of non-contradiction (Velarde, 1977). In the logic of human reasoning, some instances or states are affected by contradictions (Kamide & Wansing, 2015; Velarde, 1977). DL can solve situations in which reasoning encounters inconsistent information since it has at hand information both to believe a thing and, at the same time, to believe the opposite (Lukasiewicz & Wedin, 1971). The possible states managed by the DL are the following (Pelletier et al., 2017):

1. *Contingent statements about the future*: indicates that something was true and false in the past, so its future cannot be foreseen. For example, "tomorrow there will be a war" can be true or false, as both cases have occurred in the past.
2. *Failure of a presupposition*: to assume something that is not true. For example, "it is not a boy". If in the presupposition a possible value is assumed, then one can think that it is a boy, and there is a fault because it could also be a girl.
3. *Vagueness*: lack of clarity, precision or accuracy in natural language phenomena. E.g., in the sentence "He is bald", one cannot deny that a person with zero hairs is bald, nor can one deny that a person with 10,000 hairs is bald.

4. *Fictional discourse*: making decisions according to certain real or imaginary assumptions (non-Aristotelian imaginary logics). For example, "Cows are flying", it can be said that it is false because our beliefs tell us that cows do not fly, but it could happen that cows are being transported in an airplane, and in this case, the answer would be true.
5. *Counterfactual reasoning*: to think about what could have been and wasn't. E.g., "If I hadn't gone out, then I would have passed and now I wouldn't have to study for the recuperative". It represents something that did not happen, but that could have happened. There underlies the uncertainty, in that which could have happened.

Finally, at the University of Miami was developed a dialetheic reasoner, called JGRM3, based on *Dialetheic Logic RM3* (Pelletier et al., 2017). This logic belongs to the branch of Paraconsistent Logic (PL), that is, it allows inconsistencies and contradictions to be valid. However, PL has a weakness in the conditional, because it is not compatible with the Modus Ponens and Modus Ponendo Tollens inference rules. Instead, DL improves it because it allows Modus Ponens. It is understood that the *Modus Ponens* (affirmative mode), also called *Modus Ponendo Ponens* (a mode that by affirming affirms), states that if a term A implies the term B, and A is true; then it can be inferred that B is true (Beevers, 2020). For example, "If it is raining, then I wait for you inside the theater", and "it is raining", therefore, "I wait for you inside the theater". The reasoner JGRM3 recognizes two types of events: (i) Dialetheic Events (DE), which correspond to those where the axioms return a positive truth value (true). (ii) Nondialetheic Events (NE), those to which the axioms give a negative truth value (false).

4 | ARCHITECTURE OF OUR HYBRID RECOMMENDER SYSTEM

4.1 | General architecture of our HRS

This research develops an HRS based on the ideas proposed in Dos Santos et al. (2019). This hybrid recommender responds to several needs: the first is to be able to solve situations in which a process of reasoning of recommendations encounters inconsistent information, that is, a state of contradiction or ambiguity. The second is to exploit the information on the web structured as LD, to enrich the recommendation process with semantic information about the user (tastes, preferences, etc.) and/or the resources to be recommended (diseases, symptoms, treatments, among many others). Based on these needs, it is necessary to specify a knowledge-based RS, called *Intelligent Recommender System* (Aguilar et al., 2017; Monsalve-Pulido et al., 2020; Ricci et al., 2011), which considers the mechanisms of knowledge representation and reasoning engines, among other aspects.

The general architecture of an IRS for our HRS is defined by four layers, based on the work (Aguilar et al., 2017): (i) *Knowledge Sources*: are the sources that provide information about users, context, resources, among others. In our system, it will be mainly made up of LD sources, which provide data with semantic information; (ii) *Knowledge acquisition*: this layer is in charge of data extraction and processing. In our case, it will generate queries in SPARQL that allow identifying, filtering and extracting the information available in the LD sources to enrich the knowledge model; (iii) *Knowledge Modelling*: this layer specifies the paradigm of knowledge representation; for our case, it is represented as LD and axioms based on description/dialetheic logic; (iv) *Reasoning and verification*: this layer implements reasoning mechanisms, offering the capacity to exploit knowledge and infer recommendations. In our system, it will use a hybrid mechanism, on one hand, it uses a reasoner of description logic that allows exploiting the sources of LD to enrich the knowledge model, and on the other hand, it uses a reasoner of DL that verifies the knowledge model and evaluates the recommendations through the different dialetheic events.

Figure 1 shows the components of our HRS, made up of two groups: the first group is composed of the components that allow reasoning to extract information or infer recommendations, even in the presence of inconsistencies or ambiguities in the problem or query (PoQ), or in the data extracted from LD, called Reasoning Engines. The second group, called the Manager, is composed of the components in charge of managing all the processes necessary to reach a recommendation. They determine when and what should be exploited from the LD, either to enrich the ontologies or vocabularies of the reasoning model, or to semantically enrich the data or recommendations found.

4.1.1 | Below are described the reasoning engine components

1. *Description Logic Engine (DeLE)*: This engine is intrinsic to LD, since both the semantic structure of the data and the query mechanisms (read, create, update or delete triplets) are based on description logic. In addition, the engine allows exploiting different data sources, thanks to the LD technique that interconnects data through distributed access points or local endpoint (for personal profiles and context), or public endpoint (Dbpedia endpoint, Wikidata endpoint, among many others). This engine receives a query based on triplets and returns the data found as variables. In the following, an example: the query *SELECT? disease WHERE{? disease rdf:type dbo:Disease.}* returns the variable *disease* with the list of diseases that are available in Dbpedia endpoint.

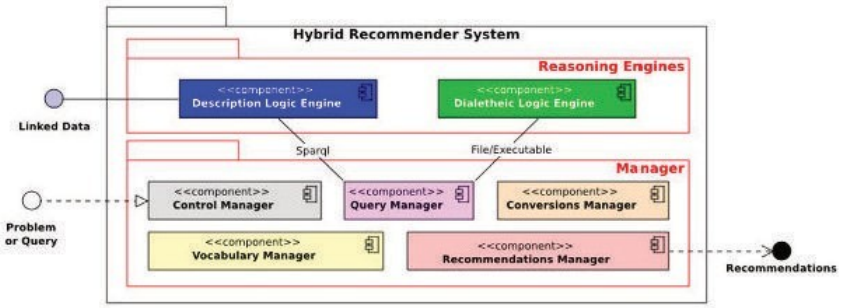

FIGURE 1 Model (diagram) of components of our hybrid recommender system

2. *Dialetheic Logic Engine (DiLE)*: This engine responds through consultations constructed as conjectures on the models described in first-order logic, where axioms and their facts are detailed, being able to detect and reason in states of ambiguity or inconsistency, thanks to the capacity offered by DL. This engine receives a query as a conjecture and a model, and returns the inferred from the conjecture on the model. In the following, an example: *fof(example, conjecture, [disease | ∼ disease])*. It checks whether or not the disease is present, which is an ambiguity, and the JGRM3 returns that it is a DE.

4.1.2 | The manager components are

1. *Vocabulary Manager (VM)*: it is responsible for identifying and selecting the vocabularies and ontologies that are necessary to process the requests received by the recommender. The manager seeks to match the terms of the request with the classes and properties of the knowledge it possesses. If this objective is not achieved, then it relies on the Query Manager to extract new knowledge from the sources of LD, which can enrich the ontologies or vocabularies of the reasoning model. This component takes the axioms present in the PoQ input and extracts the predicates, since those predicates are the concepts that need to be identified. For example, for the axiom *fof(diseaseDoctor,axiom, (! [U, Dr, S, D]: ((disease(D) & symptomDoctor(U, S, Dr) & isSymptom(D, S)) = > diseaseDoctor(U, D, Dr)))*, it selects *disease*, *symptomDoctor* and *isSymptom*. Then, those predicates are compared with the knowledge base, in order to determine if the URIs that identify and relate to the concepts are known. If not, then the Query Manager is invoked to generate queries for URIs representing these unknown concepts (more in Table 3).
2. *Query Manager (QM)*: it is responsible for preparing and generating the necessary queries to enrich or recommend the information.

- On the side of the DeLE, queries are generated based on triplets that exploit the data contained in the LD sources, relying on the knowledge base (KB) of the vocabularies and ontologies handled by the recommender and the LD sources. Specifically, The query is built through three templates, according to the type of request: (i) *Concept as URI* allows searching, using LD, the URI that represents a concept with the template *SELECT DISTINCT? URI WHERE {[] a? URI. FILTER regex(?URI, 'ontology', "i"). FILTER regex(?URI, 'CONCEPT', "i")}*, replacing *CONCEPT* with the concept to be searched; (ii) *Property related to a Concept as URI* allows searching, using LD, the URI that represents a property that is related to a URI of a known concept. *SELECT DISTINCT? URI WHERE {? i a URI_CONCEPT.? i? URI? o. FILTER regex(?URI,'ontology', "i"). FILTER regex(?URI, 'PROPERTY', "i")}*, replacing *URI_CONCEPT* with the URI of the known concept and *PROPERTY* with the property to be searched.; and (iii) *Knowledge Extraction* allows extracting all the knowledge available in the LD source, making a *SELECT* of the variables to be extracted, using the triple? *URI_CONCEPT*? *URI_PROPERTY*? *OBJECT*, and replacing? *URI_CONCEPT* and/or? *URI_PROPERTY* by the known URIs in the cases required.
- On the side of the DiLE, queries are generated based on conjectures described in the PoQ axioms, allowing to test and infer the recommendations, even if there is inconsistency or ambiguity in the PoQ or data. Specifically, each axiom of PoQ is analysed, and the predicate after each implication symbol (=>) is extracted since it will be part of the conjectures that the HRS will use to prove the model and the data. For example, in the axiom *fof(sickDoctor, axiom,(! [U,Dr,D]: ((diseaseDoctor(U, D, Dr)) => sickDoctor(U, Dr)))*, the predicate *sickDoctor(U, Dr)* is extracted and the conjecture *fof(li, conjecture,(sickDoctor[U,Dr]))* is constructed.

3. *Conversion Manager (ConM)*: It is in charge of preparing and generating the data transformations to allow the exchange of information between the different reasoners that the recommender has and the information found, which are necessary to allow the results of one reasoner to be used by the other. For example, to use the data generated by the DeLE with the DiLE, it is required to transform the data, because the DeLE returns an inferred table with the values found by each requested variable, and the DiLE needs a model with the data described as facts in First-Order Logic. These transformations are created from the reasoning model, using the knowledge base that VM reached and the knowledge extraction query that QM generates. Specifically, two types of conversions are performed, the first type converts the data of a variable or concept extracted from LD. For example, if a list of diseases is extracted, then it would generate an axiom of that concept with the form: *fof (disease_axiom, axiom,[? [D]: (disease(D))])*, and the list of facts with the form: *fof(disease523, axiom, disease[dbpedia_Zika_fever])*, where *disease523* is the 523 disease extracted from LD, and *disease(dbpedia_Zika_fever)* is the predicate *disease* with the fact that the disease is *dbpedia_Yellow_fever*. The second type converts the data of two related variables or concepts that were extracted from LD. For example, if a list of diseases and their symptoms are extracted, then it would generate an axiom of those related concepts in the form: *fof(isSymptom_type,-axiom,[! [D] disease(D) = >? [S]: (symptom(S) & isSymptom(D, S))])*, and the list of facts with the form *fof(isSymptom1357, axiom, isSymptom [dbpedia_Zika_fever, dbpedia_Fever])*, where *isSymptom1357* is the disease/symptom number 1357 extracted from LD, and *isSymptom (dbpedia_Zika_fever, dbpedia_Fever)* is the fact that *dbpedia_Zika_fever* has the symptom *dbpedia_Fever*.
4. *Recommendation Manager (RM)*: It is in charge of merging and filtering the information obtained by the reasoners, allowing it to collect and deliver the knowledge reached by all the mechanisms that make up the recommender. Specifically, RM asks ConM for the conjectures that will allow validating the reasoning model with the elements extracted through DeLE. Then, DiLE filters the recommendations; for that purpose, the DE are detected using the different conjectures. Now, with the recommendations reached, it is proceeded to evaluate the degree of closeness of each recommendation to the user profile. This evaluation consists of adding up the characteristics of the user profile that coincided with the recommendation. Finally, the recommendations are ordered from highest to lowest, according to the result of each evaluation (more details in Table 5).
5. *Control Manager (CM)*: it is responsible for all decisions of the HRS, determining when and how managers and reasoners should be invoked according to their capabilities. This manager uses meta-reasoning to make these decisions (more details in Table 2). The meta-reasoning seeks to accomplish the following objectives: (i) verify that the PoQ is correctly defined, testing it with DiLE (see Table 2); (ii) identify the concepts present in PoQ, searching for them with DeLE (see Table 3); (iii) extract the knowledge associated with the concepts identified in PoQ, extracting them with DeLE (see Table 4); (iv) verify and filter the recommendations, using DiLE (see Table 5); and (v) extract content related to the recommendations, using DeLE (see Table 6).

4.2 | Behaviour of our HRS

In general, the steps followed by the HRS can be seen in the Macro-Algorithm in Table 1.

The process begins when a user requests the recommender, providing the PoQ that he/she wishes to solve. The first thing that CM does is to run the input with DiLE, determining if the axioms are correctly described (Step 1). In case it is not correct, then the process is stopped (Step 1.1). In case it is correct, then the CM activates a series of processes that allow incorporating knowledge from the LD and finding possible recommendations (Step 2). In step 2.1, it is determined if there are ontologies and/or vocabularies that allow identifying the terms used by the PoQ. In step 2.2, the information is extracted from the LD, concerning the terms identified in the previous step. This allows the enrichment with knowledge of the possible recommendations that will be reached. In step 2.3, the inconsistencies and/or ambiguities that are present in the collected data for the PoQ are verified, and the recommendations found are obtained. In step 2.4, these recommendations are enriched with related knowledge searched through the LD sources. Finally, the enriched recommendations are delivered. A more detailed description of the different processes follows:

TABLE 1 Macro-algorithm of our hybrid recommender system

Input: PoQ
Procedure: 1. CM verifies PoQ with DiLE. 1.1 If does not correct PoQ, then the recommendation process is stopped. 2. CM activates each process. 2.1. Process: Identification of URIs using LD (see Table 3). 2.2. Process: Knowledge Extraction using LD (see Table 4). 2.3. Process: Verification and Filtering of Recommendations using DL (Table 5). 2.4. Process: Content Extraction Related to Recommendations using LD (Table 6).
Output: Enriched Recommendations

TABLE 2 Macro-algorithm of identification of URIs using linked data

Input: PoQ
Procedure: 1. CM activates VM. 2. VM checks if ToC is in KB: 2.1. VM extracts To present in PoQ. 2.2. For each ToC of PoQ: 2.2.1. If ToC is not Identified, it requires updating KB: 2.2.1.1. VM activates QM. 2.2.1.2. QM prepares the queries to extract URI. 2.2.1.3. VM invokes DeLE with the generated queries. 2.2.1.4. ToC is added to the KB.
Output: updated KB

TABLE 3 Macro-algorithm of knowledge extraction using linked data

Input: PoQ
Procedure: 1. CM needs to extract information for the PoQ using KB. 1.1. CM actives QM. 1.2. QM prepares queries to extract information associated with PoQ using KB. 1.3. CM invokes DeLE with queries to the local source (knowledge of context and user preferences). 1.4. CM invokes DeLE with queries to other sources (general knowledge).
Output: Extracted Knowledge

TABLE 4 Macro-algorithm of verification and filtering of recommendations using dialetheic logic

Input: PoQ and extracted knowledge
Procedure: 1. CM activates RM to find recommendations. 2. RM activates ConM. 2.1. ConM converts the Extracted Knowledge according to the need of DiLE. 3. RM activates QM. 3.1. QM generates the queries according to the PoQ using KB. 4. RM seeks recommendations by running DiLE. 4.1. RM verifies the data. 4.2. RM filters results. 4.3. RM orders results.
Output: Recommendations

4.2.1 | Identification of URIs using LD

Objective: to identify the terms or concepts used in the PoQ, looking for URIs of an ontology and/or vocabulary that represents them.
General description: The process shown in Table 2, starts when CM receives the PoQ and activates the VM to recognize the characteristics of that PoQ (Step 1). In step 2, VM verifies if the terms or concepts (ToC) present in PoQ are related to a vocabulary or ontology known by the HRS in its KB. This relationship is identified through a URI, which allows extracting new knowledge from the sources of LD. For example, a disease is represented by the URI http://dbpedia.org/ontology/disease. To do this, step 2.1 extracts each ToC in PoQ (see VM component). Then, each ToC extracted from PoQ is checked (Step 2.2). If ToC is not known by HRS in its KB, then VM must find a URI that represents the ToC (Step 2.2.1). In this case, VM activates QM (Step 2.2.1.1), so that QM prepares queries to find a URI through DeLE that represents the ToC (Step 2.2.1.2). Examples of this process are described in the QM component, specifically, in the consultation types *Concept as URI* and *Property related to a Concept as URI*. In step 2.2.1.3, VM executes DeLE with the queries received by QM, allowing it to find a URI representing ToC. Then, ToC, with its URI, is added to KB as a concept identified by HRS (Step 2.2.1.4). All these URIs identified in this process will allow the extraction of knowledge for the following process.

TABLE 5 Macro-algorithm of content extraction related to recommendations using linked data

Input: Recommendations
Procedure: 1. CM activates RM to enrich recommendations. 2. RM activates QM. 2.1. QM generates queries according to recommendations. 3. RM activates DeLE with queries. 4. RM merges and returns enriched recommendations.
Output: Enriched Recommendations

4.2.2 | Knowledge extraction using LD

Objective: to extract information to be recommended using the URIs as a mapping point between LD sources and ToC.
General description: Table 3 presents the process of Knowledge Extraction, which starts when the CM has identified each ToC with their respective URIs, and needs to extract information from the LD sources to find possible recommendations (Step 1). The CM carries out this task by activating QM (Step 1.1), so that QM prepares the queries to extract information from the LD associated with the PoQ using KB (Step 1.2). It is carried out according to the next way: using the known URIs of the previous process and the triplet for the "Knowledge Extraction" query type (see QM Component), now, in the cases where two concepts are related, it substitutes? *URI_CONCEPT* and? *URI_PROPERTY* with the respective URIs. In cases where one concept is not related to others, only? *URI_CONCEPT* is substituted. Having the queries, the CM invokes DeLE and collects information from sources, such as user preference, context, and other knowledge associated with PoQ (Steps 1.3 and 1.4).

4.2.3 | Verification and filtering of recommendations using DL

Objective: to verify the inconsistencies and/or ambiguities that are present in the collected data and the model, and generate recommendations.
General description: The process begins when the ToC has been identified, and all data have been extracted using the LD paradigm (Table 4). Therefore, the CM activates the RM to generate the recommendations (Step 1). RM carries out a series of processes to achieve the recommendations. The first one is to convert the data collected by DeLE in the previous process to the structure received by DiLE. For that, the task ConM is activated (Step 2). In Step 2.1, ConM performs this task through two types of conversions, which are described in the ConM component. The second is to activate QM (Step 3) to generate the queries based on the model that DiLE receives. To do this, QM, in Step 3.1, must deliver the queries based on conjectures according to the axioms present in PoQ. This process is described in the QM component. Finally, RM runs DiLE with PoQ, the data transformed by ConM, and the different conjectures delivered by QM (Step 4). DiLE checks the data and conjectures (Step 4.1). With the results delivered by DiLE, the recommendations are filtered (Step 4.2) and sorted (Step 4.3), as described in the RM component.

4.2.4 | Content extraction related to recommendations using LD

Objective: to enrich the recommendations with related content extracted from the LD.
General description: Table 5 shows the process of enrichment of the recommendations. In step 1, the CM activates the RM to look for information related to the recommendations reached in the previous process (Step 1). To reach this objective, the RM activates QM (Step 2), which will be in charge of generating the necessary queries to extract from the LD sources, the contents related to the recommendations (Step 2.1). It is made using the URIs of the recommendations reached from the previous process, and the triplet for the type of query "Knowledge extraction" (see QM component), replacing? URI_CONCEPT with the URI of each recommendation reached. When this URI is used, then the ambiguity is avoided with respect to the recommendations reached in previous processes. Then, RM invokes DeLE using the queries generated by QM (Step 3). Finally, the related content extracted from the LD sources is linked to the recommendations reached in the previous process, and delivered as a final result (Step 4).

5 | CASE STUDY

This section presents a case study, in which is detailed the behaviour of our system. In addition, it shows how the resources from the LD would be exploited within the recommender. Figure 2 shows the input sources for our HRS in this case study. In addition, the next assumption is carried

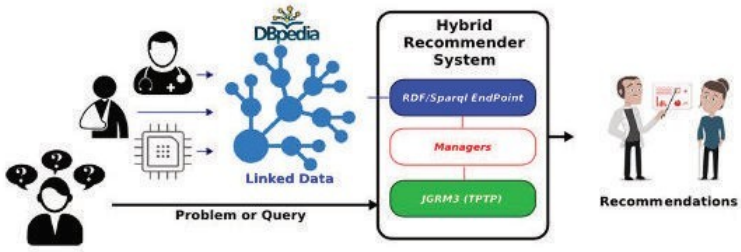

FIGURE 2 Case study of the hybrid recommender system

out: *doctors and/or body sensors only detect the different symptoms that a user may present, and such information is stored in a local LD repository, which is implemented with OpenLink Virtuoso (Open-Source Edition)*.[i] On the other hand, there are data associated with known types of diseases, and their symptoms, treatments, and causes, among others. This information is extracted from the external LD repository, called Live-Dbpedia,[ii] which is a source with updated data, since it immediately retrieves all the changes from Wikipedia.

Besides, it also needs PoQ as input, which is the problem or query to be solved. This input will be received by the CM, which is responsible for controlling the entire process carried out by the HRS. In this case, it seeks a PoQ representing a DL situation called *Fictional Discourse*. To do this, it seeks to determine if a person is sick, according to certain assumptions. In particular, our system must manage the symptoms that can or cannot be associated with a disease, the conflicts between the symptoms provided by each doctor or sensor, the lack of information, the inconsistency of the data, etc. Thus, our system allows managing the different ambiguities between the opinions of the doctors and/or information captured by the sensors. In this way, it is composed of four axioms that describe the problem, and two conjectures that represent the questions to be solved (see Figure 3):

1. Sick with D because U presents any of the S symptoms, according to Doctor Dr (see axiom diseaseDoctor). This axiom determines if any of the S symptoms is detected by Dr (symptomDoctor) in the user U. Then, it is checked if the detected symptom is associated with the D disease (isSymtom).
2. Sick with D according to the opinion of the Doctor Dr, because U presents any of the S symptoms (see axiom sickDoctor). This axiom determines if there is a disease D in the user U, according to the opinion of doctor Dr (diseaseDoctor). This axiom is based on axiom 1, because the Doctor Dr detects any of the S symptoms.
3. Sick with D because U presents any of the S symptoms, according to Doctors Dr1 and Dr2 (see axiom diseaseDoctors). This axiom also is based on axiom 1. It carries out a double check of the doctor's opinions about the symptoms, such that it determines if disease D is present in user U based on these opinions of the doctors Dr1 and Dr2.
4. Sick according to the opinion of the Doctors Dr1 and Dr2, because U presents any of the S symptoms (see axiom sickDoctors). This axiom is based on axioms 2 and 3, and it determines if there is a disease D in the user U, according to the opinions of the doctors Dr1 and Dr2 (diseaseDoctors).
5. Conjecture i: sickDoctors. User_A is sick according to the opinion of doctor_A and doctor_B.
6. Conjecture ii: diseaseDoctors. User_A is sick with D according to the opinion of doctor_A and doctor_B. D belongs to the set of diseases that are available in the knowledge base.

Finally, the HRS with the inputs defined is ready to start the extraction of the knowledge needed to reason and make its recommendations. These processes are shown below:

5.1 | Identification of URIs using LD

This process shows how the HRS identifies the ToC present in the PoQ input, and their relationships. For this, it seeks to associate each ToC to a URI that represents it (see macro-algorithm in Table 3). In this case, it is assumed that the knowledge base of the HRS knows the *user* and *doctor* ToC, and the relationships between them, using only the vocabulary FOAF (see Table 6).

The process begins when VM is activated by CM to identify ToC (see step 1 in Table 2). Then, VM obtains the predicates *disease*, *isSymptom* and *symptomDoctor*, and with them, it extracts the ToC *disease*, *symptom* and *doctor* (step 2.1 in Table 2). It does not identify the ToC *disease* and

```
fof(diseaseDoctor,axiom,(
    ! [U,Dr,S,D] : (
        (  disease(D) & symptomDoctor(U, S, Dr) & isSymptom(D,S)  )
        => diseaseDoctor(U, D, Dr)
    )
)).

fof(sickDoctor,axiom,(
    ! [U,Dr,D] : (
        ( diseaseDoctor(U, D, Dr) )
        => sickDoctor(U, Dr)
    )
)).

fof(diseaseDoctors,axiom,(
    ! [U,Dr1,Dr2,D] : (
        ( diseaseDoctor(U, D, Dr1) & diseaseDoctor(U, D, Dr2)  )
        => diseaseDoctors(U, D, Dr1, Dr2)
    )
)).

fof(sickDoctors,axiom,(
    ! [U,Dr1,Dr2,D] : (
        ( diseaseDoctors(U, D, Dr1, Dr2) )
        => sickDoctors(U, Dr1, Dr2)
    )
)).

%%%%%% Conjecture
fof(i,conjecture,( sickDoctors(user_A, doctor_A, doctor_B) )).
fof(ii,conjecture,( diseaseDoctors(user_A, D, doctor_A, doctor_B) )).
```

FIGURE 3 Input problem or query for the hybrid recommender system

TABLE 6 Hybrid recommender system knowledge base on the terms or concepts (ToC)

ToC		URI
user		foaf:Person
doctor		foaf:Person
opinionDoctor		foaf:Document
publications		foaf:publications
maker		foaf:maker
Subject	**Property**	**Object**
user	publications	opinionDoctor
opinionDoctor	maker	Doctor

Note: foaf: http://xmlns.com/foaf/0.1/.

```
sparql = SPARQLWrapper2("http://live.dbpedia.org/sparql")
sparql.setQuery("""
SELECT DISTINCT ?URI
WHERE {
   [] a ?URI
   FILTER regex(?URI,'ontology', "i" ) .
   FILTER regex(?URI,'disease', "i" ) .
}
""")
```

FIGURE 4 Query to find an ontology for the terms or concepts: disease

symptoms, and the relationship between them (step 2.2.1 in Table 2). VM solves this problem with the invocation to *QM* to prepare queries to extract information from the LD through the *DeLE* (step 2.2.1.1 in Table 2). In this way, for each unidentified ToC, an ontology is sought to describe it. Figure 4 shows a query defined by QM (step 2.2.1.2 in Table 2), a process that was previously described in the QM component, for extracting the URI. This query searches for an ontology for the ToC *disease* that is contained in the source of LD Live-Dbpedia, obtaining as a result the URI to represent *diseases*: "http://dbpedia.org/ontology/disease" (step 2.2.1.3 in Table 2).

Finally, VM uses LD techniques and the information contained in the LD sources (step 2.2.1.4 in Table 2), to identify and relate the terms of the PoQ input (see Table 7).

The processes described in this section are automated in Python. The first process (VM component) extracts the predicates from the axioms described in PoQ. Those predicates are transformed into ToC, which are the terms to be searched in the LD source. The second process uses the templates described in QM as "Concept as URI" or "Property related to a Concept as URI", where the words CONCEPT or PROPERTY are replaced by the ToC that needs to identify its URI. Then, the search query is executed in the LOD source endpoint (e.g. http://live.dbpedia.org/sparql). This process is repeated with all the ToC that need to be identified, and if any of these ToC is not associated with a URI, then the system stops its execution and indicates the problem.

5.2 | Knowledge extraction using LD

In this case, it extracts information from the sources of LD to enrich the knowledge about the ToC identified from the PoQ entry (see macro-algorithm in Table 3). This process starts when CM activates QM to generate two queries (steps 1.1 and 1.2 in Table 3), these queries are generated following the process described in the QM component to extract knowledge. On one hand, it generates a query that extracts from the local source of LD the symptoms detected in the *user_A* by doctors or sensors (Figure 5).

On the other hand, it generates a query that extracts a list of diseases with their symptoms, from the Live-Dbpedia source of LD (see Figure 6).

Finally, CM executes the queries (steps 1.3 and 1.4 in Table 3). Table 8 shows a small part of the data extracted about the diseases, with their symptoms.

TABLE 7 New terms or concepts identified and related to their URIs using linked data

ToC		URI
disease		dbo:disease
symptom		dbo:symptom
isSymptom		dbo:symptom
Subject	**Property**	**Object**
disease	isSymptom	symptom
opinionDoctor	isSymptom	symptom

Note: dbo: http://dbpedia.org/ontology/.

```
sparql = SPARQLWrapper2("http://localhost/sparql")
sparql.setQuery("""
PREFIX local: <http://localhost/>
SELECT ?symptom ?doctor
WHERE {
   local:user_A foaf:publications ?opinionDoctor .
   ?opinionDoctor foaf:maker ?doctor .
   ?opinionDoctor dbo:symptom ?symptom .
}
""")
```

FIGURE 5 Query to extract the symptoms detected in the user_A

```
sparql = SPARQLWrapper2("http://live.dbpedia.org/sparql")
sparql.setQuery("""
SELECT ?disease ?symptom
WHERE{
     ?disease dbo:symptom ?symptom .
}
order by ?disease
""")
```

FIGURE 6 Query that generates the list of diseases and their symptoms

TABLE 8 Diseases with their symptoms extracted from linked data

Disease	Symptom
dbpedia_Volvulus	dbpedia_Bloating
dbpedia_Volvulus	dbpedia_Constipation
dbpedia_Zika_fever	dbpedia_Conjunctivitis
dbpedia_Zika_fever	dbpedia_Fever

```
fof(isSymptom_type,axiom,(
    ! [D] :
      ( disease(D)
     => ? [S] :
          ( symptom(S)
          & isSymptom(D,S) ) ) )).

fof(isSymptom1, axiom, isSymptom(dbpedia_17Hydroxysteroid_dehydrogenase_III_deficiency , dbpedia_Hypothyroidism) ).
                       :                                :                                :
fof(isSymptom1333, axiom, isSymptom(dbpedia_Volvulus , dbpedia_Bloating) ).
fof(isSymptom1334, axiom, isSymptom(dbpedia_Volvulus , dbpedia_Constipation) ).
fof(isSymptom1335, axiom, isSymptom(dbpedia_Volvulus , dbpedia_Bloody_stool) ).
fof(isSymptom1336, axiom, isSymptom(dbpedia_Volvulus , dbpedia_Abdominal_pain) ).
                       :                                :                                :
fof(isSymptom1356, axiom, isSymptom(dbpedia_Zika_fever , dbpedia_Conjunctivitis) ).
fof(isSymptom1357, axiom, isSymptom(dbpedia_Zika_fever , dbpedia_Fever) ).
fof(isSymptom1358, axiom, isSymptom(dbpedia_Zika_fever , dbpedia_Maculopapular_rash) ).
fof(isSymptom1359, axiom, isSymptom(dbpedia_Zika_fever , dbpedia_Arthralgia) ).
```

FIGURE 7 Information on diseases and their symptoms converted to dialetheic logic engine

```
%%%%%% Conjecture
fof(i,conjecture,( sickDoctors(user_A, doctor_A, doctor_B) )).
fof(ii,conjecture,( diseaseDoctors(user_A, D, doctor_A, doctor_B) )).
```

FIGURE 8 Conjectures to verify and filter the data

This process is automated in Python, for which it uses the template described in QM "Knowledge Extraction", where it replaces the word URI_PROPERTY with the URI of the ToC that needs to extract the knowledge. Then, it proceeds to execute the search query on the endpoint of the LOD source (e.g. http://live.dbpedia.org/sparql). This process is repeated with all the ToCs that need to extract knowledge.

5.3 | Verification and filtering of recommendations using DL

This process verifies the inconsistencies and/or ambiguities of the extracted data, and filters them based on the DE found by DiLE (see macro-algorithm in Table 4). However, to achieve this goal, ConM and QM must be activated. ConM (step 2.1 in Table 4) transforms the data extracted by DeLE (see Table 8) to DiLE specifications, based on axioms and facts. This transformation process has been explained in the ConM component. Figure 7 shows part of the result of the conversion of diseases and their symptoms, where the axiom isSymptom_type represents the relationship between disease and symptom, and the rest are the facts that represent the extracted data.

Figure 8 shows the conjectures extracted from the PoQ by the QM (step 3.1 in Table 4). With these conjectures, DiLE will be able to reason in the next step with the model and its data, allowing filtering of recommendations.

Now, the RM can invoke the DiLE to verify and filter the recommendations (step 4 in Table 4). In the case of conjecture I (see Figure 8), DiLE determines that user_A is sick based on the opinion of doctor_A and doctor_B. In the case of conjecture II (see Figure 8), DiLE tests every disease (varying the value of D in conjecture II) that it has in the knowledge base, to verify and filter the diseases to be recommended. Figure 9 shows the verification of two diseases (dbpedia_Zika_fever and dbpedia_Volvulus) with conjecture ii. For dbpedia_Zika_fever DiLE determines that it is a disease to be recommended (YES), since in both the doctors determine that there are symptoms associated with the disease. For dbpedia_Volvulus DiLE determines that it is not a disease to recommend (NO), since only doctor_B opined that it has symptoms associated with that disease.

Finally, Table 9 shows the result achieved, after DiLE filtered and sorted the recommended diseases that is, the cases that gave YES with conjecture II.

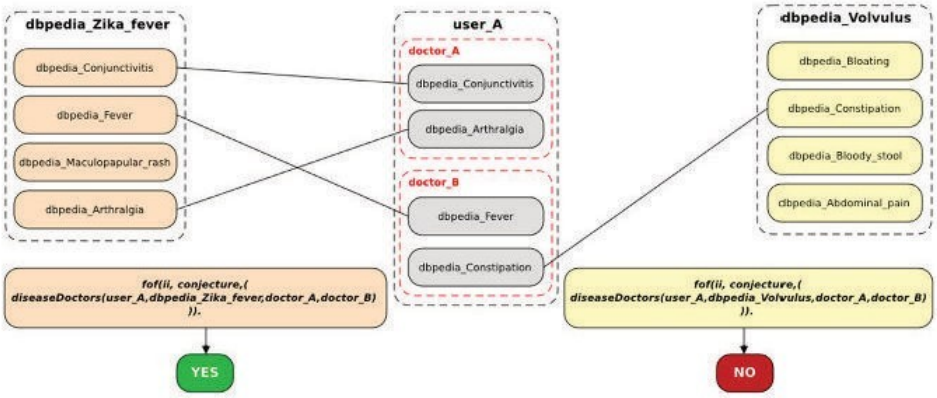

FIGURE 9 Comparison of the result of two conjectures with respect to the data

TABLE 9 Recommendation achieved with dialetheic logic engine

Recommendation	Ranking
dbpedia_Zika_fever	3
dbpedia_Chikungunya	2
dbpedia_Measles	2
dbpedia_Rheumatic_fever	2
dbpedia_Trichinosis	2

```
sparql = SPARQLWrapper2("http://live.dbpedia.org/sparql")
sparql.setQuery("""
SELECT ?property ?object
WHERE {
   dbr:Zika_fever ?property ?object .
}
""")
```

FIGURE 10 Query to extract data associated with Zika_fever

The processes described in this section are automated in Python. They are the most complex processes, and are the core of the HRS. The first process (ConM component) transforms the enriched properties to axiom; in this particular case, the isSymptom property (see PoQ in the case study), which has associated two concepts: disease and symptom (concepts that were obtained using the URI_PROPERTY in the previous section), being as follows: isSymptom(disease, symptom). The second process executes the JGRM3 reasoner with the information held by PoQ and the enriched properties transformed into axioms. Finally, the items recommended by the reasoner are stored in a list and sorted by their ranking.

5.4 | Content extraction related to recommendations using LD

In this case, it is extracted new information from the LD sources related to the URI representing each disease reached as a recommendation (see macro-algorithm in Table 5). For this, RM activates QM to generate a query that allows extracting such information (step 2 in Table 5). This query is generated following the process described in the QM component to extract knowledge. Figure 10 shows the query that searches and extracts all the content related to the URI that identifies Zika Fever, such as symptoms, treatments, cause, among others (step 2.1 in Table 5).

RM collects all the information extracted with the previous Zika Fever query (steps 3 and 4 in Table 5), and associates it with the URI of said disease (see Table 10). This process of extraction and association is repeated with each of the diseases recommended by the HRS.

TABLE 10 Extracted linked data content on Zika fever

Disease	Property	Value
dbpedia_Zika_fever	Duration	Less than a week
	Deaths	None during the initial infection
	Prevention	Decreasing mosquito bites, condoms
	Diagnosis	Testing blood, urine, or saliva for viral RNA or blood for antibodies
	Complications	During pregnancy can cause microcephaly, Guillain-Barré syndrome
	Symptom	Conjunctivitis
		Fever
		Maculopapular_rash
		Arthralgia
	Differential diagnosis	Leptospirosis
		Measles
		Malaria
		Chikungunya
		Dengue
	Treatment	Supportive_care

Finally, the RM delivers all collected information to the user.

Also, this process is automated in Python based on the template described in QM "Knowledge Extraction", for which it replaces the word URI_CONCEPT with the URI of the item recommended by the system. Then, it proceeds to execute the search query at the endpoint of the LOD source (e.g. http://live.dbpedia.org/sparql). This process is repeated with all the recommended items.

6 | ANALYSIS OF RESULTS

6.1 | Analysis of the HRS

In this section, we analyse the behaviour of our HRS and evaluate the recommendations achieved using the information extracted from LD. In general, the process is composed of three steps: (1) General description of the data extracted from the sources of LD with DeLE. (2) Recommendations reached with the DE detected by DiLE. (3) Assessment of the recommendations reached through the hybrid reasoning mechanism. To show in detail each step, the data and the model described in the previous section will be used.

6.1.1 | General description of the data extracted from the sources of LD with DeLE

Table 11 shows the state of the knowledge about diseases, symptoms, and the relationship between diseases/symptoms. In the column *Before*, the knowledge of the HRS before the extraction of the external sources of LD is indicated, where only the profile of the user is known, with its 4 symptoms detected by doctor_A (dbpedia_Conjunctivitis, and dbpedia_Arthralgia) and doctor_B (dbpedia_Constipation and dbpedia_Fever). In the *After* column, the knowledge extracted by HRS from external sources of LD based on Dbpedia is indicated, where the system now recognizes 523 diseases, 544 symptoms, and 1359 disease/symptom relationships.

The computational procedure to obtain the knowledge in Table 11 involves several processes, such as detection, extraction, and transformation of data. The strongest complexity lies in the ability to detect each concept needed in the model studied and the relationship of these concepts with the URI that represents it. Once this knowledge is obtained, the extraction and transformation are quite simple and fast.

6.2 | Recommendations reached with the DE detected by DiLE

In the second step (see Table 12), it presents the DE detected by DiLE. The DE represents the ambiguities found by the reasoner through conjecture II (diseaseDoctors). In the particular case of this problem, these DE represent the diseases (items) to be recommended, since DiLE identifies

in each of the possible recommendations their state of ambiguity, that is, if there are or no symptoms associated with each disease. Specifically, Table 12 presents two things, in the first one, it is observed for each symptom (1.dbpedia_Conjunctivitis, 2.dbpedia_Arthralgia, 3. dbpedia_Constipation and 4.dbpedia_Fever) of the user_A the number of diseases associated with those symptoms. On the other hand, it is observed the number of diseases that DiLE has recommended when detecting DE cases.

In this step, the computational costs to reason and determine the recommended items (see Table 12) are quite high, because the DL reasoning engine (JRM3) in the first stage performs an exhaustive process of verification of the model and the data to be used; and in the second stage, it makes tests to determine the cases of ambiguities and to find the possible items to be recommended.

6.2.1 | Assessment of the recommendations reached through the hybrid reasoning mechanism

Finally, step 3 assesses the recommendations achieved. On one hand, the concept of "optimal completeness" is used to validate that the recommendations follow the characteristics of the user profile. On the other hand, Precision, Recall and *F*1-measure are used to determine the quality of the HRS.

Optimal Completeness (OC): A retrieval strategy is optimally complete if it ensures the retrieval of the most similar case, if any, which satisfies all the hard constraints in a given query (McSherry, 2006). In our case, the recommendation given by the HRS is considered optimally complete regarding the profile of a user, if the characteristics of the recommended items (R_C) are similar to the characteristics associated with the user profile (U_C), satisfying all the hard constraints of a given query. The optimal completeness is:

$$OC = R_C / U_C.$$

The completeness value would be between zero and one, such that the closer the value is to one, then they are more similar. Table 13 shows in detail the diseases recommended by the HRS for the joint decision by doctors A and B, according to the symptoms detected by each one in user_A, where the total of such symptoms is 4 that is, $U_C = 4$. Zero represents that this symptom is not present in the disease, and one represents the opposite. Let us see, in detail, the calculation for the disease dbpedia_Zika_fever: we observe that 3 symptoms of the user_A are associated with this disease (the real symptoms of the diseases were extracted from the LD sources), then, R_C is equal to 3. Therefore, the completeness value would be ¾ = 0.75.

TABLE 11 Summary of data collected by description logic engine from the linked data sources

	Before	After
Diseases	0	523
Symptoms	4	544
Relationship disease/symptom	0	1359

TABLE 12 Dialetheic contradiction level of the user

user_A	Symptoms	1	2	3	4	Recommendations/DE
	Disease	3	3	10	86	5

TABLE 13 Optimal completeness of recommendations submitted by our hybrid recommender system

user_A							
	Symptoms						
Disease	1	2	3	4	R_C	U_C	OC
dbpedia_Zika_fever	1	1	0	1	3	4	0.75
dbpedia_Chikungunya	0	1	0	1	2	4	0.5
dbpedia_Measles	1	0	0	1	2	4	0.5
dbpedia_Rheumatic_fever	0	1	0	1	2	4	0.5
dbpedia_Trichinosis	1	0	0	1	2	4	0.5

Now, to evaluate the quality of the HRS, *we use the metrics Accuracy (P), Recall I and F1-measure (F1)*. These metrics are calculated from a contingency table that classifies items with respect to the information needs of the user (see Table 14) (Ricci et al., 2011), distinguishing two groups: *relevant* or *not relevant*. In this case, the relevant items are the diseases that present some of the symptoms associated with the user's profile. Furthermore, the items are also classified according to whether they are recommended to the user (*recommended*) or not (*not recommended*).

Table 15 presents the quality evaluation of the HRS using the recommendations reached in 11 users. However, we have extended the number of users, despite the high computational cost due to the DL reasoning engine based on JRM3. In the *Recommended* column, it indicates the number of diseases recommended by the HRS. In *Relevant*, it indicates the number of diseases that have an associated symptom present in the user profile. Finally, the values reached in the metrics in *P*, *R* and *F*1 are presented. About *P*, it is interpreted that the HRS always recommends diseases that are relevant to the user. For *R*, when the value is higher, more user-relevant diseases have been recommended by HRS. For *F*1, when the value is higher, then the HRS has the best behaviour, according to the *P* and *R* metrics.

In general, our HRS has a high *P*, recommends all the relevant diseases, and an acceptable *F*1 value (0.83). In general, the value of recall is not very bad, although in some cases the value is low (0.54). That is, in some cases the sensitivity is not very good, such as the total amount of relevant recommendations successfully given is not high. It can be due to a lack of information about the disease, among other things. However, the main point is that our system can eliminate the information do not relevant for all cases (see column *tn* in Table 15). The results for 100 users follow the same trend as for the 11 users shown in Table 15. Thus, the HRS always manages to reach a $P = 1$ since the inference engine always recommends relevant information about the diseases. For *R*, the average is equal to 0.75, which indicates that almost all the relevant information is recommended, but despite the management of uncertainty and linked data, there is still some relevant information that is not completely linked by our HRS. Finally, the average of *F*1 is 0.86, which is affected by the value of R.

6.3 | Analysis of the PoQ input

In this section, we analyse some models of the PoQ input to our HRS, to study the different states of DL using LD. This article only tested the dialetheic state called *Fictional Discourse*, where the model "*It seeks to determine if a person is sick according to certain assumptions of Doctors*" was implemented. Also, for the "*Fictional Discourse*" case can be studied:

1. I have the D disease because I have the same symptoms as another user.
2. I will take the M medicine because I feel sick with D.

TABLE 14 Contingency table

	Recommended	Not recommended
Relevant	True-Positive (tp)	False-Negative (fn)
Not relevant	False-Positive (fp)	True-Negative (tn)

TABLE 15 Quality of the hybrid recommender system

User	Recommended	Relevant	Tp	fp	fn	tn	*P*	*R*	*F1*
user_1	26	50	26	0	24	473	1	0.54	0.7
user_2	28	50	28	0	22	463	1	0.6	0.75
user_3	5	7	4	0	2	520	1	0.72	0.83
user_4	22	38	22	0	16	485	1	0.6	0.75
user_5	86	100	86	0	14	423	1	0.86	0.92
user_6	14	23	14	0	9	500	1	0.58	0.73
user_7	17	24	17	0	7	499	1	0.73	0.84
user_8	5	7	5	0	2	520	1	0.72	0.83
user_9	5	5	5	0	0	518	1	1	1
user_10	12	20	12	0	8	503	1	0.6	0.75
user_11	43	51	43	0	8	472	1	0.84	0.91
Average							1	0.72	0.83

For the "*Contingent statements about the future*" case can be studied:

1. I'll be sick tomorrow...
2. Tomorrow, Will I be sick with the D disease?
3. Next month, Will I have the S symptoms of disease D?

For the "*Failure of a presupposition*" case:

1. If I have symptoms, then I'm sick.
2. If I have the S symptom is that I'm sick with D.

For the "*Vagueness*" case:

1. He is very sick
2. He's a little sick.
3. He has a lot of symptoms

Or, for the "*Counterfactual reasoning*" case:

1. If I had not gotten wet, then I would not be sick and I would not have to take medicine now.
2. If I had taken the medicine, then I would not be sick and now I would be cured.
3. If I had gone to the doctor, then I would not be sick and now I'd be in the P place.

In general, the implementation of all these PoQs requires the definition of axioms and assumptions for each case. In addition, it is required to use a source of LD with the medical history of the users, the symptoms, diseases, drugs used, and vaccines, among others, with their respective dates. Let us analyse each case:

1. *Contingent statements about the future*: Determine whether or not a person might have a disease in the future, knowing that the disease occurred repeatedly or not. For example, acne (dbpedia_Acne) appears in people at certain times in their lives.
2. *Failure of a presupposition*: Even if we know that a person has a disease that does not repeat itself in life, it again had occurred. For example, varicella (dbpedia_Chickenpox) when occurs in a person with little aggressiveness, the body does not generate the necessary antibodies to be immune all life long, and the disease can occur again.
3. *Vagueness*: How to know when a person had aggressively an illness or not. In the example of varicella, it can be crucial, to know if such a disease could present again.
4. *Counterfactual reasoning*: By knowing what the patient had in the past or not, it would allow building scenarios of what might or might not have happened. A person who has not been vaccinated against a disease is more likely to have it in the future.

6.4 | Comparative analysis

This section carries out a comparison with the works in the literature. For that, in this paper are identified the following four evaluation criteria:

1. C1: Mechanisms used to obtain recommendations.
2. C2: Utilization of LD as a source of Knowledge.
3. C3: Recommendation of complementary information.
4. C4: Utilization of a Knowledge Model with inconsistency/ambiguity states.

In Table 16, the *Evaluation* column presents the metrics used for the evaluation. In the *Criteria* column for C1, VS is Vector Space, GB is Graph-Based, CBF is Content-Based Filtering, CF is Collaborative Filtering. Similarly, in the *Criteria* columns for C2, C3 and C4, ✓ means the criterion is met, and x means the criterion is not met.

For criterion C1, the reviewed works show that the mechanisms used to obtain the recommendations fall into two groups. In the first group, the mechanisms are focused on representing and enriching the knowledge for the recommendations. In this case, the vast majority of the papers are based on graphs, except for Musto, Franza, et al. (2018) and Zitouni et al. (2017), which are based on vectors to represent and characterize the recommendations. In the second group, the mechanisms are focused on filtering the recommendations, using only one type of RS (Chicaiza et al., 2017; Musto,

TABLE 16 Comparative analysis with previous works

Work	Evaluation	Criteria			
		C1	C2	C3	C4
Musto, Basile, et al. (2017)	*F*1-measure, mean average precision (MAP) and intra-list diversity (ILD)	GB, CBF	✓	x	X
Musto, Lops, et al. (2017)	*F*1-measure and normalized discounted cumulative gain (nDCG)	GB, CBF, CF	✓	x	X
Chicaiza et al. (2017)	Not indicated	GF, CBF	✓	x	X
Pereira et al. (2018)	Acceptance criteria, precision	GF, CBF, CF	✓	x	X
De Angelis et al. (2017)	nDCG@n	GF, CBF, CF	✓	x	X
Fogli et al. (2018)	Precision, novelty, non-obviousness and serendipity	GF, CBF, CF	✓	✓	X
Musto, Narducci, et al. (2018)	Transparency, persuasion, engagement, trust and effectiveness	GF, CBF	✓	x	X
Natarajan et al. (2020)	F-measure and precision	CF	✓	x	X
García-Sánchez et al. (2020)	F-measure and precision	CBF	✓	x	X
Zitouni et al. (2017)	Mean absolute error (MAE)	VS, CBF	✓	x	X
Musto, Franza, et al. (2018)	*F*1@k-measure	VS, CBF	✓	x	X
Selvan et al. (2019)	Precision, recall, accuracy and *F*1-measure	GB, CBF	✓	x	X
Sebbaq et al. (2020)	Similarity measures	CBF	✓	✓	X
Ammar et al. (2021)	Precision	CBF	✓	x	X
Chew et al. (2021)	Data sparsity and root mean square error	CF	✓	x	X
Our approach	Optimal completeness, precision, recall and *F*1-measure	GB, CBF	✓	✓	✓

Basile, et al., 2017; Musto, Franza, et al., 2018; Musto, Narducci, et al., 2018; Rahayu et al., 2022; Selvan et al., 2019; Zitouni et al., 2017), or combining various RSs (De Angelis et al., 2017; Fogli et al., 2018; Musto, Lops, et al., 2017; Pereira et al., 2018) that combine CBF with CF.

For C2, all works use LD, whether the sources are based on the LD paradigm or transformed into LD. Besides, these sources are very varied, ranging from public or private sources to general or specific sources. For C3, it is observed that only (Fogli et al., 2018; Sebbaq et al., 2020) and our proposal take advantage of these interrelations to offer complementary information extracted from the same sources of LD, allowing the extension of the information that is presented to the users about the recommendations reached (see Sections .4.2.4 and 5.4). On the other hand, for C4, only our work shows capabilities to solve inconsistencies and/or ambiguities that are present in the reasoning model, and this is achieved thanks to the DL reasoner. Other fundamental aspects are the flexibility of the proposed architectures to incorporate other reasoning methods (e.g. fuzzy using paradigms such as fuzzy cognitive maps (Aguilar, 2001)), feature engineering processes (Pacheco et al., 2014), or migrate the architecture to an autonomic scheme (Monsalve-Pulido et al., 2020). Our architectural proposal can be easily extended with these characteristics.

Concerning the evaluation, all papers present a set of metrics to evaluate the different characteristics of their RSs, except for (Chicaiza et al., 2017). Our approach uses the same metrics as other RSs (see Table 17), like the papers (Ammar et al., 2021; Fogli et al., 2018; García-Sánchez et al., 2020; Natarajan et al., 2020; Pereira et al., 2018; Selvan et al., 2019) that use Precision metric, (Selvan et al., 2019) that use Recall, or (García-Sánchez et al., 2020; Musto, Basile, et al., 2017; Musto, Franza, et al., 2018; Musto, Lops, et al., 2017; Natarajan et al., 2020; Selvan et al., 2019) that use *F*1-measure, but each paper has been used in a different context and dataset to solve the recommendations. Now, we compare these papers with our proposal using the value reported for these metrics (see Table 17). When comparing the papers with respect to their metrics, our work outperforms all the papers by a wide range, with the exception of (Selvan et al., 2019), where our work is better in the Precision metric and inferior in the Recall and *F*1-measure metrics, but it can also be observed that these difference are minimal. It is also important to highlight that there is no previous work that evaluates the quality of the generated ontology (see works, Chew et al., 2021; Mahdi & Hadi, 2021; Selvan et al., 2019), only McSherry (2006) and our work proposes a metric to evaluate it (completeness).

In synthesis, all works use LD as a source of knowledge because of its variety and semantics, either to characterize the elements to be recommended and/or the user's profiles. However, only Fogli et al. (2018), Sebbaq et al. (2020) and our approach take advantage of the LD to extract complementary information for the recommendation, which is very interesting because the user when receiving the recommendations will need to know them in-depth to make a decision. Finally, what makes our work unique with respect to the rest, is the capability acquired by using a DL reasoner, which allows reasoning in states of ambiguity and inconsistencies, which is one of the most difficult problems to face in RSs and in the LD paradigm.

The main aspect of introducing DL is that our HRS guarantees to deliver only relevant information in the recommendation process since it eliminates non-relevant information for all cases. For this task, the disambiguation process is crucial, since it allows recommending only relevant information at a given moment ($P = 1$), filtering out the non-relevant. This is vital in a recommendation process, since for certain contexts, it is mandatory to guarantee this (for example, in diagnostic tasks).

Regarding the relevant information that is not recommended in certain cases, that is that it is outside the recommendation, two aspects could be analysed. On the one hand, extend the LD process using dialetheic logic in this process (in principle, to guarantee the handling of uncertainty

TABLE 17 Comparative analysis with previous work using the same metrics as our approach

	Metrics			
Work	**Optimal completeness**	**Precision**	**Recall**	***F*1-measure**
Musto, Basile, et al. (2017)	x	x	x	0.550
Musto, Lops, et al. (2017)	x	x	x	0.569
Pereira et al. (2018)	x	0.750	x	x
Fogli et al. (2018)	x	0.820	x	x
Natarajan et al. (2020)	x	0.564	x	0.560
García-Sánchez et al. (2020)	x	0.856	x	0.792
Musto, Franza, et al. (2018)	x	x	x	0.653
Selvan et al. (2019)	x	0.957	**0.785**	**0.865**
Ammar et al. (2021)	x	0.500	x	x
Chew et al. (2021)	x	0.929	x	x
Our approach	0.750	**1**	0.720	0.830

The bold values represent the best results obtained with each metric.

during the semantic enrichment process). On the other hand, seek to extend the LD process with exhaustive information retrieval strategies, seeking to avoid leaving out any information related to the topic to be recommended. From there, the process of filtering the information recovered through the uncertainty management mechanism would then be carried out.

7 | CONCLUSIONS

In this research, it was developed an HRS based on the Description/Dialetheic Logic, which provides an innovative architecture for the management of LD in contexts of ambiguous reasoning, taking advantage of the data generated by the different sources of LD, either created by people, sensors or applications, allowing the integration of all this information to be exploited by the recommender. The utilization of LD allows enriching the knowledge of the concepts involved in the reasoning, by extracting from the different sources the semantic information, and then, relating them to those concepts. In our case study, the symptoms associated with the different diseases were obtained from the sources of LD. On the other hand, the reasoning mechanisms based on DL allow the recommender to handle situations of contradiction or inconsistency, which is one of the most difficult problems in order to determine what information to offer to users.

The modular architecture of this recommender allows including reasoners with different logics, increasing its capacity to give richer answers in different scenarios. Currently, the recommender has a description logic reasoner based on LD, and a DL reasoner based on RM3 logic. Likewise, the architecture specifies a set of managers that can be replaced or modified to improve or adapt to new reasoning mechanisms.

In the case study, it is shown how the HRS orchestrates the reasoners and managers to extract and process the knowledge obtained from the LD, and thus respond to the recommendation needs considering the states of ambiguity or inconsistency. In particular, it describes how concepts associated with PoQ are identified (see Section V.A), which are then used to extract information from LD sources (see Section 5.2). Having all the necessary knowledge to reason, it is processed with the DL reasoner that identifies the ambiguous cases, and reaches the recommendations based on the preferences of each user (see Section 5.3). Finally, it extracts from LD content related to each recommendation achieved (see Section 5.4). In each process is detailed the behaviour of the algorithms executed by the HRS, to arrive at the recommendations.

In comparison with other RS, all recommenders use description logic as an intrinsic mechanism for the exploitation of LD, and with respect to solve the ambiguities and/or inconsistencies, only our work has this capability (see Section 6.2). In our approach, we use a DL engine based on RM3 logic, which allows reasoning in the states of ambiguities or inconsistencies. Another detail is related to the extraction and enrichment, all the recommenders consider the use of LD as a source of knowledge, because of its variety and semantic, either to characterize the elements to be recommended and/or the user profiles. However, only one work and our proposal take advantage of the paradigm of LD to offer complementary information extracted from the same sources of LD, allowing the extension of the information that is presented to the users.

Future works will test the HRS by exploiting the different states handled by the DL, such as when the presuppositions fail (for diagnosis or detection of faults), in vagueness (lack of clarity in language), in counterfactual reasoning (thinking what could have been) and/or in contingencies about the future that are keys to handling historical data (that indicate that something was true and false in the past). Also, other data sources will be tested to observe system behaviour in these situations, and tests and evaluations will be made to determine the computational cost when using LD (when extracting and transforming data) and DL (when reasoning with JRM3). Particularly. by increasing the number of users who need recommendations, the computational cost increases since each recommendation needs to run the DL reasoning engine (JRM3), due to this engine is responsible for determining the possible ambiguities and items to recommend (process analysed in section V.A.2). A future work is to test other

DL engines to analyse their computational costs and the quality of their reasoning results, and to compare them with respect to the one used in this work.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Ricardo Dos Santos https://orcid.org/0000-0001-8752-8550
Jose Aguilar https://orcid.org/0000-0003-4194-6882

ENDNOTES

[i] http://vos.openlinksw.com
[ii] http://live.dbpedia.org

REFERENCES

Aguilar, J. (2001). A fuzzy cognitive map based on the random neural model. Lecture Notes in Computer Science, Vol. *2070*, pp. 333–338.
Aguilar, J. (2011). *Temporal logic from the chronicles paradigm: Learning and reasoning problems, and its applications in distributed systems*. LAP Lambert Academic Publishing.
Aguilar, J., Jerez, M., & Rodriguez, J. (2018). CAMeOnto: Context awareness meta ontology modeling. *Applied Computing and Informatics*, *14*(2), 202–213. https://doi.org/10.1016/j.aci.2017.08.001
Aguilar, J., Valdiviezo-Díaz, P., & Riofrio, G. (2017). A general framework for intelligent recommender systems. *Applied Computing and Informatics*, *13*(2), 147–160. https://doi.org/10.1016/j.aci.2016.08.002
Ammar, N., Bailey, J., Davis, R., & Shaban-Nejad, A. (2021). Using a personal health library-enabled mHealth recommender system for self-Management of diabetes among underserved populations: Use case for knowledge graphs and linked data. *JMIR Formative Research*, *5*(3), e24738. https://doi.org/10.2196/24738
Basile, P., Greco, C., Suglia, A., & Semeraro, G. (2019). Bridging the gap between linked open data-based recommender systems and distributed representations. *Information Systems*, *86*, 1–8. https://doi.org/10.1016/j.is.2019.07.001
Beevers, T. (2020). Vagueness-induced counterexamples to modus Tollens. *Proceedings of the Aristotelian Society*, *120*(3), 405–416. https://doi.org/10.1093/arisoc/aoaa005
Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J, Lerer A., Sheets, D. (2006). Tabulator: Exploring and analyzing linked data on the semantic web. Proceeding 3rd International Semantic Web User Interaction Workshop. pp. 159–174.
Chew, L., Haw, S., & Subramaniam, S. (2021). A hybrid recommender system based on data enrichment on the ontology modelling. *F1000Research*, *10*, 937. https://doi.org/10.12688/f1000research.73060.1
Chicaiza, J., Piedra, N., Lopez-Vargas, J., & Tovar-Caro, E. (2017). Recommendation of open educational resources. An approach based on linked open data. *Proceedings of the IEEE global engineering education conference*, pp. 1316-1321. https://doi.org/10.1109/educon.2017.7943018
De Angelis, A., Gasparetti, F., Micarelli, A., & Sansonetti, G. (2017). A social cultural recommender based on linked open data. *Proceeding of the 25th Conference on User Modeling, Adaptation and Personalization*, pp. 329–332. https://doi.org/10.1145/3099023.3099092
Dos Santos, R., & Aguilar, J. (2018). Enlazado de datos. In J. Aguilar (Ed.), *Introducción a la Mineria Semántica. Fondo Editorial Universidad Nacional Experimental del Táchira* (pp. 177–216). Fondo Editorial Universidad Nacional Experimental del Táchira.
Dos Santos, R., Aguilar, J., & Puerto, E. (2021). A meta-learning architecture based on linked data. *Proceedings of the 2021 XLVII Latin American computing conference (CLEI)*. pp. 1–10. https://doi.org/10.1109/CLEI53233.2021.9640223
Dos Santos, R., Aguilar, J., & Rodríguez, T. (2019). Una revisión de la literatura sobre datos enlazados. *Revista Ingeniería al Día*, *5*(1), 54–82.
Fogli, A., Micarelli, A., & Sansonetti, G. (2018). Enhancing itinerary recommendation with linked open data. *Communications in Computer and Information Science*, *850*, 32–39. https://doi.org/10.1007/978-3-319-92270-6_5
García-Sánchez, F., Colomo-Palacios, R., & Valencia-García, R. (2020). A social-semantic recommender system for advertisements. *Information Processing & Management*, *57*(2), 102153. https://doi.org/10.1016/j.ipm.2019.102153
González-Eras, A., Dos Santos, R., & Aguilar, J. (2022). Evaluation of digital competence profiles using dialetheic logic. *International Journal of Artificial Intelligence in Education*, *1-29*. https://doi.org/10.1007/s40593-021-00286-8
Jiménez, C., Jerez, M., Aguilar, J., & García, R. (2019). Linked data and dialetheic logic for localization-aware applications. *Contemporary Engineering Sciences*, *12*(3), 103–116. https://doi.org/10.12988/ces.2019.9515
Kamide, N., & Wansing, H. (2015). *Proof theory of N4-related paraconsistent logics* (Vol. 54). College Publications.
Laenen, K., & Moens, M. (2020). A comparative study of outfit recommendation methods with a focus on attention-based fusion. *Information Processing & Management*, *57*(6), 102316. https://doi.org/10.1016/j.ipm.2020.102316
Lukasiewicz, J., & Wedin, V. (1971). On the principle of contradiction in Aristotle. *The Review of Metaphysics*, *24*(3), 485–509.
Mahdi A., Hadi A. (2021) The current state of linked data-based recommender systems, *Proceedings of the 2nd information technology to enhance e-learning and other application*, 154–160, doi: https://doi.org/10.1109/IT-ELA52201.2021.9773738
McSherry, D. (2006). Completeness criteria for retrieval in recommender systems. In *Advances in case-based reasoning (ECCBR 2006)* (Vol. 4106, pp. 4109–4129). Springer. https://doi.org/10.1007/11805816_2
Monsalve-Pulido, J., Aguilar, J., Montoya, E., & Salazar, C. (2020). Autonomous recommender system architecture for virtual learning environment. *Applied Computing and Informatics*. https://doi.org/10.1016/j.aci.2020.03.001
Musto, C., Basile, P., Lops, P., de Gemmis, M., & Semeraro, G. (2017). Introducing linked open data in graph-based recommender systems. *Information Processing & Management*, *53*(2), 405–435. https://doi.org/10.1016/j.ipm.2016.12.003

Musto, C., Franza, T., Semeraro, G., de Gemmis, M., & Lops, P. (2018). Deep content-based recommender systems exploiting recurrent neural networks and linked open data. *Proceedings of the 26th conference on user modeling, adaptation and personalization*. pp. 239–244. https://doi.org/10.1145/3213586.3225230

Musto, C., Lops, P., de Gemmis, M., & Semeraro, G. (2017). Semantics-aware recommender systems exploiting linked open data and graph-based features. *Knowledge-Based Systems*, *136*, 1–14. https://doi.org/10.1016/j.knosys.2017.08.015

Musto, C., Narducci, F., Lops, P., de Gemmis, M., & Semeraro, G. (2018). Linked open data-based explanations for transparent recommender systems. *International Journal of Human-Computer Studies*, *121*, 93–107. https://doi.org/10.1016/j.ijhcs.2018.03.003

Najafabadi, M., Mohamed, A., & Onn, C. (2019). An impact of time and item influencer in collaborative filtering recommendations using graph-based model. *Information Processing & Management*, *56*(3), 526–540. https://doi.org/10.1016/j.ipm.2018.12.007

Natarajan, S., Vairavasundaram, S., Natarajan, S., & Gandomi, A. (2020). Resolving data sparsity and cold start problem in collaborative filtering recommender system using linked open data. *Expert Systems with Applications*, *149*, 113248. https://doi.org/10.1016/j.eswa.2020.113248

Oliveira G., Araújo F., Capretz M. (2019) PLDSD: Personalized linked data semantic distance for LOD-based recommender systems. *Proceedings of the 21st international conference on information integration and web-based applications & services*, pp. 294–303. https://doi.org/10.1145/3366030.3366041

Pacheco, F., Rangel, C., Aguilar, J., Cerrada, M, & Altamiranda, J (2014). Methodological framework for data processing based on the data science paradigm, *Proceedings of the XL Latin American computing conference*, doi: https://doi.org/10.1109/CLEI.2014.6965184.

Pelletier F. J., Sutcliffe G., Hazen A. P. (2017). Automated reasoning for the Dialetheic logic RM3. *Proceeding 30th International Florida Artificial Intelligence Research Society Conference*, pp. 110–115.

Pereira, C. K., Campos, F., Ströele, V., David, J. M. N., & Braga, R. (2018). BROAD-RSI—Educational recommender system using social networks interactions and linked data. *Journal of Internet Services and Applications*, *9*(1), 1–28. https://doi.org/10.1186/s13174-018-0076-5

Pulcini, G., & Varzi, A. C. (2018). Paraconsistency in classical logic. *Synthese*, *195*(12), 5485–5496. https://doi.org/10.1007/s11229-017-1458-0

Rahayu, N. W., Ferdiana, R., & Kusumawardani, S. S. (2022). A systematic review of ontology use in E-learning recommender system. *Computers and Education: Artificial Intelligence*, *3*, 100047. https://doi.org/10.1016/j.caeai.2022.100047

Ricci, F., Kantor, P. B., Rokach, L., & Shapira, B. (2011). *Recommender Systems Handbook*. Springer. https://doi.org/10.1007/978-0-387-85820-3

Ristoski, P. (2019). *Exploiting semantic web knowledge graphs in data mining* (Vol. 38). IOS Press.

Rodriguez, T., Dos Santos, R., Aguilar, J. (2017). Metodología para el desarrollo de aplicaciones web utilizando datos enlazados. *Proceeding Conferencia Nacional de Computación, Informática y Sistemas*, pp. 114-122.

Sebbaq H., el Faddouli N., Bennani S. (2020) Recommender system to support MOOCs teachers: Framework based on ontology and linked data. *Proceedings of the 13th international conference on intelligent systems: Theories and applications*. https://doi.org/10.1145/3419604.3419619

Selvan, N. S., Vairavasundaram, S., & Ravi, L. (2019). Fuzzy ontology-based personalized recommendation for internet of medical things with linked open data. *Journal of Intelligent Fuzzy Systems*, *36*(5), 4065–4075. https://doi.org/10.3233/jifs-169967

Van Rossum B., Frasincar F. (2019) Augmenting LOD-based recommender systems using graph centrality measures. Lecture Notes in Computer Science, Vol. *11496*. pp. 19–31. https://doi.org/10.1007/978-3-030-19274-7_2

Velarde, J. (1977). La Lógica dialéctica. *Teorema: Revista internacional de filosofía*, *7*(2), 129–140.

Yochum, P., Chang, P., Gu, G., & Zhu, M. (2020). Linked open data in location-based recommendation system on tourism domain: A survey. *IEEE Access*, *8*, 16409–16439. https://doi.org/10.1109/ACCESS.2020.2967120

Zitouni, H., Meshoul, S., & Taouche, K. (2017). Improving content based recommender systems using linked data cloud and FOAF vocabulary. *Proceedings of the international conference on web intelligence*. pp. 988-992. https://doi.org/10.1145/3106426.3120963

AUTHOR BIOGRAPHIES

Ricardo Dos Santos obtained his degree in Systems Engineering in 2006 and Master's in Computer Science in 2016, at the Universidad de Los Andes, Venezuela. He is currently a PhD candidate in Applied Sciences in the Universidad de Los Andes, Venezuela and Associate Professor at Universidad Politécnica Territorial del Estado Trujillo "Mario Briceño Iragorry", Trujillo, Venezuela. He is a researcher at the Centro Estudios en Microelectrónica y Sistemas Distribuidos (CEMISID), and at Tepuy R+D Group, Artificial Intelligence Software Development. His research interests include artificial intelligence, linked data, big data, machine learning, semantic mining.

Jose Aguilar received his Systems Engineer degree from the Universidad de Los Andes, Venezuela, in 1987, MSc in Computer Science from Universite Paul Sabatier, France, in 1991, and PhD in Computer Science from Universite Rene Descartes, France, in 1995. He completed his postdoctoral studies at the University of Houston, USA, in 2000, Laboratoire d'analyse et d'architecture des systemes (LAAS), France in 2011. Currently, he has a MSCA Postdoctoral Fellowship at the Universidad de Alcala, Spain. He is a Full Professor at the Universidad de Los Andes, Venezuela and Universidad EAFIT, Colombia. His research interests include artificial intelligence, big data, emerging computing and intelligent environments. He has published more than 650 papers in journals, books and proceedings of international conferences.

How to cite this article: Santos, R. D., & Aguilar, J. (2022). A hybrid recommender system based on description/dialetheic logic and linked data. *Expert Systems*, e13143. https://doi.org/10.1111/exsy.13143

8.6 Anexo 4.B: Evaluation of digital competence profiles using dialetheic logic

International Journal of Artificial Intelligence in Education
https://doi.org/10.1007/s40593-021-00286-8

ARTICLE

Evaluation of Digital Competence Profiles Using Dialetheic Logic

Alexandra González-Eras[1,2,3] · Ricardo Dos Santos[2,3] · Jose Aguilar[2,3,4,5]

Accepted: 9 November 2021

Abstract
Professional profiles are unstructured documents where the knowledge and experience of the editor predominate, presenting inconsistencies and ambiguities in terms of the competencies they contain, making complicated the recognition of knowledge and skills necessary for the proposal of university study programs. Also, the identification of knowledge and skills in digital academic profiles present difficulties due to their inconsistencies. This work proposes analyzing the contradictions or ambivalences found in the academic and professional competencies published in digital media (for example, web pages or social networks) through a model of axioms based on dialetheic logic. Notably, the model considers five types of natural language phenomena: Vagueness or ambiguity, presupposition failure, counterfactual reasoning, fictional discourse, and contingent statements about the future. In addition, the model uses lexical and semantic similarity measures in its analysis process. The dialetheic model is validated using several performance measures to determine its capability to find ambiguity in a competence ontology described using description logic. The results show that dialetheic logic is required to accurately interpret digital academic and professional profiles using computational reasoning mechanisms. The model applies in a Spanish context for computer science jobs, with the possibility to apply in other languages or domains, such as English, French, etc. Our model is a contribution for competencies management, which is useful for the automatic curriculum design, competencies validation in learning processes, among other uses.

Keywords Natural language processing · Dialetheic logic · Professional competencies · Ambiguous reasoning

✉ Jose Aguilar
aguilar@ula.ve

Extended author information available on the last page of the article

Published online: 21 January 2022

Introduction

At present, competence management finds complications in understanding the real meaning of competence in unstructured digital professional and academic profiles, where its significance depends directly on the knowledge and perception of the publisher. Thus, one of the main limitations is competence interpretation, leading to more than one meaning. For example, we find skills that describe different cognitive levels and processes simultaneously, generating that the publisher determines erroneous assumptions about these skills. In this way, we find ambiguous gaps in the alignment of competence terms, which constitute a natural language problem, affecting the construction of competence models.

In general, the lack of clarity of competencies causes ambivalences and misunderstandings, which do not allow them to create specific training or curricula (Kravcik et al., 2018). An educational syllabus requires definitions of domain and scope of knowledge and skills to acquire competencies (Guevara et al., 2017) correctly. On the other hand, universities require to update academic profiles with new labor market needs (Elchamaa et al., 2019). But if job requirements are ambiguous, how can academic profiles accomplish the desired competencies (Kondratova et al., 2017) and guarantee good validation processes for acquiring competencies (Gluga et al., 2013)? Thus, it is necessary to propose models for competence ambiguity recognition in the professional and academic profiles (Ramsauer, 2020).

Semantic models like ontologies modeling the relationships between skills and knowledge, providing a theoretical and contextual framework for creating professional and academic profiles (Miranda et al., 2017). Despite this, ontologies face lexical ambiguity problems between terms and concepts, causing ambivalences when unifying information from multiple sources (De Leenheer et al., 2007). Furthermore, ontologies use formal languages, such as the Description Logic, to describe concepts and their relationships by assigning them true or false values (Hassan et al., 2012; Malzahn et al., 2013; Montuschi et al., 2015; Sateli et al., 2017; Sikos, 2017), which limits the ability of an ontology to represent the meaning of ambiguous terms (Guo et al., 2016). Consequently, ontologies are not efficient for the representation of competence ambiguity in academic and professional profiles.

On the other hand, dialetheic logic is paraconsistent logic, which considers that logic - formulae can be true, false, or both. Applied to formal models, the arguments (axioms) in dialetheic logic allow contradictions and ambivalences that present several truth values simultaneously (Pulcini, 2018). In this sense, dialetheic logic enables the representation of contradictory or ambivalent events; consequently, dialetheic logic can be a valid alternative to model computationally the ambiguity presented by competencies.

There are investigations where the principles of dialetheic logic are the foundation for solving problems, mainly in decision support (Zamansky, 2019), providing a non-rigid notion of consistency on large knowledge bases and ontologies (Pelletier et al., 2017). On the other hand, there are methods for ontological

representation to deal with cases of uncertainty (Dubois, 2012) (Perozo et al., 2013), Vagueness (Lukasiewicz, 2008), and imprecision (Faes, 2019). An extension of description logic is carried out for all of them, defining a set of rules for probabilistic and Vagueness cases without testing in real contexts.

According to Pelletier et al. (2017), lexical ambiguity of five natural language phenomena is related to Vagueness, failure of a presupposition, counterfactual reasoning, fictitious discourse, and contingent statements about the future. Each phenomenon generates contradictions in the statements that make it impossible to obtain a single truth value. These phenomena are presented in the context of professional and academic profiles when publishers create ambivalent phrases according to their knowledge, experience, and beliefs. In this way, professional and academic profiles contain ambiguous competence statements, whose truth values can be true or false or both.

This work aims to develop a knowledge model to analyze the ambiguities of academic and professional profiles from dialetheic logic. First, we identify the cases of ambiguity in the competencies, and then, we create a knowledge model based on dialetheic axioms called the Dialetheic Model (DM). After, we apply the DM over a set of terms of competence belonging to a Competencies Ontology (OC) defined in General Descriptive Logic. Subsequently, using measures of Robustness and Entropy, we analyze the two models OC and DM, to determine their capabilities to recognize dialetheic terms.

Thus, the main contribution of this paper is the proposition of a model to analyze academic and professional competence profiles, using dialetheic logic and description logic, being an essential topic for educational technology. Intelligent educational systems and learning environments can use this DM to improve their reasoning capabilities; by computer-supported learning tools to support the learning processes via their personalization; by instructional design methods, curriculum administration systems, and intelligent tutoring systems to acquire and model the knowledge and skills, among other utilizations. In this way, the model contributes to the development of intelligent educational systems, allowing adequate management and validation of competencies during learning processes, curriculum management, among others.

Background

One of the competence management processes is the construction of professional profiles, which arise the set of competencies of the ideal individual to successfully fill a job position (Guo et al., 2016). With this information, companies plan the training their employees require, and universities develop their academic profiles (Malzhan et al., 2013). Due to the business environment dynamics and the non-standardization of the core competencies of most companies (Elchamaa et al., 2019), the competencies have different interpretations gathering in the job offers. In this way, universities receive ambiguous competencies from the work environment, complicating the alignment between academic and professional profiles (Ramsauer, 2020).

On the other hand, the university degree programs build learners' competencies, managing the appearance of new positions and the growing need for experts in certain areas (De Leenheer et al., 2007). At the same time, the university must align with learning goals defined in standards and frameworks (e.g., ACM, EQF, etc.) to comply with the regulations of government entities. However, the competencies of these professional bodies vary in their descriptors, granularity, specificity, and structure (Gluga et al., 2013).

The panorama narrated in the previous paragraphs generates ambiguities in the skills and knowledge within the models of competencies (Miranda et al., 2017), and consequently, in the descriptions of the degree programs (Kondratova et al., 2017) and the syllabuses (González et al., 2017). This section describes the theoretical aspects related to this problem and the proposed resolution.

Description Logic and Representation of Competencies

Description logic plays a crucial role in the ontological models that currently support competence management, like OWL Lite and OWL DL (Horrocks et al., 2003). Description logic generally provides the first-order formalism with a well-established and straightforward declarative semantic to capture the meaning of the most popular features of structured knowledge representations (Lukasiewicz, 2008).

However, the Description logic is not reliable for representing the information uncertain, vague or imprecise, which produces inconsistencies in the processes of ontological reasoning due to its inability to handle ambiguity (Sikos, 2017). For instance, it is complicated to describe the existing state, future outcome, or more than one possible outcome when there is incomplete evidence or inconsistent knowledge (Bourahla, 2015).

The ontological models for the management of competencies present Vagueness and uncertainty in their structure at different levels:

- Terminology: It is impossible to establish what class an individual of the ontology belongs to due to its lexical ambiguity (Malzhan et al., 2013). This case is widespread in those competence models where an ontological population is made from unstructured sources, such as professional profiles (Janev, 2011). For example, in the case of the individual, "software development" may belong to the domain of the concept "application" and the domain of the concept "software" at the same time.
- Structural: Where concepts' semantic ambiguity of ontologies causes alignments and mixtures of ontologies that do not present the domain of the analyzed competencies (Gluga et al., 2013). For instance, the concepts "design" and "sketch" are synonymous but may belong to domains of different concepts, such as "knowledge" or "synthesis."

Consequently, competence management requires efficient, and at the same time, flexible, ontological models for the detection and treatment of ambiguous phenomena.

Dialetheic Logic for the Ambiguity of the Competencies

Formal languages for knowledge representation, such as description logic, cannot show context contradiction cases (Aguilar, 2011; Gutierrez et al., 2017) because their truth values can be either true or false. In contrast, DMs describe linguistic ambiguities through axioms that can be both (Pelletier et al., 2017). For example, the affirmation that a person is in a room when he is walking through the door can say that it is true and false at the same time (Pelletier et al., 2017).

Such assumptions are analyzed by dialetheic logic RM3 through translations of RM3 formulae to classical First-Order Logic (FOL). Defining two translation functions consistent with each other: translation *trs* (dialetheic truth being true or both) and anti-translation *atrs* (dialetheic falsity, being false or both) (Pelletier et al., 2017; Sutcliffe & Pelletier, 2019). The unified truth value of the two translations represents the dialetheic possibility that the formula RM3 is both true and false, which confirms the contradiction existing in the formulae analyzed (Arruda, 1980).

Typically, the automated reasoning systems for dialetheic logic can support several languages, such as clause normal form (cnf) and first-order form (fof). Mainly, fof is the language used by the reasoner of RM3 in its internal processes for the translations of RM3 formulae to classical first-order logic (FOL) to use existing first-order reasoning systems (Sutcliffe et al., 2012; Sutcliffe & Pelletier, 2019).

RM3 reasoning analyzes different natural language phenomena, such as term competencies ambiguity, detecting various contradictions defined by formal axioms. Thus, RM3 becomes a viable option for ontological contradiction analysis described by general description logic axioms, especially related to relationships or concepts, due to specific situations defined by the context. For instance, the OC ontology presents ambiguities regarding the knowledge and skill terms caused by the ambiguity of the profiles. The following section explains the axioms in Dialetheic logic applied to competencies ambiguity phenomena in the OC model.

Knowledge Model

The DM model contains hypotheses corresponding to the five dialetheic phenomena: Vagueness, failure of a presupposition, counterfactual reasoning, fictitious discourse, and contingent statements about the future. We apply the descriptions of each axiom on the terms of competence, knowledge, and skill, belonging to documents of a profiles collection analyzed by experts (González-Eras & Aguilar, 2018). These terms belonging to the ontological population of the OC model, following a method developed in (González-Eras & Aguilar, 2019), with the support of knowledge bases of knowledge and skills definitions: DISCO II (for knowledge), BLOOM (for skill) (González-Eras & Aguilar, 2019).

In the following subsections, for each dialetheic phenomenon, we first analyze the axioms using term examples. We present the inference process using natural language and description logic, and finally, the RM3 description is composed of three components: axioms, which correspond to the dialetheic rules that define them; facts, which are the entries to the model from the instances extracted from the digital

Table 1 Linguistic patterns of vagueness

Term	Linguistic pattern	Interpretation according to the pattern	Editor's interpretation
Hardware knowledge	NC-SP-NC	Knowledge	Skill
Java expert	NC-SP-NC	Knowledge	Skill
Software development	NC-SP-NC	Knowledge	Competence

academic or professional profiles; and conjectures, which are activated during reasoning to perform the interpretation of digital academic or professional profiles (Pelletier et al., 2017).

Vagueness

Vagueness corresponds to a lack of clarity, precision, or accuracy in natural language (Sorensen, 2018). The linguistic patterns of nominal and verbal phrases that identify skills and knowledge competencies may be the same (homonyms). Table 1 shows three examples of the ambivalence of these patterns, which are considered noun phrases of the form NC-SP-NC and NC-SP-NC-AQ[1], representing the knowledge component. However, these terms can be interpreted as a skill (*Java expert* and Hardware knowledge) or a competence (Software development) (González-Eras & Aguilar, 2019). In this way, the linguistic structure of the nominal phrases is ambivalent, according to the editor's interpretation of knowledge and skill.

In particular, we propose the following axiom for the vagueness problems explained in Table 1:

> If (the term T has a pattern P as knowledge) and (P is interpreted as a Skill (C1) or Competence (C2)), then (T has an ambivalent pattern).

Figure 1 describes the axiom for the case of the term "hardware knowledge," considering the ambiguity in the language patterns of the nominal and verbal phrases for the model's first axiom, which can be interpreted as knowledge, skill, or competence. According to the facts, the conjecture is true because "hardware knowledge" has a pattern of knowledge. Still, it is interpreted as a skill, which is not understood in the description logic OC model, given its inability to reason about ambivalent facts.

Table 2 shows the axiom in RM3 format (Dialetheic Logic). As can be seen, the axiom *"term hasAmbivalentPattern"* establishes the relationship between the linguistic patterns of the terms, depending on whether T (term) has a pattern P that represents knowledge, but that, when it is interpreted is different (as skill (C1) or competence (C2)), so that there is an ambivalence. Thus, although

[1] CONLL format, where NC: noun, SP: preposition and AQ: adjective.

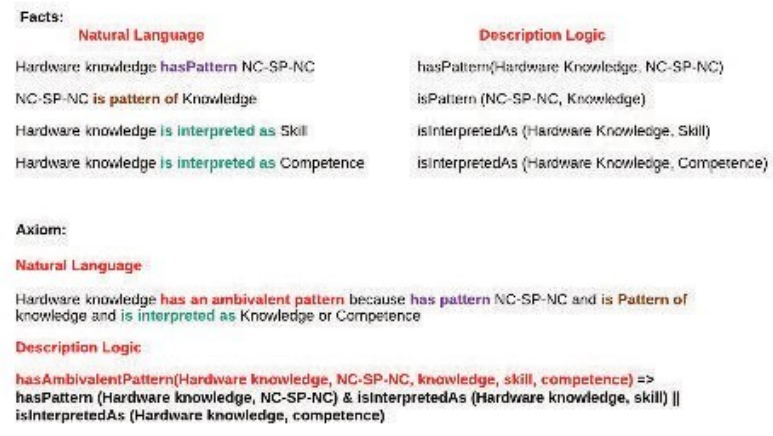

Fig. 1 Vagueness analysis process for the term "Hardware Knowledge"

Table 2 Vagueness axioms in RM3 format

If the term T has a pattern P as knowledge and is interpreted as Skill C1 or Competence C2, then T has an ambivalent pattern.	
Axioms	fof(hasAmbivalentPattern,axiom,(! [T,P,C1,C2] : ((hasPattern(T, P) & isInterpretedAs(T, C1) & isInterpretedAs(P, C2) & isDifferent(C1,P) & isDifferent(C2,P)) => hasAmbivalentPattern(T, P)))).
Facts	fof(hasPattern1, axiom, hasPattern(hardware_knowledge, nc_sp_nc)). fof(isInterpretedAs1, axiom, isInterpretedAs (hardware_knowledge, competence)). fof(isInterpretedAs2, axiom, isInterpretedAs(hardware_knowledge, skill)). fof(isPattern1, axiom, isPattern(nc_sp_nc, knowledge)). fof(isDifferent1, axiom, isDifferent(knowledge, competence)). fof(isDifferent2, axiom, isDifferent(knowledge, skill)).
Conjecture	If "SZS status Theorem for FOF" term has ambivalent pattern fof(conjeture1,conjecture, (hasAmbivalentPattern(hardware_knowledge, nc_sp_nc))).

the term linguistic pattern indicates a nominal phrase that corresponds to knowledge, the term is interpreted as a skill or competence.

The model starts with the facts proposal such as *fof(hasPattern1, axiom, hasPattern (hardware_knowledge, nc_sp_nc))*, on which the axioms perform the interpretations, from basic axioms as *fof(isInterpretedAs2, axiom, isInterpretedAs(hardware_knowledge, skill))*, until arrives at the conjecture, which is an axiom that interprets the facts based on the basic axiom *fof(conjeture1,conjecture,(hasAmbivalentPattern(hardware_knowledge,nc_sp_nc)))*.

Contingent Statements About the Future

The statements analyze future events, actions, etc. (Peter & Hasle, 2015). This phenomenon occurs in verbal phrases that generally describe competencies and skills. In this case, the phrase is formed by several verbs that, considering their synonyms, are found in different skill levels and cognitive processes, which do not establish what skill the competence will develop shortly. For example, according to the thesaurus Bloom described in González-Eras and Aguilar, (2019), for the competence of Table 3, "*Design and manage systems*," the word "*design*" belongs to the cognitive level 3 and the word "*manage*" to the cognitive level 5, both within different cognitive processes (lower and higher, respectively). Therefore, if this competence is eventually needed, it is ambiguous to establish the teaching mechanisms to achieve it. Even the learning evaluation process is unclear at what level and cognitive process must be considered the competence.

To formalize this contradiction, we propose the following axioms for the contingent statement problems of the examples in Table 3.

> Problem 1: If (the term Th is synonymous with the thesaurus term Tb) and (Th and Tb have different cognitive levels Nc1 and Nc2), then (Th belongs to several cognitive levels).
> Problem 2: If (the term Th1 is synonymous with the term Th2) and (Th1 and Th2 belong to different cognitive levels Nc1 and Nc2), then (Th1 and Th2 have several cognitive levels).
> Problem 3: If (the term T is synonymous with the terms Th1 and Th2) and (Th1 and Th2 belong to different cognitive levels Nc1 and Nc2), then (T has several cognitive levels).

Figure 2 presents the inference process of the axioms, in the case of the term's "design" and "plan," considering the ambiguity between terms of skills when they belong to two different levels and cognitive processes. The reference thesaurus for this analysis is the BLOOM thesaurus explained in González-Eras and Aguilar (2019), in which each cognitive level has associated a set of related terms. So, a term in a given cognitive level can be synonymous with a term that belongs to a different cognitive level. Thus, the term is ambiguous because its synonyms belong to two different cognitive levels and, therefore, to two different cognitive processes.

For example, for the first case, the axiom *"termBelongsSeveralCognitiveLevels"* establishes that if the term Th is synonymous with Tb, and the cognitive levels of Th (Nc1) and Tb (Nc2) are different, then they can belong to several cognitive levels (Nc1 and Nc2). In this way, the contradiction of term Th is identified on what cognitive level it belongs to; consequently, since Th belongs to several cognitive levels, then it is established that it also belongs to several cognitive processes.

In Table 4, we present the DM of the axioms starting with the fact to establish that their cognitive levels are different (with *fof(isDifferent2, axiom, isDifferent (synthesis, application))*). Then, the synonymy relation is established between the terms (with *fof(isSynonymous2, axiom, isSynonymous (design, plan))*), and of the membership of each term to a cognitive level (with *fof(belongsCognitiveLevel1, axiom,*

Table 3 Cases of contingent statements about the future in profiles terms due to cognitive levels contradiction

Verbal phrase	Cognitive level 1	Cognitive level 2	Cognitive process 1	Cognitive process 2
Design and manage systems	Design 3	Manage 5	Inferior	Superior
Operate and maintain computer centers	Operate 3	Maintain 6	Inferior	Superior

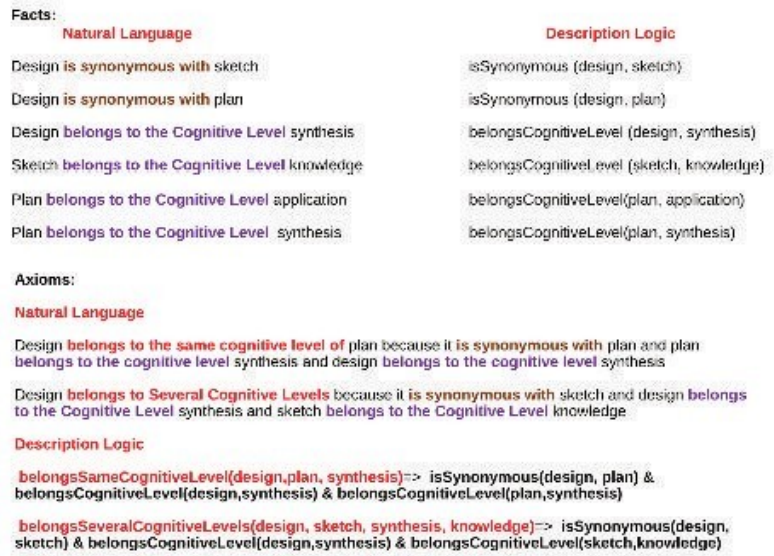

Fig. 2 Analysis process of contingent statements about the future in the synonyms "design," "plan," and "sketch" due to their belonging to different cognitive levels

belongsCognitiveLevel (design, synthesis))). In this way, as shown in Fig. 2, the knowledge base for interpretation is built for the conjecture *fof(conjecture, conjecture,(termsBelongSeveralCognitiveLevels (design, plan)))*, which has a value of true because "design" and "plan" are synonymous and belongs to different cognitive levels ("synthesis" and "application," respectively).

Fictional Discourse (Speech)

According to the people's beliefs, the statements imply making decisions related to particular real or imaginary assumptions (Eklund, 2017). In the case of competencies and their knowledge and skill components, it is common for the profile editor to place these three components under sections of a document, such as the description, the occupational field, and not precisely as competencies, knowledge, or skills. Table 5 shows some cases founded in (González-Eras & Aguilar, 2019), where the profile editor placed the competence "Plan and manage computer projects" as an antecedent. A similar case concerns of knowledge topic "*Industrial process control*," set in the competence section. Consequently, it depends a lot on the interpretation and knowledge of the editor to recognize a competence or its knowledge and skill components, which can generate a fiction in the writing of the academic or professional profile.

Table 4 Axioms of contingent statements about the future in profile terms in RM3 format

Problem 1: If Th is synonymous with the thesaurus term Tb and Th and Tb have different cognitive levels Nc1 and Nc2, then Th belongs to several cognitive levels. Problem 2: If the related verb Th1 is synonymous with the corresponding verb Th2 and Th1 and Th2 belong to different cognitive levels Nc1 and Nc2, then Th1 and Th2 have several cognitive levels. Problem 3: If T is synonymous with related verbs Th1 and Th2 and Th1 and Th2 belong to different cognitive levels Nc1 and Nc2, then T has several cognitive levels.	
Axioms	fof(termBelongsSeveralCognitiveLevels, axiom,(! [Th,Tb,Nc1,Nc2] : ((isSynonymous(Th,Tb) & belongsCognitiveLevel(Th,Nc1)& belongsCognitiveLevel(Tb,Nc2) & isDifferent(Nc1,Nc2)) => termBelongsSeveralCognitiveLevels(Th)))). fof(termsRelatedVerbBelongsSeveralCognitiveLevels,axiom,(! [Th1,Th2,Nc1,Nc2] : ((belongsCognitiveLevel (Th1,Nc1)& belongsCognitiveLevel(Th2,Nc2) & isDifferent(Nc1,Nc2)) => termsRelatedVerbBelongsSeveralCognitiveLevels(Th1, Th2)))). fof(termsBelongsSeveralCognitiveLevels, axiom,(! [Th1,Th2] : ((termsRelatedVerbBelongSeveralCognitiveLevels(Th1, Th2) | termBelongsSeveralCognitiveLevels (Th1) | termBelongsSeveralCognitiveLevels (Th2)) => termsBelongSeveralCognitiveLevels (Th1,Th2)))).
Facts	fof(isDifferent1, axiom, isDifferent(synthesis, knowledge)). fof(isDifferent2, axiom, isDifferent(synthesis, aplication)). fof(isDifferent3, axiom, isDifferent(aplication, synthesis)). fof(isSynonymous1, axiom, isSynonymous(design, sketch)). fof(isSynonymous2, axiom, isSynonymous(design, plan)). fof(isSynonymous3, axiom, isSynonymous(plan, sketch)). fof(belongsCognitiveLevel1, axiom, belongsCognitiveLevel(design, synthesis)). fof(belongsCognitiveLevel2, axiom, belongsCognitiveLevel(sketch,synthesis)). fof(belongsCognitiveLevel22, axiom, belongsCognitiveLevel(sketch,knowledge)). fof(belongsCognitiveLevel3, axiom, belongsCognitiveLevel(plan, aplication)).
Conjecture	If "SZS status Theorem for FOF" terms belongs several cognitive levels fof(conjetura,conjecture, (termsBelongSeveralCognitiveLevels(design, plan))). fof(conjetura,conjecture, (termBelongsSeveralCognitiveLevels (design))).

Table 5 Cases of Fictitious speech in profiles terms due to their meaning and location

Term	Component	Document section
Industrial process control	Knowledge	Competence
Development of computer applications	Knowledge	Career profile
Plan and manage computer projects	Knowledge	Antecedent

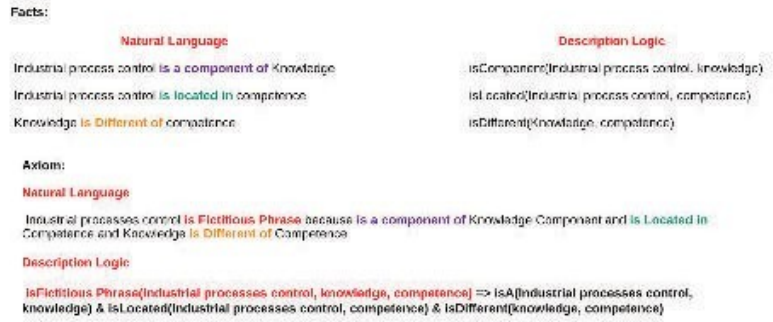

Fig. 3 Fictitious phrase analysis process for the term "industrial process control"

Table 6 Axioms of fictitious terms in RM3 format

If the term T is located in the section of document C1 and T is a component C2, and C1 is different from C2, then T is a fictitious phrase.	
Axiom	fof(isFictitiousPhrase, axiom,(! [T,P,C1,C2] : ((isLocated(T, C1) & isComponent(T, C2) & isDifferent(C1,C2)) => isFictitiousPhrase(T)))).
Facts	fof(isComponent1, axiom, isComponent(industrial_processes_control, knowledge)). fof(isComponent2, axiom, isComponent(industrial_process_control, competencies)). fof(isLocated1, axiom, isLocated(industrial_process_control, competencies)). fof(isDifferent1, axiom, isDifferent(competencies, knowledge)). fof(isDifferent2, axiom, isDifferent(knowledge, competencies)).
Conjecture	If "SZS status Theorem for FOF" is Fictitious Phrase fof(conjetura,conjecture, (isFictitiousPhrase(industrial_process_control, competencies))).

In particular, we propose the following axiom for this problem, according to the examples in Table 5. In this case, the description logic fails to represent the contradiction of the facts; for instance, the term "industrial process control" is a knowledge component, but it is located as a competence.

> If (the term T is located in the section of document C1) and (T is a component C2) and (C1 is different from C2), then (T is a fictitious phrase).

Figure 3 presents the inference process of the axioms for the term *"industrial_process_control"* and the component *"competencies"*, establishing the contradiction in the narrative of a profile. When the term T is located in the section of document C1, being a component of C2, and C1 is different from C2, it causes an unreal statement about the term T. Consequently, the phrase meaning that contains T is altered, generating a fictional speech in the profile. As a consequence, the editor places the competence and knowledge terms under sections with no relationships.

In Table 6, we present the axiom "isFictitiousPhrase" starting with the fact that the term is a component of *"knowledge"* (with *fof(isComponent1,axiom,isComp*

Table 7 Cases of Presupposition failure due to the contradiction of interpretation by experts of profiles terms

Term	Presupposition	Interpretation by the expert (pattern)
Develop computer applications	Career profile	Skill
Develop computer programs	Competence	Skill
Plan and manage computer projects	Antecedent	Skill
Hardware Control	Antecedent	Knowledge
Java knowledge	Experience	Skill

onent(industrial_process_control, knowledge))), which is located in the "*competencies*" section of the document (with *fof(isLocated1, axiom, isLocated (industrial_process_control, competencies))*), being different "*knowledge*" and "*competencies*" (*with fof(isDifferent2, axiom, isDifferent(knowledge, competencies))*). Based on the facts, the conjecture *fof(conjecture, conjecture, (isFictitiousPhrase (industrial_process_control, competencies))* has a value of true, because at the same time "*industrial_process_control*" is a "*knowledge*" component and is identified as a "*competence*".

Presupposition Failure

The statement implies the assumption of something that is not really true (Beaver, 1997), applied to competencies when the term is misused in a profile section, in such a way that the term presupposition is wrong. According to the editor's interpretation, it is of a type, but it is another type. For example, in Table 7, the term "*Develop computer applications*" is assumed as a "*Career profile*", when in fact, it is interpreted as a "*Skill*" (González-Eras & Aguilar, 2019). Similarly, "*Hardware control*" is supposedly an "*Antecedent*," being a "*Knowledge*," and so for the other cases. Thus, for each term, the assumption made by the profile editor is wrong concerning the expert's interpretation.

To formalize this contradiction, we propose the following axiom for this problem, according to the examples in Table 7.

> If (the term T is located in the section of document C1) and (T has a pattern C2) and (C1 is different from C2), then (T is a Presupposition failure).

Figure 4 presents the inference process of the axioms for the term *"java_knowledge"* and the description logic model of the axiom of this problem. It does not adequately represent the ambiguity of the term "Java knowledge," which has a knowledge pattern and is located/used as a skill. That establishes the contradiction in using the term T, located in the document section C1, having a pattern C2 different from C1. In this way, the profile editor's assumption about T fails because he misuses the term in the document.

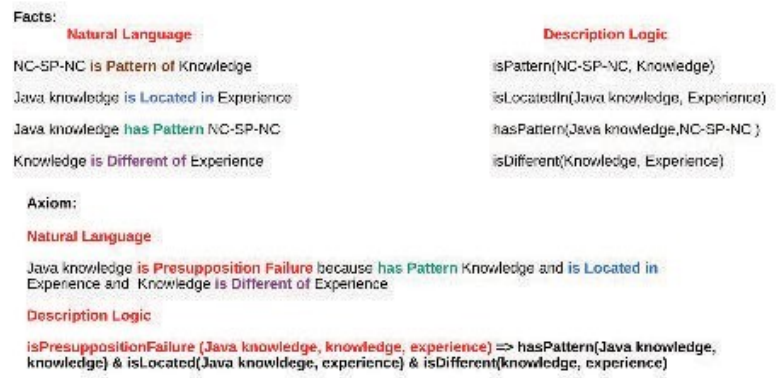

Fig. 4 Presupposition failure analysis process for the term "java knowledge"

Table 8 Presupposition failure axioms in RM3 format

If the term T is located in the section of document C1 and T has a pattern C2, and C1 is different from C2, then T is a Presupposition failure.	
Axiom	fof(isPresuppositionFailure, axiom,(! [T,P,C1,C2] : ((hasPattern(T, P) & isLocatedIn(T, C1) & isPattern(P, C2) & isDifferent(C1,C2)) => isPresuppositionFailure (T)))).
Facts	fof(hasPattern 1, axiom, hasPattern (java_knowledge, nc_aq)). fof(isLocatedIn 1, axiom, isLocatedIn(java_knowledge, experience)). fof(isPattern 1, axiom, isPattern (nc_aq, knowledge)). fof(isDifferent 1, axiom, isDifferent(experience, knowledge)).
Conjecture	If "SZS status Theorem for FOF" term is Presupposition Failure fof(conjetura,conjecture, (isPresuppositionFailure (java_knowledge))).

In Table 8, we present the axiom in dialetheic logic "isPresuppositionFailure" starting with the fact that the term has a pattern of knowledge "nc_aq" (*fof(hasPattern1,axiom,hasPattern(java_knowledge, nc_aq))*), which is located in the "*experience*" section of the document (*fof(isLocatedIn 1, axiom, isLocatedIn(java_knowledge, experience))*), being different "*knowledge*" and "*experience*" (*fof(isDifferent 1, axiom, isDifferent(experience, knowledge))*). Based on the facts, the conjecture *fof(conjetura,conjecture, (isPresuppositionFailure (java_knowledge)))* has a value of true because at the same time "*java_knowledge*" has a pattern of "*knowledge*" that is identified as an "*experience*".

Counterfactual Reasoning

Considering the meaning of the causal statements can be explained in terms of counterfactual conditionals of the form: "If A had not occurred, then C would not have occurred" (Menzies, 2001). In the context of competencies, counterfactual reasoning

Table 9 Cases of counterfactual reasoning due to belonging of term to a domain according to a similarity measure

Term	Topic	Domain	Similarity
Software	Software debugging	Programming	0.53
	Software installation	IT installation and configuration	0.50
	Software Application Development	Software development	0.41

Fig. 5 Counterfactual reasoning analysis process for the terms "software", "programming", and "software debugging"

applies in the assumptions made when aligning terms of competencies with the terms of thesauri according to lexical similarity measures, establishing thresholds to determine the similarities. We will propose the following hypothesis: "A term and a topic of a competence thesaurus belong to the same domain of knowledge when the measure of similarity between them exceeds the limit of 0.45" (González-Eras & Aguilar, 2019). As shown in Table 9, for the three proposed cases, two belong to the same domain because the similarity measure exceeds the threshold of 0.45. But, if we change the limit value to 0.51, we see that only the case "software" versus "*Programming*" meets the hypothesis. In general, the threshold value is subjective, causing errors and ambivalences in interpreting the belonging of a term of a domain of knowledge.

Table 10 Counterfactual reasoning axioms in RM3 format

If the term T has a measure of similarity Ms with a topic Tr greater than the threshold Us, then it belongs to the root topic of thesaurus TD.	
Axioms	fof(termBelongsTopic,axiom,(![T,Tr,Ms,Us,TD] : ((relationMeasure(T, Tr, Ms, TD) & isGreaterThan(Ms, Us)) => termBelongsTopic(T, TD)))). fof(termBelongsSeveralTopics,axiom,(![T,TD1,TD2] : ((termBelongsTopic (T, TD1) & termBelongsTopic (T, TD2) & isDifferent(TD1, TD2)) => termBelongsSeveralTopics (T)))).
Facts	fof(relationMeasure1, axiom, relationMeasure (software, programming, ms0_48, td1)). fof(relationMeasure2, axiom, relationMeasure (software, software_installation, ms0_30, td11)). fof(relationMeasure 3, axiom, relationMeasure (software, software_debugging, ms0_52, td12)).
	fof(threshold, axiom, threshold = ms0_45). fof(isGreaterThan1, axiom, isGreaterThan(ms0_48 , threshold)). fof(isGreaterThan2, axiom, isGreaterThan (ms0_52 , threshold)). fof(isDifferent 1, axiom, isDifferent (td1 , td12)).
Conjecture	If "SZS status Theorem for FOF term Belongs Topic fof(conjetura ,conjecture, (termBelongsTopic (software, td1))). fof(conjetura ,conjecture, (termBelongsTopic (software, td12))). If "SZS status Theorem for FOF" term Belongs Several Topics fof(conjetura ,conjecture, (termBelongsSeveralTopics(software))).

According to the examples in Table 9, we propose the following axiom for this problem,

If (the term T has a measure of similarity Ms with a topic Tr greater than the threshold Us), then (it belongs to the root topic of the thesaurus TD).

Figure 5 describes the operation of the axioms for the case of the term "*software.*" In this case, we address the contradiction in belonging a knowledge term to a thesaurus topic due to the thresholds used in the similarity measures. In this case, using DISCO II as a reference thesaurus (González-Eras & Aguilar, 2015; González-Eras et al., 2018). Mainly, the axiom "*termBelongsSeveralTopics*" requires compliance with the axiom "*termBelongsTopic*." With these relationships, it is described that a term T belongs to several topics of the thesaurus if the measure of similarity is greater than the established threshold.

Table 10 shows the dialetheic axioms for this contradiction, starting with the facts *fof(relationMeasure1, axiom, relationMeasure (software, programming, ms0_48, td1))* and *fof(relationMeasure 3, axiom, relationMeasure (software, software_debugging, ms0_52, td12))*, which defines that the term "*software*" has a similarity measure of 0.48 with "programming" and of 0.52 with "*software_debugging*." Another fact is that similarity measures of 0.48 and 0.52 are more significant than the threshold (0.45), and also that the facts "td1" and "td12" are "different." In this way, as shown in Fig. 5, the knowledge base for interpretation is built according to the axiom *fof(conjecture, conjecture, (termBelongsTopic (software, td12)))*, which is the base axiom for the *conjecture fof(conjecture, conjecture, (termBelongsSeveralTopics (software))). Considering the term "software," the result is* true because *"software"* belongs to the topics "programming" and "software debugging."

Experimentation

In this section, we will analyze the capacity of our DM model to detect ambiguities in the OC ontology proposed in (González-Eras & Aguilar, 2019). Thus, we first describe the OC and then evaluate its quality using metrics of the ontological scope, particularly the completeness and Robustness (González-Eras & Aguilar, 2018). Later, we analyze the present ambiguities in the OC using DM and compare it with the entropy metric calculated for the OC. The objective is to analyze the professional and academic profiles from two perspectives: the first is the number of relevant terms that each profile contributes to the OC (see Section 4.1); and, the second, to establish the ambiguity that exists in the terms provided by each profile according to the DM, compared with the Entropy of the OC (see Section 4.2). In this way, the description logic complements the DM to analyze the profiles more in-depth (see Section 4.3).

We consider the terms found in the OC ontology, taken for semantic relevance from a corpus of academic and professional profiles regarding the experiment data. Each document is analyzed to detect knowledge and skill terms using linguistic patterns (NC_SP_NC, NC_AQ). Generally, terms belong to profile sections such as description, objectives, roles, and competencies. They are labeled according to the document section where they are found and their real meaning. Thus, the dataset contains information for the processes performed in both models (OC and DM).

The Ontological Model (OC)

To select relevant terms to the ontological population of the OC defined in González-Eras (2019), the dataset elements are aligned with the DISCO II (Tc1,…, Tc15) and BLOOM (Th1,…, Th6) thesauri, comparing the terms of the thesauri against the terms found in the profiles, using two classes of similarity measures. First, using lexical similarity measures, such as Levenshtein and Sorensen Coefficient, comparing pairs of terms at their characters' level and obtaining those pairs with the most significant similarity (according to the U_L threshold). Second, we compare the chosen pairs with the term's ancestors, siblings, and children of the thesaurus subtree with the pair is aligned, selecting those that obtain the most remarkable similarity. As a final step, we establish a measure of relevance (Score(id_i, C_j)) of each term according to its frequency in the collection of profiles (through Okapi BM25 ranking function (Robertson, 2009)); being a chosen term to fill the ontology if it exceeds the relevance threshold (U_R >= 0,3). Thus, the term set is obtained to fill the OC (refer to (, 2019) for more detail).

To establish OC quality, we calculate Completeness and Robustness measures to determine if the ontology is complete and robust regarding how many relevant terms have been obtained from the profiles to fill the ontology (González-Eras & Aguilar, 2018).

Completeness The OC is considered complete regarding the professional profile id_i if it contains all relevant terms extracted from this profile (see Eq. (1)).

$$Completeness(OC, id_i) = \frac{\sum_{i=1}^{m} Trelevant(id_i) \cap Terms(OC)}{\sum_{i=1}^{m} Trelevant(id_i)} \quad (1)$$

Where $Trelevant(id_i)$ are the terms whose $Score(id_i,C_j)$ is in the range defined by the threshold ($U_R >= 0.3$) and $Terms(OC)$ are the candidate terms to integrate the OC.

Score(id_i,C_j) The relevance value of a term consists of its position within the collection of analyzed terms. Specifically, the relevance value of a term C_j in the profile id_i is given by:

$$Score(id_i, C_j) = \sum_{j=1}^{n} IDF(C_j).\frac{f(C_j, id_i).(k_1 + 1)}{f(C_j, id_i) + k_1.(1 - b + b.\frac{|D|}{avgdl})} \quad (2)$$

Where, $f(C_j, id_i)$ is the frequency of appearance of the term C_j in the profile id_i; |D| is the number of terms in the profile id_i; $avgdl$ is the average length of the profiles in the collection; k_1 and b are parameters to adjust the length differences of the profiles; $IDF(C_j)$ is the weight given to the term C_j in the collection, and n is the number of profiles in the collection.

Robustness An OC is robust about the set of professional and academic profiles if its C_j terms are relevant for the profile id_i (see Eq. 3).

$$Robustness(OC, id_i) = \frac{\sum_{i=1 \& C_j \in Terms(OC)}^{m} Score(id_i, C_j)}{|Terms(OC)|} \quad (3)$$

Finally, this paper uses the entropy metric to analyze the information contained in an ontology. The Entropy allows estimating the amount of information that some concepts contribute to a specific target concept (Haque & Chiang, 2019). This paper uses this idea to say that an ontology, when it is very ambiguous, does not contain useful information. So, intuitively we can think that when the Entropy of the ontology decreases significantly, there is less uncertainty in its content.

Entropy it determines the amount of information that the OC ontology contains. Considering that the terms of the profiles can be ambiguous, the Entropy defines the uncertainty that each profile id_i introduces to the OC (Mendonça et al., 2020) (see Eq. (4)).

$$H_{OC}(id_i) = \sum_{j=1}^{k} P(x_j) log_2 P(x_j) \quad (4)$$

Where: P(x_j) corresponds to the probability of that term $x_j \in id_i$ included in the OC (it is a relevant term) is not both true and false, and k is the number of terms in the profile.

We consider that if the value of the Entropy $H_{OC}(id_i)$ is zero, then the profile id_i does not introduce uncertainty.

The Dialetheic Model (DM)

The DM model recognizes two types of events: dialetheic terms, which correspond to those in which the axioms return a positive (true) value of truth, and non-dialetheic terms when the axioms give a negative (false) value of truth. Based on these terms, we define for each profile id_i, $sd_{ji}=1$ when the DM recognizes the dialetheic event j (ambivalent term). The following measure is used to evaluate the DM:

Robustness The DM is considered robust concerning a professional or academic profile id_i if it recognizes all its dialetheic terms. Equation (5) presents the Robustness measure, where n_i represents the number of dialetheic terms in the profile id_i.

$$Robustness(DM, id_i) = \sum_{j=1}^{n_i} \frac{sd_{ji}}{n_i} \tag{5}$$

For this calculation, previously, the terms of competence, knowledge, and skill in the profiles are labeled as dialetheic (ambiguous) when they do not match with their linguistic patterns; they are in the wrong section of the document, among other reasons. Otherwise, they are labeled as non-dialetheic terms. These contradictions are due to the appearance of five natural language phenomena in the formulation of competencies in the profiles. For example, fictional narratives contradict the interpretation of a term and its location in the document (for example, profiles 2 to 11). The robustness metric allows determining if all dialetheic terms are recognized. So, with this metric, the capability to recognize dialetheic terms by the DM model is determined.

Finally, it is possible to calculate the proportion of terms recognized by the DM on all relevant terms in the OC (Pr_DM) and compare them with the Entropy. If the proportion of ambiguous cases detected by DM is close to the entropy value of the ontology, then we could say that DM is identifying the ambiguities present in the ontology. Thus, we can evaluate the capability of the DM to detect the ambiguity of an ontology through its dialetheic terms, which a classical approach cannot consider.

Results and Discussion

This section presents the experiment results, validating the OC quality through the completeness and robustness measures explained in Sect. 4.1. Next, the comparison of the results of the DM application in OC and the Entropy of the OC.

Quality of the OC

The quality of the OC was presented in previous works regarding its completeness and Robustness (González Eras, 2018). In general, for 71.44% of the profiles, the ontology has High completeness (0.5 to 1), and for 22.85% of the profiles, Medium completeness (0.3 to 0.5); Low-level completeness (0 to 0.3) corresponds to two profiles (5.71%). Thus, on average, all relevant terms are used to populate the OC, indicating the usefulness of the process. Regarding Robustness, 28.57% of the profiles have a High relevance, while 68.57% have a medium relevance. In this case, the ontological model contains more than 80% of the relevant components of profiles.

We have included this section about the quality of OC to indicate that when using metrics from the domain of description logic, no problems are found in the ontology. In particular, the metrics used say that the competence dataset is well described by the ontological model defined by the OC. However, this ontological model cannot determine if the terms included are contradictory, mainly if they present ambiguities typical of natural language phenomena. A first approximation to clarify the above is to apply the entropy metric to the ontological model.

Quality of the DM

Table 11 presents the results of the robustness measure in the DM for the different phenomena of the natural language studied in this work. Thus, in case 1, 93% of the profiles have robustness values ranging between 0.8 and 1, which indicates that DM recognizes all the terms labeled as dialetheic. This trend is maintained for cases 2, 3, and 4, where around 90% of the profiles have robustness values between 0.8 and 1, indicating DM recognizes the relevant dialetheic terms in the profiles. Also, in case 5, the threshold does not affect the robustness value of each profile, identifying the dialetheic terms in the profiles regardless of the threshold used to determine the relevant terms (which will determine if there are more or fewer terms in the OC). So, this factor that serves to analyze the scalability of DM is well covered by it. Finally, we found profiles in that the robustness measure is maintained for the five cases (id20). The Robustness calculation does not apply in some profiles because they have not labeled dialetheic terms (n/a). For example, id12 and id15 for Case 3; and id 29 for cases 1, 2, 3, and 4. In general, we can conclude that the terms labeled with dialetheic for not following the linguistic patterns, for being in the wrong section of the document, among other reasons, are also recognized by the DM as dialetheic terms.

Comparison of the DM with the Entropy of the OC

Table 12 presents the calculation of the Entropy of the profiles for the OC, where the Entropy is zero when it is considered that all the relevant terms of the profile id_i do not introduce uncertainty. The results show that a significant majority tend to zero (between 0.5 and zero).

Table 11 Robustness calculation for the DM

id	Case 1	Case 2	Case 3	Case 4	Case 5				
					U1 0.2	U2 0.3	U3 0.4	U4 0.5	U5 0.6
1	0.86	0.83	0.82	0.81	0.83	0.81	0.77	0.78	0.78
2	0.8	0.87	0.8	0.8	0.8	0.8	0.81	n/a	n/a
3	0.8	1.00	0.87	0.67	0.8	0.8	0.8	0.8	0.8
4	0.88	0.67	0.88	0.88	0.75	0.8	0.8	0.8	0.8
5	0.87	1.00	0.93	0.83	0.83	0.8	0.8	0.8	0.8
6	0.8	1.00	0.8	0.8	0.8	0.8	0.8	0.8	n/a
7	0.87	0.83	0.93	0.93	0.83	0.8	0.8	0.8	0.8
8	0.85	0.89	0.85	0.85	0.75	0.75	0.73	0.75	0.73
9	0.9	1.00	0.8	0.7	0.83	0.8	0.8	0.8	n/a
10	0.8	1.00	0.8	0.8	0.8	0.8	0.8	n/a	n/a
11	0.9	1.00	0.8	0.9	0.8	0.82	0.8	0.8	1
12	0.76	0.83	n/a	0.86	0.76	0.8	0.8	0.8	0.9
13	0.84	0.95	0.84	0.84	0.84	0.84	0.84	0.8	0.8
14	0.8	0.67	0.96	0.8	0.8	0.8	0.8	0.77	n/a
15	0.8	1.00	n/a	0.8	0.8	0.8	1	n/a	n/a
16	0.67	1.00	0.89	0.87	0.88	0.87	0.87	0.87	0.8
17	0.86	0.67	0.8	0.9	0.8	0.8	0.8	n/a	n/a
18	0.91	1.00	0.86	0.8	0.8	0.8	0.8	n/a	n/a
19	0.68	0.87	0.86	0.86	0.85	0.87	0.85	0.8	0.8
20	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
21	0.81	0.81	0.59	0.79	0.81	0.83	0.85	0.8	0.8
22	0.87	0.87	0.87	0.8	0.8	0.79	0.8	n/a	n/a
23	0.81	n/a	0.86	0.8	0.8	0.8	n/a	n/a	n/a
24	0.87	1.00	0.81	0.75	0.84	0.87	0.87	0.87	0.8
25	0.82	1.00	0.8	0.84	0.87	0.87	0.87	0.8	0.8
26	0.7	0.89	0.75	0.7	0.75	0.73	0.73	0.78	1
27	0.87	0.83	0.87	0.87	0.79	0.72	0.77	0.77	0.77
28	0.89	0.95	0.81	0.8	0.86	0.84	0.86	0.88	0.8
29	n/a	n/a	n/a	n/a	0.87	0.87	0.87	0.87	0.8
30	0.8	1.00	0.8	0.8	0.8	0.8	n/a	n/a	n/a
31	0.83	0.83	0.83	0.83	0.83	0.8	0.8	0.8	n/a
32	0.87	0.67	0.87	0.87	0.8	0.8	0.8	0.8	0.8
33	0.79	0.89	0.69	0.8	0.88	0.87	0.82	0.83	0.84
34	0.85	1.00	0.82	0.83	0.85	0.86	0.88	0.82	0.85
35	0.81	0.83	0.68	0.8	0.79	0.77	0.8	0.8	0.8

Also, Table 12 presents terms proportion recognized by the DM on the total of terms relevant in the OC for each profile ($Pr\_DM(id_2)$). Regarding this value of the DM, it is zero when it does not recognize the relevant terms as dialetheic (ex. id_2), and this measure tends to one when it recognizes a significant number of the terms

Table 12 Comparison between the entropy of the OC and the DM based on the uncertainty of ambiguous terms recognized

id	$H_{OC}(id_i)$	$Pr\_DM(id_i)$
1	0.5	0.39
2	0	0
3	0.39	0.32
4	0	0.16
5	0.39	0.28
6	0	0
7	0.19	0.09
8	0	0
9	0	0.14
10	0	0.14
11	0	0.23
12	0.26	0.20
13	0.06	0.08
14	0	0.09
15	0	0
16	0.40	0.38
17	0.5	0.34
18	0	0
19	0	0.08
20	0	0.09
21	0.53	0.49
22	0	0.14
23	0.53	0.53
24	0.31	0.47
25	0	0.14
26	0	0.19
27	0.53	0.47
28	0	0.16
29	0.19	0
30	0.35	0.52
31	0.19	0.09
32	0.5	0.46
33	0.46	0.39
34	0.44	0.41
35	0.53	0.47

as dialetheic terms (ex. id_{13}). In this table, our DM follows the entropy metrics. For example, for the profiles id_2 and id_8 the Entropy of both models is zero, which indicates that there is no uncertainty in the OC, which is detected for Pr_DM when its value is zero that means that the terms are not dialetheic. For the rest of the profiles,

Table 13 Comparison of our proposal with other works

Work	Model	Competence Component	Disambiguation	Thesauri	Sources	Validation
(Dorn & Pichlmair, 2007)	Ontology	Knowledge	Average	WordNet	Job offers	Expert
(Paquette et al., 2012)	Ontology	Skill	LPO Axiom	Linked Data competence ontology	Profiles	Precision and relevance
(Fazel-Zarandi & Fox, 2010)	Ontology	Knowledge	Similarity measure	Competence ontology	Corpus	Precision, Recall
(Malzhan et al., 2013)	Ontology	Knowledge	Semantics	Lexicon and Onomasticon	Web text	Precision relevance
(Mendonça et al., 2015)	Ontology	Knowledge	ACO Algorithm	----	Web text	Precision
(Gil-Vallejo et al., 2018)	Patterns	Verbs	Coseno Dice measures	Adesse Wordnet Sensem	Verb Cases	Pearson Ratio
(Miranda et al., 2017)	Ontology	Knowledge skills	---	Europass	Curriculums	Fuzzy-based approach
(Gugla et al., 2013)	Ontology	Skill learning outcomes	By experts	ACM, IEEE, TEQSA	---	Precision
Our proposal	Ontology and dialetheic logic	Knowledge and skill	Dialetheic Axioms and Similarity Measures	DISCO II BLOOM	Job and academic offers	Dialetheic events Completeness Robustness

both values are close, determining that dialetheic terms correlated with the Entropy of the OC.

This result is fascinating since it indicates that our DM can determine the uncertainty present in an ontology. Also, it allows us to analyze which types of ambiguity are present in the ontology (something that cannot be done with the entropy metric).

Comparison with Other Works

Table 13 compares our proposal concerning related works selected due to their proposal to model competencies and/or handle the phenomena of lexical ambiguity. For this, we consider the following analysis criteria:

- The knowledge model establishes the information semantic structure.
- Competence components considered determining the elements involved in the analysis (for example, skills, knowledge).
- Disambiguation strategy presents methods to deal with the lexical ambiguity in the information units (synonyms, homonyms, hyponyms, and hypernyms).
- Thesauri indicate the knowledge bases to support the disambiguation process.
- Data sources indicate the origin of the information in each work.
- Metrics validation determines methods to the results verification process.

Dorn and Pichlmair (2007) present an information system for storing, evaluating, and reasoning students' competencies at universities based on a competence ontology. The system produces competence profiles for job applications based on HR-XML to enable an exchange of data. Paquette et al. (2012) compare competencies defining according to a structured competence model based on a domain ontology to provide a context for recommendations. Fazel-Zarandi and Fox (2010) present a formal ontology for competence management and consider three reasoning problems related to Human Resources Management: determining the set of skills of an individual, conducting competence gap analysis, and determining whether an individual satisfies a set of requirements. Malzhan et al. (2013) define a human resource manager model based on a conceptual model encompassing the market level, the social context, and the relationships between competencies. This model is the basis for an ontology-based decision support system for human resource managers presented in the same article. Mendonça et al. (2015) study ontology comparison (alignment) strategies based on their similarities. This problem is approached as an optimization one, and they propose an ant colony algorithm for analyzing the multiple ontology combinations. Gil-Vallejo et al. (2018) carry out a comparative analysis of the similarity between the verbal meanings in Spanish in the cognitive and linguistic fields. The results of the comparison show a significant correlation between the verbal similarities of both perspectives. Miranda et al. (2017) propose an ontology-based model for the representation of competencies to support several scenarios. The proposed model integrates representations of job offers and demands to support recruiting

initiatives and develop employability strategies. Finally, Gugla et al. (2013) describe the CUSP (Course and Unit of Study Portal) system, which supports the design of degree programs. CUSP exploits a semantic mapping approach that gives a flexible and scalable way to map learning goals from multiple accrediting sources.

Most works use ontologies to represent the knowledge and skill components for the knowledge model, limiting these works to Description logic. Our proposal uses a DM to the recognition of cases of contradiction. We have compared its results against a Description model, and it improves the capabilities of recognition of the context. Many works coincide in analyzing knowledge components, while others examine actions and verbs concerning the components analyzed. Our work considers knowledge, skill, and competence studied from the five natural language phenomena.

Concerning disambiguation strategies, several methods use vector space models with similarity measures and algorithms, which align the components with thesauri to eliminate the ambiguity of the elements analyzed. In our proposal, we also use similarity measures against thesauri. Still, additionally, we carry out a study of the ambiguity of the terms based on dialetheic logic, according to the five cases mentioned in Pelletier et al. (2017), which allows us to obtain another perspective of the academic and professional profiles. Most of the thesauri have target languages like English and German, which is a limitation for analyzing and comparing the competence components in the Spanish language. In our proposal, we consider the multilingualism of DISCO II thesaurus as an advantage to the replicability of our DM, not only in Spanish but in other languages. Regarding the model's validation, the measures of Robustness and Entropy allow determining the capabilities of our proposal for the management of ambivalence and contradiction. Typically, other works use measures like precision or experts' criteria.

Conclusions and Future Works

The present work proposes a model for representing ambiguity and contradiction in academic and professional competencies, based on axioms defined for five natural language phenomena: Vagueness, contingent statements about the future, fictional discourse, failure in the presupposition, and counterfactual reasoning. The importance of this analysis is due to the competency's ambiguity in the academic and labor contexts, difficulty terms alignment, and in this way, to identify common competencies and those that represent new requirements.

Although other investigations use different mechanisms for contradiction analysis, the use of Dialetheic Logic, and specifically RM3, allow identifying the ambiguity of description logic axioms and their representation in a framework of paraconsistency. The axioms of RM3 enable the implementation of dialetheic hypotheses from the Description Logic axioms.

In addition, the metrics determine the ability of DM to recognize dialetheic terms and the uncertainty that these terms bring to the OC model. Consequently, the DM in terms of Robustness shows that it is robust. Regarding the Entropy of OC, understood as an ontology uncertainty measure, the proportion of dialetheic terms

recognized by the DM follows that value, which can be interpreted as recognizing that degree of ambiguity in the OC.

The model proposed in this work can be part of automatic systems for developing competencies throughout a career program, supporting the validation and monitoring of the fulfillment of competencies in the curriculum subjects (intelligent tutoring systems). It can also contribute to detecting lexical ambiguities in the standards and frameworks used in the development of degree programs (instructional design systems) and in intelligent learning environments to develop flexible learning paths for students within and across subjects. Thus, our proposal provides a solution for lacking the university's capacity to control automatic acquisition of job requirements and their integration in the issues throughout the entire degree program.

The results obtained allow the correct interpretation of digital academic and professional profiles. Remarkably, this proposal can be used as an extension of the formal representation of competencies based on Description logic, find ambiguities and contradictions in the skill and knowledge components, and provide a new point of view of professional and academic profiles from the dialetheic logic. The validation of the DM through the measures of Entropy and Robustness allows determining the model's capability to find the presence of dialetheic events in the profiles.

The model works in a Spanish context for the computer science domain. According to the linguistic characteristics of the language where our model is applied, it can be extended to other languages and fields considering the adaptation of the linguistic patterns, which identify the knowledge, skill, and competence components.

Future work considers integrating the proposed knowledge model with semantic models, such as ontologies based on linked data (Jiménez et al., 2019). Also, context-aware ontologies (Aguilar et al., 2018b) allow a deeper analysis of the competencies using information obtained from the Web (Puerto et al., 2012; Rodriguez et al., 2010). In addition, the generations of new experiments apply the proposed DM in other contexts and areas of knowledge, using thesauri and related knowledge bases, for example, in intelligent learning environments like in (Aguilar et al., 2018a).

References

Aguilar, J. (2011). *Temporal Logic from the Chronicles Paradigm: learning and reasoning problems, and its applications in Distributed Systems*. LAP Lambert Academic Publishing

Aguilar, J., Cordero, J., & Buendía, O. (2018). Specification of the autonomic cycles of learning analytic tasks for a smart classroom. *Journal of Educational Computing Research, 56*(6), 866–891

Aguilar, J., Jerez, M., & Rodríguez, T. (2018). CAMeOnto: Context awareness meta ontology modeling. *Applied Computing and Informatics, 14*(2), 202–213

Arruda, A. (1980). A survey of paraconsistent logic. *Studies in logic and the foundations of mathematics* 99, 1–41. https://doi.org/10.1016/S0049-237X(09)70477-X

Beaver, D. I. (1997). Presupposition. *Handbook of logic and language* (pp. 939–1008). North-Holland

Bourahla, M. (2015). Reasoning over vague concepts. *Lecture Notes in Computer Science, 9120*, 591-602

De Leenheer, P., De Moor, A., & Meersman, R. (2007). Context dependency management in ontology engineering: a formal approach. *Lecture Notes in Computer Science, 4380*, 26-56

Dorn, J., & Pichlmair, M. (2007). A Competence Management System for Universities. In European Conference on Information Systems (pp. 759-770)

Dubois, D., & Prade, H. (2012). *Possibility theory: an approach to computerized processing of uncertainty*. Springer Science & Business Media

Elchamaa, R., Mbaya, A., Moalla, N., Ouzrout, Y., & Bouras, A. (2019). Ontology for Continuous Learning and Support. *Enterprise Interoperability VIII* (pp. 191–202). Springer

Eklund, M. (2017). *Fictionalism. The Stanford Encyclopedia of Philosophy*. (E. Zalta). Stanford University

Faes, M. G., & Moens, D. (2019). Recent Trends in the Modeling and Quantification of Non-probabilistic Uncertainty. *Archives of Computational Methods in Engineering, 27*, 633–671. https://doi.org/10.1007/s11831-019-09327-x.

Fazel-Zarandi, M., & Fox, M. S. (2010). Reasoning about skills and competencies. In L. M. Camarinha-Matos, X. Boucher, H. Afsarmanesh (Eds.), *Collaborative Networks for a Sustainable World* (336, pp. 372-379)

Gil-Vallejo, L., Castellón, I., & Coll-Florit, M. (2018). Similitud verbal: Análisis comparativo entre lingüística teórica y datos extraídos de corpus. *Revista Signos, 51*(98), 310–332

González-Eras, A. G., & Aguilar, J. (2015). Semantic architecture for the analysis of the academic and occupational profiles based on competencies. *Contemporary Engineering Sciences, 8*, 1551–1563

González-Eras, A., Buendia, O., Aguilar, J., Cordero, J., & Rodriguez, T. (2017). Competences as services in the autonomic cycles of learning analytic tasks for a smart classroom. In *International Conference on Technologies and Innovation* (pp. 211-226). Springer

González-Eras, A., & Aguilar, J. (2018). *Esquema para la actualización de Ontologías de Competencias en base al Procesamiento del Lenguaje Natural y la Minería Semántica* (pp. 433–447). Revista Ibérica de Sistemas e Tecnologias de InformaçãoE17

González-Eras, A., & Aguilar, J. (2019). Determination of professional competencies using an alignment algorithm of academic and professional profiles, based on competence thesauri and similarity measures. *International Journal of Artificial Intelligence in Education, 29*(4), 536–567

Gluga, R., Kay, J., & Lever, T. (2013). Foundations for modeling university curricula in terms of multiple learning goal sets. *IEEE Transactions on Learning Technologies, 6*(1), 25–37

Guevara, C., Aguilar, J., & González-Eras, A. (2017). The model of adaptive learning objects for virtual environments instanced by the competencies. *Advances in Science, Technology and Engineering Systems Journal, 2*(3), 345–355

Guo, S., Alamudun, F., & Hammond, T. (2016). RésuMatcher: A personalized résumé-job matching system. *Expert Systems with Applications, 60*, 169–182

Gutiérrez-Basulto, V., Jung, J. C., Lutz, C., & Schröder, L. (2017). Probabilistic description logics for subjective uncertainty. *Journal of Artificial Intelligence Research, 58*, 1–66

Hassan, F. M., Ghani, I., Faheem, M., & Hajji, A. A. (2012). Ontology matching approaches for eRecruitment. *International Journal of Computer Applications, 51*(2), 39–45

Haque, E., & Chiang, F. (2019). Restoring consistency in ontological multidimensional data models via weighted repairs. *Procedia Computer Science, 159*, 1085–1094

Horrocks, I., Patel-Schneider, P. F., & Van Harmelen, F. (2003). From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics, 1*(1), 7–26

Janev, V., & Vranes, S. (2011). Ontology-based competency management: the case study of the Mihajlo Pupin Institute. *Journal of Universal Computer Science, 17*(7), 1089–1108

Jiménez, C., Jerez, M., Aguilar, J., & García, R. (2019). Linked data and dialetheic logic for localization-aware applications. *Contemporary Engineering Sciences, 12*(3), 103–116

Kravcik, M., Wang, X., Ullrich, C., & Igel, C. (2018). Towards competence development for industry 4.0. In *International Conference on Artificial Intelligence in Education* (pp. 442-446). Springer

Kondratova, I., Molyneaux, H., & Fournier, H. (2017). Design considerations for competency functionality within a learning ecosystem. In *International Conference on Learning and Collaboration Technologies* (pp. 124-136). Springer

Lukasiewicz, T., & Straccia, U. (2008). Managing uncertainty and Vagueness in description logics for the semantic web. *Journal of Web Semantics, 6*(4), 291–308

Malzahn, N., Ziebarth, S., & Hoppe, H. U. (2013). Semi-automatic creation and exploitation of competence ontologies for trend aware profiling, matching and planning. *Knowledge Management & E-Learning: An International Journal, 5*(1), 84-103

Mendonça, M., Perozo, N., & Aguilar, J. (2015). An approach for Multiple Combination of Ontologies based on the Ants Colony Optimization Algorithm. In *Asia-Pacific Conference on Computer-Aided System Engineering (APCASE)*, (pp. 140-145)

Mendonça, M., Perozo, N., & Aguilar, J. (2020). Ontological emergence scheme in self-organized and emerging systems. *Advanced Engineering Informatics, 44*, 101045

Menzies, P. (2001). *Counterfactual theories of causation*. Stanford Encyclopedia of Philosophy (E. Zalta), Stanford University

Miranda, S., Orciuoli, F., Loia, V., & Sampson, D. (2017). An ontology-based model for competence management. *Data & Knowledge Engineering, 107*, 51–66

Montuschi, P., Lamberti, F., Gatteschi, V., & Demartini, C. (2015). A semantic recommender system for adaptive learning. *IT Professional, 5*, 50–58

Paquette, G., Rogozan, D., & Marino, O. (2012). Competency comparison relations for recommendation in technology enhanced learning scenarios. In *RecSysTEL 2012 Proceedings* (pp. 23-34)

Perozo, N., Aguilar, J., Terán, O., & Molina, H. (2013). A Verification Method for MASOES. *IEEE Transactions on Cybernetics, 43*(1), 64–76

Pelletier, F. J., Sutcliffe, G., & Hazen, A. P. (2017). Automated reasoning for the dialetheic logic RM3. In *30th International Florida Artificial Intelligence Research Society Conference* (pp. 110-115)

Peter, O., & Hasle, P. (2015). *Future Contingents. Stanford Encyclopedia of Philosophy*. (E. Zalta), Stanford University

Pulcini, G., & Varzi, A. C. (2018). Paraconsistency in classical logic. *Synthese, 195*(12), 5485–5496

Puerto, E., Aguilar, J., & Rodriguez, T. (2012). Automatic learning of ontologies for the semantic web: experiment lexical learning. *Revista Respuestas, 17*(2), 5–12

Ramsauer, C. (2020). Competencies of production in SMEs in assembly industries in a digital volatile business environment. *Tehnički Glasnik, 14*(3), 388–395

Robertson, S., & Zaragoza, H. (2009). *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc

Rodriguez, T., Puerto, E., & Aguilar, J. (2010). Dynamic semantics ontological framework for web semantics. In *9th WSEAS Intl. Conference on Computational Intelligence, Man-Machine Systems and Cybernetics* (pp. 91-98)

Sateli, B., Löffler, F., König-Ries, B., & Witte, R. (2017). ScholarLens: extracting competences from research publications for the automatic generation of semantic user profiles. *PeerJ Computer Science, 3*, e121

Sikos, L. F. (2017). *Description Logics in Multimedia Reasoning*. Springer International Publishing

Sorensen, R. (2018). *Vagueness. Stanford Encyclopedia of Philosophy* (E. Zalta). Stanford University

Sutcliffe, G., Schulz, S., Claessen, K., & Baumgartner, P. (2012). The TPTP typed first-order form with arithmetic. *Lecture Notes in Computer Science, 7180*, 406-419

Sutcliffe, G., & Pelletier, F. (2019). JGXYZ: An ATP system for gap and glut logics. *Lecture Notes in Computer Science, 11716*, 526-537

Zamansky, A. (2019). On recent applications of paraconsistent logic: an exploratory literature review. *Journal of Applied Non-Classical Logics, 29*(4), 382–391

Authors and Affiliations

Alexandra González-Eras[1,2,3] · Ricardo Dos Santos[2,3] · Jose Aguilar[2,3,4,5]

Alexandra González-Eras
acgonzalez@utpl.edu.ec

Ricardo Dos Santos
ricardojdsg@gmail.com

[1] Departamento de Ciencias de la Computación y Electrónica, Universidad Técnica Particular de Loja, Cdla. Universitaria San Cayetano Alto, 1101608 Loja, Ecuador

[2] Centro de Estudios en Microelectrónica y Sistemas Distribuidos, Facultad de Ingeniería, Universidad de Los Andes, 5101 Mérida, Venezuela

[3] Tepuy R+D Group. Artificial Intelligence Software Development, Mérida, Venezuela

[4] GIDITIC, Escuela de Ingeniería, Universidad EAFIT, Medellín, Colombia

[5] Departamento de Automática, Universidad de Alcalá, Alcalá de Henares, Spain

8.7 Anexo 4.C: Análisis de las contradicciones en las competencias profesionales en los textos digitales usando Lógica Dialéctica

risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Iberian Journal of Information Systems and Technologies

Recebido/ Submission: 15/09/2019
Aceitação/Acceptance: 03/11/2019

Análisis de las contradicciones en las competencias profesionales en los textos digitales usando lógica dialéctica

Alexandra González-Eras[1,2], Ricardo Dos Santos[2], Jose Aguilar[2]

acgonzalez@utpl.edu.ec, ricardojdsg@gmail.com, aguilar@ula.ve

[1] Universidad Técnica Particular de Loja, Departamento de Ciencias de la Computación y Electrónica, San Cayetano Alto, 1101608, Loja, Ecuador.

[2] CEMISID, Facultad de Ingeniería, Universidad de Los Andes, Núcleo La Hechicera, 51010, Mérida, Venezuela.

Pages: 150–163

Resumen: Este trabajo propone el análisis de la contradicción o ambivalencia que se encuentran en los perfiles profesionales publicados en medios digitales (por ejemplo, páginas web o redes sociales), mediante un modelo de axiomas fundamentados en dos tipos de fenómenos dialécticos, el fallo de la presuposición y el razonamiento contrafáctico. Además, el modelo considera medidas de similitud léxicas y semánticas en su proceso de análisis El modelo se valida contra un modelo no dialéctico tomado como referencia, usando tesauros de conocimientos y habilidades, y métricas de Completitud y Robustez como medida de rendimiento. Los resultados obtenidos muestran que la utilización de modelos dialécticos de este tipo se requiere, si se quiere tener una adecuada interpretación de los perfiles profesionales digitales usando mecanismos de razonamiento computacional.

Palabras-clave: procesamiento de lenguaje natural, lógica dialéctica, razonamiento ambiguo, DISCO, BLOOM, RM3.

Analysis of the contradictions in the professional competences in digital texts using dialectical logic

Abstract: This work proposes the analysis of the contradiction or ambivalence found in the professional profiles published in digital media (for example, web pages or social networks), by means of a model of axioms based on two types of dialectical phenomena, the failure of the presupposition and the counterfactual reasoning. In addition, the model considers lexical and semantic similarity measures in its analysis process. The model is validated against a non-dialectical model taken as a reference, using thesauri of knowledge and skills, and metrics of Completeness and Robustness as a measure of performance. The obtained results show that the use of dialectical models of this type is required, in order to have an adequate interpretation of digital professional profiles using computational reasoning mechanisms.

Keywords: natural language processing, dialectical logic, ambiguous reasoning, DISCO, BLOOM, RM3.

1. Introducción

En la actualidad, la gestión de competencias encuentra complicaciones para entender cuál es el significado real de una competencia en un perfil profesional digitalizado. Una de las principales limitaciones del análisis de competencias es la interpretación que una competencia puede tener en diferentes contextos, lo cual puede conllevar a más de un significado al mismo tiempo. Se han utilizado modelos semánticos en un intento de representar las competencias y sus principales componentes, la habilidad y el conocimiento, pero estos esquemas no pueden manejar la complejidad dialéctica de las competencias (Sateli, 2017).

En particular, desde el ámbito de la Web Semántica, las ontologías han sido usadas para modelar las competencias en base a la Lógica Descriptiva, la cual describe una proposición o evento con valores de verdadero o falso, pero no ambos a la vez. En consecuencia, estos modelos no son eficientes para la representación de la ambigüedad en las competencias profesionales. Por otro lado, la lógica dialéctica permite modelar estos aspectos de ambigüedad, tal que la ambivalencia es permitida y, por lo tanto, no limita el análisis de las competencias.

El propósito del presente trabajo es determinar el alcance de la lógica dialéctica en el análisis de competencias. Para ello, se desarrolla un modelo dialéctico para la representación de la ambigüedad semántica en las competencias, presentes en los perfiles profesionales digitales. En primer lugar, se identifican los casos de ambigüedad que se pueden presentar desde la óptica de la lógica dialéctica en las competencias, para después crear un modelo de conocimiento basado en axiomas dialécticos. Con el modelo dialéctico y medidas de similitud léxicas y semánticas, se propone un esquema de análisis de competencias para perfiles profesionales digitales. Posteriormente, se realizan experimentos con el esquema de análisis de competencias sobre perfiles profesionales recolectados del Internet, para analizar los casos donde el modelo dialéctico es útil. Para la evaluación del modelo dialéctico se usan medidas de rendimiento sobre los axiomas, basadas en la Completitud y Robustez del modelo para el reconocimiento de sucesos dialécticos.

2. Lógica Dialéctica y la ambigüedad de las competencias

La gestión de competencias en los contextos académicos y laborales se basan en procesos dinámicos diferentes, debido a las diversas interpretaciones que ambos contextos tienen acerca de las competencias (Malzhan, 2013), lo que impide establecer alineamientos entre los perfiles académicos y profesionales (Dorn, 2007). Esto en parte se debe la ambigüedad presente en los anuncios de los perfiles profesionales.

Los lenguajes formales de representación del conocimiento, como la Lógica de Primer Orden (First Order Logic) o la Lógica Descriptiva (Descriptive Logic), no representan

los casos de contradicción existentes en un contexto (Aguilar, 2011), Puerto, 2012). En cambio, los modelos de Lógica Dialéctica describen las ambigüedades lingüísticas mediante axiomas que pueden ser verdaderos o falsos al mismo tiempo (Maybee, 2016). De esta manera, los axiomas dialécticos permiten que las contradicciones y ambivalencias sean válidas dentro de un modelo formal (Pulcini, 2018). Según (Sutcliffe, 2017), existen los siguientes fenómenos dialécticos: vaguedad, fallo de una presuposición, razonamiento contrafáctico, discurso ficticio y declaraciones contingentes sobre el futuro. En particular, en este trabajo trabajaremos los casos de ambigüedad dialéctica siguientes:

- ***Fallo de una presuposición*** implica la suposición de algo que no es realmente cierto (Beaver, 1997). En el caso del ejemplo de la Tabla 1, se muestra el verbo "diseñar" y sus sinónimos, según el tesauro BLOOM definido en (González-Eras & Aguilar, 2019). Cada sinónimo aporta un significado diferente a la frase, ya que cada uno pertenece a un nivel cognitivo diferente (inferior, superior). Por ejemplo, en el caso de la competencia "diseñar aplicaciones móviles", según los sinónimos del verbo, la competencia se encuentra en varios niveles cognitivos a la vez. Estas contradicciones representan fallos de presuposición, ya que generan múltiples valores de verdad según la base de conocimiento (tesauro) que se utilice como contexto de referencia, lo que puede producir planteamientos erróneos de competencias y dominios de competencias en los perfiles profesionales.

Habilidad	Sinónimo	Nivel cognitivo	Proceso Cognitivo
Diseñar	Bosquejar	Conocimiento	Inferior
	Trazar	Aplicación	Inferior
	Planear	Síntesis	Superior

Tabla 1 – Casos de Fallo de una presuposición

- ***Razonamiento contrafáctico*** es el caso donde el significado de las afirmaciones causales se puede explicar en términos de condicionales contrafactuales de la forma "Si A no hubiera ocurrido, C no habría ocurrido" (Menzies, 2001). En el contexto de las competencias, se aplica el razonamiento contrafáctico en los supuestos o hipótesis que se realizan al alinear términos de competencias con los términos de tesauros según medidas de similitud léxicas, y establecer umbrales para determinar similitudes altas. Siguiendo el análisis realizado en la Tabla 2, vamos a plantear la siguiente hipótesis: "Un término y un tópico de un tesauro de competencias pertenecen al mismo dominio de conocimiento cuando la medida de similitud entre ellos supera el límite de 0.45". Como se observa en la Tabla 2, para los tres casos propuestos, dos pertenecen al mismo dominio porque la medida de similitud supera el umbral de 0.45. Pero si cambiamos el valor límite a 0.51, vemos que solo el caso "software" versus "Programación" cumple con la hipótesis. Ahora, el valor del umbral es subjetivo, causando errores y ambivalencias en la interpretación de la pertenencia de un término a un dominio de conocimiento.

Término	Tópico	Dominio	Similitud
Software	depuración de software	Programación	0.53
	Instalación de software	Instalación y configuración de TI	0.50
	Desarrollo de aplicaciones de software	Desarrollo de software	0.41

Tabla 2 – Casos de Razonamiento contrafáctico

3. Modelo de conocimiento

A continuación, se presentan los casos de ambigüedad dialéctica de fallo en la presuposición y razonamiento contrafáctico (Tablas 1 y 2), usando el modelo RM3 propuesto en (Sutcliffe, 2017). Los casos dialécticos se describen basados en 3 componentes: axiomas, que corresponden a las reglas dialécticas que los definen; hechos, que son las entradas al modelo desde las instancias extraídas de los perfiles profesionales digitales; y conjeturas, que se activan durante el razonamiento para realizar la interpretación de los perfiles profesionales digitales.

Para ello, se analizan los dos casos de ambigüedad dialéctica aplicados a términos de conocimiento y habilidad en dos tesauros, DISCO II (para conocimiento) y BLOOM (para habilidad) (González-Eras & Aguilar, 2019), con el fin de identificar la ambigüedad semántica presente para el alineamiento entre términos de competencias y los tópicos en los tesauros.

3.1. Caso 1: Tesauro DISCO II

En el primer caso, abordamos la contradicción que existe en cuanto a la pertenencia de un término de conocimiento a un tópico de un tesauro, tomando a DISCO II como tesauro de referencia (González-Eras & Aguilar, 2018). Los axiomas se definen entorno al siguiente problema (que fue explicado en la Tabla 2):

Si el término T tiene una medida de similitud Ms con un tópico Tr mayor al umbral Us, entonces pertenece al tópico raíz del tesauro TD.

En la Tabla 3 se observan los 4 axiomas que lo describen, los cuales están relacionados entre sí, de tal forma que para que se cumpla un axioma, deben cumplirse los axiomas relacionados. Por ejemplo, el axioma *"terminoPerteneceTopicos"* requiere del cumplimiento de los axiomas *"términoPerteneceTopico"*, *"terminoPerteneceVariosTopicos"* y *"terminoPerteneceAlgunTopico"*. Con estas relaciones, se describe que un término T pertenece a varios tópicos del tesauro si la medida de similitud es mayor que el umbral establecido.

Problema: Si el término T tiene una medida de similitud Ms contra un tópico Tr mayor al umbral Us entonces pertenece al tópico raíz del tesauro.	
Axiomas	fof(terminoPerteneceTopico,axiom,(! [T,Tr,Ms,Us] : ((medidaRelacion(T, Tr, Ms) & esMayor(Ms, Us)) => terminoPerteneceTopico(T, Tr)))). fof(terminoPerteneceAlgunTopico,axiom,(! [T,Tr] : ((terminoPerteneceTopico(T , Tr)) => terminoPerteneceAlgunTopico(T)))). fof(terminoPerteneceTopicos,axiom,(! [T,Tr1,Tr2] : ((terminoPerteneceTopico(T , Tr1) & terminoPerteneceTopico(T , Tr2)) => terminoPerteneceTopicos(T,Tr1,Tr2)))). fof(terminoPerteneceVariosTopicos,axiom,(! [T,Tr1,Tr2] : ((terminoPerteneceTopicos(T,Tr1,Tr2)) => terminoPerteneceVariosTopicos(T)))).
Hechos	fof(medidaRelacion1, axiom, medidaRelacion(software, programacion, s0_48)). fof(medidaRelacion2, axiom, medidaRelacion(software, depuracion_de_software, s0_52)). fof(medidaRelacion3, axiom, medidaRelacion(software, instalacion_de_software, s0_30)). fof(umbral, axiom, umbral = s0_45). fof(esMayor1, axiom, esMayor(s0_48 , umbral)). fof(esMayor2, axiom, esMayor(s0_52 , umbral)). fof(esMayor3, axiom, esMayor(s0_30, umbral)).
Conjeturas	Si "SZS status Theorem for FOF" término pertenece al tópico fof(conjetura1,conjecture, (terminoPerteneceTopico(software,programacion))). Si "SZS status Theorem for FOF" término pertenece a algún tópico fof(conjetura2,conjecture, (terminoPerteneceAlgunTopico(software))). Si "SZS status Theorem for FOF" término pertenece a los dos tópicos fof(conjetura3,conjecture, (terminoPerteneceTopicos(software,programacion,depuracion_de_software))). Si "SZS status Theorem for FOF" término pertenece a varios tópicos fof(conjetura4,conjecture, (terminoPerteneceVariosTopicos(software))).

Tabla 3 – Axiomas Caso 1

3.2. Caso 2: tesauro BLOOM

Para el segundo caso, consideramos la ambigüedad que existe entre términos de habilidades cuando pertenecen a dos niveles cognitivos distintos, esto se da debido los sinónimos que tiene un término, y a los niveles cognitivos que pertenecen estos sinónimos. El tesauro con el cual realizamos este análisis es con el tesauro BLOOM que se explica en (González-Eras & Aguilar, 2019), el cual presenta estas contradicciones. En particular, proponemos axiomas para los siguientes problemas, tal como se explica en la Tabla 1:

Si el término Th es sinónimo del término del tesauro Tb y Th y Tb tienen el mismo nivel cognitivo Nc, entonces Th tiene un solo nivel cognitivo Nc.

Si el término Th es sinónimo del término del tesauro Tb y Th pertenece al nivel cognitivo Nc1 y Tb pertenece al nivel cognitivo Nc2, entonces Th tiene varios niveles cognitivos.

En la Tabla 4 presentamos el siguiente modelo dialéctico: el axioma *"terminoPerteneceIgualNivelCognitivo"* establece la relación de inclusión del término dentro del grupo de términos pertenecientes a un nivel cognitivo, según si Th es sinónimo del término del grupo Tb; y el axioma *"terminoPerteneceVariosNivelesCognitivos"* establece que el término Th al ser sinónimo del término Tb, que pertenece a un diferente grupo de nivel cognitivo, también se considera dentro del grupo de nivel cognitivo de Tb. De esta forma, se identifica la contradicción del término Th sobre el nivel cognitivo al que pertenece.

Problema:	**Problema 1: Si el término Th es sinónimo del término del tesauro Tb y Th y Tb tienen el mismo nivel cognitivo Nc, entonces Th tiene un solo nivel cognitivo Nc.** **Problema 2: si el término Th es sinónimo del término del tesauro Tb y Th pertenece al nivel cognitivo Nc1 y Tb pertenece al nivel cognitivo Nc2 entonces Th tiene varios niveles cognitivos.**
Axiomas	fof(terminoPerteneceIgualNivelCognitivo,axiom,(![Th,Tb,Nc] : ((esSinonimo(Th,Tb) & perteneceNivelCognitivo(Th,Nc) & perteneceNivelCognitivo(Tb,Nc)) => terminoPerteneceIgualNivelCognitivo(Th, Tb, Nc)))). fof(terminoPerteneceVariosNivelesCognitivos,axiom,(![Th,Tb,Nc1,Nc2] : ((esSinonimo(Th,Tb) & perteneceNivelCognitivo(Th,Nc1) & perteneceNivelCognitivo(Tb,Nc2)) => terminoPerteneceVariosNivelesCognitivos(Th, Tb, Nc1, Nc2)))).
Hechos	fof(esSinonimo1, axiom, esSinonimo(disenar, bosquejar)). fof(esSinonimo2, axiom, esSinonimo(disenar, planear)). fof(perteneceNivelCognitivo1, axiom, perteneceNivelCognitivo(disenar, sintesis)). fof(perteneceNivelCognitivo2, axiom, perteneceNivelCognitivo(bosquejar,conocimiento)). fof(perteneceNivelCognitivo3, axiom, perteneceNivelCognitivo(planear, aplicacion)).
Conjeturas	Si "SZS status Theorem for FOF" término pertenece a igual nivel cognitivo fof(conjetura1,conjecture, (terminoPerteneceIgualNivelCognitivo(disenar,planear,sintesis))). Si "SZS status Theorem for FOF" término pertenece a varios niveles cognitivos fof(conjetura2,conjecture, (terminoPerteneceVariosNivelesCognitivos(disenar, bosquejar, sintesis, conocimiento))).

Tabla 4 – Axiomas Caso 2

3.3. Validación del modelo

El proceso de validación inicia con la aplicación de los axiomas de los casos 1 y 2 para cada término de habilidad y conocimiento de los perfiles profesionales, utilizando como punto de comparación los tesauros DISCO II y BLOOM) (ver (González-Eras & Aguilar 2015), (González-Eras & Aguilar 2019) para más detalles de los tesauros), considerando en el caso 1 una medida de similitud léxica para establecer la relación existente entre el término de conocimiento del perfil profesional (T) y los términos del tesauro DISCO II (Tr), según el coeficiente de Sorensen (ecuación (1)), que determina cercanías según la similitud de los pares de caracteres de los términos (Alqadah, 2011); y, en el caso 2 se determina la pertenencia del término de habilidad a un nivel cognitivo (tópico raíz del subárbol del tesauro BLOOM), si se encuentra incluido dentro de su grupo de verbos relacionados (nivel 1 del tesauro BLOOM) o de sus sinónimos (nivel 2 del tesauro BLOOM).

Como resultado, se reconocen dos tipos de sucesos: los sucesos dialécticos que corresponden a aquellos en los cuales los axiomas retornan un valor de verdad positivo (verdadero); y, los sucesos no dialécticos aquellos a los que los axiomas dan un valor de verdad negativo (falso). En base de estos sucesos, definimos para cada perfil el nivel de contradicción dialéctica de un perfil $X(sd_j,id_j)$ como el número de sucesos dialécticos (términos ambivalentes) que contiene el perfil id_j (ver ecuación (2)); y, el nivel de no contradicción de un perfil $Y(snd_j,id_j)$ como el número de sucesos no dialécticos que contiene (ver ecuación(3)).

$$Sim_{lex}(T, Tr) = \frac{2 x |pares(T) \cap pares(Tr)|}{|pares(T) + pares(Tr)|} \quad (1)$$

$$X(sd_i, id_i) = \sum_{i=1}^{n} sd_i \quad (2)$$

$$Y(snd_i, id_i) = \sum_{i=1}^{n} snd_i \quad (3)$$

Para evaluar el modelo dialéctico se usan las siguientes medidas:

Completitud. El modelo dialéctico *MD* es considerado completo con referencia al perfil profesional o académico id_i, si reconoce todos los sucesos dialécticos sd_{ji} del perfil id_i. Para el Caso 1 se calcula esta métrica para varios umbrales establecidos, como se indica en el apartado 3.1, correspondientes a 0.2, 0.3, 0.4, 0.5, 0.6; mientras que para el Caso 2 se calcula para el resultado obtenido al aplicar los axiomas. La Ecuación (4) presenta la definición de Completitud, en donde n es el número de sucesos en id_i.

$$Completitud(MD_j, id_i) = \sum_{i=1}^{n} \frac{sd_i}{n} \quad (4)$$

Robustez. El modelo dialéctico MD se considera robusto con referencia al conjunto de perfiles profesionales y académicos, si los sucesos dialécticos reconocidos sd_i son relevantes para el perfil id_i, en función del nivel del árbol donde fueron detectados. Dada la ambigüedad de los sucesos dialécticos sd_i (pueden presentarse en diferentes niveles del árbol a la vez), el nivel del árbol de un sd_i se asigna según la mayor frecuencia del nivel que más cerca se encuentre de la raíz. Una vez obtenido el nivel para sd_i, se calcula la robustez del MD para id_i según las ecuaciones (5) y (6)

$$Robustez(MD_j, id_i) = \sum_{i=1}^{n} \frac{Robustez(sd_i, id_i)}{n} \quad (5)$$

$$Robustez(sd_i, id_i) = \sum_{i=1}^{n} \frac{N - nivel_{sdi} + 1}{N} \quad (6)$$

Donde, *Robustez* (sd_i, id_i) es la diferencia entre N (altura máxima del árbol del tesauro donde se ubican los sucesos del perfil id_i) y el nivel asignado a sd_i en el árbol.

4. Experimentación

Para el experimento, se toman como entrada 15 perfiles profesionales en español: 10 ofertas de carrera obtenidas de portales universitarios (id_1,...,id_{10}), y 5 ofertas laborales obtenidas de portales de empleo en Internet (id_{21}...id_5), donde se usó esta nomenclatura para proteger la identidad de los autores de los perfiles. De cada perfil, fueron seleccionados los textos que se encontraban bajo apartados como: descripción, objetivos, competencias. Además, se utilizaron dos tesauros para la fase de alineamiento: DISCO II (Tc_1...,Tc_{15}) y BLOOM (Th_1...,Th_6).

Completitud del modelo dialéctico

La Tabla 5 presenta los resultados de la aplicación del modelo dialéctico *MD* sobre los perfiles, con respecto a la medida de Completitud. En el Caso 1 se observa una relación inversamente proporcional entre el umbral y el valor de completitud de cada perfil, es decir, a medida que el valor del umbral aumenta, el valor de completitud disminuye tendiendo a cero cuando el umbral se acerca a 1. Por ejemplo, los umbrales U1 (0.2) y U2 (0.3) presentan una gran cantidad de perfiles en los que la medida de completitud es igual a 1 (reconoce todos los sucesos existentes en el perfil como dialécticos), en comparación con los existentes en el umbral U6(0.6), donde los perfiles tienen un valor de completitud de cero y no aplica (n/a: según el modelo, los perfiles no presentan sucesos dialécticos). En consecuencia, podemos afirmar que si el umbral es alto (cercano a 1) el modelo no reconoce a toda la población de sucesos dialécticos que existe en los perfiles, mientras que si el umbral es bajo la población de sucesos dialécticos reconocidos aumenta; debido a la dispersión provocada por el uso de la medida léxica, en el cálculo de la similitud de los sucesos de un perfil contra un tesauro.

Por otra parte, en el Caso 2 observamos que cerca del 50% de los perfiles tienen un valor de completitud superior a 0.5, lo que indica que el modelo reconoce una gran cantidad de sucesos dialécticos en los perfiles, debido a la sinonimia que presentan los términos de habilidad (verbos) según el tesauro Bloom. Cabe mencionar que existen perfiles en donde se detecta que no existen sucesos dialecticos por el umbral usado (ver en el Caso 1 el id2 para el umbral U6), o no existen términos de habilidad en el perfil a evaluar (en el Caso 2 el perfil id13), para el análisis con el modelo MD (n/a).

id	Caso 1					Caso 2
	U1 0.2	U2 0.3	U4 0.4	U5 0.5	U6 0.6	
1.	1	1	0.8	0.7	0.2	0.57
2.	1	1	0.5	n/a	n/a	1.00
3.	1	1	1	1	1	0.50
4.	1	1	1	0.8	0.5	0.40
5.	1	1	1	1	1	0.50
6.	1	1	1	1	n/a	1.00
7.	1	1	1	1	1	1.00
8.	1	1	1	1	0.5	1.00
9.	1	1	1	1	n/a	0.50
10.	1	1	1	n/a	n/a	1.00
11.	1	0.8	0.5	0.18	0.16	0.50
12.	1	1	0.5	n/a	n/a	0.50
13.	0.3	0.3	n/a	n/a	n/a	n/a
14.	0.83	0.81	0.8	0.6	0.5	0.80
15.	1	0.9	0.5	0.5	0.5	0.43

Tabla 5 – Resultados de la Completitud del modelo dialéctico MD

La Figura 1 presenta la comparación de resultados de la completitud del modelo dialéctico MD (Caso 1) contra la completitud del modelo descriptivo propuesto en (González-Eras & Aguilar, 2018), para ello se escoge el U2 (0.3) porque el 90% de los perfiles alcanzan un valor de completitud superior al 80%; que implica que el modelo MD reconoce una mayor cantidad de sucesos ambiguos dado que identifica las contradicciones y ambivalencias de los perfiles en función de los tesauros y la medida de similitud léxica. En cambio, en el modelo descriptivo vemos que el reconocimiento disminuye en vista de que en la comparación con los términos del tesauro se usa una medida híbrida que considera la estructura del subárbol donde se ubica el suceso dialéctico detectado, reduciéndolo a un solo nivel para el suceso.

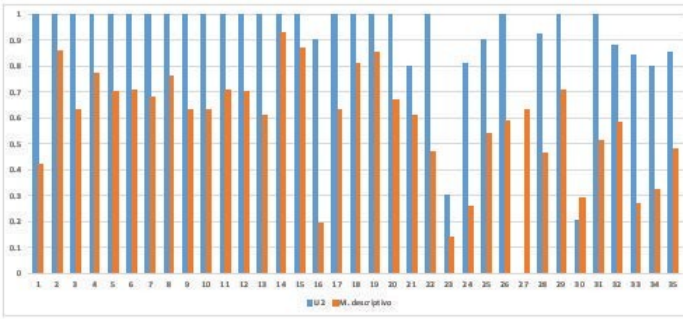

Figura 1 – Comparación de modelos dialéctico y descriptivo según su Completitud

Robustez del modelo dialéctico

La Tabla 6 presenta los resultados de la aplicación del modelo dialéctico *MD* sobre los perfiles, con respecto a la medida de Robustez, en el Caso 1, donde existe una relación directamente proporcional entre el umbral y el valor de robustez de cada perfil, es decir a medida que el valor del umbral aumenta hacia 1, el valor de robustez aumenta. Por ejemplo, en los umbrales U1 (0.2) y U2 (0.3) los valores de la medida oscilan entre el 0.5 y 0.8, y para los umbrales U3 (0.4), U4 (0.5) y U5 (0.6) los valores de la medida se mantienen entre 0.7 y 0.8, y en algunos casos tienden hacia 1. En definitiva, podemos afirmar que si el umbral es bajo (cercano a 0) la robustez del modelo MD es baja, lo que implica que los sucesos dialécticos reconocidos en el perfil se encuentran en los niveles más bajos del árbol del tesauro y, por lo tanto, no son relevantes; mientras que, si el umbral aumenta hacia 1, la robustez del modelo es alta, ya que los sucesos dialécticos reconocidos en el perfil se encuentran en los niveles más altos del árbol y, en consecuencia, son relevantes. Esta tendencia se mantiene en el Caso 2, donde el 90% de los perfiles tienen una medida de Robustez superior a 0.5, lo que indica que el modelo reconoce una gran cantidad de sucesos dialécticos relevantes en los perfiles, debido a que se encuentran en los niveles más altos del tesauro Bloom. Así mismo, en algunos perfiles no aplica el cálculo de la Robustez porque no presentan sucesos dialécticos.

id	Caso 1					Caso 2
	U1 0.2	U2 0.3	U3 0.4	U4 0.5	U5 0.6	
1.	0.63	0.71	07	0.8	0.8	0.83
2.	0.8	0.8	0.9	n/a	n/a	0.67
3.	0.6	0.7	0.8	0.8	0.8	100
4.	0.75	0.8	0.8	0.8	0.8	0.67
5.	0.73	0.8	0.8	0.8	0.8	100

id	Caso 1					Caso 2
	U1 0.2	U2 0.3	U3 0.4	U4 0.5	U5 0.6	
6.	0.8	0.8	0.8	0.8	n/a	1.00
7.	0.73	0.8	0.8	0.8	0.8	0.83
8.	0.65	0.65	0.73	0.8	0.8	0.89
9.	0.73	0.8	0.8	0.8	n/a	1.00
10.	0.6	0.8	0.8	n/a	n/a	1.00
11.	0.61	0.63	0.75	0.8	0.8	0.81
12.	0.7	0.7	0.8	n/a	n/a	0.67
13.	0.6	0.6	n/a	n/a	n/a	n/a
14.	0.64	0.7	0.7	0.7	0.8	1.00
15.	0.67	0.7	0.78	0.8	0.8	1.00

Tabla 6 – Cálculo de la Robustez del modelo dialéctico MD

La Figura 2 presenta la comparación de resultados de la robustez del modelo dialéctico MD contra la robustez del modelo descriptivo propuesto en (González-Eras & Aguilar, 2018), usando el umbral U2 (0.3), el cual aplica a todos los casos con valores de Robustez superiores a 0.5 (Alto). Observamos que, en general, el modelo dialéctico considera relevantes a una gran cantidad de sucesos dialécticos de los perfiles que tienen una frecuencia de aparición alta en uno de los niveles altos del árbol del tesauro. Por otra parte, el modelo descriptivo es más restrictivo porque asigna la relevancia a un suceso en función de la frecuencia de aparición del mismo en el nivel 1 del árbol, sin considerar los niveles intermedios.

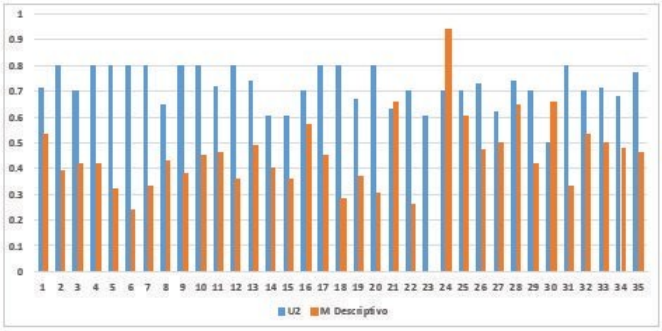

Figura 2 – Comparación de modelos dialéctico y descriptivo según su Robustez

6. Comparación con otros trabajos

La Tabla 7 presenta una comparación de la propuesta con trabajos anteriores considerando los siguientes criterios: modelo conocimiento usado, componentes de competencia considerados, estrategia de desambiguación, Tesauros utilizados como fuente, y métricas de validación del modelo propuesto. Se puede observar, que existen coincidencias entre los trabajos y nuestra propuesta en el uso de medidas de similitud léxicas y semánticas, pero la nuestra agrega el modelo dialéctico para analizar los sucesos ambivalentes. En cuanto al uso de información extra, muchos usan tesauros, lexicones o enlazado de datos como punto de referencia para establecer similitudes, los cuales combinan con medidas de similitud para el tratamiento de casos de desambiguación. En cuanto a la validación del modelo, las medidas de Completitud y Robustez de nuestra propuesta son las únicas que permiten identificar casos de ambivalencia y contradicción que los otros modelos no reconocen, y que generan sesgos en la comparación de competencias en función de sus componentes de habilidad y conocimiento. Finalmente, la mayoría usa ontologías, nuestra propuesta es la única que propone un conjunto de axiomas que permiten el reconocimiento de casos de contradicción.

Trabajo	Modelo	Componente de competencia	Desambiguación	Tesauros	Fuente de datos	Validación
(Hassan, 2012)	ontología	Conocimiento	Promedio	WordNet	Ofertas laborales	Un experto
(Paquette, 2012)	ontología	Habilidad	Axioma LPO	LinkedData ontología competencia	Perfiles	Precisión y relevancia
(Fazel 2013)	ontología	Conocimiento	Medida Similitud	Ontología competencia	Corpus	Precisión Recall
(Malzhan 2013)	ontología	Conocimiento	Semántica	Lexicón y Onomasticón	Texto de la Web	Precisión y relevancia
(Mendonça, Aguilar, Perozo 2015)	ontología	Conocimiento	Algoritmo ACO	-----	Texto de la Web	Precisión
(Gil-Vallejo, 2018)	patrones	Verbos	Coseno Dice	Adesse Wordnet Sensem	Casos verbales	Pearson, Ratio
Nuestra propuesta	ontología y Lógica dialéctica	Conocimiento y habilidad	Axiomas dialécticos y Medidas Similitud	DISCO II BLOOM	Ofertas laborales y académicas	Sucesos dialécticos Completitud Robustez

Tabla 7 – Comparación de la propuesta con otros trabajos

7. Conclusiones y trabajos futuros

El presente trabajo propone un modelo para la representación de la ambigüedad y contradicción en las competencias profesionales, fundamentado en axiomas definidos

según dos casos de ambigüedad dialéctica: el fallo en la presuposición y el razonamiento contrafáctico. Los resultados obtenidos permiten la interpretación correcta de perfiles profesionales digitales, al poder ser usada esta propuesta como una extensión de los modelos de representación formal de las competencias basados en lógica descriptiva, usando sus componentes de habilidad y conocimiento. La validación del modelo a través de las medidas de Completitud y Robustez, permite determinar la presencia de sucesos dialécticos y no dialécticos en los perfiles, además de su relevancia según tesauros de habilidad y conocimiento.

Los trabajos futuros se orientarán hacia la integración del modelo de conocimiento propuesto con modelos semánticos, como ontologías basadas en datos enlazados, de tal forma que permitan un análisis más profundo de la información sobre competencias obtenidas desde la Web, tal que las contradicciones en el lenguaje natural en las competencias sean correctamente analizadas en los perfiles profesionales. Además, se pretende generar axiomas para los casos de contradicción no analizados en este artículo (vaguedad del lenguaje natural, discurso ficticio y declaraciones contingentes sobre el futuro), y mejorar los procesos de validación de los resultados del modelo dialéctico con otras métricas de calidad, que reflejen la robustez del modelo dialéctico.

Referencias

Aguilar J. (2011) Temporal Logic from the Chronicles Paradigm: learning and reasoning problems, and its applications in Distributed Systems, LAP Lambert Academic Publishing.

Alqadah, F., & Bhatnagar, R. (2011). Similarity measures in formal concept analysis. Annals of Mathematics and Artificial Intelligence, 61(3), 245-256.

Beaver, D. I. (1997). Presupposition. In Handbook of logic and language (pp. 939-1008). North-Holland.

Dorn, J., & Pichlmair, M. (2007). A Competence Management System for Universities. In ECIS (pp. 759-770).

Fazel-Zarandi, M. (2013). Representing and Reasoning about Skills and Competencies over Time (Doctoral dissertation).

Gil-Vallejo, L., Castellón, I., & Coll-Florit, M. (2018). Similitud verbal: Análisis comparativo entre lingüística teórica y datos extraídos de corpus. Revista signos, 51(98), 310-332. Rocha, Á. (2012). Framework for a Global Quality Evaluation of a Website. *Online Information Review*, *36*(3), 374-382. doi:10.1108/14684521211241404

González-Eras, A. G., & Aguilar, J. (2015). Semantic Architecture for the Analysis of the Academic and Occupational Profiles Based on Competencies, Contemporary Engineering Sciences, 8, 1551—1563.

González-Eras, A., & Aguilar, J. (2018). Esquema para la actualización de Ontologías de Competencias en base al Procesamiento del Lenguaje Natural y la Minería Semántica. Revista Ibérica de Sistemas e Tecnologias de Informação, (E17), 433-447.

González-Eras, A., & Aguilar, J. (2019) Determination of professional competencies using an alignment algorithm of academic and professional profiles, based on competence thesauri and similarity measures, International Journal of Artificial Intelligence in Education.

Hassan, F. M., Ghani, I., Faheem, M., & Hajji, A. A. (2012). Ontology matching approaches for e-recruitment. International Journal of Computer Applications, 51(2).

Maybee, J. E. (2016). Hegel's dialectics.

Mendonça, M., Perozo, N., & Aguilar, J. (2015, July). An approach for Multiple Combination of Ontologies based on the Ants Colony Optimization Algorithm. In Asia-Pacific Conference on Computer Aided System Engineering (APCASE), (pp. 140-145).

Menzies, P. (2001). Counterfactual theories of causation. Stanford Encyclopedia of Philosophy.

Paquette, G., Rogozan, D., & Marino, O. (2012). Competency comparison relations for recommendation in technology enhanced learning scenarios. CEUR Workshop Proceedings.

Pulcini, G., & Varzi, A. C. (2018). Paraconsistency in classical logic. Synthese, 195(12), 5485-5496.

Puerto E., Aguilar, J., & Rodriguez T (2012). Automatic Learning of Ontologies for the Semantic Web: experiment lexical learning, Revista Respuestas, 17(2), 5-12.

Sateli, B., Löffler, F., König-Ries, B., & Witte, R. (2017). ScholarLens: extracting competences from research publications for the automatic generation of semantic user profiles. PeerJ Computer Science, 3, e121.

Sutcliffe, G., Pelletier, F. J., & Hazen, A. P. (2017, May). Automated Reasoning for the Dialetheic Logic RM3. In The Thirtieth International Flairs Conference.

8.8 Anexo 5.A: A meta-learning architecture based on linked data.

A Meta-Learning Architecture based on Linked Data

Ricardo Dos Santos
CEMISID, Universidad de Los Andes, Mérida, Venezuela.
ricardojdsg@gmail.com

Jose Aguilar
CEMISID, Universidad de Los Andes, Mérida, Venezuela
GIDITIC, Universidad EAFIT, Medellín Colombia
Universidad de Alcalá, Dpto Automática, España
aguilar@ula.ve

Eduard Puerto
GIA, Universidad Francisco de Paula Santander, Cúcuta, Colombia
eduardpuerto@ufps.edu.co

2021 XLVII Latin American Computing Conference (CLEI) | 978-1-6654-9503-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/CLEI53233.2021.9640223

Abstract—**In Machine Learning (ML), there is a lot of research that seek to automate specific processes carried out by data scientists in the generation of knowledge models (predictive, classification, clustering, etc.); however, an open problem is to find mechanisms that allow conferring the ability of self-learning. Thus, a meta-learning mechanism is required to allow ML techniques to self-adapt in order to improve their performance in problem solving, and even in some cases, to induce the learning algorithm itself. In this context, our research defines a meta-learning architecture using Linked Data (LD) for the automatic generation of knowledge models. Specifically, this intelligent architecture is formed by the layers of Knowledge Sources, Meta-Knowledge and Knowledge Modelling, to unify all processes to guarantee a Meta-Learning process. The Knowledge Sources layer is responsible for providing semantic knowledge about the processes of generation of knowledge models; the Meta-Knowledge layer is responsible for controlling the different processes and strategies for the automatic generation of knowledge models; and finally, the Knowledge Modelling layer is responsible for executing ML tasks defined by the Meta-Knowledge layer, among which are the tasks of feature engineering, ML algorithm configuration, model building, among others. Additionally, this article presents a case study to analyze the behavior of the different layers of the architecture, to generate knowledge models. Thus, the main contribution of this research is the definition of a Meta-Learning architecture for ML techniques, which takes advantage of the semantic information described as LD when generating the knowledge models. The preliminary results are very encouraging**

Keywords—Meta-Learning, Meta-Knowledge, Linked Data, Machine Learning

I INTRODUCTION

In the last years, a lot of research has been developed in the area of Machine Learning (ML), with special emphasis on generating ML models for very specific tasks. For this, it has been detailed algorithms and mechanisms that allow optimizing the results, but all these processes are carried out manually by the developer, such that the expert must define the hyper-parametrization techniques, load balancing strategies of the classes, etc. Particularly, in the ML processes are necessary the interventions of experts or data scientists, who is in charge of carrying out the tasks of information extraction from the data sources, data pre-processing, selection of the ML algorithms, adjustments of the hyper-parameters of the ML algorithms, among other things. However, for some of these tasks, there are algorithms that can help, but the developer must still choose which one to use (for example, in the case of hyper-parametrization).

Therefore, an intelligent architecture that allows managing these ML tasks, which exploits the experiences from the experts or data scientists, or metadata about the use of ML techniques, is required. That is, such an architecture must be able to correctly handle data by satisfying three principles [1]: i. data-centric, ii. data-driven, and iii. data-aware. The concept that covers this capacity is Meta-Learning [2]. In general, a meta-learning mechanism is a scheme that allows ML techniques to learn how to learn, through tasks that quickly adapt the ML techniques to new environments [2], [3]. However, Meta-Learning needs knowledge of the different ML processes to adequate the ML techniques to the requirements of the problems to be solved. In general, it needs a Meta-Knowledge that includes knowledge of the data sources, knowledge of the data features (Meta-Feature) and knowledge of the knowledge models (Meta-Model).

On the other hand, in the different ML processes, a great amount of information is generated, which is often not exploited. Besides, this information must be interconnected with the information that is scattered on the web to add semantic to these processes. This would allow exploiting the environment knowledge for the ML model generation, which in turn, would help to optimize each one of the ML processes with new strategies, ideas, etc. A solution that can be used for this type of problem is the paradigm of Linked Data (LD) [4]. LD has the ability to semantically enrich data, facilitates the exchange of data, among other things, which allows exploiting the web distributed knowledge in the processes of generation and optimization of knowledge models. All this thanks to the LD mechanisms that allow identifying, describing, connecting and relating the different semantic elements in the web [5], [6], [7].

A meta-learning architecture that takes advantage of the LD characteristics is interesting because it would allow automating the generation of knowledge models using the semantic information in the web. Thus, the different components of a meta-learning mechanism will be enriched with this source to improve their performance.

This research aims to define a Meta-Learning Architecture using LD. To achieve this goal, the proposed architecture is composed of three layers: Knowledge Sources, Meta-Knowledge and Knowledge Modelling. The first layer provides the knowledge sources enriched with semantic information,

978-1-6654-9503-5/21/$31.00 ©2021 IEEE

deployed as LD, to provide knowledge about the processes of generation of knowledge models. The second layer manages all the Meta-Knowledge about ML, in order to control the different processes and strategies for the generation of knowledge models. Finally, the third layer executes the tasks for the generation of knowledge models, such as feature engineering, algorithm configuration, model building, among others. Thus, the main contributions of this work are:

- The definition of a meta-learning architecture that harnesses the capabilities of LD, with the aim of providing semantic information to the process of generating knowledge models.
- The specification of the mechanisms to automate the generation of ML models from the statement of the problem until the optimization of the generated models by exploiting LD during the meta-learning process.
- The specification of the knowledge model for the architecture, which is enriched by the LD paradigm.

This article is organized as follows: Section 2 describes related works, Section 3 describes the Meta-Learning Architecture using LD; and Section 4 describes several case studies where the Meta-Learning Architecture is used. Section 5 compares our approach with other architectures; and finally, Section 6 presents the conclusions and future works of this research.

II THE META-LEARNING ARCHITECTURE

A General Architecture

An ML architecture requires following a set of steps or processes to generate a knowledge model. In our architecture, it follows the three phases proposed by the MIDANO methodology [8], [9], which are: in phase 1, the sources for knowledge extraction are identified. In phase 2, the data are prepared, i.e., the data available in the knowledge sources are processed using feature engineering tasks. Finally, in phase 3, different tasks are implemented to generate the required knowledge models, such as algorithm configuration, and model building and integration. Thus, our architecture carries out the different tasks of data experts or scientists for the generation of ML models, who is in charge of tasks such as information extraction from data sources, data processing, selecting of ML algorithms, adjusting of the hyper-parameters of the ML algorithms, among others. For that, our architecture is structured in three levels of knowledge (see Fig. 1): i) Knowledge Sources: This level performs what is indicated in phase 1 of MIDANO (identification of the data sources). In addition, it provides information collected at all levels of the architecture about previously created models (see Section III.B.i). For example, it provides data on the settings used in the different ML tasks in other similar models, provides the dataset to be used in the generation of the knowledge model, among others. This level is composed by the Linked Data Module (LDM). ii) Meta-Knowledge: At this level, all the knowledge related to the processes and strategies to generate the knowledge models is handled, i.e., the role of the expert or data scientist is played (see Section III.B.ii). This level is composed by the modules Meta-Learning (MLM), Meta-Feature (MFM), Meta-DataSet (MDSM) and Meta-Model (MMM). iii) Knowledge Modelling: In this level, phases 2 and 3 of the MIDANO methodology are carried out (see Section III.B.iii). This level is composed of the modules Feature Engineering (FEM), Tuning (TM), Model Building (MBM) and Model Integration (MIntM). In general, the architecture follows the steps determined by the MLM module of the Meta-Knowledge layer, as can be seen in the Macro-Algorithm in Table I.

TABLE I: MACRO-ALGORITHM OF THE ARCHITECTURE

Input: Problem.
Procedure: 1. Process: MLM identifies the data source for the problem (see Table III) using LDM. 2. MLM activates each ML process: 2.1. Process: activates FEM (see Table VII) 2.2. Process: activates TM (see Table VIII) 2.3. Process: activates MBM (see Table IX) 2.4. Optional Process: activates MIntM (see Table X) 3. MLM repeats step 2 for each new model to be considered. 4. MLM returns the most optimal knowledge model.
Output: Knowledge Model.

The process begins when the architecture receives the problem to be solved. MLM proceeds to identify the data sources to be used to solve the problem using LD (Step 1). MLM then proceeds to activate the ML processes to generate the knowledge model (Step 2). In step 2.1, the Dataset is prepared to be consumed for the generation of the ML model (See Table VII). In step 2.2, the configuration of the ML algorithm is created that will be used in the generation of the ML model (See Table VIII). In step 2.3, the ML model is built and evaluated (See Table IX). In step 2.4, the ML models that have been previously evaluated are integrated (See Table X). In step 3, MLM determines if all or some of the processes of step 2 have to be repeated to create a new model. Finally, in step 4 the optimal knowledge model for the problem received is returned.

B. Levels of knowledge of the Architecture

i) Knowledge Sources: This level is composed by the sources that provide information about the necessary elements in the ML processes, such as Dataset to be used, Features (knowledge extracted from the dataset), Hyper-parameters, knowledge Models to build, Validation techniques, among others. In our architecture, it will be mainly made up of LD sources, which provide data with semantic information. In addition, to represent the knowledge generated in the different processes of the architecture, the DataBench Ontology [10] is used (see Fig. 2). This ontology is composed of three modules: *1) Benchmarking Module:* it includes elements to measure the experiments from the perspective of the generation ecosystem, covering the hierarchy of indicators to measure individual benchmarking experiments, ranging from technical to business aspects. Among the labels available are the following: Benchmark, BenchmarkingDomain, BenchmarkingIndicator, BenchmarkingOrganisation, DataBenchmarkingIndicator, BigDataApplicationFeatures (DataSize, DataType, AnalyticsType, others), etc. *2) Technical Experimentation Module:* it describes the elements and process of ML

experimentation, using MLS (ML Schema) [1] as base. MLS includes labels for the overall ML process, such as mls:Study, mls:Experimentation, mls:Data, algorithmic process and generated models (mls:Task, mls:Algorithm, mls:HyperParameter, mls:Model, mls:ModelEvaluation, mls:EvaluationMeasure), among others. *3) Data Module:* it describes the experimentation data, ranging from metadata aspects to the provenance of the data used, using DCAT (Data Catalog Vocabulary) [2]. DCAT expands the structure of the mls:Data class of the MLS ontology, adding classes such as dcat:Catalog, dcat:Resource, dcat:Dataset, dcat:Distribution, dcat:DataService, dcat:CatalogRecord, among others.

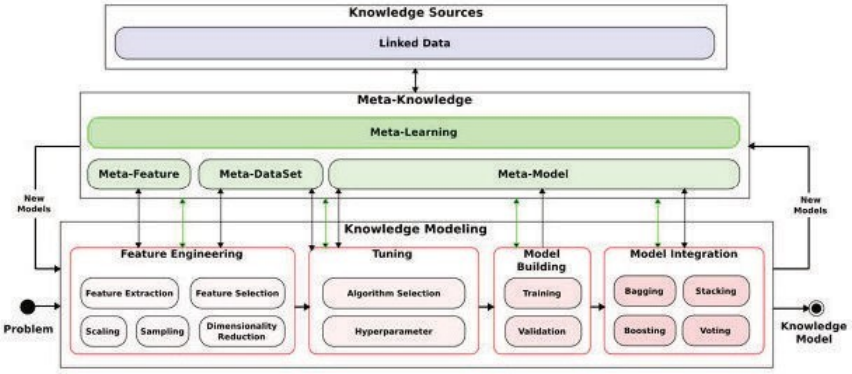

Fig. 1: ML/LD Architecture

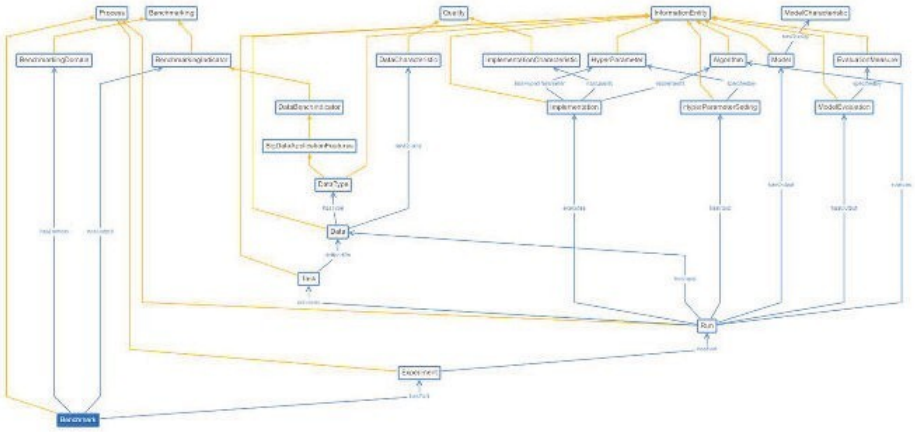

Fig. 2: The DataBench ontology [10]

The Knowledge-Sources module is described below:

1. LDM: This module uses the LD paradigm to search information about the elements that make up the ML Processes and Meta-Knowledge level, offering query mechanisms (read, create, update or delete triplets) to the rest of the components of the architecture. Thus, thanks to the LD technique, the sources of this module are connected with other data sources to exploit

[1] http://www.w3.org/2016/10/mls/

[2] https://www.w3.org/TR/vocab-dcat/

the knowledge that is distributed in the different data sources. At the following, an example reading information from the Meta-Knowledge level (AKM): the query *SELECT ?algorithm, ?dataset WHERE{ ?algorithm rdf:type mls:Algorithm. ?algorithm mls:hasInput ?dataset.}* returns the variables algorithm and dataset with the list of used algorithms that are available at the specific source. On the other hand, Table II shows the macro-algorithm that governs LDM, this process starts when it receives a query from the other modules LDM analyses and processes the query, adding information specific (Step 1). Then, the query is executed to consume the information from the AKM of due to LD (Step 2). Finally, the response is processed and sent to the requesting module (Step 3).

TABLE II: MACRO-ALGORITHM OF LDM TO PROCESS THE QUERIES OF THE MODULES OF THE ARCHITECTURE

Input: Query
Procedure: 1. LDM analyses and processes the query. 2. LDM executes the query using AKM and LD. 3. LDM processes the response.
Output: Response to query

ii) Meta-Knowledge: This level manages all the knowledge related to the processes, tasks and strategies for the utilization of ML techniques to build knowledge models. Specifically, it describes all the elements that make-up the ML, plans and organizes the processes to be executed, analyzes and evaluates the strategies used in the different processes and/or tasks, and discovers new knowledge based on experimentation, among others. The level is composed by the next modules:

1. MLM: This module analyses the responses obtained in all the processes, with the objective of learning and improving future decisions. Specifically, this module is responsible for making all the general decisions of the architecture (see Table I), i.e., it is responsible for invoking the rest of the modules specialized in each task. It is also responsible for receiving and characterizing the problem to be solved, and identifying the data source to be used to solve it (see Table III). In order to make these decisions, MLM uses the information stored by each module in the different processes in AKM, for this reason, every time it needs information, it is enough to request it to the LDM module.

TABLE III: MACRO-ALGORITHM OF MLM TO IDENTIFY THE DATASET FOR THE PROBLEM

Input: Problem.
Procedure: 1. MLM characterises the problem. 2. Process: Asks MDSM for possible Datasets for the problem to be solved (see Table 5). 3. MLM establishes the Dataset to be used.
Output: Dataset

The macro-algorithm in Table III starts when MLM receives the problem to be solved. MLM analyses the problem and characterizes it (Step 1). Then, MLM proceeds to activate MDSM by requesting possible Datasets to be used for the characterized problem (Step 2). Finally, MLM selects the Dataset to be used to solve the problem (Step 3).

2. MFM: This module is responsible for managing the knowledge about general properties of the features from the datasets (such as *mls:FeatureCharacteristic, mls:DatasetCharacteristic, mls:DataCharacteristic,* etc.). Particularly, this module specifies the information that needs to be generated from the Datasets, such as standard statistical measures, theoretical measures that relate attributes and their classes, correlation between data, among many others. In addition, it is also responsible for specifying the processes needed to clean and transform the Dataset. All the information generated and the mechanisms used for this purpose are sent to LDM to be stored in AKM.

Table IV shows the macro-algorithm that MFM performs on the Dataset. MFM analyzes the Dataset (Step 1), if it has previously processed the Dataset (Step 2). If so, then it searches in LDM for the recipe with the processes that must be carried out in the Dataset to prepare it for the construction of the ML models (Step 2.1). Else, the Dataset is processed using the knowledge of the MDSM (Step 3), doing the following: I) The general characteristics of the Dataset are determined (Step 3.1), such as the number of columns, the relationship between columns, etc. ii) The specific characteristics of each data in the Dataset are determined (Step 3.2), such as the distribution of the data, among others. iii) The recipe is generated with the processes that must be carried out in the Dataset to prepare it for the construction of the ML models (Step 3.3). Among the processes that would be described in the recipe are feature selection and extraction, missing values correction, normalization, dimensionality reduction and many more. iv) The information generated is sent to LDM. (Step 3.4). Finally, the recipe for optimizing the Dataset is returned.

TABLE IV: MACRO-ALGORITHM OF MFM TO CHARACTERIZE THE DATASET

Input: Dataset and Problem Characterized.
Procedure: 1. MFM analyzes Dataset. 2. If MFM has previously processed the Dataset. 2.1. Process: Requests LDM to search for the recipe associated to the Dataset with the processes to be performed (see Table II). 3. Else, MFM processes the Dataset using the knowledge of MDSM. 3.1. Determines the general characteristics of the Dataset. 3.2. Determines the characteristics of each data of the Dataset. 3.3. Generates the recipe with the processes to be carried out with the Dataset. 3.4. Process: Sends to LDM the information generated on the Dataset (see Table II).
Output: Recipe for optimizing the Dataset.

3. MDSM: This module is responsible for managing the specific knowledge about the data that is used to generate an ML model. For this, it keeps a record that describes the datasets and their data as the number of instances, attributes, classes, author, creation date, modification date, etc. MDSM uses standards (ontologies and vocabularies) to represent and describe these data, such as: i) DCAT: it allows defining catalogues with descriptions of the datasets using tags such as *dcat:Catalog* (collections of metadata about a dataset or data services), *dcat:Dataset* (Information about a specific dataset), *dcat:Data*

(Information about a specific data) and *dcat:Distribution* (They represent the ways to access a dataset, such as a download, Web page, Web Service, Web API or SPARQL endpoint). DCAT reuses properties of the Dublin Core[3], and FOAF[4] and SKOS[5] vocabularies. ii) VoID (Vocabulary of Interlinked Datasets)[6]: deals with metadata on RDF datasets, such as LD, allowing search, query, crawling and indexing. It also works in conjunction with DCAT. In addition, MDSM is able to search for Datasets in the data sources known by the architecture, and in turn, extract the metadata of the Datasets found. Table V shows the process that MDSM carries out when it receives a request to search Datasets for a problem. First, it analyses the characteristics of the problem (Step 1). Then, it searches for Datasets that satisfy the minimum characteristics to solve the problem (Step 2). Once it has the list of Datasets found, then it verifies if the datasets has not been previously processed (Step 3). If it has not been processed (Step 3.1), then: a) Extract the metadata from the Dataset (Step 3.1.1); b) Analyze the metadata to determine the information to be associated with the AKM (Step 3.1.2), c) Send the Collected information to LDM (Step 3.1.3).

TABLE V: MACRO-ALGORITHM OF MDSM TO OBTAIN THE DATASETS FOR THE PROBLEM

Input: Problem Characterized.
Procedure: 1. MDSM analyzes the problem characterized by MLM. 2. MDSM searches Datasets for the problem. 3. For each Dataset of the found Datasets: 3.1. If MDSM has not previously processed the Dataset. 3.1.1. Extracts metadata from the Dataset. 3.1.2. Analyzes the metadata of the Dataset. 3.1.3. Process: Sends to LDM the information extracted on the Dataset (see Table II).
Output: Datasets.

4. MMM: This module is responsible for managing knowledge about the training and validation of ML models. To do this, it keeps in AKM all the characteristics of ML models with their rules and processes for their creation (*mls:Model, mls:ModelCharacteristic*). Furthermore, it records all the validations and tests carried out on these models, with the aim of studying their quality and comparison with other ML models (*mls:ModelEvaluation, mls:Quality, mls:EvaluationMeasure, mls:EvaluationProcedure*). For example, this module can determine the need to create predictive models that strategically take advantage of the combination of various ML models, using ensemble techniques such as Bagging, Boosting, Stacking, among others. MMM responds to two types of requests (See Table VI): a) If it receives a request from the TM module (Step 1), then it proceeds to analyze the characteristics of the problem and the Dataset to determine which types of algorithms are suitable for solving the problem (Step 1.1). Then, LDM is asked for information from other models previously generated for problems similar to this one, obtaining information that describes how to build, test and evaluate the possible models to be built (Step 1.2). Finally, the recipes for building the knowledge models are generated (Step 1.3). b) If the request is received from MIntM (Step 2), then MMM requests to LDM information from the models previously generated to solve this specific problem (Step 2.1) and information from the combination models previously generated in other similar problems (Step 2.2). Then, with this information, it determines the strategies that are necessary to build the combination models for this specific problem (Step 2.3). Finally, it generates the recipes to build the knowledge models for this problem (Step 2.4).

TABLE VI: MACRO-ALGORITHM OF MMM TO OBTAIN THE KNOWLEDGE MODEL BUILDING RECIPES

Input: Optimized Dataset, Problem Characterized.
Procedure: 1. If MMM receives the request from TM. 1.1. Analyzes the problem characterized and Dataset. 1.2. Process: Asks LDM for information on previously generated models (see Table II). 1.3. Generates the recipes for knowledge model building. 2. Else, MMM receives the request from MIntM. 2.1. Process: Asks LDM for information on previously generated models (see Table II). 2.2. Process: Asks LDM for information on previously generated combination models (see Table II). 2.3. Determines the model combination strategies. 2.4. Generates the recipes for knowledge model building.
Output: Recipes for knowledge model building.

iii) Knowledge Modeling: This level executes and records all the processes of the ML. Specifically, data is processed by applying feature engineering; training algorithms are selected and used to build knowledge models; which are evaluated and integrated. The level is composed of the next modules.

1. FEM: This module is responsible for preparing the appropriate input Dataset for the ML model, using as knowledge base the information that has LDM. To do this, it follows the instructions of the MFM. Table VII presents the process for the preparation of the Dataset, for the construction of the ML models, which starts when FEM asks MFM for detailed instructions to transform the dataset for the knowledge model generation process (Step 1). Then, FEM executes the processes described in the instructions given by MFM (Step 2). Among the possible processes to be executed are: feature selection and extraction, missing values elimination, normalization, dimensionality reduction, etc. Finally, the optimized Dataset is returned.

TABLE VII: MACRO-ALGORITHM OF FEM

Input: Dataset, Problem Characterized.
Procedure: 1. Process: Asks MFM for instructions on how to optimize the Dataset (see Table IV). 2. FEM executes the instructions indicated by MFM.
Output: Optimized Dataset for ML.

[3] https://dublincore.org/
[4] http://www.foaf-project.org/
[5] https://www.w3.org/TR/skos-primer/
[6] https://www.w3.org/TR/void/

2. TM: This module is responsible for selecting the ML algorithm and preparing its configuration, using the knowledge provided by the MLM about experiences in previous runs, which allows it to identify the appropriate algorithms to solve the problem. The TM process starts (see Table VIII) when TM asks MMM for a list of possible algorithms based on the Dataset to be used and the characteristics of the problem (Step 1). Then, it proceeds to set the ML algorithm to be used for the problem (Step 2). TM generates the configuration of the selected ML algorithm (Step 3), which includes the following: i) Identify the hyper-parameters required by the ML algorithm. ii) Defines appropriate initial values of each hyper-parameter. iii) Identifies the parameters to be optimized in the training of the model. Finally, TM will return the configuration for the algorithm to be used. For example, when configuring an SVM (Support Vector Machines) with a linear kernel, it has the hyper-parameter C (Cost) that controls the bias-variance balance and the predictive capacity of the model. So, MLM must provide the possible values to use to achieve the best model.

TABLE VIII: MACRO-ALGORITHM OF TM

Input: Optimized Dataset, Problem Characterized.
Procedure: 1. Process: Asks MMM for a list of possible algorithms and their characteristics (see Table VI). 2. TM sets the ML algorithms to be configured. 3. TM generates the configuration for the ML algorithm.
Output: Algorithm Configuration.

3. MBM: This module is responsible for training and validating the ML models, using as a knowledge base the information provided by TM on the configuration of the algorithm to be used. The process begins (see Table IX), when MBM prepares the configuration for the construction of the ML model using the algorithm configuration and the optimized Dataset (Step 1). When the configuration for the construction of the model is available, the model is constructed (Step 2): i) The model is trained according to the indications described in the configuration (Step 2.1). ii) The trained model is tested (Step 2.2). iii) The parameters of the model are adjusted according to the results of the test (Step 2.3). Step 2 is repeated as many times as the configuration for the construction of the model is indicated (Step 2.4). Finally, MBM evaluates the model achieved in the construction (Step 3) and returns the model.

TABLE IX: MACRO-ALGORITHM OF MBM

Input: Optimized Dataset, Algorithm Configuration
Procedure: 1. MBM prepares the configuration for the construction of the ML model. 2. MBM builds the ML model. 2.1. The model is trained. 2.2. The model is tested. 2.3. The model parameters are adjusted. 2.4. Step 2 is repeated according to the configuration. 3. MBM evaluates the achieved ML model.
Output: Evaluated model.

4. MIntM: This module is optionally activated by MLM, when the ML models that have been previously created must be integrated, using techniques such as Bagging, Boosting, Stacking, among others. Specifically, the module allows using the basic models previously created as building blocks to design more complex models when combined, due to these basic models do not work as well by themselves, either because they have a high bias (for example, low degree of freedom models) or because they have too much variance to be robust (for example, high degree of freedom models). So, MIntM seeks to reduce bias and/or variance through an ensemble learning model. Table X shows the ensemble learning process, which starts when MIntM asks MMM for the integration strategies that will be used to build those models (Step 1). In Step 2, the different configurations are prepared according to the type of strategy to be applied. In step 3, the ensemble model is built and evaluated according to the specifications described in the configuration (Steps 3.1 and 3.2). Finally, MIntM returns the ensemble model.

TABLE X: MACRO-ALGORITHM OF MINTM

Input: Evaluated models.
Procedure: 1. Process: asks MMM for possible ensemble model strategies (see Table VI). 2. MIntM specifies the configuration of the ensemble models to be built based on the strategies. 3. MIntM builds the ensemble models. 3.1. Generates the ensemble model. 3.2. Evaluates the ensemble model. 3.3. Step 3 is repeated according to the number of ensemble models.
Output: ensemble model(s).

III CASE STUDY

This section presents a case study, which shows the capabilities of the proposed architecture.

A. Experimental Context

The objective of this case study is to show the activation process of the different modules of the architecture to create a knowledge model that solves a specific problem. In general, the context of this case study is the development of a knowledge generation service that will be deployed in a smart city, which takes advantage of the information from the sensors and actuators that are dispersed in the city. Specifically, our architecture must determine the optimal model to predict the destination of cab drivers in the city of Porto in Portugal, with the purpose of providing this information to the location-based services deployed in the city.

The data source used to generate the knowledge models of this experiment is data obtained from taxis operating in the city of Porto[7]. In this source is collected the trajectories of 442 taxis circulating in the city from July 1st 2018 to June 30th 2019, with a total of 17,106 records. Each record is composed of 9 attributes: i. *TRIP_ID (String):* id of each trip. ii. *CALL_TYPE (char):* identifies how the service was requested (A from the central, B from the taxi stand or C from a random street). iii. *ORIGINCALL (integer):* id that identifies the phone number that requested the service (if CALL_TYPE is different from A,

[7] https://www.kaggle.com/crailtap/taxi-trajectory

the value is NULL). *iv. ORIGINSTAND (integer):* id of the taxi stop (if CALL_TYPE is different from B, the value is NULL). *v. TAXI_ID (integer):* id of the taxi that made the trip. *vi. TIMESTAMP (integer):* identifies the beginning of the trip (in seconds). *vii. DAYTYPE (char):* identifies the kind of day of the trip (A normal day, B holiday or C a day before the holiday). *viii. MISSING_DATA (Boolean):* True if one (or more) locations are missing from the GPS data stream, otherwise it is False. *ix. POLYLINE (String):* contains a list of GPS coordinates with longitude and latitude, where each item represents 15 seconds of travel time. The first element of the list represents the beginning of the trip and the last element corresponds to the destination of the trip. Finally, this case study is built with the information generated in [11].

B. Utilization of the Meta-learning Architecture

The process starts when MLM receives the request to find an optimal model to predict the destination of taxi drivers in the city (see Fig. 3). Then, MLM identifies and establishes that the source to use is a Dataset with the information of the taxis operating in the city of Porto. Then, having the data source for the problem, MLM starts to activate each ML process (see steps 1, 2, 3 and 4 in Fig. 3), which can be invoked repeatedly according to the variety of models that MLM will consider (see step 3 in Table I). Each of these processes is detailed in the following sections.

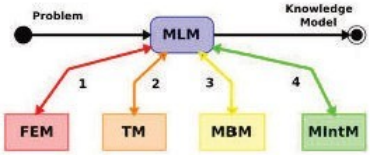

Fig. 3: Activity Diagram of the Architecture Workflow

C. Activation of FEM

This process aims to prepare the dataset that will be used in the generation of the knowledge models. This process is described below (see Fig. 4):

1 The MLM module activates FEM to perform Feature Engineering tasks to the dataset, such as feature selection and extraction, missing values processing, normalization, dimensionality reduction, many more.
2 The FEM module receives this request and asks MFM for a detailed recipe of the tasks that need perform on the dataset to optimize it.
3 Then, MFM analyzes the dataset and checks if it has been previously processed. If it has been previously processed, then it asks LDM for the recipe associated to that dataset. Else, it determines the characteristics of the dataset and of the data that compose it, and with this information, it generates the recipe to optimize the dataset. Finally, it sends the recipe to LDM to be associated with the dataset in the AKM.
4 LDM processes the query received and executes it on AKM. In this case, the query was to extract the associated recipe to the dataset or to store the generated recipe to the dataset.
5 Returns the information of the query executed by AKM.
6 Returns the recipe to optimize the dataset.
7 FEM takes the recipe and executes the different tasks indicated by MFM to optimize the dataset.
8 Finally, FEM returns the optimized dataset for the knowledge models to be generated.

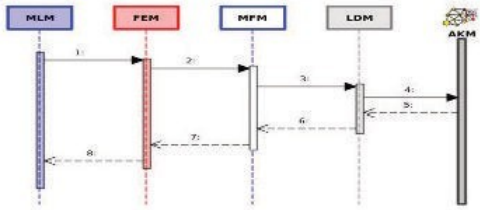

Fig. 4: Sequence diagram of FEM activation

In this particular case, the tasks indicated in the recipe for optimizing the dataset were as follows:

- MISSING_DATA: deletes all records where one (or more) locations are missing in the GPS data stream, i.e., when this attribute is TRUE.
- TIMESTAMP: This attribute represents the seconds from midnight on January 1st, 2018 until the time of its entry, however, this value provides little information for the models to be generated. Therefore, the TIMESTAMP is transformed into several characteristics such as: i. WEEK_DAY. ii. MONTH. iii. MONTH_DAY. iv. YEAR. v. YEAR_WEEK. vi. DATE. vii. HOUR. viii. MINUTE. For example, a TIMESTAMP equal to 1388745716 will be WEEK_DAY=4, MONTH=1, MONTH_DAY=3, YEAR=2018, YEAR_WEEK=1, DATE=2018-01-03, HOUR=10 and MINUTE=11.
- CALL_TYPE and DAYTYPE: these two attributes are composed of three categories (A, B and C), each attribute is converted into three new characteristics that represent in binary form the category of that attribute. For example, if CALL_TYPE is A, then the characteristics would be CALL_TYPE_A is one (1), and CALL_TYPE_B and CALL_TYPE_C are zero (0). In the same way, it would be carried out with DAYTYPE.
- POLYLINE: this attribute as it is composed of a list of GPS coordinates taken every 15 seconds, can be used to get the following characteristics: i. TRIP_START: First coordinate in the list. ii. TRIP_END: Last coordinate in the list. iii. TRIP_TIME: 15 seconds * (Number of Coordinates - 1). iv. TRIP_DISTANCE: It contains the total geodesic distance calculated from all consecutive pairs of coordinates. v. AVERAGE_SPEED: TRIP_DISTANCE / TRIP_TIME. vi. TOP_SPEED: max(Geodetic distance between consecutive coordinate pairs) / 15 seconds.

D. Activation of TM

The objective of this process is to select the ML algorithm and prepare its configuration for the construction of the

knowledge model. A description of how it works is given below (see Fig. 5):

1 The process starts when MLM activates TM to select the ML algorithm and prepare its configuration.
2 TM asks MMM for a list of possible algorithms to be used.
3 MMM analyzes the characterized problem and dataset. Then, MMM asks LDM for information on previously generated models.
4 Run the search query in AKM.
5 Returns the information of the query executed by AKM.
6 Returns recipe for configuring ML algorithms
7 TM receives the following algorithms with the configurations used in previous runs: SVR (Support Vector Regression), DBN-C (Deep Belief Network - Classification), DBN-R (Deep Belief Network - Regression), ANN-C (Artificial Neural Network - Classification), ANN-R (Artificial Neural Network - Regression), kNN-R (K-Nearest Neighbor - Regression) and NB (Naive Bayes).
8 TM after analyzing the algorithms received, it establishes that it will use SVR for this run. Finally, it generates the configuration: i) Hyper-parameters: kernel = RBF (Radial Basis Function), C = 100 and epsilon = 0.1. ii) Metrics: MAE (Mean Absolute Error) and RMSE (Root Mean Square Error). Returns to MLM the configuration of the algorithm to use.

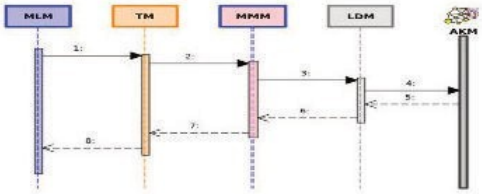

Fig. 5: Sequence diagram of TM activation

E. Activation of MBM

This process aims to train, test and evaluate ML models using the information generated in the previous processes. The following is a description of this process (see Fig. 6):

1 MLM activates MBM to generate the knowledge model. Then, MBM prepares the configuration for the ML model building. An example in Python, for this case of construction of the model with the SVR algorithm, using the dataset and hyper-parameters obtained in previous processes, is shown below:

```
from sklearn.svm import SVR
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_dataset, y_dataset)
model_SVR = SVR(kernel='rbf', C=100, epsilon=0.1)
model_SVR.fit(X_train, y_train)
```

2 MBM when building the knowledge model is kept in a loop, where it trains, tests and adjusts the parameters until it reaches the values indicated in the algorithm configuration.
3 Finally, MBM before returning the achieved knowledge model, evaluates it using the metrics established in the previous process (see section IV.B.2). In this specific case, using the SVR algorithm, it obtains an average of MAE equal to 2.10 km and RMSE of 2.73 km (more info in [11]). Below is a python example to obtain the metrics:

```
from sklearn import metrics
import numpy as np
y_pred = model_SVR.predict(X_test)
MAE = metrics.mean_absolute_error(y_test, y_pred)
RMSE = np.sqrt(metrics.mean_squared_error(y_test, y_pred))
```

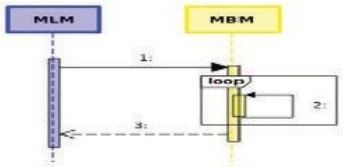

Fig. 6: Sequence diagram of MBM activation

F. Activation of MIntM

The objective of this process is to generate an ensemble ML model by using the information from the previously evaluated models. It should be noted that this process is optional. This process is described in detail below (see Fig. 7):

1 MLM activates MintM when it is necessary to generate an ensemble ML model.
2 MIntM asks MMM for possible ensemble learning strategies.
3 MMM asks LDM for information stored in AKM about previous generated models.
4 LDM processes and executes the query using the knowledge available in AKM.
5 Return the information found in AKM.
6 Then, LDM returns the information of all the models previously built for this problem (Table IV of article [11]).
7 MMM asks LDM for information on the strategies used in the previously generated combination models.
8 LDM executes the query in AKM.
9 AKM returns the information found.
10 LDM returns information on the strategies used in the ensemble models built previously.
11 MMM analyses the extracted information, determining that it gives better results with DBN-C when the taxi is before 30% of the full trajectory, otherwise SVR offers better results (see Table XI). Therefore, a bagging approach would be ideal to obtain a better model, specifically, it is indicated as a strategy to create an ELM (Ensemble Learning Model) [11].
12 MIntM builds the ensemble models indicating the strategies received by MMM. Table XI shows the results obtained in the model using ELM.

13 Finally, MintM delivers the best ensemble prediction model found.

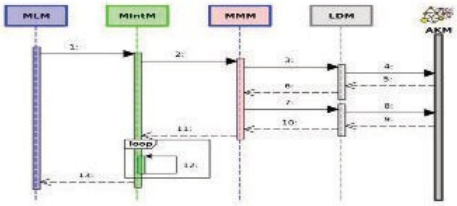

Fig. 7: Sequence diagram of MIntM activation

TABLE XI. PERFORMANCE COMPARISON BETWEEN DIFFERENT PREDICTORS [11]

Model	Metric (km)	Percentage of Whole Trajectory Completion									Average (km)
		10%	20%	30%	40%	50%	60%	70%	80%	90%	
SVR	MAE	3.44	3.27	2.98	2.64	2.21	1.76	1.29	0.85	0.42	2.10
	RMSE	4.32	4.13	3.81	3.42	2.92	2.38	1.77	1.19	0.59	2.73
DBN-C	MAE	3.08	3.06	3.00	2.91	2.80	2.68	2.54	2.42	2.28	2.75
	RMSE	3.92	3.88	3.82	3.73	3.60	3.44	3.26	3.10	2.87	3.51
ELM	MAE	3.11	3.14	3.01	2.68	2.24	1.78	1.30	0.86	0.42	2.06
	RMSE	3.99	4.02	3.83	3.44	2.93	2.39	1.78	1.20	0.60	2.69

IV COMPARISON WITH PREVIOUS WORKS

This section presents a comparison with related works. For this purpose, we are going to use several evaluation criteria based on the general processes carried out in the MIDANO methodology: I. Phase 1: focuses on the definition of data sources. II. Phase 2: focuses on the treatment of the Datasets. III. Phase 3: focuses on the generation of knowledge models. The qualitative evaluation criteria are specified below: C1: Data source based on LD. C2: Datasets processing mechanisms using Meta-Learning. C3: Mechanisms for the generation of knowledge models using LD. C4: Mechanisms for the generation of knowledge models using Meta-Learning. In Table XII, when a Criterion column has ✓ means the criterion is met.

TABLE XII. COMPARATIVE ANALYSIS WITH PREVIOUS WORKS

Works	Criteria			
	C1	C2	C3	C4
[12], [13], [14], [15], [16], [17]				✓
[18], [19], [20], [21]		✓		
[22]		✓		✓
[23]	✓			
[24], [25]	✓		✓	
Our approach	✓	✓	✓	✓

For the criteria C1 and C3 related to the LD paradigm, it is observed that the works [24], [25], [23] and our approach, seek to link the data to ontologies that allow organizing the information in a semantic way, except for [23] in the criterion C3, since this work does not specify mechanisms that use the data for the model generation process. Moreover, in [24], they specify an API that provides operators that facilitate the task of filtering and extracting previously organized information. In [25], they use their own query language called HyQL. In our case, the Meta-Knowledge layer defines the rules and/or algorithms for filtering and extracting the data related to the ML processes.

Concerning Meta-Learning, for criterion C2, it is observed that the works [18], [19], [20], [21], [22] and our approach present mechanisms for the problems of extraction/selection of meta-features that improve the process of meta-learning. In [18] is specified a method to analyze the effect on predictions of different meta-features. [19] defines an automatic approach for the analysis and selection of descriptors in a given audio context. In [20] is specified a mechanism that automatically learns atomic descriptors/features, discovering or selecting features using various feature engineering methods. In [21], it is described a Feature Selection algorithm recommender. In [22] is presented an abstraction called Meta-Data to find the relationship between the inputs and the processes for the model generation. In our case, macro-algorithms are described that take advantage of the information taken from LD by the components Meta-DataSet, Meta-Feature and Meta-Learning.

For C4, it is observed that the works [12], [13], [14], [15], [16] and [17] present Meta-Learning algorithms that allow optimizing the hyper-parameters of the ML algorithms. In [22], they look at meta-learning from another perspective. They propose the use of Meta-Modelling of complete problems, and the meta-learning algorithms what they do is to look for the most suitable model according to the characteristics of the tasks to be solved. In our case, macro-algorithms are specified that use LD, and are managed by the Meta-Knowledge layer through the Meta-Feature, Meta-Model and Meta-Learning components. This feature allows the architecture to be adapted to any ML problem.

With respect to the quantitative comparison, no such comparison is specified because the reviewed works present partial solutions to different characteristics associated with the architecture described in this paper. Furthermore, our work only presents a functional design of the architecture. In summary, no works found in the literature merge the concepts of ML with LD and Meta-Learning. In our case, a general architecture was specified that manages ML processes by taking advantage of LD and Meta-Learning. Particularly, our approach can take advantage of automation schemes such as those presented in [26], [27], [28], [29], in order to make the construction processes of solutions based on machine learning techniques completely autonomous.

V CONCLUSIONS

In this paper, we present an architecture that combines Meta-Learning with the LD paradigm. The first contribution achieved with the meta-learning architecture based on LD is to provide a semantic description of the data, processes and tools used, which allows optimizing the generation of future ML models. The second contribution achieved is the specification of mechanisms to automate the generation of ML models, by using a Meta-

Learning based manager that exploits all the information collected and linked from the ML processes. Finally, the last contribution is the knowledge model of the architecture based on the LD paradigm, where all the information collected from the processes carried out in the architecture is stored and semantically linked, called AKM.

Our architecture is composed of three layers for the generation of knowledge models. Specifically, the Knowledge Sources layer manages the internal knowledge by adding the dataset for the problem, and the external knowledge because it adds semantic information to the Datasets, which are used to generate the knowledge models. In the Meta-Knowledge layer are defined the strategies and algorithms for the construction of the knowledge models. Finally, the Knowledge Modeling layer allows the generation of models adapted to the different requirements of the problems.

This research, in comparison to other works, is the first that exploit the mix of LD and Meta-Learning in the domain of ML, there are only works that associate ML with LD or ML with Meta-Learning. In our approach, we combine the three areas into an overall architecture for ML, taking advantage of the capabilities offered by each area.

In future works, the modules of the proposed architecture will be implemented, which will allow the definition of case studies to evaluate their quality and robustness. In addition, due to the complexity and computational resources needed of the Meta-Learning architecture to automate all ML processes, we will study the implementation of a Federated Meta-Learning with LD that allows distributing the Meta-Knowledge generated in the architecture.

REFERENCES

[1] O. Muthu, "Intelligent Architectures for Intelligent Computing Systems", in proceeding Design, Automation & Test in Europe Conference, pp. 318-323, 2021.

[2] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people", Behavioral and brain sciences, vol. 40, 2017.

[3] H. Yao, X. Wu, Z. Tao, Y. Li, B. Ding, R. Li, and Z. Li, "Automated relational meta-learning", In proceeding International Conference on Learning Representations (ICLR 2020), 2020.

[4] R. Dos Santos, J. Aguilar, T. Rodriguez, "Una Revisión de la Literatura sobre Datos Enlazados", Revista Ingeniería al Día, vol. 5, no 1, pp. 54–82, 2019. ISSN: 2389-7309

[5] T. Rodriguez, R. Dos Santos, J. Aguilar, "Metodología para el desarrollo de Aplicaciones Web utilizando Datos Enlazados", in proceeding *Conferencia Nacional de Computación, Informática y Sistemas (CONCISA 2017)*, pp. 114-122, 2017.

[6] T. Berners-Lee, "Linked Data", 2006. Available in: <https://www.w3.org/DesignIssues/LinkedData.html>.

[7] R. Dos Santos, J. Aguilar, "Enlazado de Datos", in: Aguilar J. (ed), Introducción a la Minería Semántica. Fondo Editorial Universidad Nacional Experimental del Táchira, pp. 177-216. ISBN 978-980-426-008-7, 2018.

[8] J. Aguilar, M. Cerrada, F. Hidrobo. A Methodology to Specify Multiagent Systems. Lecture Notes in Computer Science, vol 4496. pp 92-101, 2007.

[9] F. Pacheco, J. Aguilar, C. Rangel, M. Cerrada, and J. Altamiranda, "Methodological framework for data processing based on the Data Science paradigm", in proceeding XL Latin American Computing Conference (CLEI), September 2014.

[10] M. Grobelnik, and M. Skjane, "D5.1: Initial Evaluation of DataBench Metrics", European DataBench Project, November 2019. Available in: <https://www.databench.eu/wp-content/uploads/2020/02/databench-d5.1-initial-evaluation-of-databench-metrics-ver.2.0.pdf>.

[11] X. Zhang, Z. Zhao, Y. Zheng, and J. Li, "Prediction of taxi destinations using a novel data embedding method and ensemble learning", IEEE Transactions on Intelligent Transportation Systems, vol. 21 no 1, 68-78, 2019.

[12] J. Rajasegaran, S. Khan, M. Hayat, F. S. Khan, and M. Shah, "iTAML: An Incremental Task-Agnostic Meta-learning Approach", in proceeding IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13588-13597, 2020.

[13] X. Li, L. Yu, Y. Jin, C. W. Fu, L. Xing, and P. A. Heng, "Difficulty-aware meta-learning for rare disease diagnosis", in proceeding International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, pp. 357-366, October 2020.

[14] S. Flennerhag, A. A. Rusu, R. Pascanu, F. Visin, H. Yin, and R. Hadsell, "Meta-learning with warped gradient descent", in proceeding International Conference on Learning Representations (ICLR 2020), 2020 arXiv:1909.00025.

[15] K. Lee, S. Maji, A. Ravichandran, and S. Soatto, "Meta-learning with differentiable convex optimization", in proceedings IEEE Conference on Computer Vision and Pattern Recognition, pp. 10657-10665, 2019.

[16] Z. Cao, T. Zhang, W. Diao, Y. Zhang, X. Lyu, K. Fu, and X. Sun, "Meta-seg: A generalized meta-learning framework for multi-class few-shot semantic segmentation", IEEE Access, vol 7, pp. 166109-166121, 2019.

[17] T. Elsken, B. Staffler, J. H. Metzen, and F. Hutter, "Meta-Learning of Neural Architectures for Few-Shot Learning", in proceeding IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12365-12375, 2020.

[18] B. Bilalli, A. Abelló, and T. Aluja-Banet, "On the predictive power of meta-features in OpenML", International Journal of Applied Mathematics and Computer Science, vol. 27, no 4, pp. 697-712, 2017.

[19] M. Jiménez, J. Aguilar, J. Monsalve-Pulido, and E. Montoya, "An automatic approach of audio feature engineering for the extraction, analysis and selection of descriptors", International Journal of Multimedia Information Retrieval, vol. 10, no 1, pp. 33-42, 2021.

[20] E. Puerto, J. Aguilar, R. Vargas, and J. Reyes, "An A2p deep learning architecture for the discovery and the selection of features", Neural Processing Letters, vol. 50, no 1, pp. 623-643, 2019.

[21] A. R. S. Parmezan, H. D. Lee, and F. C. Wu, "Metalearning for choosing feature selection algorithms in data mining: Proposal of a new framework", Expert Systems with Applications, vol. 75, pp. 1-24, 2017.

[22] T. Hartmann, A. Moawad, C. Schockaert, F. Fouquet, and Y. Le Traon, "Meta-Modelling Meta-Learning", in proceeding ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS), pp. 300-305, September 2019.

[23] M. Ruta, F. Scioscia, G. Loseto, A. Pinto, and E. Di Sciascio, "Machine learning in the internet of things: a semantic-enhanced approach", Semantic Web, vol. 10, no 1, pp. 183-204, 2019.

[24] A. Mohamed, G. Abuoda, A. Ghanem, Z. Kaoudi, and A. Aboulnaga, "RDFFrames: Knowledge Graph Access for Machine Learning Tools", 2020. arXiv preprint arXiv:2002.03614.

[25] M. Moreno, V. Lourenço, S. Fiorini, P. Costa, R. Brandão, D. Civitarese, and R. Cerqueira, "Managing Machine Learning Workflow Components", in proceeding IEEE 14th International Conference on Semantic Computing (ICSC), pp. 25-30, February 2020.

[26] M. Araujo, J. Aguilar, H. Aponte, "Fault detection system in gas lift well based on artificial immune system", in proceeding International Joint Conference on Neural Networks, vol. 3, pp. 1673-1677, 2003.

[27] J. Aguilar, M. Cerrada, G. Mousalli, F. Rivas, F. Hidrobo, "A Multiagent Model for Intelligent Distributed Control Systems", Lecture Notes in Computer Science, vol 3681, pp. 191-197, 2005.

[28] J. Terán, J. Aguilar, M. Cerrada, "Integration in industrial automation based on multi-agent systems using cultural algorithms for optimizing the coordination mechanisms", Computers in Industry, vol. 91, pp. 11-23, 2017.

[29] J. Vizcarrondo, J. Aguilar, E. Exposito and A. Subias, "MAPE-K as a service-oriented architecture," IEEE Latin America Transactions, vol. 15, no. 6, pp. 1163-1175, June 2017.

8.9 Anexo 5.B: An Autonomous Meta-Learning Architecture for Transfer Learning based on Linked Data

Dos Santos, R., Aguilar. (2025). An Autonomous Meta-Learning Architecture for Transfer Learning based on Linked Data. EN REVISTA

8.10 Anexo 5.C: An Explainable Feature Generation Approach for Classification Models Using CNNs

Dos Santos, R., Aguilar. (2025). An Explainable Feature Generation Approach for Classification Models Using CNNs. EN REVISTA

8.11 Anexo 5.D: A synthetic Data Generator for Smart Grids based on the Variational-Autoencoder Technique and Linked Data Paradigm

A synthetic Data Generator for Smart Grids based on the Variational-Autoencoder Technique and Linked Data Paradigm

Ricardo Dos Santos
CEMISID. Facultad de Ingeniería
Universidad de Los Andes
Mérida 5101, Venezuela
ricardojdsg@gmail.com

Jose Aguilar
Dpto Automática, Universidad de Alcalá, Alcalá de Henares, Spain;
CEMISID, Universidad de Los Andes, Mérida, Venezuela;
GIDITIC, Universidad EAFIT, Medellín, Colombia
jose.aguilar@uah.es

Maria D. R-Moreno
Dpto Automática, Universidad de Alcalá,Alcalá de Henares, Spain;
TNO, Intelligent Autonomous Systems Group (IAS),
The Hague, The Netherlands
malola.rmoreno@uah.es

***Abstract*—In a smart environment like the smart grids, it is necessary to have knowledge models that allow solving emerging problems. However, large datasets are required to automatically create these models, which in the vast majority of cases are not available. Therefore, in these environments is essential to have a data generator for each context. In this paper, we propose a synthetic data generation system based on the variational autoencoder (VAE) technique and linked data paradigm, to create larger datasets from small datasets acting as samples. Specifically, a Linked Data-based dataset extractor is proposed, which allows obtaining samples of data in a particular context. Then, a VAE is trained with these samples of data available in a specific context, to learn the latent distribution that characterizes the dataset, allowing the production of new records that are similar to those of the original dataset. Finally, several case studies in the field of energy management are carried out. In one of them, the process that follows our approach is described in detail; and then, another 3 more are considered to evaluate its ability to automate the data generation process for smart energy management. The results show how our synthetic data generation system can be used to obtain synthetic datasets in different energy contexts.**

***Keywords*—Synthetic Data Generator, Linked Data, Deep Learning, Variational Autoencoders, Smart Grid**

I. INTRODUCTION

A smart environment is a paradigm in which people are empowered or strengthened by the use of digital environments that are aware of their presence and context, that are sensitive, adaptive, and responsive to their needs, habits, gestures and emotions [1]. Smart environments such as smart grids require knowledge models that allow them to make decisions, organize activities, automate processes, among other things [1, 2]. However, as these problems are emerging, they require automatically to create these models [3], which may lead to the need for large datasets, which in most cases are not available. Therefore, in these environments, it is essential to have a data generator for each situation.

Data generators require two components, the first component is an algorithm or model that when fed with information from the desired situation, then it automatically extracts the characteristics that represent it. A solution to this problem is the use of Deep Learning (DL) techniques. DL is a set of Machine Learning (ML) algorithms that attempts to model high-level abstractions in data using iterative, multiple nonlinear transformations of data, which are useful in classification and recognition tasks, among other applications [4, 5]. Among the DL algorithms, we have the Variational AutoEncoder (VAE), which provides a probabilistic way to learn the latent space representations of the input data [6]. Specifically, an encoding network or recognition model is used to discover a probability distribution for each latent attribute. Then, a decoding network or generative model is used that takes these latent attributes and attempts to recreate the original input. The second component is to find a dataset that represents the desired context. This second component must generate automatically synthetic data of that situation. One way to solve this type of problem is by using the Linked Data (LD) paradigm. LD has the ability to facilitate the entailment of data, among other things, allowing the exploitation of distributed knowledge on the web [7].

Some similar research on synthetic data generators based on ML techniques are the following: Islam et al. [6] develop a vehicular collocation generator, seeking to correct the imbalance they had in the original dataset, since the dataset consists of only 625 crash events for over 6.5 million non-crash events. They used a VAE to encode all events in a latent space. After training, the model was able to successfully separate crashes from non-crashes. Finally, to generate the data, they sampled the latent space containing the crash data. Also, Chen et al. [9] implemented two vehicle trajectory generators for autonomous driving simulation and traffic analysis tasks in data-scarce cities or regions. In particular, a VAE generator, called TrajVAE, and a Generative Adversarial Generator (GAN), called TrajGAN, were developed. In the experiments with both generators, they demonstrated that the TrajVAE model effectively addresses the problem and its trajectories turn out to be more similar to the original trajectories. Salim et al. [10] developed a data generator using VAE in the healthcare environment, specifically, patient records were generated given a particular diagnosis. By training the model, the underlying patterns in the diagnoses and the patients suffering from them are discovered, and with this information, new patient records are generated. Finally, Wan et al. [13] proposed a VAE-based synthetic data generation method for the unbalanced learning problem, being effective for high-dimensional data,

IEEE 978-1-6654-7671-3/22/$31.00 ©2022 IEEE

such as images. This generation process is composed of two steps. First, VAE samples some values of the latent variable that are likely to produce the original data. Second, new samples are generated from the conditional distribution of the data given the latent variable. This method generates datasets similar to the original, but not exactly the same.

On the other hand, the generation of data in the context of smart grids is very important. For example, in the context of monitoring tasks of a data-driven smart energy management system, it is necessary a permanent update of the models of identification, detection, and diagnosis, among others [8]. It is very important for the Internet of Things (IoT), and Big Data where the sources and characteristics of new data from energy can be very unbalanced, dynamics, among other aspects [14, 17, 18, 19]. In order to update the data-driven models, the generation of synthetic data is very useful to improve, e.g., the energy consumption or production prediction. This is the reason for this preliminary study in this specific context.

In the context of smart grids, there are some works in synthetic data generation. For example, Zhang et al. [20] propose a data-driven approach to synthetic dataset generation by utilizing deep generative adversarial networks (GAN) to learn the conditional probability distribution of features of the real dataset and generate samples based on the learned distribution. Also, Kababji and Srikantha [21] define a framework for generating synthetic labeled load (e.g., appliance) patterns and usage habits for the context of smart meter data. They use GAN and kernel density estimators. The work [22] presents a framework for validating synthetic distribution data sets using statistical and operational metric targets. Asre et al. [24] define a framework for synthetic energy consumption data generation based on the Time-GAN techniques. Finally, Wiest et al. [23] describe a methodology for the generation of synthetic load profiles for industrial, commercial and agricultural customers. The methodology is based on a stochastic linear regression model, which obtains a set of parameters from the datasets for each type of customer to enable the production of new synthetic load profiles.

As can be seen in the works, the VAE algorithm is widely used in the generation of artificial data, because it offers good results from small samples of data. However, all works start from a specialized dataset prepared for a particular context/situation. In this paper, we seek to go a step further in the process of automating the generation of artificial/ synthetic data. On the other hand, in the context of the smart grids, neither approach exploits linked data to extract more useful information for the definition of synthetic data. In particular, we propose a synthetic data generator that exploits the benefits of the LD paradigm for the automatic detection and extraction of data samples for smart grids, and the learning capability of DL algorithms, such as the VAE algorithm, to know the latent distribution in the data. Thus, the main contributions of this work are:

- Define an architecture for the automatic generation of datasets for smart grids.
- Specify an automatic mechanism for the automatic detection and extraction of samples of data based on LD.
- Specify a synthetic data generator method based on VAE using the data sample extracted with LD.

This article is organized as follows: Section 2 describes the architecture of the synthetic data generator using LD and; Section 3 describes in detail a classical case study in the context of smart grids, the generation of data about the energy consumption of the clients, showing how our approach works in this context. Section 4 aims to evaluate the ability to automate the data generation process in different domains required for smart energy management. For this, three other relevant case studies are presented in the field of energy management, such as the generation of data about the energy consumption of devices and lights in a home, the production of renewable energy, and the consumption by sectors (residential, industrial, academic, government, etc.). Finally, Section 5 presents the conclusions and future work of this research

II. SYNTHETIC DATA GENERATION SYSTEM

A. Architecture

The generation of synthetic data involves a set of complex processes that allow identifying, extracting, transforming and learning the relevant features of the dataset to be generated. Thus, we propose an architecture composed of the following modules (see Fig. 1): DataSet Acquisition (DSA), Data Preparation (DP) and Synthetic Data Generation (SDG).

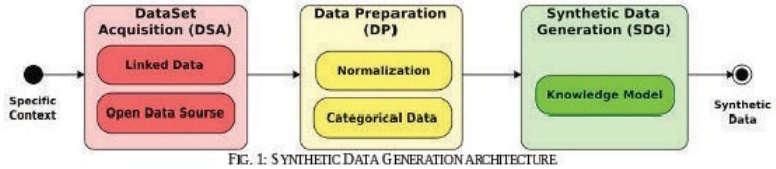

FIG. 1: SYNTHETIC DATA GENERATION ARCHITECTURE

B. DSA Module

The objective of this process is to find samples of data for the given context. Specifically, the samples of data are obtained using search mechanisms based on LD, taking advantage of Open Data Sources (ODS) o endpoints, such as those officially provided in countries like the United States (https://www.data.gov/), Spain (https://datos.gob.es/) and Europe (https://data.europa.eu), among many others. All this is possible because these data sources publish all the metadata of their datasets using the CKAN standard (https://ckan.org/), which allows queries using the Sparql language. For example, the query *SELECT ?id ?title ?title ? url WHERE{ ?id dct:title ?title. ?id dcat:accessURL ?url.}*.

Table 1 shows the DSA macro-algorithm. This process starts by analyzing the context of the required samples of data, obtaining the keywords to search for the data samples (Step 1). Then, the LD query that will be used to search for the datasets in the ODS is prepared (Step 2). Finally, the

query is executed and a list of datasets is received, and the dataset that best fits the search is selected (Step 3).

TABLE 1: MACRO-ALGORITHM OF DSA TO SEARCH SAMPLES OF DATA.

Input: Specific Context
Procedure: 1. The context of the required data is analyzed in order to obtain the search keywords. 2. The search query of the samples of data is prepared. 3. The search in ODS is run and the samples of data that best fits the search are selected.
Output: data sample

C. DP Module

The objective of this process is to optimize the sample of data, seeking to transform the data into a more optimal representation for DL-based models since these models work with binary (digital) or continuous (analog) data, which typically range from [0 to 1] or [-1 to 1]. Also, it can carry out a feature engineering process [11]. Specifically, attributes with numeric data and high variance are normalized, and attributes with textual or numeric data representing a specific finite set of categories or classes are processed as Categorical Data. Table 2 shows the DP macro-algorithm, this process begins by analyzing the data to determine the processes that will be required for each column of the sample of data (Step 1). In the case of columns with numeric data with many different values, it proceeds to normalize them (Step 2). For columns with textual or numeric data that can be represented in categories or classes, it proceeds to categorize them (Step 3).

TABLE 2: MACRO-ALGORITHM OF DPM TO OPTIMIZE THE SAMPLE OF DATA.

Input: Sample Dataset
Procedure: 1. The sample of data is analyzed. 2. The attributes with numeric data and high variance are normalized. 3. The attributes with textual and numeric data with finite values are categorized.
Output: Preprocessed sample of Data

D. SDG Module

The objective of this process is to generate the synthetic data from the sample of data optimized in the previous module. In this process, a knowledge model is built and trained that automatically extracts and learns the characteristics of the sample of data. Table 3 shows the SDG macro-algorithm, the process starts by configuring and building the knowledge model that will learn the latent characteristics in the sample of data (Step 1). Then, it proceeds to train the knowledge model using the data sample (Step 2). Finally, the synthetic dataset is generated using the previously created and trained knowledge model (Step 3).

TABLE 3: MACRO-ALGORITHM OF SDG FOR THE GENERATION OF SYNTHETIC DATA.

Input: Preprocessed Sample Dataset
Procedure: 1. The knowledge model with the desired configuration is built. 2. The knowledge model representing the sample of data is trained. 3. The synthetic dataset is generated with the knowledge model.
Output: Synthetic Dataset

1 https://pypi.org/project/fuzzywuzzy/

III. INITIAL CASE STUDY

This section presents a detailed description of a case study, which shows a functional version of the modules of the synthetic data generation system, allowing the construction of a dataset for a specific context. This case study is in the smart grid context.

A. Experimental Context

The context in which this case study is developed is focused on showing a functional version of the different modules of the proposed system for a smart grid context, allowing to automate the generation of synthetic data with minimal human intervention. In general, these processes can be deployed in any smart environment, offering an automatic synthetic data generation service to solve the need for large datasets to solve emerging problems that arise in such environments. Specifically, in the context of energy management of smart grids, it is necessary to generate synthetic datasets for different tasks.

For example, suppose that the generation of data on electricity consumption in the different provinces of Spain is required. The ODSs used as search sources to obtain the samples of data for this experiment come from the open data initiative of the Government of Spain (https://datos.gob.es/), taken from 43 State Administrations, 18 Autonomous Administrations, 229 Local Administrations and 16 Universities, among others. This source contains 56,381 datasets, distributed in 22 categories, such as: Science and Technology, Commerce, Demographics, Sports, Economy, Energy, Transport or Tourism.

B. Module 1: DSA

The responsibility of this process is to search for the sample of data to be used in the generation of the synthetic data. This process begins when the information is received from the problem context that requires a dataset. For our case study, this information would be:

- Topic: electricity, energy, smart grids
- Objective: consumption, Spain

Then, the context of the problem is analyzed, and two things are generated:

1. **Keywords:** "energy| electricity| consumption| Spain".
2. ***Query in sparql language:***

```
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
SELECT DISTINCT ?url ?title
WHERE{
        ?id a dcat:Distribution.
        ?id dct:title ?title.
        FILTER regex(lang(?title), "en", "i").
        FILTER regex(?title, Keywords, "i").
        ?id dct:format ?idtype.
        ?idtype rdfs:label "CSV".
        ?idtype rdfs:label ?type.
        ?id dcat:accessURL ?url.
} LIMIT 100
```

Once the keywords and the LD query are available, the query is executed in the public data source endpoints. For example, in our ODS would be: "http://datos.gob.es/virtuoso/sparql". Table 4 shows the result of the search in an ordered manner using the FuzzyWuzzy[1] library, where the dataset with the highest

match ratio is selected. In order to calculate this ratio, the library uses the Levenshtein Distance to compare the differences between our keywords and the description or title of the datasets.

TABLE 4: RESULT OF THE SEARCH OF SAMPLES OF DATA.

Title	Ratio
Electricity consumption in municipalities and sectors of Catalonia	79.75
Installation of electrical energy production. Individualized data	72.25
Hourly electricity demand in Catalonia per MWh	71.0
Production of electrical energy. Data added	67.25
Registration of producers of electricity	65.0

C. Module 2: DP

This process prepares and optimizes the sample of data that will be used in the next module with the DL algorithms. The first step of this process analyzes the sample of data. The datasets collect the electricity consumption of the municipalities and sectors of Catalonia. The first dataset has a total of 35,980 records. Each record is composed of 9 attributes in Catalan: i. ***Any (integer):*** year of the record. ii. ***Province (String):*** name of the province, which is an administrative demarcation of Spain. iii. ***Region (String):*** name of the territorial division delimited by physical and human geographical affinities. iv. ***Municipality (String):*** name of the municipality, which is the basic local entity of the territorial organization of the State. v. ***Cod Mun. (integer):*** id identifying the municipality to which the record belongs. vi. ***Description Sector (String):*** short description of the sector that consumes the electricity, such as, for example, consumption in the industrial or residential sector, among others. vii. ***Cod Sector (integer):*** id that identifies the sector to which the record belongs. viii. ***Consumption kWh (decimal):*** electricity consumption in kilowatt-hours (kWh). ix. ***Observations (String):*** short remark about the record.

The second step identifies the numerical attributes that can be normalized. In this case, the following are identified: "Any", "Cod Mun.", "Cod Sector" and "Consumption kWh". Then, it is verified if any of these attributes contain high variance, as they would be ideal candidates to be categorized in the next step. For example, the attribute "Any" has only 8 distinct values (from 2013 to 2020) and "Cod Sector" has 6 distinct values (1, 3-7). Therefore, only the attributes "Cod Mun." and "Consumption kWh" will be normalized with the formula: Xnorm = (X - Xmin) / (Xmax - Xmin). When normalizing, values between 0 and 1 are obtained, which are optimal values for knowledge models based on neural networks. The result of some values is shown below (Table 5):

TABLE 5: RESULT OF NORMALIZATION.

Cod Mun	Cod Mun norm	Consumption kWh	Consumption kWh norm
8001	0	3839.0	0.0000009695
17138	0.25447	66485378.0	0.0169895240
25193	0.47881	1516975100.0	0.3876446800
43002	0.97480	3154255400.0	0.8060319000
43907	1.0	3913313300.0	0.9999999400

The rest of the attributes are categorized. Table 6 shows the result of the "Cod Sector" attribute when applying One Hot Encoding, which follows the next steps: i) for each value in "Cod Sector", an attribute is created with the name and associated value, e.g. "Cod Sector_1", "Cod Sector_3", "Cod Sector_4", etc.; ii) for each "Cod Sector" record, 1 is assigned to the attribute that the value belongs to and 0 to the rest, e.g. if "Cod Sector" is 1 in a record, then "Cod Sector_1" is 1, "Cod Sector_3" is 0, "Cod Sector_4" is 0, etc., iii) finally, the "Cod Sector" attribute is deleted, since it was replaced by the other attributes. This process is repeated with all the attributes to be categorized, obtaining a dataset with 1019 attributes.

TABLE 6: RESULT OF THE CATEGORIZATION OF "COD SECTOR".

Cod Sector	*Cod Sector_1*	*Cod Sector_3*	*Cod Sector_4*	*Cod Sector_5*	*Cod Sector_6*	*Cod Sector_7*
1	1	0	0	0	0	0
3	0	1	0	0	0	0
4	0	0	1	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	1	0
7	0	0	0	0	0	1

D. Module 3: SDG

The responsibility of this process is to generate the synthetic data with the characteristics of the data sample. In this module, it uses Variational AutoEncoders (VAE), implemented as a Python object with the Keras[2] and Tensorflow[3] libraries. The VAE object allows setting the following parameters: i) ***original_dim:*** number of input neurons or dimension of the input data, ii) ***intermediate_dim:*** number of neurons in the intermediate hidden layer, has a default value: 256. iii) ***latent_dim:*** number of neurons in the latent space, default value: 100. iv) ***batch_size:*** batch size, default value: 100, v) ***epochs:*** number of epochs, default value: 50, vi) ***epsilon:*** standard deviation of the tensor, default value: 0.5. With this information, we proceed to define the parameter values to configure the knowledge model that will allow generating the synthetic data. Figure 2 shows the model built with the configuration.

2 https://keras.io/

3 https://www.tensorflow.org/

```
Layer (type)                 Output Shape              Param #   Connected to
==================================================================================================
input_1 (InputLayer)         [(None, 1019)]            0         []

dense (Dense)                (None, 300)               306000    ['input_1[0][0]']

dense_1 (Dense)              (None, 10)                3010      ['dense[0][0]']

dense_2 (Dense)              (None, 10)                3010      ['dense[0][0]']

lambda (Lambda)              (None, 10)                0         ['dense_1[0][0]',
                                                                  'dense_2[0][0]']

dense_3 (Dense)              (None, 300)               3300      ['lambda[0][0]']

dense_4 (Dense)              (None, 1019)              306719    ['dense_3[0][0]']

==================================================================================================
Total params: 622,039
Trainable params: 622,039
Non-trainable params: 0
__________________________________________________________________________________________________
```

FIG. 2: CONSTRUCTED STRUCTURE OF THE VAE MODEL

Then, the model is trained and the dataset with *N* records is generated using the trained model.

Figure 3 shows partially the generated dataset, specifically four records where each column is a different variable, using the learned generation model.

```
print(model.data_generate(4))

[[0.49952593 0.49189085 0.50488687 ... 0.48032427 0.5206242  0.5003404 ]
 [0.48942506 0.47716582 0.5042785  ... 0.48819485 0.5360881  0.49854395]
 [0.5018928  0.4956892  0.5165934  ... 0.47339827 0.5332263  0.49898785]
 [0.5035304  0.49954575 0.4980771  ... 0.46986866 0.5088126  0.50419647]]
```

FIG. 3: SYNTHETIC DATA GENERATION

IV. EXPERIMENTATION IN THE CONTEXT OF SMART ENERGY MANAGEMENT

A. *A general validation*

In this section, the objective is to test various problems in the energy field that require the generation of synthetic data for the subsequent construction of knowledge models using ML techniques. In particular, in this section, we are going to create synthetic data for the following group of case studies:

i) Energy consumption behavior of devices and lights in a home,

ii) Generation of renewable energy (specifically, solar) and,

iii) Consumption by sector (residential, industrial, academic, government, etc.).

In particular, our approach was used for each case study as follows:

1. The datasets of interest were retrieved using the LD paradigm (DSA module).
2. The dataset with a greater similarity radius was chosen and preprocessed (DP module).
3. Synthetic data was generated using VAC (SDG module).

In the first module, the datasets that were selected have a similarity ratio value greater than 80% with respect to what is sought, as indicated in Table 7. In this sense, they are datasets with very relevant information for each case study.

TABLE 7: SELECTED DATASETS.

Case	Datasets	Ratio
i	Non-controllable energy consumption in a home: https://data.world/ewood/home-energy-consumption	85.32
ii	Solar power generation information: https://www.neuraldesigner.com/learning/examples/solar-power-generation	82.52
iii	Energy consumption by sector: https://www.eia.gov/electricity/	91.10

Now, to determine the quality of the data generated by the VAC, we compare the mean and variance of the synthetic data with that of the normalized clean data that comes out of the second module. An analysis of variance (ANOVA) is carried out with the aim of testing the existence of significant differences between the mean and variance of the synthetic data with that of the normalized clean data for each variable. In Table 8 is shown an example of the means and variances of some of the variables of the synthetic data and normalized clean data for the first case study.

TABLE 8. MEANS AND VARIANCES OF THE SYNTHETIC DATA AND NORMALIZED CLEAN DATA FOR THE FIRST CASE STUDY

	Synthetic data			Original data		
Metrics	Microw	Guest bath light	Fridge	Microw.	Guest bath light	Fridge
Mean	0.122	0.080	0.931	0.114	0.077	0.936
Varian.	0.009	0.001	0.052	0.007	0.003	0.048

To analyze these differences, the hypothesis of comparison of means and variances presented in Table 9 is tested in each case study. We make the assumption of a significance level of 5%. In this test, p-values smaller than 0.05 were obtained in each case study, for the mean and variances. Thus, the results obtained allow us to reject the null hypothesis H_0 since p is smaller than the level of significance chosen, so we have a statistically significant result that allows us to reject H_0. In other words, we can verify that the differences between the data are not significant.

TABLE 9. ANOVA RESULTS FOR MEANS AND VARIANCES

Null hypothesis	P-value		
	i	ii	iii
$H_0: \mu_{generated} =$ $H_1: \mu_{generated} \neq$	0.0436	0.0248	0.0491

$H_o: \sigma_{generated} >$	0.0341	0.0405	0.0211
$H_1: \sigma_{generated} =$			

According to the results, our system allows us to search for the most appropriate data for each case study, without having to manually navigate the Internet. The LD paradigm is capable of searching for datasets that are quite similar to what is desired to be studied, in such a way as to be able to have a correct initial sample of data. With that initial sample, VAC is able to build synthetic data that fits the original data very well.

The automation of our system by adding a knowledge model generation process, would allow defining an autonomous cycle of knowledge model management to follow the real-time behavior of a smart grid [12, 15, 16].

.

V. CONCLUSIONS

This paper presented a synthetic data generation system for specific contexts using a DL algorithm and the ML paradigm. Specifically, this system has three main modules, the first one is to automatically identify and extract the sample of data. The second prepares the sample of data for the DL knowledge model and, the third creates the knowledge model to generate the synthetic data.

The test context was focused on smart grids. In particular, a first detailed analysis was carried out around the generation of data about the energy consumption of the clients. In this case study, the module of automatic detection and extraction of samples of data using LD queries determined as data sources the datasets available in the open data initiative of the Government of Spain. Likewise, the data generator module based on the VAE algorithm used the dataset with greater similarity to create a knowledge model with the latent features learned from the data sample, which was then used as a model for generating the synthetic data.

Subsequently, to show the ability of our approach to automate the data generation process for smart energy management, three other case studies from that context were considered. Overall, the results show the feasibility of our approach. The use of a system like the one we propose allows us to automate the data generation process in a field of study as complex as energy, in which the permanent generation of models (in real-time) is necessary for the different tasks of supervision, optimization and control that a smart energy management system needs to carry out.

Future works test other DL algorithms for the generation of synthetic data, such as Generative Adversarial Networks (GAN) or Adversarial Variational AutoEncoder (AVAE) approaches. In addition, they will connect this system with a process of automatic generating of knowledge models (prediction, diagnosis, identification, among others) for the context of smart energy management systems.

On the other hand, in these initial experiments, it was considered to take the dataset with maximum similarity as the data sample used to train the ML model to generate the synthetic data. Future works will consider multiple datasets to generate the data sample, which will imply performing an automatic feature engineering process that allows mixing the knowledge of experts in a given domain with an analysis of the relationships between the variables in this domain (for example, using correlations) to determine the variables to be selected in the data sample. Also, future works will analyze different similarity approaches to determine the similarity between the dataset and the case under study.

Finally, for these experiments were made general consultations. For example, for the first case study, we wanted data on energy consumption in the provinces of Spain, without specifying for what time of the year. Likewise, in the case study about solar energy production, we wanted data related to solar energy generation, without indicating region, year or time of year. Future studies should analyze how to generate more specific queries, for example, data that describe stationary behaviors, or atypical behaviors in a given year, or the opposite, that do not consider them.

ACKNOWLEDGMENT

This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska Curie Grant Agreement No. 754382. María D. R-Moreno is supported by the JCLM project SBPLY/19/180501/000024 and the Spanish Ministry of Science and Innovation project PID2019-109891RB-I00, both under the European Regional Development Fund (FEDER).

REFERENCES

[1] Augusto, J., Nakashima, H., Aghajan, H. "Ambient intelligence and smart environments: A state of the art". In Handbook of Ambient Intelligence and Smart Environments", pp 3–31, 2010.

[2] Kabalci, Y., Kabalci, E., Padmanaban, S., Holm-Nielsen, J. B., & Blaabjerg, F. "Internet of things applications as energy internet in smart grids and smart environments". Electronics, vol 8. no. 9, 2019.

[3] Dos Santos, R., Aguilar, J., & Puerto, E. "A Meta-Learning Architecture based on Linked Data". In Proc. XLVII Latin American Computing Conference (CLEI), 2021.

[4] Flach, P. "Machine learning: the art and science of algorithms that make sense of data". Cambridge University Press, 2012..

[5] Hao, X., Zhang, G., & Ma, S. "Deep learning. International Journal of Semantic Computing", Vol. 10, no. 3, pp 417-439, 2016.

[6] Islam, Z., Abdel-Aty, M., Cai, Q., & Yuan, J. "Crash data augmentation using variational autoencoder". Accident Analysis & Prevention, vol. 151, 105950. doi:10.1016/j.aap.2020.105950

[7] Dos Santos, R., Aguilar, J., Rodríguez, T. (2019). Una Revisión de la Literatura sobre Datos Enlazados. Revista Ingeniería al Día, 5 (1), pp. 54–82, 2021.

[8] Aguilar J., Garces-Jimenez A., R-Moreno M, García R, "A systematic literature review on the use of artificial intelligence in energy self-management in smart buildings", Renewable and Sustainable Energy Reviews, 151, 2021

[9] Chen, X., Xu, J., Zhou, R., Chen, W., Fang, J., & Liu, C. "Trajvae: A variational autoencoder model for trajectory generation". Neurocomputing, vol. 428, pp. 332-339, 2021.

[10] Salim Jr, A. "Synthetic patient generation: A deep learning approach using variational autoencoders". arXiv preprint arXiv:1808.06444, 2018.

[11] Pacheco F., Rangel C., Aguilar J., Cerrada M., Altamiranda J., "Methodological framework for data processing based on the Data Science paradigm," In Proc. Proc. XL Latin American Computing Conference (CLEI), 2014.

[12] Vizcarrondo J., Aguilar J., Exposito E., Subias A., "MAPE-K as a service-oriented architecture," IEEE Latin America Transactions, vol. 15 (6), pp. 1163-1175, 2017.

[13] Wan, Z., Zhang, Y., & He, H. "Variational autoencoder based synthetic data generation for imbalanced learning". In Proc. IEEE symposium series on computational intelligence. 2017.

[14] A. Plageras, K. Psannis, C. Stergiou, H. Wang, B. Gupta, "Efficient IoT-based sensor BIG Data collection–processing and analysis in smart buildings", Future Generation Computer Systems, vol. 82, pp. 349-357, 2018.

[15] L. Morales, C., Ouedraogo, J. Aguilar, C. Chassot, S. Medjiah, K. Drira, "Experimental comparison of the diagnostic capabilities of

classification and clustering algorithms for the QoS management in an autonomic IoT platform". SOCA vol. 13, pp. 199–219, 2019.

[16] Aguilar J., "A general ant colony model to solve combinatorial optimization problems", Rev. colomb. comput., vol. 2, n.º 1, pp. 7-18. 2001.

[17] Aguilar, J., Bessembel, I., Cerrada, M., Hidrobo, F., Narciso, F. "Una Metodología para el Modelado de Sistemas de Ingeniería Orientado a Agentes", Revista Iberoamericana de Inteligencia Artificial, vol. 12, pp. 39-60, 2008.

[18] Aguilar, J., Cerrada, M., Hidrobo, F. "A Methodology to Specify Multiagent Systems". Lecture Notes in Computer Science, vol 4496, pp 92–101, 2007.

[19] K. Zhou, C. Fu, S. Yang, "Big data driven smart energy management: From big data to big insights", Renewable and Sustainable Energy Reviews, vol. 56, pp. 215-225, 2016.

[20] C. Zhang, S. R. Kuppannagari, R. Kannan and V. K. Prasanna, "Generative Adversarial Network for Synthetic Time Series Data Generation in Smart Grids,"In Proc. IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), pp. 1-6, 2018.

[21] S. E. Kababji and P. Srikantha, "A Data-Driven Approach for Generating Synthetic Load Patterns and Usage Habits," IEEE Transactions on Smart Grid, vol. 11, no. 6, pp. 4984-4995, 2020.

[22] V. Krishnan et al., "Validation of Synthetic U.S. Electric Power Distribution System Data Sets," IEEE Transactions on Smart Grid, vol. 11, no. 5, pp. 4477-4489, 2020.

[23] P. Wiest, D. Contreras, D. Gross and K. Rudion, "Synthetic Load Profiles of Various Customer Types for Smart Grid Simulations," IN Proc. Conference on Sustainable Energy Supply and Energy Storage Systems, 2018.

[24] S. Asre, A. Anwa. Synthetic Energy Data Generation Using Time Variant Generative Adversarial Network". *Electronics*, vol. 11, no. 3, pp. 355, 2022.

[25] "Pecan street dataset," http://www.pecanstreet.org/category/dataport/, 2018.

8.12 Anexo 5.E: A synthetic data generation system based on the variational-autoencoder technique and the linked data paradigm

A Synthetic Data Generation System based on the Variational-Autoencoder Technique and the Linked Data Paradigm

Ricardo Dos Santos[1], Jose Aguilar[1,2,3]

[1]CEMISID, Universidad de Los Andes, Mérida, Venezuela
[2]GIDITIC, Universidad EAFIT, Medellin, Colombia
[3]IMDEA Networks Institute, Madrid, Spain

Abstract: Currently, the generation of synthetic data has become very fashionable, either due to the need to create data in certain specific contexts or to study unknown scenarios among other reasons. Additionally, synthetic data is a critical component in training machine learning models in the presence of little data. This work proposes a Synthetic Data Generation System (SDGS) architecture to allow synthetic data generation to be fully automated. SDGS is based on the Variational AutoEncoders (VAE) learning technique, and has three main capabilities. The first is related to the ability to extract data samples from multiple sources using the Linked Data (LD) paradigm. The second is linked to the ability to merge data sets to increase the amount of information that can be provided to the VAE-based synthetic data generator. The last one is related to having a Feature Engineering layer to create new features by generating or extracting information from the dataset and then selecting the features that provide the best information for the VAE model. A case study is described in detail to show the new functionalities of the SDGS, such as dataset extraction from different sources using LD, dataset merging using pivots, and the application of different feature engineering methods. Finally, two metrics are used to evaluate the quality of the generated datasets in different case studies. The first one is the accuracy to analyze the performance of the models generated with the new SDGS functionalities, obtaining results above 90%. The second one is the two-Sample Hotelling's T-Squared Test to determine the quality of the synthetic data generated by the system, obtaining synthetic datasets very similar to the original datasets.

Keywords: Synthetic Data Generator, Linked Data, Variational Autoencoders,

I. INTRODUCCIÓN

A great current technological challenge is being able to take advantage of multiple datasets to generate synthetic data in specific contexts. This requires identification processes of said data sources, for which the LD paradigm can be used, as well as subsequently merging these multiple datasets into a single sample dataset for each given context [22], [23], [24]. Now, this set of data samples obtained from multiple sources requires a process of analysis of its characteristics (feature engineering) that allows optimizing the quality of the data, before going to a synthetic data generation process [19], [25], [26].

In this work, an architecture is proposed, called SDGS, which allows the generation of synthetic data considering those three aspects mentioned above, which uses the VAE learning technique to generate synthetic data. Specifically, SDGS uses the LD paradigm to identify data sources (used as an identifier of datasets for a specific context) [1]. Subsequently, it performs the fusion of the different identified datasets and a feature engineering process to identify the relevant variables. In this way, it builds a dataset for a given context. Finally, SDGS uses a VAE-based model to generate new data similar to the newly constructed dataset. In this way, it creates larger data sets from samples of small data sets [22], [23].

A. Relate works

Some similar research on dataset extraction and fusion with LD are as follows: Avazpour et al. [2] described an architecture that incorporates complex data aggregation from multiple sources, mapping and data transformation. The architecture is made up of a set of multiple components: i. The dataset components collect all the datasets transformed into CSV that come from different sources (CSV, RDF, API, among others); ii. The aggregator components allow extracting partial information from each specific CSV, III. The mapping components allow the imported data to be assigned to generate the data model of the architecture. In the work [3], the authors defined a keyword search system on federated RDF datasets. The process begins when the Mediator component receives the set of keywords specified by the user. The Mediator component uses the storage component to find the data and metadata that match the keywords. The Federated Schema Component is then used to find outer joins between the subqueries computed by each dataset. Finally, the Mediator executes the federated SPARQL query and returns the dataset with the composition of the data requested by the user from the different sources of the federated RDF datasets.

On the other hand, Rao et al. [4] presented a method for fusing linked open data from multiple sources for querying relationships between drugs and genetic disorders. Generally, the information about genes, drugs and disorders was stored in different places and in different formats such as RDF/XML, SQL and relational, among others. In this method, biomedical datasets are converted into RDF triples, normalizing the vocabulary and data URIs, and then stored as merged data. After merging, the system can be queried with SPARQL to understand the relationships between multiple entities from different datasets and extract datasets for a more specific context. Similarly, in [5], Chen proposed an LD fusion method based on the calculation of similarity and k-nearest neighbor (KNN), allowing to solve problems of entity conflicts between data sources. The LD similarity calculation effectively integrates URI nodes and blank nodes into linked data. On the other hand, the fusion of literal type nodes based on the KNN classification allows automating the fusion independently of the data sources. The KNN classifier generates a model that learns from common assignment strategies for resolving conflicts between literal nodes.

In the context of VAE, there is some research that applies feature engineering. For example, Nishimaki et al. [6] described the Localized Variational-Autoencoder (Loc-VAE) that provides neuroanatomically interpretable low-dimensional representation from 3D brain MR images. In this research, they take advantage of VAE models for feature extraction (using the latent vector Z of an image where the vector represents the extracted features). Then, they generate perturbations to each element of the latent vector and pass through the decoder, obtaining new images with the result of the perturbation. On the other hand, a feature selection method, called AVAE (Attention of Variational Autoencoder) is proposed by Van Dao et al. [7], which emphasizes the importance of the attention mechanism in the selection and evaluation of weights in the latent space. AVAE consists of CBAM (Convolutional Block Attention Module) coding layers, and can learn what and where to emphasize or suppress. CBAM is a lightweight CNN with only two convolutional layers: Channel Attention Module (CAM) and Spatial Attention Module (SAM).

In [8], Hadipour et al. developed a molecular embedding learning approach that combines PCA (Principal Component Analysis) and VAE to integrate global and local molecular features. The approach begins by collecting the molecule data, extracting the global (molecular descriptors) and local (atomic and bond) features for each molecule. The PCA method is used to reduce the atomic feature matrix and the bond feature matrix to a PCA-based feature vector, respectively. Then, it concatenates the global and local features and it filters out the columns with zero variance. Finally, VAE is used to incorporate the global

chemical properties and the local atom and bond features. Also, Akkem et al. [21] proposed the use of VAE and Generative Adversarial Networks (GAN) to generate synthetic data for crop recommendation. This research focuses on exploring the effectiveness of VAE and GAN in producing high-quality synthetic data, facilitating improved training and evaluation of recommender systems. In addition, they performed extensive qualitative analysis on the reliability of synthetic data in various experiments, including visual comparisons such as heatmaps, scatter plots, cumulative sum per feature plots, and distribution per feature plots. The work of Marco et al. [27] explored the use of conditional variational autoencoder (CVAE) and inverse normalization transformation for data augmentation and synthetic data generation in software engineering. Eleven datasets from sources like PROMISE and ISBSG were used. The CVAE-INT model created synthetic data, showing significant results with a mean p-value above 0.90 in the Mann-Whitney test. The model achieved lower mean absolute error and root mean squared error across various datasets.

Panfilo et al. [28] introduced a data synthesis technique for both supervised and unsupervised learning on single-table and relational datasets. Utilizing generative deep learning models, the technique includes three variants: standard VAE, β-VAEs, and Introspective VAEs. The effectiveness of these variants is experimentally evaluated to determine how well they meet the quality requirements for generated data. Various performance indexes are used to capture different aspects of data quality, demonstrating the applicability of these models to relevant business cases. Kuo et al. [29] proposed enhancing the classic GAN framework with a VAE and an external memory mechanism to generate synthetic datasets that accurately reflect imbalanced class distributions in clinical variables. The method was tested on data related to antiretroviral therapy for HIV. Results show the method effectively prevents mode collapse, ensures low patient disclosure risk (0.095%), and maintains high utility for machine learning applications in healthcare. Eigenschink et al. [30] introduced a data-driven evaluation framework for generative models that produce synthetic sequential data. The framework assesses models based on five criteria: representativeness, novelty, realism, diversity, and coherence, independent of the models' internal structures. These criteria address various domain-specific requirements, allowing users to evaluate synthetic data quality across models. A review of generative models for sequential data shows that realism and coherence are crucial for natural language, speech, and audio processing, while novelty and representativeness are vital for healthcare and mobility data. Representativeness is typically measured using statistical metrics, realism through human judgment, and novelty with privacy tests. Finally, an initial SDGS has been proposed in [1], which only carries out an extraction of data from only one source to generate new data. A summary of the literature review with the articles closest to ours is presented in Table 1, showing the advantages and limitations with respect to our approach.

TABLE 1: LITERATURE REVIEW RELATED TO OUR RESEARCH

Work	Advantages	Limitations
[1]	Propose a synthetic data generator that uses the	Data extraction is performed from a single source and does not

	advantages of the LD paradigm for automatic detection and extraction of data samples for smart grids, and the learning capability of VAE algorithms.	perform a feature engineering process to improve the generation of synthetic data.
[2]	It describes an architecture that incorporates complex data aggregation from multiple sources.	All sources must be transformed to CSV before being used by the architecture, and it does not have the ability to extract datasets scattered on the web as Linked Open Data.
[3]	It presents a system for searching federated RDF datasets using SPARQL queries.	It needs to manually create UNION clauses to combine the results of queries
[4]	It presents a method of merging linked open data from multiple sources converting them into RDF triples.	It is quite difficult to build datasets with this granularity and still maintain the relationship between the features of different datasets.
[5]	It proposes a fusion method based on similarity calculation and KNN, which allows solving problems of entity conflicts between data sources.	The difficulty in queries remains due to the granularity of the information.
[6]	VAE models are used to extract features from brain images, observing the effect generated by perturbations to the latent Z vector.	The data generated by the VAE model are used to find the explainability of the extracted features.
[7]	It proposes a feature selection method that helps to acquire important features in the latent space of the VAE model.	It emphasizes the importance of the weight selection and evaluation mechanism for the VAE.
[8]	VAE is used to incorporate the global chemical properties and the local atom and bond features.	VAE is not used to generate synthetic data but as a feature fusion component.
[21]	It proposes the use of VAE and GAN to generate synthetic data for crop recommendation.	It's not defined methods that allow the datasets to be obtained automatically, no feature engineering has been applied to improve the original data.

The main difference between our approach with previous works is that we define an entire architecture for the generation of synthetic data, where aspects such as the acquisition, preparation and optimization of the dataset are covered to improve the generation of synthetic data.

B. Contributions

The main difference between the works described above with this research is that our work, before generating data, seeks to build a robust dataset with the extraction and fusion of all data from different sources. In our case, the datasets already constructed in the study area are identified using the LD paradigm. The selected datasets are then merged based on the relationships that exist between them. Subsequently, a Feature Engineering process is carried out to build a robust dataset before passing it on to the VAE, this allows us to offer new and more information from the data. Finally, our VAE is used to generate synthetic data from the created dataset. In general, the main contributions of this research are:

- It defines an SDGS architecture for the automatic generation of synthetic data.
- It specifies an automatic mechanism for dataset identification from multi-sources based on the LD paradigm.
- It specifies an automatic mechanism for the creation of a robust dataset based on multi-datasets.
- It defines a Feature Engineering module that merges, extracts and selects features.

This article is organized as follows: Section 2 shows the theoretical framework around SDGS; Section 3 describes the SDGS components for the multi-sources and multi-datasets management processes, and the feature engineering module; Section 4 describes a case study where the SDGS is tested; Section 5 presents the experiments with the SDGS in different case studies and a comparison with other architectures. Finally, Section 6 presents the conclusions of this research.

II. SDGS

The SDGS architecture generates synthetic data using the LD paradigm to identify and extract data from the Internet and the VAE technique to train a model with this sample so that this model can later be used to generate synthetic data. This architecture is composed of the following modules (see Fig. 1) [1]:

- **DataSet Acquisition (DSA):** The objective of this module is to find data samples through LD-based search mechanisms, taking advantage of Open Data Sources (ODS) or endpoints.
- **Data Preparation (DP):** The objective of this module is to optimize the data sample. It normalizes numerical attributes with high variance, and processes textual or numerical data representing a specific finite set of categories or classes.
- **Synthetic Data Generation (SDG):** The objective of this module is to generate synthetic data by training a VAE-based knowledge model, which automatically extracts and learns the features of the optimized data sample for a given context.

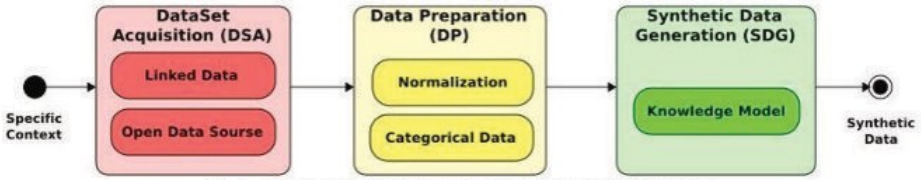

Fig. 1: SYNTHETIC DATA GENERATION ARCHITECTURE [1].

In this work, SDGS is extended with new components in order to implement its Synthetic Data Generation functionality.

III. EXTENSIONS TO THE SDGS

A. Architecture

This research extends the approach proposed in [1] (see Fig. 2). Firstly, by adding two processes to the DSA module, a first process focused on the use of multiple data sources and a second process focused on the fusion of multiple datasets. In addition, a new module, called Feature Engineering (FE), is added to analyze the characteristics of the sample datasets to be used by the SDG, allowing Feature Fusion, Feature Extraction and Feature Selection. Finally, the SDG is implemented using the VAE technique as the data generator.

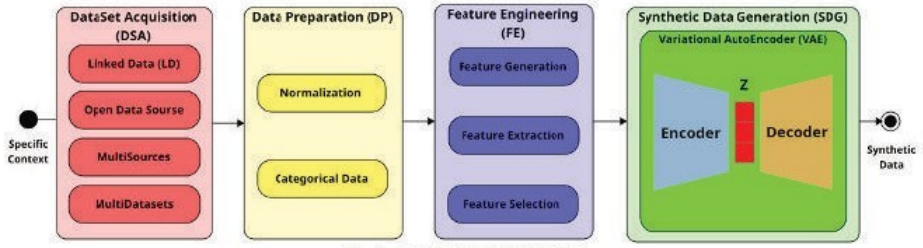

Fig 2: EXPANSION OF SDGS

B. DSA Module

The objective of this module is to find data samples for a given context using the LD paradigm. Table 2 shows the macro-algorithm of the DSA module. This module begins by analyzing the context of the required data samples, obtaining the keywords to search the data samples (Step 1). Then, the Multi-sources process is invoked to search the datasets using the keywords obtained in the previous step (Step 2). Finally, if it is decided to merge the obtained datasets with another dataset from an associated context (Step 3), then the multiple dataset process is invoked (Step 3.1).

TABLE 2: MACRO-ALGORITHM OF THE DSA MODULE.

Input: Specific Context
Procedure: 1. The context of the required data is analyzed in order to obtain the search keywords. 2. Invoke the Multi-sources process to find a Data Sample using the search keywords 3. If merging the Data Sample: 3.1 Invoke the Multi-dataset process to merge the Data Sample with the dataset from the associated context.
Output: New Dataset

B.1. Multi-sources Process in DSA

The objective of this process is to find data samples from different dataset sources. Specifically, the data samples are obtained using search mechanisms based on LD, taking advantage of Open Data Sources (ODS), searching each dataset source registered in the SDGS and selecting sample datasets that best match the specific context required. These dataset sources are officially provided by countries/regions such as Europe (https://data.europa.eu), Spain (https://datos.gob.es/), Canada (https://open.canada.ca), United States (https://www.data.gov/), among others. The metadata published by these sources follows the CKAN standard (https://ckan.org/), which allows queries using the SPARQL language. Table 3 shows the macro-algorithm of the Multi-sources Process in DSA. This process begins by preparing the search queries for each dataset source (Step 1). Then, each query is executed in each dataset source, and the list of possible datasets is obtained (Step 2). Finally, the list is ordered according to the degree of coincidence with the required context (Step 3).

TABLE 3: MACRO-ALGORITHM OF MULTISOURCES PROCESS TO SEARCH SAMPLES OF DATA.

Input: Search Keywords

Procedure: 1. Prepare the search queries with the search keywords for each dataset source based on the LD paradigm. 2. The search is executed for each dataset source and added to the list of possible data samples. 3. The data samples that best fit the search are sorted and selected.
Output: Data Sample

B.2. Multi-datasets Process in DSA

The objective of this process is to build a data sample that combines information from different datasets. Specifically, having a dataset from the main context required, another dataset belonging to a context associated with the main context is searched. Then, it searches for relations between the features of the datasets; the matches are used as pivots for the merging of both datasets. Table 4 shows the macro-algorithm of the Multi-datasets process in DSA. This process begins by invoking the Multi-sources process to find a Data Sample from the Associated Context (Step 1). Step 2 searches for the similarities of both Data Samples; this similarity is based on the name and type of each feature of the Main Data Sample and the Associated Context Data Sample. Finally, it merges the Main Data Sample and the Associated Context Data Sample using the similarities as a pivot (Step 3), generating a Merged Data Sample.

TABLE 4: MACRO-ALGORITHM OF MULTIDATASETS PROCESS TO FUSION SAMPLES OF DATA.

Input: Main Data Sample, Associated Context Search Keywords
Procedure: 1. Invoke the Multi-sources process to find a Data Sample using Associated Context Search Keywords. 2. Search for possible similarities between the characteristics of the Main Data Sample and the Associated Context Data Sample. 3. Merge both Data Samples using the similarities as a pivot.
Output: Merged Data Sample

C. DP Module

The objective of this module is to transform the sample dataset into an optimal representation for the VAE model, knowing that this type of model works optimally with data ranging between [0 and 1] or [-1 and 1], either binary (digital) or continuous (analog) data. Some of the tasks this module does is convert textual or numerical data into a specific finite set of categories (categorical data), or normalize numerical data with high variance, among other things. Table 5 shows the DP macro-algorithm, which begins by analyzing the dataset to determine the processes that will be required for each column of the data sample (Step 1). For columns with numeric data with many different values, it proceeds to normalize them (Step 2). For columns with textual or numeric data that can be represented in categories, it proceeds to categorize them (Step 3).

TABLE 5: MACRO-ALGORITHM OF THE DP MODULE TO OPTIMIZE THE SAMPLE OF DATA.

Input: Sample Dataset

Procedure:
1. The sample of data is analyzed.
2. The attributes with numeric data and high variance are normalized.
3. The attributes with textual and numeric data with finite values are categorized.
Output: Preprocessed Data Sample

D. FE Module

The objective of this module is to analyze the characteristics of the sample dataset. It specifically analyses the Preprocessed Data Sample generated by the DP module, obtaining new information from the dataset features and selecting the features that offer more information for the SDG module. Table 6 shows the macro-algorithm of the FE module. This process begins with the analysis of the dataset (Step 1). Then, in step 1.1, information is aggregated by applying Feature Generation techniques such as Interaction Feature, Polynomial Feature, Trigonometry Feature, Create Clusters, or Combine Rare Levels. In Step 1.2, information is added by applying Feature Extraction techniques such as Media-Based Feature, Median-based Feature and Quartiles-Based Feature. Finally, in step 1.3, it selects the features that offer the most information, applying Feature Selection techniques such as Permutation Feature Importance, Remove Multicollinearity, Ignore Low Variance, and Genetic Algorithm. In the next section, we will explain in detail the Feature Generation, Feature Extraction and Feature Selection techniques used in the case study to evaluate the behavior of our SDGS.

TABLE 6: MACRO-ALGORITHM OF FE TO ENHANCE THE SAMPLE OF DATA.

Input: Preprocessed Data Sample
Procedure:
1. Analyses the Preprocessed Data Sample:
1.1. Add new information generated from its characteristics.
1.2. Add new information extracted from its characteristics.
1.3. Select the features that provide the most information.
Output: Enhanced Data Sample

E. SDG Module

The objective of this module is to generate the synthetic data from the data sample optimized in the previous module. In this process, a knowledge model is built and trained that automatically extracts and learns the characteristics of the sample of data using VAE. This knowledge model is then used to generate the synthetic data. Table 7 shows the SDG macro-algorithm, the process begins by configuring and building the knowledge model that will learn the latent characteristics in the sample of data (Step 1). Step 2 consists of training the knowledge model using VAE and the data sample. Finally, the synthetic dataset is generated using the previously created and trained knowledge model (Step 3).

TABLE 7: MACRO-ALGORITHM OF SDG FOR THE GENERATION OF SYNTHETIC DATA.

Input: Enhanced Data Sample
Procedure:
1. The knowledge model with the desired configuration is built.
2. The knowledge model representing the sample of data is trained.
3. The synthetic dataset is generated with the knowledge model.
Output: Synthetic Dataset

IV. DESCRIPTION OF SDGS IN A CASE STUDY

This section presents a case study, which shows the new capabilities of the architecture.

A. Experimental Context

The objective of this case study is to show a functional version of the modules of the proposed system for a smart grid context, allowing the automated extraction of a data sample and the optimization of the information provided by the dataset. In general, this architecture can be deployed in any context for automatic synthetic data generation. Specifically, the modules will be deployed to search for datasets in the context of Energy Management, with an associated context on Territorial Indicators that will allow us to show the fusion of datasets. Additionally, in order to show the ability to identify and extract data from multiple sources, it will use three sources of datasets:

- Canada Open Government (https://open.canada.ca): it offers more than 40,000 datasets with information from Canadian provinces, territories and municipalities on government activities such as health, science and technology, economy and industry, and education, among many others.
- European Open Data (https://data.europa.eu): it offers more than 1.5 million datasets about 33 European countries with information on energy, agriculture, transport, regions and cities, etc., and
- Open Data of the Government of Spain (http://datos.gob.es): it offers more than 66 thousand datasets with information on demography, urban planning and infrastructure, rural environment, and employment, among others.

On the other hand, in this case study, different techniques were chosen for each type of feature engineering that the system has, with the intention of showing how the FE module works. Each one of the selected techniques is described below:

Regarding the Feature Generation technique, the following technique was used:

- Trigonometric features generate new features by applying the trigonometric functions tangent, sine and cosine. This process generates three features for each numerical feature in the dataset.

Also, the next Feature Extraction techniques were used [11], [13]:

- Media-Based Feature, it extracts the mean of the feature and constructs a new feature with the distance of each value of the feature with respect to its mean. This process is repeated for each numerical feature in the dataset.
- Median-Based Feature, it extracts the median of the feature and constructs a new feature with the distance of each value of the feature related to its median. This process is repeated for each numerical feature in the dataset.
- Quartiles-Based Feature extracts the quartiles of the feature (25%, 50%, 75%, 100%), where the first quartile is all values that are less than 25% of the data for that feature, the second quartile is all values greater than 25% and less than 50% of the data, the third quartile is all values greater than 50% and less than 75%, the fourth quartile is all values greater than 75% and less than 100%. With this information, it builds a new feature that indicates the quartile of each value of that feature. This process is repeated for each numerical feature in the dataset.

Finally, the next Feature Selection technique was used [13], [14]:

- Genetic Algorithm, it selects the features that provide the most information using the fitness individual's chromosome. A chromosome of an individual is represented by all the features in the dataset and it is coded in binary, 1 if present and 0 if not present in that individual. To calculate the quality of an individual, a classification model is used as the fitness function. Particularly, to build

the classification model, the Logistic Regression technique was used. The fitness of an individual is obtained based on the quality of the results of the classifier using the dataset only with the features presented in this individual. This dataset is divided into two parts, the first part is used to train the classifier and the other part is used to measure its performance. In addition, in each generation, it creates a new population, which will serve to obtain the parents for the new individuals. At this point, it applies two genetic operations: crossovers and mutations. In crossover, it copies all the features from one parent up to the crossover point, and the rest of the features are copied from a second parent, the default probability for crossover is 60%. In mutation, it selects the features to which mutation will be applied, based on a probability; the default value is 10%. At the end of this process, it calculates the fitness of the population of individuals. All these processes are repeated until the maximum number of generations is reached, the default values are 50 generations. In the last generation, it selects the individual fittest. This individual indicates which features provide the most important information.

B. Multi-sources Process

The responsibility of this process is to search for a data sample from the various data sources registered in the system. This process begins when it receives the keywords of the problem context that requires a dataset. For our case study, this information would be:

Keywords: "energy| electricity| consumption| Spain".
Endpoints registered in the system:
* EU: https://data.europa.eu/sparql
* ES: http://datos.gob.es/virtuoso/sparql
* CA: https://open.canada.ca/sparql

It prepares queries for each endpoint. For each endpoint, the queries are similar and are specified in SPARQL language:

```
PREFIX dct: <http://purl.org/dc/terms/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
SELECT DISTINCT ?id ?idtype ?type ?url ?title
WHERE{
        ?id a dcat:Distribution.
        ?id dct:title ?title.
        FILTER regex(lang(?title), "en", "i").
        FILTER regex(?title, Keywords, "i").
        ?id dct:format ?idtype.
        ?idtype rdfs:label ?type.
        ?id dcat:accessURL ?url.
} LIMIT 100
```

Once the keywords and queries are available, they are executed on their respective endpoints. The information obtained is structured with the following data: id, idtype, type, url, title, endpoint, and similarity ratio. To calculate this ratio, we use the Levenshtein distance metric to compare the differences between our keywords and the description or title of the datasets [12]. Table 8 shows the search results and similarity ratios. Finally, the selected dataset is the one with the highest similarity ratio.

TABLE 8: RESULT OF THE SEARCH OF SAMPLES OF DATA.

Title	Endpoint	Ratio

Electricity consumption in municipalities and sectors of Catalonia	EU	79.75
Electricity consumption in municipalities and sectors of Catalonia	ES	79.75
Weekly electricity consumption in Castilla y León	EU	74.6
Hourly electricity consumption in the hospitals of Castile-Leon	EU	73.6
Electricity generation by type of energy	EU	72.8
Electricity Generation	CA	72.4
Installation of electrical energy production. Individualized data	ES	72.25
Electricity Interchange 2016 Update	CA	71.8
net-zero-electricity-provincial-generation-2021	CA	71.4
Hourly electricity demand in Catalonia per MWh	ES	71.0

C. Multi-Datasets Process

The responsibility of this process is to merge datasets to generate a better sample dataset. This process begins when it receives the main dataset selected in the previous phase, and the keywords of the associated context that requires a merged dataset. For our case study, this information would be:

Keywords: "Territorial| Indicators| Spain".
Dataset: MainDataSample.csv
Similarity Threshold: 85%

Then, it requests a dataset for the Multi-sources process using the keywords of the context associated (see section IV.B). The AssociatedDataSample.csv file obtained will be merged with the MainDataSample.csv file.

To merge both datasets is necessary to identify the columns that relate them. To do that, the first step is to extract and verify the similarity of the titles and data types of the columns (variables) between both datasets. The similarity ratio of the titles is calculated and those that exceed the Similarity Threshold are selected as possible pivot columns. Then, it verifies if the possible pivot columns are of the same data type, and if they match, it selects them as pivots. Table 9 shows the pivots automatically selected by the system for this case study.

TABLE 9: SELECTED PIVOTS.

N°	Main Data Sample		Associated Data Sample		Ratio
	Title	**Datatype**	**Title**	**Datatype**	
1	Any	integer	Any	integer	100
2	Comarca	string	Comarca	string	100
3	Cod Municipi	integer	Codi Municipi Ine	integer	87

Finally, it merges the datasets using the pivots as a relationship mechanism between the datasets, obtaining the FusionatedDataSample.csv file.

D. FE Module

The responsibility of this module is to generate new features by analyzing the new dataset and selecting the features that provide the most information. This process starts by generating new features. For this case, Table 10 shows the Feature Generation obtained with Trigonometric Features. E.g., tangent, sine and cosine of Consumption kWh. These new features with trigonometric behaviors provide the VAE model with additional information at the time of the construction of the knowledge model.

TABLE 10: GENERATION OF NEW VARIABLES USING TRIGONOMETRY FEATURES

Consumption kWh	Trigonometric Feature		
	tan(Consumption kWh)	sin(Consumption kWh)	cos(Consumption kWh)
0.0000009695	0.0000009695	0.0000009695	1
0.0169895240	0.0169911588	0.0169887066	0.9998556815
0.3876446800	0.4083042838	0.3780089057	0.9258019589
0.8060319000	1.0421430222	0.7215454805	0.6923670409
0.9999999400	1.5574075191	0.8414709523	0.5403023563

Then, new features are extracted. Table 11 shows these new features, the Mean-Based Feature, the Median-Based Feature and the Quartiles-Based Feature. In this specific case, the Mean-Based Feature and Median-Based Feature calculate the distance of "Consumption kWh" with respect to its Mean (0.002746) and Median (0.000151). These two features provide the model with information on the dispersion of values in terms of the central value, either with respect to the mean and the median. The mean is usually better when the values follow a symmetric distribution and the median is better when the values are outliers, as these values distort the mean. The Quartiles-Based Feature determines the quartile to which "Consumption kWh" belongs by dividing the dataset into four equal parts; the first quartile is 25%, ranging from 0 to 0.0000341155. The second quartile is 50%, ranging from 0.0000341155 to 0.0001509117. The third quartile is 75%, ranging from 0.0001509117 to 0.0008732833. The fourth quartile is all values greater than 0.0008732833. This new feature provides information to the model on the behavior of the variables by classifying it into four groups.

TABLE 11: FEATURE EXTRACTION USING MEAN-BASED, MEDIAN-BASED AND QUARTILE-BASED FEATURES

Consumption kWh	Mean-Based Feature	Median-Based Feature	Quartiles-Based Feature
	mean(Consumption kWh)	median(Consumption kWh)	quartiles(Consumption kWh)
0.0000009695	0.0027454629	0.0001499422	1
0.0169895240	-0.0142430915	-0.0168386122	4
0.3876446800	-0.3848982475	-0.3874937682	4
0.8060319000	-0.8032854675	-0.8058809882	4
0.9999999400	-0.9972535075	-0.9998490282	4

Finally, Table 12 shows a summary of the resulting dataset, with some of the features selected using the Genetic Algorithm. At the beginning of the process, there were 864 features in the data set, and when this method was run, 422 features were selected. Furthermore, it can be observed that most of the selected features are binary, which is an advantage for the VAE model as it works very well with this type of value. Another interesting fact is that in the feature selection for this dataset, the method selected only the features that represent the year 2018 (Any_2018), since there is a lot of information around this date.

TABLE 12: PARTIAL FEATURES SELECTED USING GENETIC ALGORITHM

	Codi Municipi	Any_2018	Provincia_BARCELONA	Codi Sector_5	Codi Sector_6	DescripcionSector_CONSTRUCCIO I OBRES PUBLIQUES
0	0.0	0.0	1.0	0.0	0.0	0.0
1	0.0	0.0	1.0	0.0	0.0	0.0
2	0.0	0.0	1.0	0.0	0.0	1.0
4	0.0	0.0	1.0	0.0	1.0	0.0
5	0.0	0.0	1.0	0.0	0.0	0.0
...	....	...	...	...	...	...
35975	1.0	0.0	0.0	0.0	0.0	0.0
35976	1.0	0.0	0.0	0.0	0.0	0.0
35977	1.0	0.0	0.0	0.0	0.0	1.0
35978	1.0	0.0	0.0	0.0	1.0	0.0
35979	1.0	0.0	0.0	0.0	0.0	0.0

V. ANALYSIS OF RESULTS

This section is composed of two parts, the first part focuses on the development of experiments with our system in different case studies, and the second part focuses on comparing our work with other approaches.

A. Analysis of SDGS

The aim of this section is to analyze the performance of the SDGS in different case studies that require the generation of synthetic data. In particular, it will generate synthetic data using data samples from Energy, COVID-19, and Industry 4.0 contexts.

A.1. Context of analysis

We considered 3 contexts where an initial dataset exists. The contexts are the following:

- **Energy:** The original dataset is composed of information on electricity consumption in the different municipalities and sectors of Catalonia. In addition, it was merged with other territorial indicators such as population density, economic level, among others. This dataset was obtained thanks to the DSA Module using the LD sources that our system has (more information in III.B, IV.B and IV.C).
- **COVID-19:** The original dataset in this context is composed of information on COVID-19: confirmed cases, deaths, hospitalizations, tests, among many other variables [14]. This dataset was extracted from https://github.com/owid/covid-19-data/tree/master/public/data.
- **Industry 4.0:** This case study corresponds to a production process for the manufacturing of sandwich bread, where the client can customize the wrapper (logo, name, etc.), the quantity, the type of bread (grain bread, white bread, with sesame, with raisins, etc.), among other things. The production process involves devices like the smart conveyor belt that routes the bread to the least busy device wrapping machine to pack the bread using the correct wrapper, smart slicers that slice the bread, the smart printers, and the rest of the devices. The original dataset was composed of information on the status of each process of a production line.

In each context, 4 experiments/cases were realized,

- Two experiments without using the new FE module: One case was using a small data sample (case A) and the other case with a large data sample (case B).
- Two experiments used the new FE module: One case was using Feature Generation and Feature Extraction (case C), and the other used, additionally, Feature Selection (case D).

Regarding the metrics, we used two metrics to analyze the quality of our SDGS. The first metric was the Accuracy of the VAE model built in the different experiments (Section V.A.2). The accuracy measures the performance of the built model based on the total percentage of elements classified correctly. Specifically, the accuracy of VAE is measured by comparing each input with respect to its output, verifying how similar they are, since the learning process of this model is based on passing that input through an Encoder and Decoder, to see if the model is able to reconstruct the input. The second metric was the Two-Sample Hotelling's T-Squared Test, which allows comparing the multivariate means of different populations (datasets with multiple variables). This test allowed performing a hypothesis test to verify the existence of significant differences between two datasets. In our case, it verifies the differences between the training dataset of the VAE model and the synthetic dataset generated by the VAE model in the different experiments (Section V.A.3).

A.2. Performance of the generated VAE models

Table 13 shows the results achieved by the VAE models built with the different datasets. The accuracy obtained using the new FE module offers better results than the models built without using FE. It should also be noted that using the three types of Function Engineering the results are better, as seen in the Energy dataset with a precision of 0.953, in COVID-19 with 0.963, and in Industry 4.0 with 0.961.

TABLE 13: RESULT OF THE ACCURACY OF VAE MODELS

Dataset	Case	Sample	Features	Accuracy
Energy	A	500	95	0.903
	B	29508	95	0.912
	C	29508	864	**0.947**
	D	29508	422	**0.953**
COVID19	A	356	26	0.925
	B	10000	26	0.935
	C	10000	67	**0.957**
	D	10000	59	**0.963**
Industry 4.0	A	250	3	0.918
	B	10000	3	0.946
	C	10000	14	**0.952**
	D	10000	10	**0.961**

A.3. Quality of the Datasets generated

The Two-Sample Hotelling's T-Squared Test checks for significant differences between the dataset used in the training of the VAE model and the dataset with synthetic data generated by the VAE model. This test proposes the following hypotheses,

- Null hypothesis (H_0): the two datasets come from populations with the same features.
- Alternative hypothesis (H_1): the two datasets come from populations with different features.

In addition, the significance level was normally 5% (alpha = 0.05), knowing that the inverse distribution function of F is (1 - alpha) = 0.95. This value allowed determining when H_0 was rejected, i.e. when the probability of H_0< 0.05 or when the calculated value of F was greater than the critical value of the F distribution table when transforming Hotelling's T2 statistic into an F statistic. Rejecting H_0 indicated that at least one of the variables, or a combination of one or more variables working together were significantly different between the datasets.

Finally, we compared the two datasets (training and generated) using the Two-Samples T-Squared Test that has the Python library pingouin[1]. This library executes all the calculations associated with the test and returns the following information:

- T2: Hotelling's T-squared value.
- df1: First degree of freedom.
- df2: Second degree of freedom.
- F: F-distribution value with df1 and df2[2].
- P-value: Probability of the null hypothesis (H_0).

Table 14 shows the results obtained in the different experiments. In this case, we are going to detail the test analysis process for the first experiment since the other experiments are similar. The value 110.02386 in T2 was the result obtained in the different matrix operations of the test with both datasets. This value when is transformed the Hotelling T2 statistic into an F statistic, we obtained the value 1.04906202. Having the F value, we had two methods to determine if there were significant differences between the datasets:

- The first method required finding the critical value in the F distribution table with the degrees of freedom df1 = 95 and df2 = 904 and the default value of alpha 0.05. In this case, we used the calculator offered at https://datatab.net/tutorial/f-distribution, where it gave an F-critical of 1.268. Now, if F > F-critical then the H_0 is rejected. In this case, it was not rejected; therefore, there were no significant differences between the datasets.
- The second method required calculating the P-value. In this case, the library gave us this value (0.36056018). Now, if P-value < 0.05 then the H_0 was rejected. In this case, it was not rejected, so there were no significant differences between the datasets.

TABLE 14: TWO-SAMPLE HOTELLING'S T-SQUARED TEST RESULT

Dataset	Case	Features	Two-Sample Hotelling's T-Squared Test

[1] https://pingouin-stats.org/build/html/generated/pingouin.multivariate_ttest.html
[2] https://datatab.net/tutorial/f-distribution

			T2	F	df1	df2	P-value	Hypothesis
Energy	A	95	110.02386	1.04906202	95	904	0.36056018	H_0
	B	95	101.02386	1.06171521	95	58920	0.32111891	H_0
	C	864	895.56456	1.02137516	864	58151	0.32508918	H_0
	D	422	425.45645	1.00099831	422	58593	**0.48518434**	H_0
COVID-19	A	26	22.367299	0,82998916	26	685	**0.70947055**	H_0
	B	26	24.9877204	0.95986472	26	19973	0.52147351	H_0
	C	67	68.346573	1.01673144	67	19932	0.43904006	H_0
	D	59	57.346573	0.96915681	59	19940	**0.54289117**	H_0
Industry 4.0	A	3	1.3564567	0.45033636	3	496	0.71717036	H_0
	B	3	1.8544546	0.61808971	3	19996	0.60320568	H_0
	C	14	10.425738	0.74421147	14	19995	0.73092497	H_0
	D	10	6.725738	0.67227112	10	19989	**0.75131788**	H_0

When analyzing the experiments knowing that when P-values are less than 0.05 (tend to zero), H_0 is rejected, it is observed that the results using the FE module with the three Feature Engineering methods tend to obtain P-values closer to 1. For example, Energy with 0.48518434, COVID-19 with 0.54289117 and Industry 4.0 with 0.75131788. Only in COVID-19, the first experiment of this dataset with fewer samples performed better (0.70947055). In general, the P-value results indicated that the FE module contributes to improve the quality of the synthetic data, since SDGS was able to generate synthetic data very close to the data used to train the VAE model of the system, allowing the training of knowledge models in cases where there was a problem of insufficient data.

On the other hand, another test was carried out to evaluate the quality of the synthetic data generated. In particular, its use in the construction of classification models is verified. This test consisted of training and validating a Random Forest-based Classification model using the synthetic data, and then testing the quality of the model using the original data, to verify if it is capable of classifying them correctly. Table 15 shows the results achieved in this test using different datasets. In general, all models obtain excellent Accuracy and F1 score, both in training with the synthetic data and in the tests carried out with the original dataset. It is consistently observed that the models created with the synthetic data generated through the different processes of Feature Engineering (cases C and D) obtain better results, as seen for Energy with values of 0.936 and 0.883, in COVID-19 with 0.952 and 0.923, and in Industry 4.0 with 0.959 and 0.928, respectively. Thus, it is observed that the models with the synthetic data generated by the FE module offer better results than the models that do not use FE.

TABLE 15: RESULT OF THE ACCURACY OF CLASSIFICATION MODELS

Dataset	Case	Sample	Features	Training	Testing

				Accuracy	F1	Accuracy	F1
Energy	A	500	95	0.894	0.869	0.833	0.795
	B	29508	95	0.902	0.885	0.856	0.803
	C	29508	864	**0.923**	**0.893**	**0.877**	**0.827**
	D	29508	422	**0.936**	**0.912**	**0.883**	**0.831**
COVID19	A	356	26	0.905	0.888	0.872	0.844
	B	10000	26	0.924	0.901	0.895	0.858
	C	10000	67	**0.941**	**0.914**	**0.908**	**0.869**
	D	10000	59	**0.952**	**0.932**	**0.923**	**0.881**
Industry 4.0	A	250	3	0.903	0.892	0.864	0.815
	B	10000	3	0.927	0.903	0.889	0.862
	C	10000	14	**0.944**	**0.927**	**0.916**	**0.878**
	D	10000	10	**0.959**	**0.940**	**0.928**	**0.890**

B. Comparative analysis

This section carries out a comparison with other similar works in the literature. For that, two types of comparisons were carried out.

B.1. Qualitative comparison

In this paper are identified the following four evaluation criteria for a qualitative comparison with other works (Table 16):

- C1: Sample data acquisition mechanism.
- C2: Sample data preparation methods.
- C3: Feature engineering techniques.
- C4: Metrics used to measure the quality of synthetic dataset generation.

TABLE 16: QUALITATIVE ANALYSIS WITH PREVIOUS WORKS

Work	C1	C2	C3	C4
[9]	Dataset	Not indicated	Not indicated	Mean and Standard Deviation of

				Minutiae, False Acceptance Rate (FAR) and True Acceptance Rate (TAR)
[10]	Dataset	Not indicated	Not indicated	K-mean clustering algorithm and Area Under the Curve (AUC)
[15]	Dataset	Data Normalization and Data Scaling	Not indicated	Dice loss, IoU score, F-score, Accuracy, Recall, and Precision
[16]	Ontology	Not indicated	Not indicated	Recall, Precision, F-score, Average Precision and IuO score
[17]	Dataset	Not indicated	Not indicated	Accuracy
[18]	Dataset	Not indicated	Not indicated	Validity, Novelty, Uniqueness, Reconstruction and Score
[20]	Dataset	Not indicated	Not indicated	Discriminative score and Predictive scores
[21]	Dataset	Data Normalization and Categorical Data	Not applied	Data heatmap, Correlation, Charts by features and Accuracy
Our approach	Automatic extraction of datasets using LD and REST API	Data Normalization and Categorical Data	Feature Generation (5 techniques), Feature Extraction (3 techniques) and Feature Selection (4 techniques)	Two-Sample Hotelling's T-Squared, F1, and Accuracy

Regarding criterion C1, most of them used datasets with data samples prepared for a specific study. However, in [16], the authors used an ontology that stores all the knowledge and from there, the sample dataset was extracted for the generation of synthetic data. In our work, the datasets were automatically extracted from external sources using LD. In addition, our dataset extraction mechanism allows the fusion of several datasets for a single data sample, allowing extending the features of the synthetic dataset. With respect to C2, most of the works do not indicate what preprocessing is done with the dataset, although, due to the generation model they use, all of them should perform a normalization of the data. Additionally, in [15] was carried out a pyramidal rescaling of the input data, obtaining information from different levels of precision. In [21] and our work, several transformations were carried out.

Regarding criterion C3, in [21], they do not apply feature engineering techniques to improve the generation of synthetic data, and in the rest of the works, they do not indicate if they use any technique. In our case, the system has several methods for feature extraction, generation and selection (see sections III.D and IV.D). Concerning criterion C4, the works [9], [10], [18] and [21] used specific metrics for the type of problem to be solved. In our case, we used the two-sample Hotelling T-squared to determine in general terms the degree of similarity of the synthetic dataset with respect to the sample dataset used to train the VAE model, and the accuracy to evaluate the quality of the knowledge model. Some papers used the same metrics (e.g., [15], [17], [21]), and [15] and [16] use Recall, Precision and F-score.

B.2. Quantitative comparison

This section focuses on a quantitative comparison with other works that used the same metric as ours. We have used the same context of study in each case to carry out this test. Table 17 presents the metrics obtained in these works.

TABLE 17: QUANTITATIVE ANALYSIS WITH PREVIOUS WORKS

Work	Accuracy
[15]	96%
[17]	95%
[21]	No indicate
Our approach	92% for dataset from [15] 96% for dataset from [17]

Regarding the works [15], the results were a little lower, and that difference perhaps came from the generation model used in each case (in [15], a generative adversarial network, and in our case, a VAE technique). In [17] and our work, the values of the accuracy metric were higher than 95%, i.e., they presented very good results. In general, in our work, the results were presented in a range of values ranging from 92% to 96%, these results depended on the characteristics of each dataset used to train our synthetic data generator. Although [21] uses the same metric, the value achieved is not presented in the paper.

VI. CONCLUSIONS

The extended SDGS provides an automated architecture for the generation of synthetic data using the LD paradigm as an identifier and extract of data source, and VAE as a generator. In this work, three extensions were implemented. The multisource extension allows extracting sample data automatically from different data sources. In the case of our architecture, it has implemented an extraction mechanism based on LD queries, where queries were built using semantic tags that allow extracting datasets with the features of the context of the problem. In addition, this architecture integrates a mechanism that allows datasets to be merged, which favors the expansion of the features of a dataset. This multi-dataset extension was based on the automatic search for pivots that serve as a common point for the union of the datasets. Finally, the third extension adds to the SDGS architecture a new module with Feature Engineering processes that allowed generating and extracting of new features by applying different techniques to the dataset. Also, it allows selecting the features that offer more information to the VAE model. This module included five feature generation techniques, three feature extraction techniques and four feature selection techniques.

A case study was described in detail to show the new capabilities of the SDGS architecture using the three expansions specified in this research. Finally, two metrics were used, the first one was the accuracy for analyzing the performance of the models generated with the new SDGS functionalities, obtaining results above 90%. The second one was the Two-Sample Hotelling's T-Squared Test to determine the quality of the synthetic data generated by the system, obtaining synthetic datasets very similar to the datasets used to train the VAE models. Also, a comparative analysis was carried out with related works of generation of synthetic data, detailing the different characteristics present in each of them.

SDGS has some limitations, for example, it only feeds on datasets from repositories obtained using the LD paradigm. The second limitation is in its pivot selection mechanism for merging datasets. In addition to checking that the columns were of the same data type, only the column names were compared using the syntactic similarity of the two strings. The third limitation is that the architecture only uses default values in its different configuration parameters (e.g., for the VAE technique).

Future works will test other types of data sources for the multi-source process of the SDGS architecture, allowing the construction of sample data from databases, and knowledge graphs, among other current technologies associated with dataset repositories. For the multi-dataset process, it will test other pivot selection mechanisms with innovative techniques in text interpretation to allow semantic analysis of text strings, use of synonyms, translations, among others. Likewise, it will use the LD paradigm for the construction of a new intelligent layer, called Meta learning, which will allow the automatic adjustment of the architecture's parameters to optimize the generation of synthetic datasets according to the desired context and the knowledge model to be built. Finally, other future work should evaluate the quality of synthetic data obtained in contexts other than classification tasks, such as, for example, to build predictive or prescriptive models (in this particular area, there is very little data).

FUNDING

Jose Aguilar was partially supported by grant 22-STIC-06 (HAMADI 4.0 project) funded by the STIC-AmSud regional program.

REFERENCES

[1] Dos Santos, R., Aguilar, J., & R-Moreno, M. D. (2022). A synthetic Data Generator for Smart Grids based on the Variational-Autoencoder Technique and Linked Data Paradigm. In 2022 XVLIII Latin American Computer Conference (CLEI) https://doi.org/10.1109/CLEI56649.2022.9959918

[2] Avazpour, I., Grundy, J., & Zhu, L. (2019). Engineering complex data integration, harmonization and visualization systems. Journal of Industrial Information Integration (Vol. 16, pp. 100103). https://doi.org/10.1016/j.jii.2019.08.001

[3] Izquierdo, Y., Casanova, M. A., García, G., Dartayre, F., & Levy, C. H. (2017). Keyword Search over Federated RDF Datasets. In ER Forum/Demos (pp. 86-99). https://dblp.org/rec/conf/er/IzquierdoCGDL17

[4] Rao, G., Zhang, L., Zhang, X., Li, W., Li, F., & Tao, C. (2019). A Multi-Source Linked Open Data Fusion Method for Gene Disorder Drug Relationship Querying. In SEPDA@ ISWC (pp. 31-35). https://dblp.org/rec/conf/semweb/RaoZZLLT19

[5] Chen, Y. (2022, May). Linked Data Fusion Based on Similarity Calculation and K-Nearest Neighbor. In Journal of Physics: Conference Series (Vol. 2221, No. 1, pp. 012043). IOP Publishing. https://doi.org/10.1088/1742-6596/2221/1/012043

[6] Nishimaki, K., Ikuta, K., Onga, Y., Iyatomi, H., & Oishi, K. (2022, October). Loc-VAE: Learning Structurally Localized Representation from 3D Brain MR Images for Content-Based Image Retrieval. In 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 2433-2438). IEEE. https://doi.org/10.1109/SMC53654.2022.9945411

[7] Van Dao, T., Sato, H., & Kubo, M. (2022). An Attention Mechanism for Combination of CNN and VAE for Image-Based Malware Classification. IEEE Access (Vol. 10, pp. 85127-85136). https://doi.org/10.1109/ACCESS.2022.3198072

[8] Hadipour, H., Liu, C., Davis, R., Cardona, S. T., & Hu, P. (2022). Deep clustering of small molecules at large-scale via variational autoencoder embedding and K-means. BMC bioinformatics (Vol. 23, No. 4, pp. 1-22). https://doi.org/10.1186/s12859-022-04667-1

[9] Engelsma, J. J., Grosz, S. A., & Jain, A. K. (2022). PrintsGAN: Synthetic fingerprint generator. IEEE Transactions on Pattern Analysis and Machine Intelligence (pp. 1-14). https://doi.org/10.1109/TPAMI.2022.3204591

[10] Shah, P., Ullah, H., Ullah, R., Shah, D., Wang, Y., Islam, S., Gani A. Rodrigues, J. J. (2022). DC-GAN-based synthetic X-ray images augmentation for increasing the performance of EfficientNet for COVID-19 detection. Expert Systems (Vol. 39, No. 3, pp. e12823). https://doi.org/10.1111/exsy.12823

[11] Aguilar J, Jerez M, Exposito E., Villemur T, (2015) CARMiCLOC: Context Awareness Middleware in Cloud Computing, In *2015 Latin American Computing Conference (CLEI)*, doi: 10.1109/CLEI.2015.7360013.

[12] Morales, L., Ouedraogo, C., Aguilar, J., Chassot C, Medjiah S., Drira K. (2019) Experimental comparison of the diagnostic capabilities of classification and clustering algorithms for the QoS management in an autonomic IoT platform. Service Oriented Computing and Applications (Vol. 13, pp. 199–219)

[13] Aguilar, J.; Salazar, C.; Velasco, H.; Monsalve-Pulido, J.; Montoya, E. (2020) Comparison and Evaluation of Different Methods for the Feature Extraction from Educational Contents. Computation,(vol 8) https://doi.org/10.3390/computation8020030

[14] Quintero Y, Ardila D., Camargo E., Rivas F, Aguilar J. (2021) Machine learning models for the prediction of the SEIRD variables for the COVID-19 pandemic based on a deep dependence analysis of variables, Computers in Biology and Medicine (Vol. 134). https://doi.org/10.1016/j.compbiomed.2021.104500.

[15] Thambawita, V., Salehi, P., Sheshkal, S., Hicks, S., Hammer, L., Parasa, S., deLange T., Halvorsen P., Riegler, M. (2022). SinGAN-Seg: Synthetic training data generation for medical image segmentation. PloS one (Vol. 17, No. 5, p. e0267976). https://doi.org/10.1371/journal.pone.0267976

[16] Hoeser, T., & Kuenzer, C. (2022). SyntEO: Synthetic dataset generation for earth observation and deep learning–Demonstrated for offshore wind farm detection. ISPRS Journal of Photogrammetry and Remote Sensing (Vol. 189, pp. 163-184). https://doi.org/10.1016/j.isprsjprs.2022.04.029

[17] Pfitzner, B., & Arnrich, B. (2022). DPD-fVAE: Synthetic Data Generation Using Federated Variational Autoencoders With Differentially-Private Decoder. arXiv preprint arXiv:2211.11591. https://doi.org/10.48550/arXiv.2211.11591

[18] Ma, C., & Zhang, X. (2021, October). GF-VAE: a flow-based variational autoencoder for molecule generation. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (pp. 1181-1190). https://doi.org/10.1145/3459637.3482260

[19] Morales L., Aguilar J., Garcés-Jiménez A., Gutierrez De Mesa J., Gomez-Pulido J. (2020), "Advanced Fuzzy-Logic-Based Context-Driven Control for HVAC Management Systems in Buildings, *IEEE Access* (vol. 8, pp. 16111-16126) doi: 10.1109/ACCESS.2020.2966545.

[20] Desai, A., Freeman, C., Wang, Z., & Beaver, I. (2021). Timevae: A variational auto-encoder for multivariate time series generation. arXiv preprint arXiv:2111.08095. https://doi.org/10.48550/arXiv.2111.08095

[21] Akkem, Y., Biswas, S. K., & Varanasi, A. (2024). A comprehensive review of synthetic data generation in smart farming by using variational autoencoder and generative adversarial network. Engineering Applications of Artificial Intelligence (Vol. 131, pp. 107881). https://doi.org/10.1016/j.engappai.2024.107881

[22] Aref, S., J. Shortle, L. Sherry. (2024). Generating synthetic flight tracks for collision risk safety analysis: Variational autoencoders with a single seed track. In Proceedings of the Integrated Communications, Navigation, and Surveillance Conference, Herndon, VA.

[23] Hubert, N., Monnin, P., D'aquin, M., Monticolo, D., & Brun, A. (2024, May). PyGraft: Configurable Generation of Synthetic Schemas and Knowledge Graphs at Your Fingertips. In Semantic Web-21st International Conference, ESWC 2024. https://doi.org/10.5281/zenodo.10243209

[24] Aguilar J., Garcès-Jimènez A., Gallego-Salvador N., De Mesa J., Gomez-Pulido J., Garcìa-Tejedor A. (2919) Autonomic Management Architecture for Multi-HVAC Systems in Smart Buildings, *IEEE Access*, (vol. 7, pp. 123402-123415), 10.1109/ACCESS.2019.2937639

[25] Hoseini, S., Theissen-Lipp, J., & Quix, C. (2024). A survey on semantic data management as intersection of ontology-based data access, semantic modeling and data lakes. Journal of Web Semantics, 100819. https://doi.org/10.1016/j.websem.2024.100819

[26] Gourabpasi, A. H., & Nik-Bakht, M. (2024). BIM-based automated fault detection and diagnostics of HVAC systems in commercial buildings. Journal of Building Engineering, 87, 109022. https://doi.org/10.1016/j.jobe.2024.109022

[27] Marco R., Sakinah S. Ahmad S. (2022) Conditional Variational Autoencoder with Inverse Normalization Transformation on Synthetic Data Augmentation in Software Effort Estimation, International Journal of Intelligent Engineering and Systems, (Vol.15, No.3).

[28] Kuo N., Garcia F., Sönnerborg A., Böhm M, Kaiser R., Zazzi M., Polizzotto M., Jorm L., Barbieri S., (2023) Generating synthetic clinical data that capture class imbalanced distributions with generative adversarial networks: Example using antiretroviral therapy for HIV, Journal of Biomedical Informatics (Vol 144), https://doi.org/10.1016/j.jbi.2023.104436.

[29] Panfilo D. Boudewijn A.; Saccani S.; Coser A.; Svara B.; Rossi C. et al., (2023) A Deep Learning-Based Pipeline for the Generation of Synthetic Tabular Data," IEEE Access, (Vol. 11, pp. 63306-63323), doi: 10.1109/ACCESS.2023.3288336.

[30] Eigenschink P, Reutterer T, Vamosi S, Vamosi R., Sun C. Kalcher K. (2023) Deep Generative Models for Synthetic Data: A Survey, IEEE Access (vol. 11, pp. 47304-47320, doi: 10.1109/ACCESS.2023.3275134.

8.13 Anexo 5.F: Meta-learning Architecture for ACODAT in the Context of Agro-Industrial Production Chains of MSMEs

Fuentes, J., Dos Santos, R., Aguilar. Shi, Donghui. (2025). Meta-learning Architecture for ACODAT in the Context of Agro-Industrial Production Chains of MSMEs. EN REVISTA