UNIVERSIDAD DE MÁLAGA

ESCUELA DE INGENIERÍAS INDUSTRIALES

Máster en Ingeniería Mecatrónica

Creación de Mapas Semántico-Topológicos Para Robótica Móvil Utilizando Modelos de Aprendizaje Profundo

Building Semantic-Topological Maps for Mobile Robotics Using Deep Learning Models

Realizado por:
Jesús Moncada Ramírez

Tutorizado por:
José Raúl Ruiz Sarmiento
Javier González Jiménez

Departamento:
Ingeniería de Sistemas y Automática

Área de conocimiento:
Ingeniería de Sistemas y Automática

MÁLAGA, julio de 2025

Abstract

The ability of a mobile robot to carry out complex tasks largely depends on its level of understanding of the environment. A common option to achieve this advanced understanding —typically represented through a map— is the integration of both geometric information (spatial extent) and semantic information (identity, functionality, etc.) of the present objects. These representations can be further enriched by incorporating a division of the environment into meaningful places, as well as the relationships between these places and the objects, giving rise to the so-called semantic-topological maps. However, the construction of such maps has traditionally been limited by the need for manual annotations and the use of a closed vocabulary, which restricts both their scalability and expressiveness. In this context, recent advances in generative artificial intelligence (AI) offer a promising path to enrich these representations in a more flexible and automated manner.

This Master's Thesis explores how generative AI models —in particular, Large Language Models (LLMs) and Large Vision-Language Models (LVLMs)— can be used to automatically build semantic-topological maps while mitigating the aforementioned limitations. Specifically, two approaches are developed for place segmentation and categorization: one based on clustering of semantic descriptors contextualized through natural language processing techniques applied to the present objects, and another that directly leverages LLMs to infer the spatial structure of the environment. In addition, two strategies are proposed to infer semantic relationships between objects: a textual one, based solely on linguistic descriptions of the scene, and a multimodal one, which combines visual and textual information through LVLMs.

Experiments conducted on semantic maps generated from the ScanNet and SceneNN datasets confirm that the proposed methods effectively enrich the maps, overcoming the aforementioned limitations. Beyond the quality of the map construction, this work also evaluates the functional impact of such enriched maps on the operation of a mobile robot that uses an LLM as its reasoning engine. By comparing the LLM's performance in two configurations —one with a basic se-

mantic map (only objects) and another with an enriched map (objects, places, and relationships)— results show that the incorporation of topological information significantly improves the robot's ability to reason and act in a more informed and effective manner, especially when using smaller-scale language models.

Keywords: intelligent robotics, semantic-topological maps, deep learning, large models.

Resumen

La capacidad de un robot móvil para llevar a cabo tareas complejas depende en gran medida de su nivel de comprensión del entorno. Una opción recurrente para alcanzar esta comprensión avanzada —habitualmente representada mediante un mapa— es la integración tanto de información geométrica (extensión espacial) como semántica (identidad, funcionalidad, etc.) de los objetos presentes. Dichas representaciones pueden enriquecerse incorporando una división del entorno en lugares con un cierto significado, así como las relaciones entre estos lugares y los objetos, dando lugar a los denominados mapas semántico-topológicos. Sin embargo, la construcción de estos mapas ha estado tradicionalmente limitada por la necesidad de anotaciones manuales y el uso de un vocabulario cerrado, lo que restringe tanto su escalabilidad como su expresividad. En este contexto, los recientes avances en inteligencia artificial (IA) generativa ofrecen una vía prometedora para enriquecer dichas representaciones de forma más flexible y automatizada.

Este Trabajo de Fin de Máster explora cómo los modelos generativos de IA —en particular los *Large Language Models* (LLMs) y *Large Vision-Language Models* (LVLMs)— pueden emplearse para construir automáticamente mapas semántico-topológicos mitigando las limitaciones anteriores. En concreto, se desarrollan dos enfoques para la segmentación y categorización de lugares: uno basado en *clustering* de descriptores semánticos contextualizados mediante técnicas de procesamiento del lenguaje natural aplicadas a los objetos presentes, y otro que recurre directamente a LLMs para inferir la estructura espacial del entorno. Asimismo, se proponen dos estrategias para inferir relaciones semánticas entre objetos: una textual, basada únicamente en descripciones lingüísticas de la escena, y otra multimodal, que combina información visual y textual mediante LVLMs.

Los experimentos realizados sobre mapas semánticos generados a partir de los conjuntos de datos ScanNet y SceneNN confirman que los métodos propuestos enriquecen eficazmente los mapas, superando las limitaciones mencionadas. Más allá de la calidad en la construcción del mapa, este trabajo evalúa también el impacto funcional de dichos mapas enriquecidos en la operativa de un robot móvil

que emplea un LLM como motor de razonamiento. Al comparar el rendimiento del LLM en dos configuraciones —una con un mapa semántico básico (solo objetos) y otra con un mapa enriquecido (objetos, lugares y relaciones)— los resultados muestran que la incorporación de información topológica mejora significativamente la capacidad del robot para razonar y actuar de forma más informada y eficaz, especialmente cuando se utilizan modelos de lenguaje de menor escala.

Palabras clave: robótica inteligente, mapas semántico-topológicos, aprendizaje profundo, modelos de gran escala.

Índice

1 Introducción

Este capítulo establece el contexto general y los objetivos que guían este Trabajo Fin de Máster (TFM). La Sección 1.1 presenta la motivación del trabajo, destacando la relevancia de contar con mapas semánticos enriquecidos en el ámbito de la robótica móvil, identificando las principales dificultades asociadas a su construcción y subrayando la escasez de estudios que analicen su impacto en el rendimiento del robot, lo que justifica el enfoque adoptado. A continuación, la Sección 1.2 presenta el objetivo principal del trabajo y desglosa los objetivos específicos que han sido definidos para alcanzarlo. La Sección 1.3 describe las principales herramientas tecnológicas empleadas a lo largo del proyecto, incluyendo el lenguaje de programación, la infraestructura de ejecución y los modelos de inteligencia artificial utilizados. Finalmente, la Sección 1.4 explica la estructura del documento, resumiendo los capítulos y secciones que lo componen.

1.1. Motivación

Los robots móviles están siendo cada vez más utilizados para llevar a cabo tareas complejas en una amplia variedad de campos, que incluyen la asistencia en el hogar, la automatización industrial, la atención sanitaria y la educación [51]. Para abordar con éxito estos problemas y los retos que conllevan, los robots deben contar con capacidades cognitivas avanzadas que les permitan interpretar e interactuar con su entorno de manera inteligente.

Una forma cada vez más relevante de dar a los robots estas capacidades es la utilización de mapas del entorno semánticamente ricos —normalmente conocidos como mapas semánticos [17, 52]—, junto con un sistema de razonamiento y ejecución capaz de explotar esta información de manera efectiva [34]. Los mapas semánticos integran información tanto sobre la estructura espacial del entorno (p. ej. objetos y regiones), como sobre el contenido semántico asociado (p. ej. atributos, funcionalidades y relaciones entre objetos), lo que proporciona una

base sólida para la comprensión de un entorno de trabajo.

Un mapa de estas características puede contener información sobre las distintas áreas o lugares en que se divide. Para ello, normalmente se segmenta el espacio de trabajo en regiones o lugares significativos, lo que se conoce como segmentación de lugares (*place segmentation* o *room segmentation*) y, posteriormente, se asignan etiquetas o descripciones semánticas a cada región, lo que se conoce como categorización de lugares (*place categorization*).

También es posible que el mapa codifique relaciones entre los elementos que lo conforman. Por ejemplo, entre los objetos de un mapa pueden existir relaciones estructurales, como las relaciones de soporte (un objeto que está encima de otro), relaciones de funcionalidad (una silla sirve para sentarse y suele estar asociada a una mesa), afinidad contextual (objetos que suelen aparecer juntos, como un microondas y una encimera), usabilidad conjunta (como el carrito de la compra y la caja registradora) o incluso relaciones causales (un sensor o botón que activa una puerta automática). De forma análoga, es posible establecer relaciones entre los distintos lugares en los que ha sido segmentado el mapa, conectándolos en función de su navegabilidad. Así, si existe una conexión espacial directa que permita al robot desplazarse de un lugar a otro, ambos lugares aparecerán relacionados, es decir, conectados en el mapa.

Cuando un mapa semántico contiene los objetos del entorno acompañados de información semántica, junto con las relaciones entre ellos y los lugares donde se encuentran, así como vínculos entre los propios lugares, puede entenderse como una representación con estructura topológica, es decir, un mapa semántico-topológico (véase la Figura 1).

Por ejemplo, en un entorno como un supermercado, un mapa con estas características podría incluir una sección de lácteos (lugar), cuya función es ofrecer productos refrigerados (funcionalidad) y que contiene objetos como frigoríficos y estanterías (relaciones lugar-objeto). El mapa también podría codificar que los frigoríficos se utilizan para almacenar productos perecederos como la leche y el queso (funcionalidad), y que tienen una forma rectangular y alargada (propiedad); o que los carritos de la compra y las cajas registradoras se utilizan conjuntamente durante el proceso de compra (relación objeto-objeto). Además, el mapa puede incluir que la sección de lácteos está conectada con la sección de cárnicos por medio de un pasillo compartido (relación lugar-lugar), mientras que no está directamente conectada con la zona de cajas registradoras, ya que para acceder a esta última es necesario pasar por otras áreas intermedias.

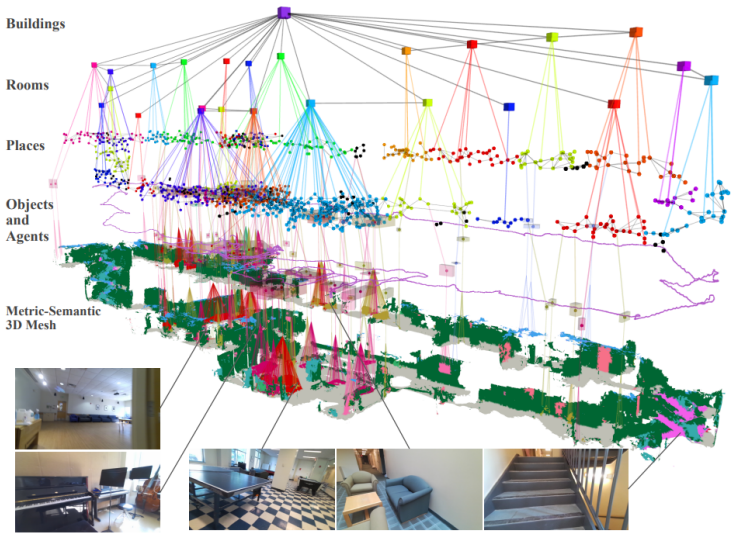


Figura 1: Ejemplo de mapa semántico-topológico propuesto en Hydra [23]. Este trabajo propone la construcción de mapas que combinan información geométrica, etiquetas semánticas y relaciones espaciales entre lugares para representar entornos complejos de forma estructurada.

Tradicionalmente, los enfoques para construir mapas semánticos han enfrentado varias limitaciones. Primero, algunos métodos dependen de la anotación manual, un proceso que consume mucho tiempo, es propenso a errores humanos y normalmente resulta en representaciones excesivamente simplificadas que no capturan la complejidad de los entornos reales.

Además, las técnicas para la construcción de estos mapas suelen funcionar bajo la restricción de vocabulario cerrado (*closed vocabulary*), que implica que el mapa solo podrá contener información incluida en un repositorio predefinido, como una ontología [58], o información vista durante la etapa de entrenamiento de un modelo de aprendizaje computacional. De esta forma, la representación semántica es ajena a una gran cantidad de conceptos que pueden ser cruciales para la operación de un robot, especialmente en aquellos casos en que se necesita autonomía a largo plazo. Esta limitación se hace patente, por ejemplo, en el proceso de detección de los objetos que forman parte de la representación semántica, una tarea que suele llevarse a cabo mediante cámaras y modelos de detección aplicados a las imágenes capturadas. Si el detector de objetos empleado ha sido entrenado con un conjunto limitado de categorías, no será capaz de detectar un gran número de objetos que puede encontrar en un entorno real y, por tanto, el mapa carecerá de dicha información. A la hora de segmentar y categorizar un

Figura 2: Ejemplo de clasificación de escenas con un modelo basado en vocabulario cerrado, tomado de Zhou et al. [65]. El sistema utiliza una red convolucional para asignar a cada imagen una etiqueta perteneciente a un conjunto fijo de 205 categorías semánticas, como "*spare bedroom*", "*messy kitchen*", "*rocky coast*" o "*darkest forest path*".

mapa en lugares también observamos esta limitación; muchos enfoques se basan en el concepto de "habitación", definen un conjunto de habitaciones predefinido y segmentan el mapa en función de dichas categorías (p. ej. "cocina", "habitación", "baño", etc.) (véase la Figura 2). Esta estrategia, además de no reflejar la complejidad del mundo real, genera inconsistencias semánticas, sobre todo en espacios amplios y multifuncionales donde pueden convivir varias actividades. En estos casos, etiquetar toda la zona con una única categoría de habitación reduce la información disponible y puede comprometer el rendimiento del robot.

En los últimos años, la riqueza de estas representaciones semánticas parece desempeñar un papel crucial en la mejora del rendimiento de los robots móviles en tareas complejas. Sin embargo, aunque diversos trabajos han demostrado que incorporar información semántica en tareas que tradicionalmente se resolvían con métodos más simples mejora el rendimiento, no existen trabajos que evalúen cómo la calidad y cantidad de esta información pueden afectar a las capacidades de razonamiento y toma de decisiones de los robots.

La aparición de técnicas avanzadas de aprendizaje profundo (en inglés *Deep Learning*, DL), como los modelos de *embeddings* de texto o imagen [12, 27, 47], y los modelos generativos de Inteligencia Artificial (IA) como los Modelos de Lenguaje de Gran Escala (en inglés *Large Language Models*, LLMs) [60, 44, 45, 37] o los Modelos de Visión-Lenguaje de Gran Escala (en inglés *Large Vision-Language Models*, LVLMs) ha supuesto una revolución en la robótica por-

que abre la puerta a abordar problemas clásicos con enfoques más flexibles y con capacidad de adaptación a contextos no vistos durante el entrenamiento. Estos modelos han sido entrenados con grandes cantidades de información del mundo real, lo que los hace de gran utilidad a la hora de generar representaciones semánticas [18], o incluso entenderlas y razonar sobre ellas [46].

Este TFM aborda las limitaciones asociadas a la construcción de mapas semántico-topológicos mediante el uso de modelos avanzados de IA generativa, como los LLMs y los LVLMs. En particular, se explora cómo estos modelos pueden enriquecer mapas semánticos incorporando información topológica desde dos enfoques complementarios: por un lado, la segmentación y categorización funcional del espacio en lugares; por otro, la inferencia de relaciones semánticas entre objetos. Para la segmentación y categorización de lugares se proponen dos métodos alternativos: el primero se basa en técnicas de *clustering* aplicadas a descriptores enriquecidos con información semántica que capturan la funcionalidad de los objetos; el segundo adopta un enfoque de extremo a extremo, en el que un LLM procesa directamente el mapa semántico para segmentarlo en regiones funcionales. En cuanto a la inferencia de relaciones semánticas entre objetos, también se presentan dos estrategias: una basada exclusivamente en LLMs que utilizan descripciones textuales de los objetos como entrada; y otra que combina información textual y visual, permitiendo a un LVLM razonar de forma multimodal sobre los vínculos existentes entre los objetos del entorno.

Además de evaluar de forma exhaustiva el rendimiento de cada una de las propuestas anteriores, este TFM también aborda una de las principales carencias de la literatura actual: la falta de evidencia que demuestre en qué medida la calidad y la cantidad de la información semántica contenida en el mapa afecta al rendimiento del robot en tareas cognitivas complejas como el razonamiento, la planificación o la toma de decisiones. Para ello, se evalúa el desempeño de un robot móvil equipado con un LLM como motor de razonamiento y decisión ante un conjunto de peticiones de un usuario en dos escenarios diferentes. En el primer escenario, el LLM cuenta con un mapa semántico con información únicamente sobre los objetos que lo conforman. En el segundo caso, el LLM cuenta con un mapa semántico enriquecido que, además de la información de los objetos presentes en la escena, contiene información topológica sobre los lugares en los que se divide el entorno y sus relaciones. En ambos casos se utilizan los mismos mapas semánticos de base, procedentes de dos conjuntos de datos de uso generalizado

disponibles públicamente: ScanNet [11] y SceneNN [20]. De esta forma, se consigue demostrar que cuanto más exacta sea la información semántica proporcionada a un robot equipado con un LLM sobre un mapa, mejor será su desempeño en tareas que requieran un entendimiento del entorno.

La propuesta basada en *clustering* de descriptores geométrico-semánticos enriquecidos con información funcional para la segmentación y categorización de lugares, junto con algunos resultados de su evaluación, ha sido aceptada como artículo en el congreso *European Conference on Mobile Robots* (ECMR)[1], que se celebrará en Padua, Italia, en septiembre de 2025. El estudio sobre el desempeño de un robot móvil equipado con un LLM para responder a consultas de un usuario sobre un mapa semántico se basa en un artículo publicado en la revista *MDPI Robotics* en febrero de 2025 [34], en el que se exploran técnicas para mejorar la capacidad de razonamiento de los LLMs sobre dichos mapas en contextos de interacción con el usuario. Los métodos propuestos, junto con su evaluación, están disponibles en un repositorio público de GitHub, el cual se detalla en el Apéndice A.

1.2. Objetivos

El propósito global de este TFM es explorar cómo los modelos de gran escala de IA generativa y vocabulario abierto pueden contribuir a la creación de representaciones semántico-topológicas en robótica móvil, al igual que demostrar que la disponibilidad de estas representaciones supone una mejora en la operativa de un robot equipado con un LLM como motor de comprensión del mapa y razonamiento sobre el mismo. A partir de este objetivo general se establecen los siguientes objetivos específicos:

1. **Estudio del estado del arte.** Revisión de las técnicas actuales para la generación de mapas semántico-topológicos, con especial atención a aquellas que emplean modelos de IA generativa. Asimismo, se realizará un análisis de métodos del estado del arte orientados a la explotación de mapas con información semántica mediante el uso de modelos a gran escala, como los LLMs y los LVLMs.

2. **Segmentación y categorización en lugares.** Diseño e implementación de procedimientos de segmentación y categorización de lugares en mapas semánticos, basados en

[1] https://ecmr2025.dei.unipd.it/

modelos de gran escala. Los métodos desarrollados abordarán los problemas existentes como la necesidad de anotaciones manuales, el vocabulario cerrado, y la división restringida al concepto de habitación.

3. **Generación de relaciones semánticas.** Diseño e implementación de técnicas de inferencia de relaciones semánticas entre los elementos de mapas semánticos, basadas en modelos de gran escala. Los sistemas enfrentarán los problemas existentes en este ámbito como el vocabulario cerrado y la dependencia de la anotación manual.

4. **Validación del impacto de mapas semánticos enriquecidos en el desempeño de un robot móvil.** Evaluación sobre la forma en que los mapas semánticos enriquecidos con información topológica empleando los desarrollos anteriores pueden afectar positivamente al desempeño de un robot móvil que los tome como referencia.

1.3. Tecnologías usadas

Este TFM ha sido desarrollado integrando diversas tecnologías ampliamente utilizadas en entornos de investigación. A continuación, se describen las principales herramientas empleadas.

Python. Todo el código del proyecto ha sido implementado en el lenguaje de programación Python, debido a su amplio ecosistema de librerías, su sintaxis sencilla y su fuerte adopción en tareas de visión por computador, procesamiento del lenguaje natural y aprendizaje profundo. Para gestionar las dependencias del proyecto y garantizar la reproducibilidad de los experimentos, se ha utilizado un entorno virtual, lo que permite mantener un entorno de ejecución limpio y controlado. En él se han instalado las bibliotecas necesarias mediante el gestor de paquetes `pip`, especificando versiones concretas para asegurar la compatibilidad entre componentes.

Control de versiones en Git y repositorios en GitHub. El control de versiones se ha llevado a cabo utilizando la herramienta Git, con repositorios alojados en la plataforma GitHub. Esto ha permitido mantener un historial completo de cambios, facilitar la depuración del código y documentar adecuadamente el desarrollo del proyecto. Además, el código ha sido publicado en repositorios públicos para fomentar la transparencia y la reutilización por parte de la comunidad investigadora.

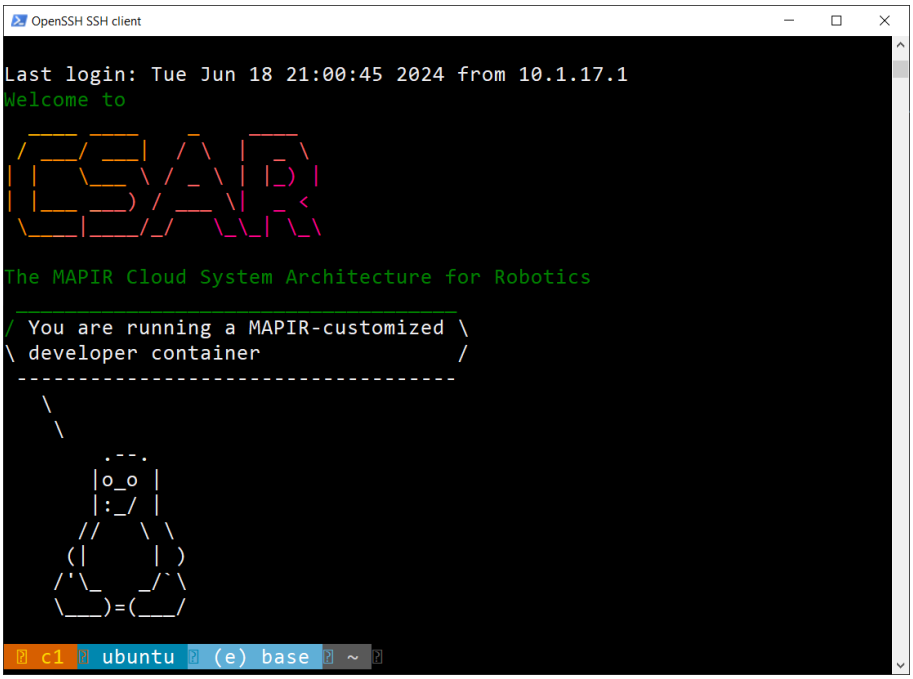


Figura 3: Mensaje de bienvenida de CSAR (*Cloud Architecture for Robotics*) al acceder por SSH.

Arquitectura de contenedores basada en Linux. Para la ejecución de los experimentos y el despliegue de los distintos componentes del sistema, se ha utilizado una arquitectura de contenedores basada en Linux proporcionada por la infraestructura CSAR (*Cloud System Architecture for Robotics*) [2], mantenida por el grupo de investigación MAPIR[2] (véase la Figura 3). Esta infraestructura permite aislar entornos de ejecución en contenedores, garantizando la consistencia entre distintas máquinas y facilitando la escalabilidad del sistema. En concreto, los experimentos se han ejecutado en el servidor *Ultra Edge*, una máquina de altas prestaciones mantenida también por el grupo de investigación MAPIR. Este sistema cuenta con un procesador AMD Ryzen Threadripper PRO 7975WX (32 núcleos a 4.0 GHz), 512 GB de memoria RAM DDR5 y tres GPUs NVIDIA RTX6000 ADA, cada una con 48 GB de memoria GDDR6, lo que le confiere una capacidad de cómputo especialmente adecuada para cargas de trabajo intensivas. El servidor opera bajo Ubuntu 22.04 LTS y dispone de CUDA 12.6, lo que garantiza la compatibilidad con los modelos y bibliotecas utilizados en este trabajo.

Modelos de *embedding* de palabras y frases. Se han empleado varios modelos de *embedding* preentrenados para obtener descriptores semánticos a partir de texto. En concreto, se han utilizado modelos como BERT [12], RoBERTa [27] y Sentence-BERT [47], todos ellos

[2] https://mapir.isa.uma.es/

disponibles a través de la plataforma Hugging Face[3]. Estos modelos están basados en arquitecturas de tipo Transformer y han sido entrenados con grandes corpus de texto mediante objetivos de enmascaramiento de palabras (*masked language modeling*) o de predicción de la siguiente oración. Sentence-BERT, en particular, adapta BERT para generar representaciones semánticas de frases completas utilizando técnicas de aprendizaje contrastivo (*contrastive learning*). Gracias a estas propiedades, los descriptores generados por estos modelos capturan información semántica sutil, útil para la tarea abordada en este trabajo.

LLMs abiertos. Además de los modelos de *embedding*, se han empleado LLMs de código abierto pertenecientes a la familia Qwen 2.5, también disponibles públicamente a través de la plataforma Hugging Face. Este tipo de modelos ha cobrado especial relevancia en los últimos meses por su capacidad para abordar tareas complejas mediante técnicas de *fine-tuning* basadas en cadenas de pensamiento (*chain-of-thought*), lo que les permite razonar de forma más estructurada y coherente en problemas que requieren múltiples pasos inferenciales, superando en algunos casos a modelos con muchos más parámetros.

LLMs propietarios. Finalmente, también se han empleado LLMs propietarios de la familia Gemini[4], desarrollados por Google. Estos modelos se han utilizado mediante una API de acceso comercial, que permite enviar solicitudes con texto de entrada, comúnmente denominadas *prompts*, y recibir respuestas generadas por el modelo. El coste del servicio se basa en el número de tokens consumidos, contabilizando tanto los tokens de entrada como los de salida, lo que requiere una gestión eficiente del diseño de dichos *prompts*.

1.4. Estructura del documento

La memoria de este trabajo se divide en los siguientes capítulos:

Capítulo 1: Introducción. Contextualiza el trabajo, recoge la motivación que impulsa la investigación, enuncia los objetivos específicos, presenta las tecnologías utilizadas y anticipa la organización del documento.

Capítulo 2: Trabajos relacionados. Revisa el estado del arte en los ejes fundamentales de este trabajo: mapeo semántico, segmentación y categorización de lugares, modelado de relaciones semánticas entre objetos y explotación de mapas semánticos basada en modelos de

[3] https://huggingface.co/
[4] https://gemini.google.com/app

gran escala como LLMs. Este análisis identifica las limitaciones de los enfoques existentes y justifica las contribuciones realizadas.

Capítulo 3: Segmentación y categorización de lugares. Detalla las metodologías propuestas para dividir el entorno en regiones funcionalmente significativas y asignarles información semántica utilizando un vocabulario abierto. Se describen tanto los métodos desarrollados como los diferentes modelos involucrados en ellos.

Capítulo 4: Inferencia de relaciones semánticas. Aborda las técnicas propuestas para la generación automática de relaciones entre los objetos de un mapa semántico pertenecientes a un vocabulario abierto.

Capítulo 5: Validación. Presenta la evaluación experimental realizada. En primer lugar, se lleva a cabo una evaluación cuantitativa de las segmentaciones de lugares, así como un análisis cualitativo tanto de las caracterizaciones funcionales de estos lugares como de las relaciones semánticas inferidas entre objetos. Además, se demuestra que un mapa semántico enriquecido con información topológica sobre los lugares del entorno y las relaciones entre objetos mejora el desempeño de un robot móvil.

Capítulo 6: Conclusiones. Resume los hallazgos más relevantes, discute las implicaciones de los resultados y señala las líneas futuras que se abren a partir de esta investigación.

2

Trabajos relacionados

Este capítulo ofrece un análisis del estado del arte en los pilares que fundamentan este TFM. La Sección 2.1 repasa las técnicas de mapeo semántico, tanto densas como a nivel de instancia, poniendo énfasis en la tendencia reciente de utilización de modelos de gran escala. La Sección 2.2 aborda los métodos clásicos y modernos de segmentación y categorización de lugares, resaltando las limitaciones de los enfoques ligados al concepto de habitación y la aparición de técnicas basadas en *affordances*, muy relacionadas con la propuesta de este trabajo. La Sección 2.3 revisa los trabajos más recientes que modelan relaciones semánticas entre objetos y regiones, desde grafos ontológicos hasta representaciones abiertas impulsadas por modelos de gran escala, subrayando los avances y retos que inspiran las contribuciones de este trabajo. Por último, la Sección 2.4 plantea la tendencia creciente de utilizar modelos de gran escala como motores de razonamiento y comprensión de representaciones semánticas por parte de robots.

2.1. Mapeo semántico

El mapeo semántico es el proceso que vincula la información geométrica de los elementos presentes en el espacio de trabajo de un robot —normalmente capturada por sensores a bordo (cámaras RGB-D, LiDAR, etc.) y agregada en forma de alguna primitiva espacial (mapas de ocupación, nubes de puntos, *bounding boxes*, vóxeles, *surfels*, etc.)— con su semántica (propiedades, funcionalidades, relaciones, etc.) [16, 52]. Cuando este proceso se centra únicamente en modelar los objetos del entorno, se le puede denominar mapeo semántico *orientado a objetos* [56].

Según la forma en que se integra la información semántica en la representación del espa-

cio, existen dos variantes principales de mapeo semántico. El mapeo semántico *denso* anota individualmente cada primitiva geométrica (p. ej. un punto 3D, un vóxel, etc.) con su semántica [33, 53], lo que permite disponer de una representación continua de la escena y conlleva un incremento significativo en los requisitos de almacenamiento y procesamiento. El mapeo semántico *a nivel de instancia*, por el contrario, agrupa las primitivas pertenecientes al mismo objeto y las mantiene como entidades discretas [42, 43], lo que mejora el consumo de memoria y permite un seguimiento individual de cada instancia.

En ambos casos, los enfoques de mapeo semántico han evolucionado en los últimos años desde el uso de redes neuronales convolucionales (*Convolutional Neural Networks*, CNNs) —como *Faster R-CNN* [48] o *Mask R-CNN* [19]— hacia modelos de vocabulario abierto, que no restringen su funcionamiento a los datos vistos durante el proceso de entrenamiento. Entre estos enfoques destacan aquellos que incorporan LVLMs [18] para detectar objetos en las imágenes de entrada, y aquellos que utilizan modelos descriptores de imágenes como CLIP [9] para extraer la semántica implícita en las escenas.

Uno de los trabajos más recientes y relevantes en nuestro discurso sobre mapeo semántico es Voxeland [32] (véase la Figura 4), que propone un marco probabilístico para la construcción incremental de mapas semánticos a nivel de instancia a partir de una secuencia de imágenes RGB-D. Voxeland funciona generando una reconstrucción tridimensional del entorno basada en vóxeles, en la que cada vóxel contiene una distribución de probabilidad sobre el conjunto de posibles instancias de objetos dentro del mapa. A su vez, cada instancia de objeto está anotada con una distribución de probabilidad en el conjunto de posibles categorías de objetos. El marco probabilístico propuesto permite disponer de una representación más realista, en la que la verdadera categoría de cada instancia de objeto debe interpretarse en términos probabilísticos, en lugar de una etiqueta determinista, reflejando así la incertidumbre inherente a la categorización de objetos [31, 35]. Además, cuando el sistema detecta una ambigüedad semántica elevada —es decir, alta entropía en la distribución de categorías— recurre a LVLM para resolverla, cuyas opiniones son posteriormente integradas en el modelo probabilístico bajo la teoría de la evidencia.

En este TFM se abordan componentes clave dentro de un sistema de mapeo semántico, como la segmentación y categorización de lugares, así como la inferencia de relaciones semánticas entre objetos. Las técnicas desarrolladas en estos ámbitos podrían integrarse de forma

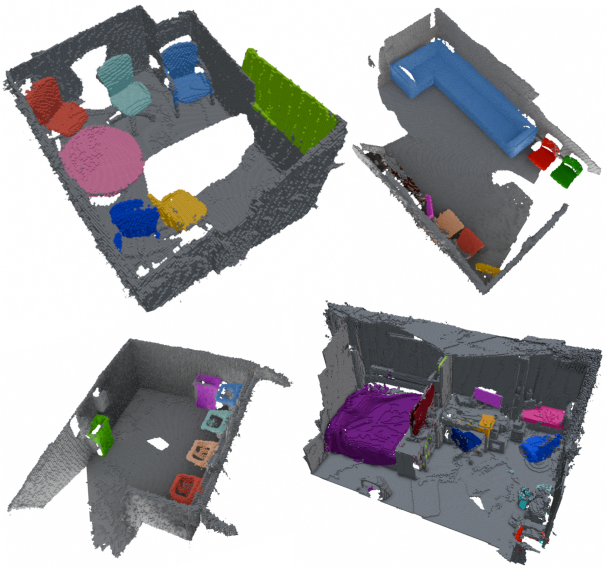


Figura 4: Reconstrucciones basadas en vóxeles generadas por Voxeland para escenas del conjunto de datos ScanNet [11]. Los vóxeles grises representan regiones sin categorizar, mientras que los vóxeles de color indican instancias de objetos distintas dentro del mapa.

natural en flujos de trabajo de sistemas de mapeo semántico del estado del arte, como Voxeland. No obstante, para asegurar que la evaluación se centre exclusivamente en los módulos propuestos en este trabajo, se ha optado por utilizar conjuntos de datos que proporcionan mapas semánticos orientados a objetos, como ScanNet [11] o SceneNN [20], en lugar de la salida de un flujo de trabajo de mapeo semántico de los comentados. Estos conjuntos proporcionan mapas generados y validados previamente, libres de ruido o errores de reconstrucción, lo que permite realizar una evaluación más precisa y aislada del rendimiento de las técnicas propuestas, sin que factores externos al diseño afecten a los resultados obtenidos.

2.2. Segmentación y categorización de lugares

Los métodos clásicos para abordar la segmentación y categorización de lugares pueden agruparse según el tipo de datos de entrada que utilizan. Algunos enfoques se apoyan únicamente en la información geométrica para explotar la disposición estructural del entorno [36, 15, 40]. De esta forma, solo las características geométricas de las distintas zonas, como habitaciones y pasillos, se emplean para segmentar y categorizar el espacio de trabajo del robot. Otros trabajos mejoran la categorización de lugares aprovechando la identificación de objetos

Figura 5: Ejemplo de categorización y segmentación de lugares en un mapa, siguiendo la propuesta de Sünderhauf et al. [55]. El método clasifica las imágenes de entrada utilizando una CNN desarrollada por Zhou et al. [65], y proyecta las etiquetas de escena resultantes sobre las celdas del mapa para asignar categorías semánticas a cada región.

y sus relaciones en la escena [17, 59]. Sin embargo, estos métodos están fuertemente ligados al concepto de habitación y consideran un número reducido de categorías de objetos y relaciones, lo que limita su capacidad para afrontar escenarios variados, poniendo de manifiesto el problema del vocabulario cerrado.

Existen enfoques que infieren directamente la categoría del lugar a partir de imágenes de entrada utilizando técnicas de aprendizaje profundo como CNNs [55, 41, 30], lo que mitiga parcialmente el problema del vocabulario cerrado (véase la Figura 5). Aunque estas técnicas ofrecen una capacidad limitada de vocabulario abierto —que depende del tamaño del conjunto de entrenamiento—, no aprovechan las ventajas del uso de modelos de lenguaje de gran escala para obtener una categorización más rica.

Recientemente han aparecido sistemas de mapeo semántico que integran mapeo 3D den-

so y semántica, como Kimera [50] o Hydra [10]. Ambos métodos ofrecen una representación jerárquica del entorno que incluye información geométrica y semántica, lo cual mejora el entendimiento del entorno por parte del robot. Sin embargo, a pesar de la riqueza espacial y semántica que alcanzan estas representaciones, siguen estando restringidas por el concepto de habitación, lo que puede limitar la comprensión de espacios abiertos o no domésticos por parte del robot.

En este contexto, este trabajo abandona esta perspectiva centrada en habitaciones, apoyándose en las relaciones semánticas entre objetos para producir una segmentación y categorización de los lugares del entorno que no tiene por qué coincidir con dichos espacios. De esta forma, la disposición de los objetos, sus relaciones y su semántica permiten crear una división del espacio de trabajo en distintas áreas funcionales, que enriquecen el mapa semántico original.

En este sentido, el enfoque seguido puede considerarse basado en *affordances*. En este tipo de enfoques, el foco se sitúa sobre las acciones que un agente puede realizar en un entorno, que están determinadas por las propiedades físicas de los objetos que encuentra y el contexto en que el agente se encuentra. Este marco ha sido explorado en direcciones distintas, como la planificación de trayectorias [4, 5] o la manipulación y el agarre [13, 62], donde las acciones disponibles se infieren directamente a partir de la geometría, la semántica y la función de los objetos.

2.3. Relaciones semánticas

Mientras que el mapeo semántico orientado a objetos, junto con la segmentación y categorización de lugares, permite identificar *qué* elementos hay en el mapa, *dónde* se encuentran y cómo se agrupan, la modelización de las relaciones entre ellos responde a la pregunta de *cómo* interactúan entre sí. Este tipo de representación añade una capa de información funcional al mapa, enriqueciendo su utilidad para tareas de razonamiento, planificación y comprensión del entorno.

Los primeros trabajos empleaban representaciones simbólicas basadas en ontologías para codificar relaciones de soporte como “está-encima-de” o “está-dentro-de”. Ejemplos destacados son las extensiones de KnowRob [57], que codificaban las relaciones en gráficos de conocimiento consultables por un razonador lógico. Sin embargo, presentan el problema del vocabulario

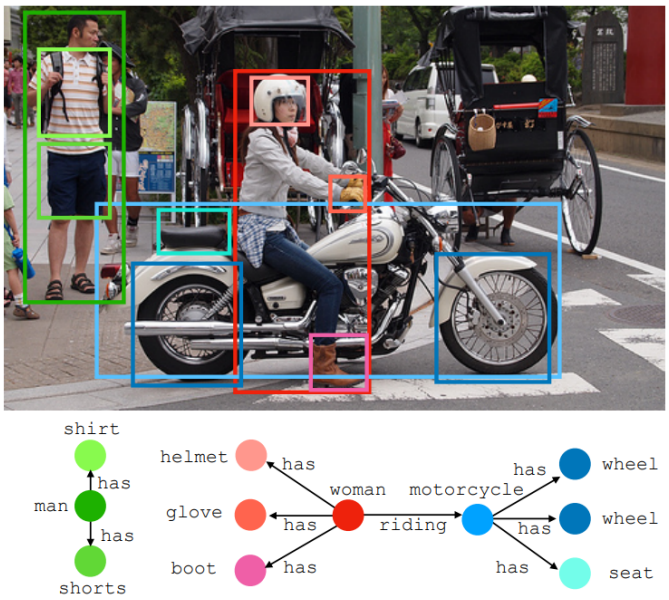


Figura 6: Ejemplo de grafo de escena tomado del trabajo de Zellers et al. [64], en el que se representan entidades ("*shirt*", "*motorcycle*", "*wheel*", etc.) y las relaciones semánticas que existen entre ellas (por ejemplo, la relación "*riding*" entre la entidad "*woman*" y la entidad "*motorcycle*").

cerrado, y la incorporación de nuevas relaciones exigía una laboriosa anotación manual.

Con la popularización de la visión por computador basada en aprendizaje profundo surgieron los grafos de escena 2D (*2D scene graphs*), representaciones basadas en imágenes en las que los nodos son objetos detectados en la imagen y las aristas codifican relaciones espaciales y semánticas aprendidas a partir de grandes conjuntos de datos (véase la Figura 6). Entre los pioneros se encuentra el trabajo de Johnson et al. [25] y la arquitectura de Neural Motifs [64], que demostraron gran capacidad para inferir relaciones de soporte ("sobre", "junto a", etc.) en fotografías estáticas.

La extensión de estos grafos de escena a 3D (*3D scene graph*) dio lugar a nuevos métodos para la construcción de estas representaciones [3], entre los cuales destaca el ya mencionado Kimera [50], donde se integra con *Simultaneous Localization and Mapping* (SLAM). En este último trabajo, las aristas modelan soporte, contención o proximidad entre instancias volumétricas. Aunque estas propuestas aportan coherencia geométrica, siguen dependiendo de un vocabulario cerrado de predicados aprendidos durante el entrenamiento de un determinado modelo.

Para romper estas limitaciones, en los últimos años se ha propuesto la utilización de modelos de gran escala. Por ejemplo, ConceptGraphs [18] utiliza un LVLM para describir los diferentes objetos de la escena y generar relaciones abiertas entre pares de instancias, lo que permite inferir vínculos funcionales de vocabulario abierto sin necesidad de entrenar un modelo específico. De forma análoga, Chen et al. [9] construyen gráficos 3D con predicados libres y razonamiento textual, y Chandhok et al. [7] emplean GPT-4 para sintetizar descripciones relacionales de escenas interiores.

Otros autores han explorado relaciones orientadas a los ya mencionados *affordances*. Por ejemplo, Pandey et al. [38] generan grafos donde las aristas denotan la posibilidad de "agarrar", "empujar" o "sentarse en", mientras que Xu et al. [63] etiquetan piezas concretas de objetos con las acciones que habilitan. Estas representaciones han mostrado su utilidad en la planificación de trayectorias [8] y manipulación [5].

Como se puede comprobar, la mayoría de los trabajos anteriores presentan la restricción del vocabulario cerrado de relaciones, o son aplicables únicamente en dominios muy específicos. El enfoque abordado en este trabajo es muy similar al utilizado por ConceptGraphs [18] y se encuentra en la línea de los métodos de vocabulario abierto. Mediante el aprovechamiento de modelos de gran escala como los LLMs, se generan relaciones abiertas entre objetos cercanos, integrándolas en el mapa semántico enriquecido para su posterior explotación por parte del robot.

2.4. Explotación de mapas semánticos con LLMs

Una tendencia creciente con la llegada de los LLMs y sus extraordinarias capacidades de razonamiento es aprovechar estos modelos para convertir indicaciones en lenguaje natural recibidas por un usuario en planes de acción del robot. Los primeros estudios [1, 54, 26, 21] dotaron a los LLMs con representaciones semánticas para lograr este fin, aunque estaban limitados a escenas de una sola habitación (véase la Figura 7). Dado que los LLMs operan sobre texto, el mapa semántico debe transformarse a un formato textual estructurado. Para ello, lo más habitual es recurrir a estructuras JSON sin estandarizar, debido a su simplicidad y compatibilidad directa con estos modelos.

Más recientemente, SayPlan [46] aborda la limitación del tamaño del mapa, permitiendo una planificación basada en LLMs en entornos más extensos y complejos. Dado un mapa se-

Figura 7: Ilustración del trabajo de Ahn et al. [1], que muestra la diferencia entre las respuestas generadas por un LLM sin contexto (izquierda) y las respuestas obtenidas cuando el LLM dispone de una representación semántica del entorno y es el motor de planificación de un robot (derecha).

mántico preconstruido y una consulta del usuario, SayPlan utiliza LLMs para identificar el subconjunto más relevante del mapa para la tarea, planificar las acciones del robot y replanificar en caso de que el plan inicial falle.

Una de las limitaciones de los enfoques basados en estos modelos es la dependencia depositada en los LLMs y sus respuestas, que se sabe que a veces incluyen información incorrecta, imprecisa o inventada [24], lo que puede provocar un comportamiento erróneo del robot. Para mitigar esta limitación, han surgido propuestas que, en el contexto de la planificación asistida por LLMs, introducen mecanismos de control como la auto-reflexión [29] o la colaboración entre múltiples agentes [39], con el objetivo de refinar y verificar las respuestas generadas [34].

En este trabajo se evalúa cómo la calidad de un mapa semántico enriquecido contribuye al desempeño de un robot móvil equipado con un LLM como motor de razonamiento, por lo que esta técnica cobra un papel fundamental.

3

Segmentación y categorización de lugares

En este capítulo se presentan las diferentes alternativas desarrolladas en este TFM para segmentar un mapa semántico previamente construido en lugares coherentes y, posteriormente, categorizarlos asignándoles una etiqueta simple y una descripción en lenguaje natural basada en un vocabulario abierto. La Sección 3.1 formaliza el problema y presenta la notación utilizada a lo largo del capítulo. A continuación, la Sección 3.2 describe la primera propuesta, cuya segmentación se basa en dos etapas. En primer lugar, se construyen descriptores que integran la información geométrica y semántica de cada objeto para obtener una segmentación inicial mediante *clustering* (véase la Sección 3.2.1). Después, esta segmentación inicial se refina dividiendo aquellos grupos que presentan una elevada variabilidad semántica y fusionando aquellos que, además de estar próximos, poseen una gran similitud semántica (véase la Sección 3.2.2). Una vez definida la segmentación, cada lugar es categorizado utilizando un LLM, que asigna de manera automática tanto una etiqueta concisa como una descripción en lenguaje natural a partir de los objetos que lo componen (véase la Sección 3.2.3). Por su parte, la Sección 3.3 presenta una alternativa basada exclusivamente en el uso de un LLM, que realiza la segmentación y categorización del mapa de manera conjunta. Por último, la Sección 3.4 explica los aspectos de implementación de los métodos explicados dentro del repositorio de código del proyecto.

3.1. Formulación del problema

En esencia, un mapa semántico $\mathbf{m}$ de un entorno 3D con N objetos puede definirse como:

$$\mathbf{m} = \{\mathbf{o}_1, \ldots, \mathbf{o}_N \mid \mathbf{o}_i = (c_i, d_i, l_i, \mathbf{p}_i^{\text{object}}), \forall i \in \{1, \ldots, N\}\} \tag{1}$$

donde:

- $c_i \in \mathbb{R}^3$ representa la localización del i-ésimo objeto (p. ej. , el centroide de su *bounding box*).
- $d_i \in \mathbb{R}^3$ representa las dimensiones del i-ésimo objeto (p. ej. , el tamaño de su *bounding box*).
- $l_i \in \mathcal{L}$ es la etiqueta semántica asignada al i-ésimo objeto, donde $\mathcal{L}$ es el conjunto de todas las clases semánticas posibles (p. ej. , "mesa", "silla", "jarrón", etc.).
- $\mathbf{p}_i^{\text{object}}$ es un conjunto de propiedades adicionales opcionales que describen el i-ésimo objeto.

Nótese que $\mathbf{p}_i^{\text{object}}$ puede emplearse para almacenar información complementaria de los objetos que resulte relevante en el contexto del problema en cuestión, como incertidumbres en la clasificación, múltiples etiquetas semánticas con sus puntuaciones de confianza o incluso relaciones con otros objetos del mapa.

En este contexto, la tarea de segmentación de lugares consiste en, dado un mapa semántico previamente construido de una escena, organizar los objetos en subconjuntos distintos —denominados lugares— que agrupen objetos del mapa relacionados tanto geométricamente como semánticamente. La relación geométrica entre objetos puede ser simplemente la cercanía, mientras que la relación semántica depende de otros factores como el contexto del problema o el tipo de mapa que se pretende segmentar. En este trabajo se ha considerado que dos objetos están semánticamente relacionados si sirven una funcionalidad similar. Por ejemplo, los objetos "caja registradora" y "datáfono" podrían considerarse semánticamente relacionados al cumplir ambos la misma finalidad de procesar pagos, aunque lo hagan de manera diferente. De esta forma, el proceso crea grupos de objetos que sirven una funcionalidad concreta para el robot en el mapa, facilitando una interpretación del entorno basada en el uso práctico de las distintas áreas.

Formalmente, dado $\mathbf{m}$, una función de segmentación de lugares $\psi(\cdot)$ produce un conjunto de lugares $\mathcal{P}$:

$$\mathcal{P} = \psi(\mathbf{m}) = \{P_k \mid k \in K\} \tag{2}$$

donde K es el conjunto de índices de lugares, y cada lugar P_k consiste en un subconjunto de los objetos del mapa:

$$P_k = \{\mathbf{o}_j \mid j \in J_k \subseteq \{1, \ldots, N\}\} \tag{3}$$

Nótese que J_k es el conjunto de índices de los objetos que pertenecen al k-ésimo lugar, y $\{1, \ldots, N\}$ es el conjunto de todos los índices de objeto.

Una vez definido el conjunto de lugares junto con los objetos que los componen, es posible enriquecer cada uno de ellos con información semántica adicional, ampliando así la utilidad del mapa. A este proceso se le denomina comúnmente categorización de lugares. Una estrategia eficaz para llevarla a cabo consiste en utilizar los propios objetos presentes en cada lugar como base para generar esta información semántica. Formalmente, dado un mapa segmentado en lugares $\mathcal{P} = \{P_k \mid k \in K\}$, una función de categorización de lugares $\phi(\cdot)$ asigna a cada lugar P_k un conjunto de propiedades semánticas $\mathbf{p}_k^{\text{place}}$:

$$\phi(P_k) = \mathbf{p}_k^{\text{place}} \tag{4}$$

Estas propiedades pueden incluir, por ejemplo, una etiqueta $t_k \in \mathbf{p}_k^{\text{place}}$ que identifique el lugar de forma general, o una descripción en lenguaje natural $d_k \in \mathbf{p}_k^{\text{place}}$ que ofrezca una explicación más detallada.

Por ejemplo, supongamos un mapa semántico $\mathbf{m}$ que contenga los objetos: “mesa”, “silla”, “pantalla”, “pizarra”, “proyector”, “cama”, “almohada” y “armario”. La segmentación podría agrupar estos elementos en dos lugares: P_1, conteniendo los objetos “mesa”, “silla”, “pantalla”, “pizarra” y “proyector”; y P_2, conteniendo los objetos “cama”, “almohada” y “armario”, suponiendo que, además de estar semánticamente relacionados entre sí, como se puede comprobar, están próximos en el espacio. A continuación, la categorización podría asignar a P_1 la etiqueta t_1 = “sala de reuniones” y la descripción $d_1 =$ “Espacio destinado a reuniones y presentaciones, equipado con mesa, sillas, pantalla, pizarra y proyector”, mientras que a P_2 podría asignar la etiqueta $t_2 =$ “dormitorio” y la descripción $d_2 =$ “Entorno destinado al descanso, con cama, almohada y armario”.

En resumen, el objetivo de la función de segmentación de lugares $\psi(\cdot)$ es generar una segmentación $\mathcal{P}$ tal que los objetos de cada lugar P_k sean geométrica y semánticamente coherentes, manteniéndose al mismo tiempo una baja similitud entre objetos pertenecientes a lugares distintos. Por su parte, la finalidad de la función de categorización $\phi(\cdot)$ es asignar a cada lugar el conjunto de características semánticas $\mathbf{p}_k^{\text{place}}$ que describa de la manera más adecuada la región correspondiente del mapa, enriqueciéndolo de cara a su posterior aprovechamiento.

3.2. Segmentación basada en *clustering* y categorización basada en LLMs

La primera propuesta para segmentar un mapa semántico en lugares consiste en aplicar *clustering* a la escena, construyendo para cada objeto descriptores que integran tanto su información geométrica como semántica, con el fin de obtener una partición inicial del mapa. Esta división inicial se refina posteriormente dividiendo aquellos grupos que presentan una alta variabilidad semántica y fusionando los que, además de estar próximos, muestran gran similitud semántica. Una vez definida la segmentación final, cada lugar resultante se categoriza mediante un LLM, que les asigna una etiqueta simple y una descripción en lenguaje natural. La Figura 8 ofrece un resumen general de todo el proceso.

3.2.1. Segmentación geométrico semántica

Para obtener la partición inicial, se construye para cada objeto un descriptor que integra tanto su información geométrica, ligada a su posición en el mapa, como su información semántica, ligada a su categoría semántica. De este modo, los objetos se proyectan en un espacio de características donde la proximidad implica tanto cercanía espacial como similitud semántica. A partir de esta representación, es posible aplicar *clustering* para identificar y agrupar conjuntos de objetos que conformen lugares coherentes en la escena.

Formalmente, para cada objeto del mapa semántico $\mathbf{o}_i \in \mathbf{m}$ se define un vector descriptor $\mathbf{f}_i \in \mathbb{R}^{3+d'}$ compuesto por dos partes:

$$\mathbf{f}_i = \begin{bmatrix} \mathbf{c}_i \\ \mathbf{s}_i \end{bmatrix} \tag{5}$$

donde:

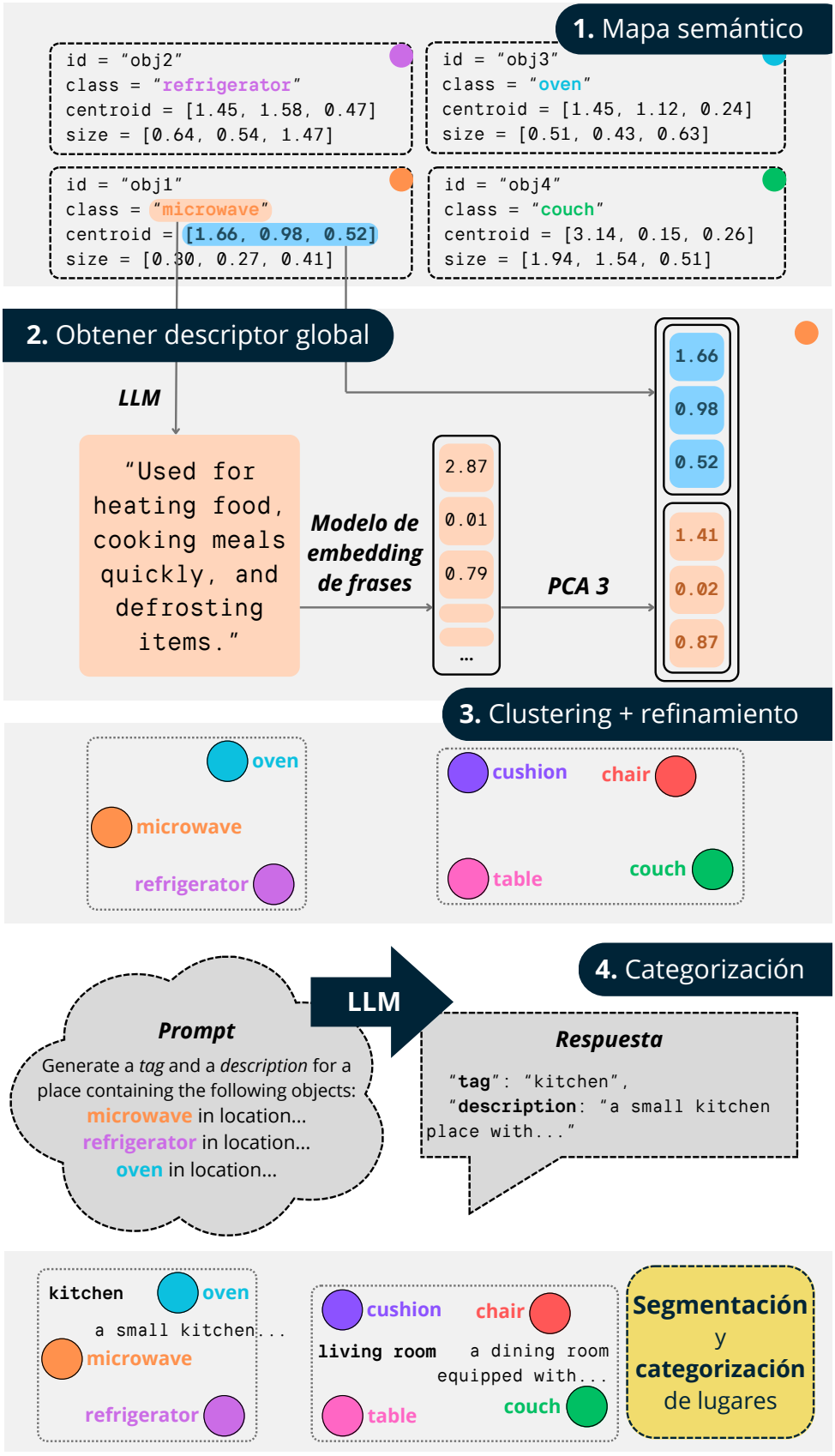


Figura 8: Esquema general del método propuesto para segmentar un mapa semántico en lugares mediante *clustering*, seguido de la categorización de cada lugar utilizando LLMs.

- $\mathbf{c}_i \in \mathbb{R}^3$ es el descriptor geométrico del objeto, p. ej. el centroide de su *bounding box*.
- $\mathbf{s}_i \in \mathbb{R}^{d'}$ es el descriptor semántico del objeto.

El descriptor semántico $\mathbf{s}_i$ puede obtenerse de varias maneras y determina cómo se utiliza la información semántica en el contexto del problema. Un enfoque simple es aplicar una función de *embedding* de palabras preentrenada $W : \mathcal{L} \to \mathbb{R}^d$ (p. ej. BERT [12] o RoBERTa [27]) a la etiqueta semántica del objeto $l_i \in \mathcal{L}$:

$$\mathbf{s}_i = \text{PCA}_{d'}(W(l_i)) \tag{6}$$

donde $\text{PCA}_{d'}(\cdot)$ denota la proyección a $\mathbb{R}^{d'}$ mediante Análisis de Componentes Principales (del inglés *Principal Components Analysis*, PCA) [28], usada para reducir la dimensión del descriptor.

No obstante, este enfoque prioriza implícitamente la semántica codificada en la función de *embedding*, lo cual puede no ser óptimo en el contexto de la aplicación en cuestión donde el robot opera. Por ejemplo, $W(\cdot)$ podría considerar que los objetos "caja registradora" y "nevera" son semánticamente similares por ser dispositivos electrónicos, aunque en el contexto de la segmentación de lugares, dichos objetos suelen pertenecer a regiones distintas y carecen de funcionalidad compartida; por tanto, no deberían quedar próximos en el espacio semántico (véase la Figura 9).

Para solventar esta limitación, se propone generar una descripción contextual en lenguaje natural T_i para cada objeto $\mathbf{o}_i$ utilizando un LLM $\mathcal{M}$. Concretamente, se facilita al LLM un *prompt* $\text{p}_{\text{context}}$ que incluye la etiqueta semántica l_i y solicita una descripción breve sobre el objeto adaptada a la tarea en cuestión, es decir, que incluya información de las funciones que puede desempeñar el objeto:

$$T_i = \mathcal{M}(\text{p}_{\text{context}} \| l_i) \tag{7}$$

Posteriormente, se obtiene el descriptor semántico aplicando un modelo de *embedding* de frases $S : \text{Text} \to \mathbb{R}^d$ (p. ej. Sentence-BERT [47]) a la frase contextual T_i:

$$\mathbf{s}_i = \text{PCA}_{d'}(S(T_i)) \tag{8}$$

donde nuevamente PCA reduce la dimensión del descriptor. De este modo, el descriptor semántico del objeto refleja su funcionalidad en el mapa, sin depender de la semántica implícita capturada por el modelo de *embedding* de palabras para su clase semántica (véase la Figura 10).

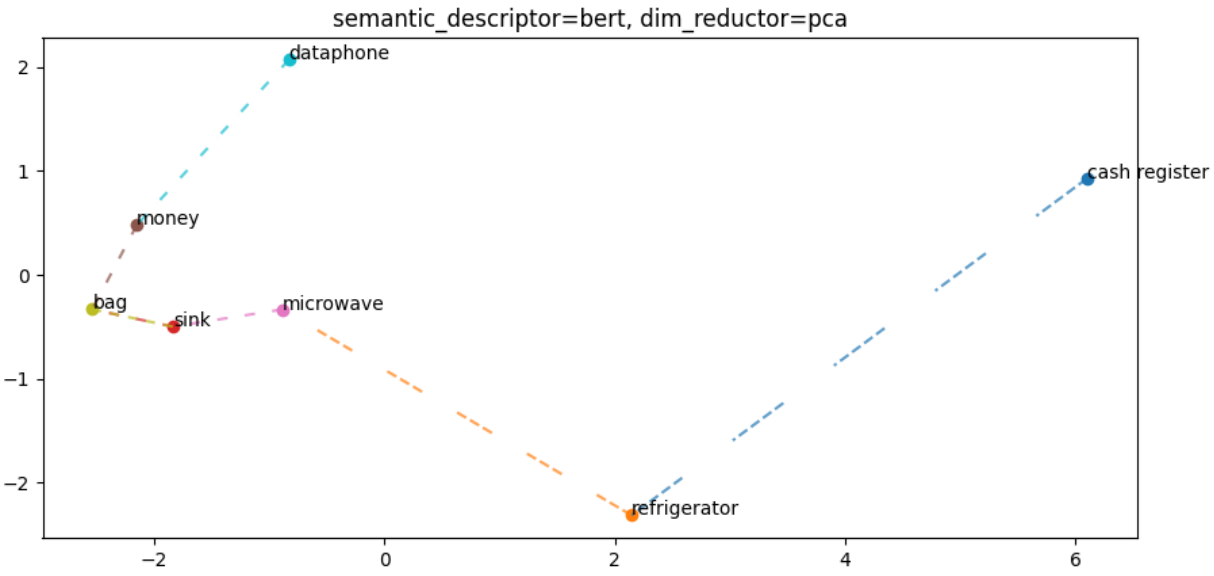


Figura 9: Representación en el espacio de 2 dimensiones de los descriptores semánticos obtenidos con BERT de las etiquetas semánticas "*dataphone*", "*money*", "*bag*", "*sink*", "*microwave*", "*refrigerator*" y "*cash register*". Cada descriptor semántico se representa con un color diferente. La línea del color de cada descriptor apunta al descriptor más cercano. Se comprueba que descriptores cercanos no implican una similitud funcional, p. ej. el descriptor más cercano al del objeto "*cash register*" es el del objeto "*refrigerator*".

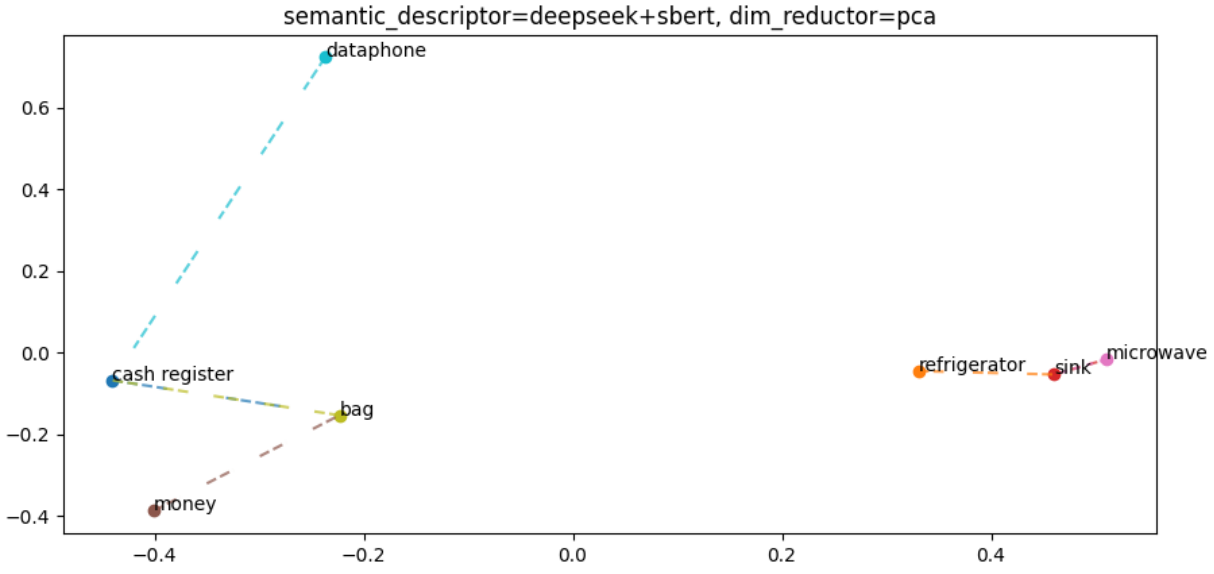


Figura 10: Representación en el espacio de 2 dimensiones de los descriptores semánticos obtenidos con el método de descriptores contextualizados para las mismas etiquetas semánticas que en la Figura 9. Descriptores cercanos implican objetos con funcionalidad similar.

Una vez calculada cada parte del descriptor, estas se normalizan por separado para garantizar que la escala de sus componentes sea consistente y comparable en el proceso de *clustering*. En particular, se aplica una normalización estándar de cada componente del descriptor —geométrico y semántico— de forma independiente. Dado el conjunto de descriptores geométricos de los objetos $\{\mathbf{c}_i \in \mathbb{R}^3\}_{i=1}^{N}$, se normaliza cada dimensión restando la media y dividiendo por la desviación típica:

$$\mathbf{c}'_i = \frac{\mathbf{c}_i - \boldsymbol{\mu}_c}{\boldsymbol{\sigma}_c} \tag{9}$$

donde $\boldsymbol{\mu}_c$ y $\boldsymbol{\sigma}_c$ son, respectivamente, la media y la desviación típica calculadas componente a componente sobre el conjunto completo de vectores $\mathbf{c}_i$. De manera análoga, los descriptores semánticos $\mathbf{s}_i \in \mathbb{R}^{d'}$ se normalizan según:

$$\mathbf{s}'_i = \frac{\mathbf{s}_i - \boldsymbol{\mu}_s}{\boldsymbol{\sigma}_s} \tag{10}$$

Esta normalización asegura que ninguna de las dos partes domine la distancia total por su escala, lo que permite combinar adecuadamente la información geométrica y semántica en la distancia utilizada por el algoritmo de *clustering*.

Con los descriptores normalizados, se procede al proceso de *clustering* para agrupar los objetos en lugares. En este trabajo se ha optado por utilizar DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) [14] porque permite detectar clústeres de forma y tamaño arbitrarios, además de identificar automáticamente regiones de baja densidad como ruido, lo cual resulta especialmente útil en mapas semánticos con estructuras complejas. Además, al no requerir el número de clústeres como entrada, DBSCAN se adapta de forma natural a escenarios en los que el número de lugares es desconocido a priori.

DBSCAN identifica regiones de alta densidad conectada, agrupando los puntos que cumplen ciertos criterios de proximidad y densidad, y marcando como ruido aquellos que no pertenecen a ninguna de estas regiones. El comportamiento del algoritmo depende fundamentalmente de dos parámetros: `epsilon` y `min_samples` (también llamado `min_points`). El parámetro `epsilon` determina el radio dentro del cual se buscan vecinos para cada punto, es decir, establece la distancia máxima a la que otro punto debe encontrarse para ser considerado parte del vecindario. De esta forma, el vecindario de un punto incluye a todos los puntos cuya distancia a él es menor o igual que `epsilon` según la métrica utilizada. Este concepto es clave en DBSCAN, ya que un punto solo se considera *núcleo* si su vecindario contiene al

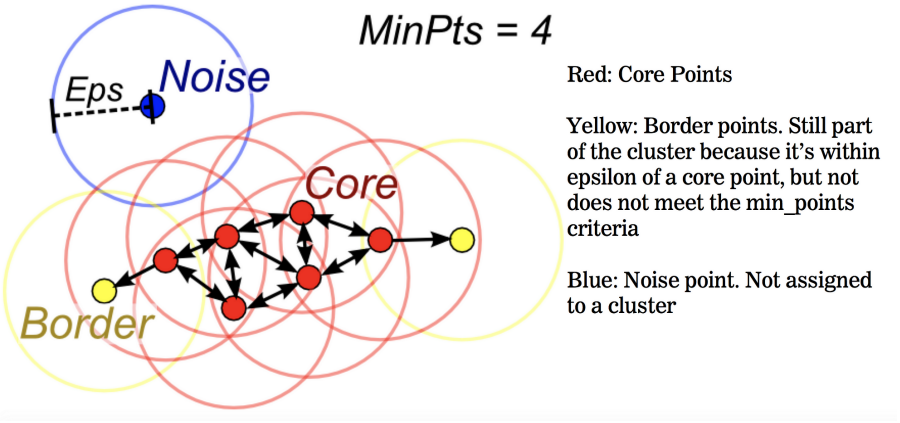


Figura 11: Imagen explicativa del algoritmo de DSBCAN, recalcando los parámetros `epsilon` y `min_samples`, además de los puntos *núcleo*, *frontera* y *ruido*.

menos `min_samples` puntos (incluyéndose a sí mismo). Los puntos núcleo actúan como anclas desde las que se expanden los clústeres, conectando con otros puntos núcleo o de frontera hasta formar regiones densas (véase la Figura 11[5]). Cuanto menor es `epsilon`, más restrictiva será la condición de vecindad, lo que tiende a generar clústeres más pequeños y a aumentar la cantidad de puntos marcados como ruido; por el contrario, un valor grande puede provocar la fusión de regiones distintas en un único clúster. Por su parte, el parámetro `min_samples` define el número mínimo de vecinos dentro del radio `epsilon` necesarios para formar una región densa. Cuanto mayor sea este valor, mayor densidad local se exige para consolidar un grupo, mientras que valores bajos permiten detectar agrupaciones más pequeñas y menos densas.

Además, para aplicar el algoritmo DBSCAN, este trabajo propone una distancia ponderada entre descriptores de objetos que da mayor peso a la proximidad espacial que a la similitud semántica. Esta elección evita que el algoritmo agrupe objetos que, aunque sean funcionalmente similares, se encuentren muy alejados entre sí en el espacio físico, lo que llevaría a segmentaciones poco coherentes desde el punto de vista geométrico. Dados dos descriptores $\mathbf{f}_i = [\mathbf{c}_i^\top, \mathbf{s}_i^\top]$ y $\mathbf{f}_j = [\mathbf{c}_j^\top, \mathbf{s}_j^\top]$, la distancia $d_{\text{clustering}}(\mathbf{f}_i, \mathbf{f}_j)$ utilizada en DBSCAN se define como:

$$d_{\text{clustering}}(\mathbf{f}_i, \mathbf{f}_j) = \|\mathbf{c}_i - \mathbf{c}_j\|_2 + \alpha_s \cdot \frac{\|\mathbf{s}_i - \mathbf{s}_j\|_2}{\sqrt{d'}} \tag{11}$$

donde:

- α_s es el peso asignado a la parte semántica.

[5] https://elutins.medium.com/dbscan-what-is-it-when-to-use-it-how-to-use-it-8bd506293818

- d' es la dimensión del descriptor semántico tras la reducción.

En este trabajo, los objetos identificados como ruido por su baja densidad local se tratan como clústeres individuales en la nueva segmentación, preservando así su singularidad.

3.2.2. Refinamiento

Aunque el agrupamiento descrito anteriormente resulta efectivo, en la práctica puede provocar dos efectos indeseados. En primer lugar, puede aparecer la *sobresegmentación*, generando clústeres muy pequeños que carezcan de significado en una aplicación real. Estos clústeres suelen corresponderse con grupos aislados de objetos muy similares entre sí, o incluso con la misma categoría, que podrían fusionarse en una región geométrica y semánticamente más coherente. En segundo lugar, también puede darse la *subsegmentación*, originando clústeres demasiado grandes y sin una relación semántica clara. En este caso se trata de zonas con alta densidad de objetos poco relacionados entre sí, donde los vínculos semánticos entre subconjuntos no son lo suficientemente fuertes para justificar una partición en clústeres diferentes. Para atajar ambos problemas se propone un refinamiento que divide los clústeres grandes con alta variabilidad semántica y dispersión geométrica, y fusiona los que están próximos y presentan gran similitud semántica.

División de clústers. En primer lugar, se seleccionan como candidatos a dividir aquellos clústeres cuyo llamado *valor de división* excede un umbral; este valor depende de la variabilidad semántica de los objetos y de la dispersión geométrica de sus centroides. Formalmente, dado un clúster que representa un lugar $P_k = \{\mathbf{o}_1, \ldots, \mathbf{o}_n\}$, el valor de división σ_k se define como:

$$\sigma_k = \lambda \cdot \mathrm{Var}_{\mathrm{sem}}(P_k) + (1 - \lambda) \cdot \mathrm{Dist}_{\mathrm{avg}}(P_k) \tag{12}$$

donde la varianza semántica se calcula como:

$$\mathrm{Var}_{\mathrm{sem}}(P_k) = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{s}_i - \bar{\mathbf{s}}\|_2^2 \tag{13}$$

y la dispersión geométrica media entre los objetos se calcula como:

$$\mathrm{Dist}_{\mathrm{avg}}(P_k) = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \|\mathbf{c}_i - \mathbf{c}_j\|_2 \tag{14}$$

Nótese que el parámetro $\lambda \in [0, 1]$ controla el compromiso entre variabilidad semántica y dispersión geométrica.

Los clústeres cuyo valor σ_k supera un cierto umbral θ_{split} se dividen. La división se realiza volviendo a aplicar DBSCAN sobre P_k utilizando los descriptores globales $\mathbf{f}_i$, lo que genera un nuevo conjunto de clústeres:

$$P_k \to \{P_k^{(1)}, \ldots, P_k^{(m)}\}, \quad m > 1 \tag{15}$$

De nuevo, los objetos etiquetados por DBSCAN como ruido pasan a formar clústeres individuales.

Para la re-aplicación de DBSCAN sobre el subconjunto P_k, se reduce el valor de `epsilon` (el radio de vecindad que define cuándo dos puntos se consideran vecinos) porque al segmentar únicamente la porción del mapa con alta variabilidad semántica se requiere un alcance más pequeño; de lo contrario, al usar el mismo `epsilon` que en el *clustering* global, esa región volvería a agruparse en un único clúster y no se revelarían sus agrupaciones internas. Al mismo tiempo, se fija `min_samples = 2`, porque se pretende que, en este espacio acotado, la presencia de dos objetos mutuamente vecinos (según el nuevo `epsilon`) ya satisfaga el requisito mínimo de densidad para definirse como núcleo; de este modo, incluso pares de puntos muy cercanos generan nuevos clústeres en lugar de quedar marcados como ruido, lo que permite capturar divisiones muy finas dentro de la región previamente *subsegmentada*.

Fusión de clústers. Para la fusión de clústeres se parte del conjunto $\mathcal{P} = \{P_k\}_{k \in K}$ obtenido en la etapa anterior. Primero se calcula el centro geométrico de cada clúster haciendo el promedio entre los centroides de sus objetos:

$$\bar{\mathbf{c}}_k = \frac{1}{|P_k|} \sum_{\mathbf{o}_i \in P_k} \mathbf{c}_i \tag{16}$$

Después, para cada par (P_k, P_l) se evalúa la distancia euclídea entre sus centros geométricos:

$$d_g(P_k, P_l) = \|\bar{\mathbf{c}}_k - \bar{\mathbf{c}}_l\|_2 \tag{17}$$

Si dicha distancia es inferior a un umbral $\theta_{\text{merge},g}$, el par se marca como candidato a fusión.

A continuación se comprueba la similitud entre clústeres marcados para la unión. Se calcula el centroide semántico de cada clúster promediando los descriptores semánticos de sus objetos:

$$\bar{\mathbf{s}}_k = \frac{1}{|P_k|} \sum_{\mathbf{o}_i \in P_k} \mathbf{s}_i \tag{18}$$

y se evalúa la similitud entre clústeres mediante el coseno:

$$\text{sim}_{\cos}(P_k, P_l) = \frac{\bar{\mathbf{s}}_k \cdot \bar{\mathbf{s}}_l}{\|\bar{\mathbf{s}}_k\|_2 \, \|\bar{\mathbf{s}}_l\|_2} \tag{19}$$

Si la similitud supera un umbral $\theta_{\text{merge},s}$, los clústeres se consideran semánticamente cercanos y se fusionan en un nuevo clúster P_m que contiene todos los objetos de P_k y P_l.

Este proceso iterativo de división y fusión se puede repetir hasta que se cumpla una condición predefinida, como un número máximo de iteraciones, la ausencia de cambios en los clústeres tras la ejecución de ambos pasos, o que ninguna operación de división o fusión sea ya aplicable.

3.2.3. Categorización basada en LLMs

Tras la ejecución de las etapas anteriores, cada clúster resultante $P_k \in \mathcal{P}$ contiene un subconjunto de objetos y define un lugar en el mapa. Para enriquecer estos lugares, se puede llevar a cabo una etapa de categorización que asigne a cada lugar una etiqueta breve y una descripción en lenguaje natural, por ejemplo.

Para ello, en este trabajo se hace uso de las capacidades de compresión del lenguaje natural de los LLMs. En concreto, se construye un *prompt* $\text{p}_{\text{categorize}}$ cuya instrucción solicita al modelo que genere tanto la etiqueta t_k como la descripción d_k del lugar, atendiendo a la información de los objetos presentes, que debe ser incluida. Formalmente, siendo $\mathcal{M}$ el LLM, para cada clúster P_k se realiza:

$$(t_k, d_k) = \mathcal{M}(\text{p}_{\text{categorize}} \| \{\mathbf{o}_i \in P_k\}) \tag{20}$$

donde $\|$ denota concatenación.

El *prompt* $\text{p}_{\text{categorize}}$ es en realidad una plantilla de prompt (*prompt template*) que contiene marcadores donde se inserta la información específica del problema en cada iteración, en este caso el listado de los objetos del lugar con sus centroides, dimensiones y clases semánticas. En términos simplificados, este *prompt* puede plantearse como: "Describe, proporcionando una etiqueta corta y una descripción más extendida, un lugar que contiene los siguientes objetos: [...]". El *prompt* involucrado en este proceso, $\text{p}_{\text{categorize}}$, se encuentra en el Apéndice B.1. En este *prompt*, al igual que en el resto de *prompts* empleados en este trabajo, se han empleado las directrices de ingeniería de *prompts* (*prompt engineering*) explicadas en el Apéndice C.

3.3. Segmentación y categorización basada en LLMs

La segunda propuesta para segmentar y categorizar un mapa semántico previamente construido consiste en utilizar un LLM para llevar a cabo, de manera conjunta, ambas etapas. En lugar de construir descriptores semánticos para cada objeto y posteriormente aplicar un algoritmo de *clustering*, esta aproximación propone que el LLM reciba un *prompt* que detalle la tarea a realizar junto con el mapa semántico completo, de manera que el modelo pueda obtener directamente tanto la segmentación en lugares como su categorización correspondiente.

Con este propósito, se construye un prompt, $\mathrm{p}_{\text{segment\_categorize}}$, cuya instrucción principal guía al modelo en la segmentación del entorno. En particular, se le indica que las agrupaciones deben formarse no solo según la proximidad geométrica de los objetos, sino también atendiendo a su similitud semántica en términos de funcionalidad. Asimismo, el *prompt* especifica que para cada grupo segmentado debe asignarse una etiqueta concisa, junto con una breve descripción que refleje el uso o propósito del espacio, en función de los objetos que contiene. Formalmente, la operación conjunta de segmentación y categorización de lugares realizada por el LLM $\mathcal{M}$ puede definirse como:

$$(\mathcal{P}, \mathcal{P}_{\text{desc}}) = \mathcal{M}(\mathrm{p}_{\text{segment\_categorize}} \| \mathbf{m}) \tag{21}$$

donde:

- $\mathcal{P} = P_1, \ldots, P_K$ es el conjunto de lugares segmentados, con $P_k = \{\mathbf{o}_j \mid j \in J_k\}$.
- $\mathcal{P}_{\text{desc}} = \{(t_1, d_1), \ldots, (t_K, d_K)\}$ es el conjunto de etiquetas t_k y descripciones d_k asignadas a cada lugar P_k.

Al igual que en los procedimientos previos, el *prompt* $\mathrm{p}_{\text{segment\_categorize}}$ actúa como una plantilla donde se inserta la información del mapa semántico considerado, $\mathbf{m}$. De forma simplificada, este *prompt* puede plantearse como: "Segmenta el siguiente mapa semántico en lugares atendiendo a los siguientes criterios: [...]; para cada lugar proporciona la lista de objetos que contiene, además de una etiqueta concisa y una descripción [...]". Su versión completa se encuentra en el Apéndice B.2; al igual que el resto de *prompts* de este trabajo, sigue las directrices del Apéndice C sobre ingeniería de *prompts*.

Esta segunda estrategia ofrece varias ventajas en comparación con el enfoque basado en *clustering* de descriptores semánticamente enriquecidos. En primer lugar, se aprovechan las

capacidades de razonamiento de los LLMs actuales para generar segmentaciones considerando el contexto global del mapa. Este contexto permite al modelo interpretar y procesar la información de forma más profunda y coherente, teniendo en cuenta las relaciones espaciales y funcionales entre los objetos. En segundo lugar, se elimina la necesidad de ajustar manualmente múltiples hiperparámetros en cada fase del proceso, como el peso semántico en la distancia de *clustering* o los umbrales de división o fusión, ya que estos quedan implícitamente integrados en la operación de inferencia del LLM. Además, este enfoque aporta una mayor flexibilidad, por la posibilidad de adaptar la forma de agrupar los objetos modificando simplemente la instrucción en lenguaje natural proporcionada al modelo, con la posibilidad de adaptar el *prompt* en función del tipo de mapas que se pretende segmentar y categorizar.

No obstante, esta estrategia también presenta algunas desventajas importantes. Por un lado, este enfoque requiere disponer de un LLM con las capacidades suficientes para procesar el mapa completo, lo que conlleva una elevada demanda de recursos computacionales o, en su defecto, el acceso a un modelo propietario a través de una API de pago. El enfoque basado en *clustering* de descriptores contextualizados también recurre a un LLM para obtener una frase que describa la funcionalidad de cada objeto, aunque en este caso puede utilizarse un modelo mucho más pequeño debido a la simplicidad de la tarea. Por otro lado, la dependencia de la calidad del *prompt* es alta; una formulación ambigua o incompleta de la instrucción puede llevar a resultados subóptimos o inesperados, requiriendo un proceso iterativo de ajuste. Además, los LLMs están sujetos a alucinaciones, lo que significa que pueden generar información incorrecta o inventada que afecte negativamente la precisión y fiabilidad de las segmentaciones obtenidas.

3.4. Implementación

Todos los procesos para la segmentación y categorización de lugares se han implementado y automatizado en un proyecto de Python, cuyo código completo está disponible en el Apéndice A.1. Para garantizar la correcta ejecución del proyecto sin interferencias, es recomendable crear y utilizar un entorno virtual de Python, donde se pueden instalar fácilmente todas las dependencias necesarias especificadas en el fichero `requirements.txt`, utilizando la herramienta `pip`. De este modo, se garantiza la reproducibilidad de los experimentos y la compatibilidad de las versiones de cada paquete utilizado.

El *script* principal del proyecto es `main_places.py`. Su función es procesar los distintos mapas semánticos de los conjuntos de datos empleados, aplicando los algoritmos de segmentación y categorización previamente descritos. De este modo, divide cada mapa en lugares significativos y les asigna etiquetas semánticas en lenguaje natural utilizando vocabulario abierto. Este *script* ha sido diseñado con un enfoque flexible, lo que permite ejecutar de forma independiente las distintas etapas del proceso según los argumentos especificados. Además, todos los datos generados, así como las interacciones con los LLMs, se almacenan en disco para facilitar su análisis posterior y su reutilización en fases subsiguientes.

El proyecto también incorpora varios *scripts* auxiliares que han resultado fundamentales para el desarrollo y la evaluación de los métodos propuestos:

- `inspect_clusters.py`: ofrece herramientas de visualización en 2D y 3D para explorar segmentaciones y categorizaciones de los mapas semánticos, facilitando la detección de aciertos y limitaciones en cada enfoque.

- `inspect_semantics.py`: permite representar en 2D o 3D los descriptores geométrico-semánticos calculados para cada objeto del método de la Sección 3.2. Esta visualización resultó fundamental para evaluar el impacto del contexto en la representatividad y utilidad de dichos descriptores.

- `check_places_ground_truth.py`: verifica la validez del *ground truth* de segmentación de lugares para cada mapa utilizado, comprobando que cada objeto de mapa semántico se encuentra en un lugar y visualizando las segmentaciones para asegurar su correctitud. El *ground truth* empleado para evaluar las segmentaciones se describe en detalle en el Capítulo 5.

- `evaluate_places.py`: compara las segmentaciones generadas por los distintos métodos con las segmentaciones de referencia, evaluando su calidad mediante métricas de *clustering*. Los resultados obtenidos en la evaluación así como las métricas se presentan en el Capítulo 5.

Junto con los *scripts* principales, el proyecto se organiza en una serie de paquetes especializados que implementan funcionalidades clave de forma modular:

- `embedding`: contiene utilidades para trabajar con modelos de *embedding* de palabras y frases, integrando técnicas de Procesamiento de Lenguaje Natural (*Natural Language Processing*, NLP).
- `llm`: incluye herramientas para la interacción con LLMs y LVLMs de última generación, tanto ejecutados en local como accesibles a través de APIs propietarias.
- `prompt`: define clases para la construcción y gestión de los *prompts* empleados en las distintas etapas de los métodos propuestos, disponibles en el Apéndice B.
- `semantic`: ofrece utilidades específicas para la segmentación semántica de lugares, incluyendo algoritmos de *clustering*, reducción de dimensionalidad y generación de descriptores semánticos.
- `show`: proporciona herramientas para la visualización y generación de gráficos, tablas y otros elementos ilustrativos.
- `utils`: agrupa funciones auxiliares para la manipulación de archivos, estructuras de datos y otras operaciones de soporte.
- `voxeland`: contiene herramientas para procesar los mapas semánticos en el formato empleado en el proyecto.

4

Inferencia de relaciones semánticas

En este capítulo se presenta la metodología desarrollada en este TFM para inferir relaciones semánticas entre los objetos de un mapa semántico previamente construido. La Sección 4.1 formaliza esta tarea, ampliando la formulación de la segmentación de lugares introducida en la Sección 3.1, y define los distintos tipos de relaciones semánticas entre objetos considerados en este trabajo (véase la Sección 4.1.1). A continuación, se describen dos enfoques basados en modelos de gran escala para inferir relaciones semánticas abiertas entre pares de objetos, superando las restricciones impuestas por los métodos tradicionales basados en predicados cerrados. La Sección 4.2 presenta un método que utiliza un LLM para inferir relaciones a partir de descripciones textuales de los objetos, que incluyen atributos como clase semántica, posición y tamaño. Por su parte, la Sección 4.3 incorpora un contexto visual adicional mediante el uso de LVLMs, que reciben como entrada una imagen donde aparece el par de objetos en la escena, además de la información textual de dichos objetos en el mapa. Finalmente, la Sección 4.4 aborda los detalles de implementación del proceso de inferencia de relaciones, en el contexto del código desarrollado en este proyecto.

4.1. Formulación del problema

Sea $\mathbf{m}$ el mapa semántico introducido en la Sección 3.1, es decir:

$$\mathbf{m} = \{\mathbf{o}_1, \ldots, \mathbf{o}_N\}, \qquad \mathbf{o}_i = (c_i, d_i, l_i, \mathbf{p}_i^{\text{object}}), \tag{22}$$

donde cada $\mathbf{o}_i$ representa un objeto del mapa con su localización c_i, dimensiones d_i, etiqueta semántica l_i y propiedades adicionales $\mathbf{p}_i^{\text{object}}$.

En este contexto, la tarea de inferencia de relaciones semánticas consiste en, dado un mapa semántico previamente construido, identificar un conjunto de relaciones dirigidas entre pares de objetos, basadas tanto en la disposición espacial de estos objetos como en su significado semántico. Formalmente, una función de inferencia de relaciones $\varphi(\cdot)$ toma un mapa semántico $\mathbf{m}$ como entrada y produce un conjunto de relaciones $\mathcal{R}$:

$$\mathcal{R} = \varphi(\mathbf{m}) = \left\{ \mathbf{r}_{ij}^{(k)} \;\middle|\; i, j \in \{1, \ldots, N\},\ i \neq j,\ k \in \mathcal{K}_{ij} \right\}, \tag{23}$$

donde $\mathcal{K}_{ij}$ es el conjunto de índices de las relaciones existentes entre el objeto $\mathbf{o}_i$ y el objeto $\mathbf{o}_j$, y cada relación individual se modela como:

$$\mathbf{r}_{ij}^{(k)} = \left(\mathbf{o}_i,\ t_{ij}^{(k)},\ p_{ij}^{(k)},\ \mathbf{o}_j \right), \tag{24}$$

con:

- $\mathbf{o}_i$ el objeto origen de la relación.
- $\mathbf{o}_j$ el objeto destino de la relación.
- $t_{ij}^{(k)}$ el tipo de relación.
- $p_{ij}^{(k)}$ un predicado semántico procedente de un vocabulario abierto.

Según esta formulación, entre dos objetos del mapa pueden establecerse múltiples relaciones semánticas, cada una asociada a un tipo de relación y predicado distintos. El número total de relaciones dirigidas entre un par de objetos $(\mathbf{o}_i, \mathbf{o}_j)$ está determinado por la cardinalidad del conjunto $\mathcal{K}_{ij}$, es decir, $|\mathcal{K}_{ij}|$.

El objetivo de la función de inferencia de relaciones $\varphi(\cdot)$ es generar un conjunto de relaciones que sean geométricamente plausibles —a partir de las posiciones y dimensiones de los objetos implicados— y, al mismo tiempo, que el tipo de relación $t_{ij}^{(k)}$ y su predicado asociado $p_{ij}^{(k)}$ sean semánticamente coherentes con las etiquetas semánticas l_i de los objetos.

4.1.1. Tipos de relaciones consideradas

En este TFM se han considerado diferentes tipos de relaciones semánticas entre objetos, en concordancia con las clasificaciones propuestas por trabajos recientes del estado del arte. A continuación se describen brevemente estos tipos, ilustrados con ejemplos en la Figura 12:

- Espaciales: describen la disposición geométrica entre objetos, como “encima de”, “debajo de”, “dentro de” o “cerca de”. Idealmente, para inferirlas se necesita únicamente la localización y el tamaño de los objetos involucrados, sin necesidad de conocimiento adicional sobre su naturaleza física.

- Estructurales: representan vínculos físicos, de ensamblaje o de dependencia material entre objetos, como “parte de”, “unido a”, “soportado por” o “conectado a”. A diferencia de las relaciones espaciales, que describen únicamente configuraciones geométricas relativas, las relaciones estructurales implican una conexión física o funcional directa entre los elementos. Este tipo de relaciones ha sido utilizado en representaciones simbólicas de conocimiento, como las desarrolladas en KnowRob [57] o en el enfoque propuesto por Armeni et al. [3].

- Funcionales: expresan posibles usos, funciones o formas de interacción compartidas entre objetos, como “se utiliza con”, “necesario para utilizar”, o “permite manipular”. Este tipo de relaciones es especialmente relevante en enfoques basados en *affordances*, que buscan modelar las posibles acciones que el entorno ofrece a un agente, como USA-Net [5].

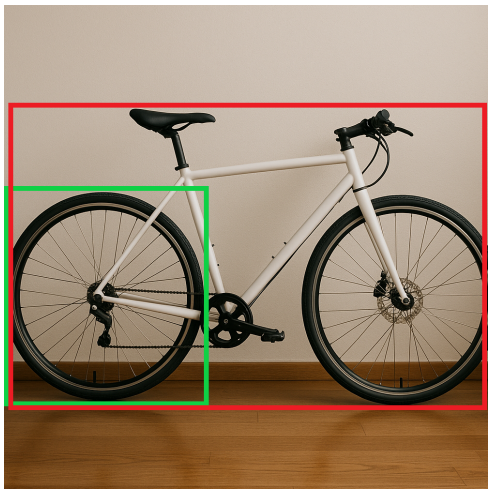
- Causales: representan interacciones entre objetos en las que uno provoca un cambio de estado o un efecto sobre otro, como “activa”, “bloquea”, “abre” o “enciende”. Este tipo de relaciones permite modelar las dinámicas causa-efecto en el entorno.

La clasificación de las relaciones en distintos tipos permite filtrar y seleccionar aquellas que resulten más relevantes según el uso previsto del mapa. Si bien el tipo de relación $t_{ij}^{(k)}$ se elige dentro de un vocabulario finito predefinido, el predicado específico $p_{ij}^{(k)}$ asociado a cada relación no está limitado a un conjunto cerrado de categorías, por lo que puede considerarse de vocabulario abierto.

4.2. Inferencia de relaciones basada en LLMs

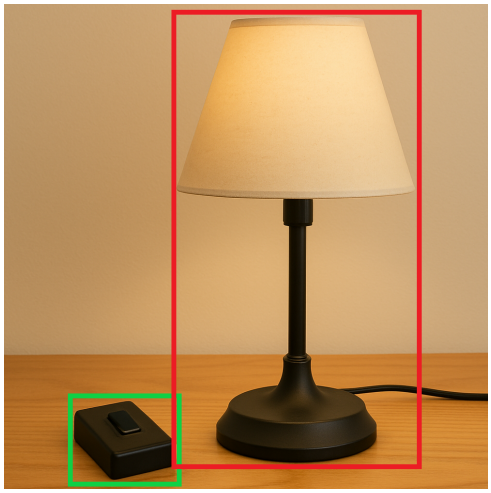
El proceso de inferencia de relaciones semánticas desarrollado en este trabajo se inspira en el enfoque propuesto por ConceptGraphs [18], y se basa en el uso de un LLM para identificar relaciones de vocabulario abierto entre pares de objetos en el mapa. Para reducir el número

(a) Relación espacial: el libro “encima de” la mesa.



(b) Relación estructural: la rueda “parte de” la bicicleta.

(c) Relación funcional: el ratón “se utiliza con” el teclado.



(d) Relación causal: el interruptor “enciende” la lámpara.

Figura 12: Ejemplos ilustrativos de diferentes tipos de relaciones semánticas entre objetos.

de consultas y centrarse en relaciones potencialmente relevantes, solo se consideran como candidatos aquellos pares de objetos cuya proximidad espacial en el mapa semántico sugiere una posible interacción significativa.

De esta forma, primero se identifican los pares de objetos $(\mathbf{o}_i, \mathbf{o}_j)$ cuya proximidad espacial sugiere la existencia de una posible relación semántica. Para ello, se utiliza una condición de cercanía basada en la distancia mínima entre sus *bounding boxes* alineadas con los ejes (*Axis-Aligned Bounding Boxes*, AABB). Esta distancia se calcula considerando los centros $c_k = (x_k, y_k, z_k)$ y las dimensiones $d_k = (d_{xk}, d_{yk}, d_{zk})$ de las cajas de cada objeto $\mathbf{o}_k$. La distancia mínima γ_{ij} entre los objetos $\mathbf{o}_i$ y $\mathbf{o}_j$ se define como:

$$\gamma_{ij} = \left\| \text{máx}\left(\mathbf{0}, |c_i - c_j| - \frac{d_i}{2} - \frac{d_j}{2}\right) \right\|_2 \tag{25}$$

donde las operaciones se aplican por componente, y el vector $\mathbf{0}$ garantiza que no se obtengan distancias negativas en caso de solapamiento entre las cajas. Un par de objetos se considera candidato si la distancia γ_{ij} es inferior o igual a un umbral predefinido δ_{max}.

Para cada par de objetos candidatos se construye un *prompt* $\text{p}_{\text{relationship\_llm}}$ que se proporciona al modelo con el objetivo de inferir las relaciones existentes entre ellos. La instrucción contenida en el *prompt* solicita al modelo que genere una o más relaciones dirigidas desde $\mathbf{o}_i$ hacia $\mathbf{o}_j$, y desde $\mathbf{o}_j$ hacia $\mathbf{o}_i$, cada una definida por un predicado semántico $p_{ij}^{(k)}$ y un tipo de relación $t_{ij}^{(k)}$ perteneciente a una lista predefinida. Este *prompt* actúa como una plantilla en la que, en cada iteración, se inserta la información específica de ambos objetos: sus etiquetas semánticas (l_i, l_j), la localización de sus centroides (c_i, c_j) y sus dimensiones (d_i, d_j). Formalmente, ante una consulta con este *prompt*, el LLM $\mathcal{M}$ devuelve el conjunto de relaciones semánticas dirigidas detectadas entre ambos objetos en ambas direcciones:

$$\left\{\mathbf{r}_{ij}^{(k)}\right\}_{k\in\mathcal{K}_{ij}} \cup \left\{\mathbf{r}_{ji}^{(k)}\right\}_{k\in\mathcal{K}_{ji}} = \mathcal{M}(\text{p}_{\text{relationship\_llm}} \| l_i, l_j, c_i, c_j, d_i, d_j) \tag{26}$$

donde $\mathcal{K}_{ij}$ y $\mathcal{K}_{ji}$ representan los conjuntos de índices de las relaciones inferidas desde $\mathbf{o}_i$ hacia $\mathbf{o}_j$ y viceversa, respectivamente. El *prompt* utilizado en este proceso se detalla en el Apéndice B.3 y ha sido diseñado siguiendo las directrices de ingeniería de *prompts* descritas en el Apéndice C.

Por ejemplo, supóngase que dos objetos del mapa semántico, $\mathbf{o}_7$ y $\mathbf{o}_{19}$, han sido seleccionados como candidatos a partir de su proximidad espacial, con categorías semánticas “cafetera”

(*coffee machine*) y "taza" (*cup*), respectivamente. El *prompt* utilizado en el proceso de inferencia de relaciones, $p_{relationship\_llm}$, se completaría con la información específica de ambos objetos:

```
[...]
- object1_id: "obj7"
- object1_name: "coffee machine"
- centroid1: [3.05, 1.82, 0.95]
- size1: [0.42, 0.38, 0.90]
- object2_id: "obj19"
- object2_name: "cup"
- centroid2: [3.10, 2.20, 0.94]
- size2: [0.10, 0.10, 0.12]
[...]
```

El fragmento anterior representa únicamente la parte dinámica del *prompt* $p_{relationship\_llm}$, es decir, no incluye la instrucción principal ni los ejemplos explicativos incorporados en la plantilla, entre otros. Ante esta petición, el LLM $\mathcal{M}$ devolvería el conjunto de relaciones semánticas dirigidas entre los objetos, en ambas direcciones, siguiendo el formato especificado en el *prompt* para que pueda ser interpretado automáticamente por el programa desarrollado. Un posible resultado de esta inferencia sería:

```
[   {
        "source_id": "obj19",
        "source": "cup",
        "target_id": "obj7",
        "target": "coffee machine",
        "type": "functional",
        "predicate": "is filled by"
    },
    {
        "source_id": "obj7",
        "source": "coffee machine",
        "target_id": "obj19",
        "target": "cup",
        "type": "functional",
        "predicate": "dispenses into"
    },
```

```
    {
        "source_id": "obj7",
        "source": "coffee machine",
        "target_id": "obj19",
        "target": "cup",
        "type": "structural",
        "predicate": "has resting platform for"
    },
    {
        "source_id": "obj19",
        "source": "cup",
        "target_id": "obj7",
        "target": "coffee machine",
        "type": "spatial",
        "predicate": "is under"
    }, [...]
]
```

Nótese que el LLM devuelve las relaciones de acuerdo con la notación explicada. Cada relación $\mathbf{r}_{ij}^{(k)}$ está caracterizada por su objeto de origen $\mathbf{o}_i$ (`source_id` en la respuesta), su objeto de destino $\mathbf{o}_j$ (`target_id`), el tipo de relación $t_{ij}^{(k)}$ (`type`) y el predicado de vocabulario abierto $p_{ij}^{(k)}$ (`predicate`) generado por el modelo en lenguaje natural.

4.3. Inferencia de relaciones basada en LVLMs

La inferencia de relaciones basada en LVLMs sigue el mismo planteamiento general que la basada en LLMs, pero incorpora una ventaja fundamental: la inclusión de información visual en la entrada del modelo. Esta capacidad permite al modelo interpretar las relaciones entre objetos de forma más rica y precisa, al tener en cuenta no solo posiciones, dimensiones y categorías semánticas, sino también la apariencia visual de ambos objetos dentro de la escena.

De manera análoga al caso de LLMs, primero se determina qué pares de objetos $(\mathbf{o}_i, \mathbf{o}_j)$ pueden participar en una relación semántica. Para ello, se aplica la misma métrica de proximidad basada en la distancia mínima entre las cajas delimitadoras alineadas con los ejes, definida en la Ecuación (25). Si $\gamma_{ij} \leq \delta_{\max}$, el par de objetos es marcado como candidato.

Una vez seleccionados los pares de objetos candidatos, se construye un *prompt* específi-

co para esta variante, denotado como $\mathrm{p}_{\text{relationship\_lvlm}}$, que sigue una estructura muy similar al *prompt* $\mathrm{p}_{\text{relationship\_llm}}$ descrito anteriormente, con una diferencia fundamental. En este caso, el *prompt* no solo incluye la información textual de los objetos —como su clase semántica (l_i, l_j), posición (c_i, c_j) y dimensiones (d_i, d_j)—, sino que también incorpora una imagen en la que ambos objetos aparecen de forma clara dentro de la escena.

La selección de una imagen en la que ambos objetos sean claramente visibles es un paso clave para asegurar una inferencia de relaciones efectiva mediante LVLMs. Si se proporciona al modelo una imagen en la que los objetos no se observan con claridad —por ejemplo, por estar parcialmente ocultos o fuera del campo de visión—, el modelo no podrá aprovechar la información visual, y el resultado será similar al obtenido mediante un LLM dotado de puramente textual. Para elegir la imagen más adecuada, denotada como I^*, este trabajo propone maximizar la visibilidad conjunta de los objetos $\mathbf{o}_i$ y $\mathbf{o}_j$ en su proyección bidimensional sobre cada imagen disponible:

$$I^* = \arg\max_{I \in \mathcal{I}} \Big[\text{AreaProyectada}(\mathbf{o}_i, I) + \text{AreaProyectada}(\mathbf{o}_j, I) \Big]. \tag{27}$$

donde $\mathcal{I}$ representa el conjunto de imágenes disponibles de la escena, y $\text{AreaProyectada}(\mathbf{o}_k, I)$ es una función que devuelve el área del objeto $\mathbf{o}_k$ que se proyecta, y por tanto es visible, en la imagen I.

Los conjuntos de datos utilizados para la validación en este trabajo proporcionan tanto las nubes de puntos anotadas por objeto como vídeos reales de la escena, junto con las poses de cámara correspondientes a cada *frame*. Dado este contexto, se define la función $\text{AreaProyectada}(\mathbf{o}_i, I)$ como el número de puntos de la nube correspondiente al objeto $\mathbf{o}_i$ que se proyectan dentro del campo de visión del *frame* I del vídeo. Esta estrategia se basa en la suposición razonable de que, cuanto mayor sea el número de puntos proyectados en la imagen, mayor será el área aparente del objeto y, por tanto, su visibilidad en dicho *frame*. Cabe destacar que, aunque en este trabajo se ha definido $\text{AreaProyectada}(\cdot)$ en función del número de puntos proyectados sobre imágenes reales, este cálculo podría adaptarse fácilmente a otros entornos. Por ejemplo, si la escena estuviera disponible en un simulador 3D, sería posible ajustar de forma programática la pose de la cámara para obtener una vista óptima en la que ambos objetos aparezcan claramente visibles, y capturar una imagen sintética desde dicha posición.

Una vez se dispone de la imagen óptima I^*, esta se incluye en el *prompt* $\mathrm{p}_{\text{relationship\_lvlm}}$,

Figura 13: Imagen óptima de ejemplo I^* incluida en el *prompt* del método de inferencia de relaciones basado en LVLMs.

junto con la información textual de los objetos entre los que se desea inferir relaciones, que se pasa al LVLM $\mathcal{M}_\mathcal{V}$ Formalmente, la Ecuación (26) se transforma en:

$$\left\{\mathbf{r}_{ij}^{(k)}\right\}_{k\in\mathcal{K}_{ij}} \cup \left\{\mathbf{r}_{ji}^{(k)}\right\}_{k\in\mathcal{K}_{ji}} = \mathcal{M}_\mathcal{V}(\mathrm{p}_{\text{relationship\_lvlm}} \| l_i, l_j, c_i, c_j, d_i, d_j \| I^*) \tag{28}$$

El *prompt* involucrado en este proceso, $\mathrm{p}_{\text{relationship\_lvlm}}$, se encuentra disponible en el Apéndice B.4 y sigue las directivas de ingeniería de *prompts* del Apéndice C.

Por ejemplo, supóngase que se desea inferir la relación entre dos objetos de un mapa semántico, $\mathbf{o}_{15}$ y $\mathbf{o}_{31}$, correspondientes a las categorías "mesa" (*table*) y "silla de escritorio" (*desk chair*). El *prompt* $\mathrm{p}_{\text{relationship\_lvlm}}$ se completaría con la imagen óptima I^* en la que ambos objetos aparezcan claramente visibles (que podría ser la de la Figura 13), además de la información textuual de ambos objetos (categoría semántica, posición y dimensiones):

```
[...]
- object1_id: "obj15"
- object1_name: "table"
- centroid1: [-3.76, 4.33, 0.86]
- size1: [0.44, 0.27, 0.13]
- object2_id: "obj31"
```

```
- object2_name: "desk chair"
- centroid2: [-4.11, 4.86, 0.72]
- object2_size: [0.40, 0.36, 0.49]
- image_path: "scene_deskchair_frame00123.jpg"
[...]
```

Ante esta entrada multimodal, el LVLM generaría una respuesta con el mismo formato que en el método basado en LLMs, conteniendo las relaciones dirigidas entre ambos objetos, por ejemplo:

```
[   {
        "source_id": "obj31",
        "source": "desk chair",
        "target_id": "obj15",
        "target": "table",
        "type": "functional",
        "predicate": "is used with"
    },
    {
        "source_id": "obj15",
        "source": "table",
        "target_id": "obj31",
        "target": "desk chair",
        "type": "spatial",
        "predicate": "is in front of"
    }, [...]
]
```

En la Sección 5.4 se presenta un análisis comparativo de ambos métodos de inferencia de relaciones, destacando sus respectivas ventajas, limitaciones y el tipo de relaciones que son capaces de capturar con mayor precisión.

4.4. Implementación

Al igual que en el proceso de segmentación y categorización de lugares, la inferencia de relaciones entre objetos está completamente automatizada en el proyecto Python descrito en el Apéndice A.1. En este caso, el *script* principal es `main_relationships.py`, encargado de

procesar cada mapa semántico de un conjunto de datos de entrada para inferir relaciones relevantes entre pares de objetos, aplicando los algoritmos descritos anteriormente. Este *script*, al igual que `main_places.py`, ha sido diseñado con un enfoque modular y flexible, lo que permite ejecutar cada fase del proceso de forma independiente según los argumentos especificados por el usuario. Para llevar a cabo esta tarea, `main_relationships.py` se apoya en los mismos paquetes clave del proyecto:

- `llm`: para realizar llamadas a LLMs y LVLMs, tanto de código abierto como de servicios propietarios.
- `prompt`: encargado de encapsular y generar los *prompts* utilizados en ambos métodos, descritos en detalle en el Apéndice B.
- `utils`: que proporciona utilidades para la manipulación de datos, gestión de archivos y otras funciones auxiliares.
- `voxeland`: para la carga, interpretación y gestión de los mapas semánticos en el formato requerido por el proyecto.

5 Validación

En este capítulo se evalúan y validan los métodos propuestos en este TFM, analizando además cómo la incorporación de información topológica en un mapa semántico —dando lugar a un mapa semántico-topológico— puede mejorar el rendimiento de un robot móvil que utiliza un LLM como motor de razonamiento. En primer lugar, la Sección 5.1 detalla la configuración experimental utilizada, abordando tanto la selección de mapas semánticos y conjuntos de datos (véase la Sección 5.1.1), como los detalles de implementación (véase la Sección 5.1.2) y las métricas empleadas para evaluar cuantitativamente los métodos de segmentación de lugares (véase la Sección 5.1.3). A continuación, la Sección 5.2 presenta un análisis cuantitativo de los métodos propuestos para segmentar lugares, utilizando las métricas previamente descritas. Posteriormente, la Sección 5.3 evalúa cualitativamente la categorización de lugares basada en LLMs, independientemente del método de segmentación empleado. La Sección 5.4 analiza cualitativamente las distintas alternativas propuestas para la inferencia de relaciones semánticas entre objetos. Finalmente, la Sección 5.5 evalúa la capacidad de un LLM para responder a preguntas del usuario sobre una escena, comparando dos escenarios: uno en el que el modelo dispone únicamente de un mapa semántico convencional (con información de objetos, posiciones, tamaños y clases), y otro en el que se le proporciona un mapa semántico-topológico enriquecido con información estructurada sobre lugares y relaciones, construido mediante las técnicas presentadas en este trabajo.

5.1. Configuración experimental

Los experimentos desarrollados en este trabajo se han realizado sobre un conjunto de mapas semánticos derivados de dos conjuntos de datos 3D ampliamente utilizados, ScanNet y SceneNN. Para la implementación de los métodos propuestos, se han empleado principalmente modelos de código abierto, con la excepción de algunos LLMs de familia Gemini, desarrollados

Figura 14: Ejemplos de escenas del conjunto de datos ScanNet [11] con segmentación semántica por objeto. Cada color representa un objeto distinto con una etiqueta semántica asociada. Las imágenes muestran la diversidad estructural y la riqueza en tipos de objetos presentes en el conjunto de datos.

por Google. La evaluación de las segmentaciones de lugares se ha llevado a cabo mediante métricas estándar en la literatura de *clustering*, que permiten medir la calidad de las agrupaciones generadas.

5.1.1. Conjuntos de datos y mapas semánticos

Para la evaluación de los métodos propuestos se han utilizado 10 escenas extraídas de dos repositorios ampliamente reconocidos y de acceso público: ScanNet [11] y SceneNN [20]. El uso combinado de ambos conjuntos de datos introduce una mayor heterogeneidad en los mapas semánticos, tanto en el tamaño de las escenas como en el número y variedad de objetos presentes. Esta diversidad permite analizar con mayor rigor la robustez y capacidad de generalización de los métodos propuestos en contextos variados y potencialmente desafiantes.

ScanNet (véase la Figura 14) es un conjunto de datos de gran escala compuesto por reconstrucciones 3D anotadas de escenas interiores, junto con datos RGB-D, anotaciones semánticas y poses de cámara para cada una de las casi 1 500 escenas disponibles. De este conjunto, se han seleccionado cinco escenas de apartamentos de una sola planta (`scene_0000`, `scene_0101`, `scene_0392`, `scene_0515` y `scene_0673`), todas caracterizadas por contener un elevado número de objetos.

Figura 15: Ejemplos de escenas del conjunto de datos SceneNN [20] con segmentación semántica por objeto. Cada color identifica un objeto distinto etiquetado manualmente. La imagen refleja la variedad de configuraciones espaciales, categorías de objetos y densidad de escenas presentes en este conjunto.

Por su parte, SceneNN (véase la Figura 15) también ofrece reconstrucciones 3D anotadas de espacios interiores junto con datos RGB-D, poses de cámara y anotaciones semánticas de referencia. En este caso, se han seleccionado cinco escenas de menor tamaño en comparación con las de ScanNet, provenientes de diferentes entornos: una sala de estar (`011`), una oficina (`030`, `078`, `086`) y un dormitorio (`096`).

Dado que cada conjunto de datos proporciona la información en un formato diferente, fue necesario unificar la representación para poder ejecutar y evaluar de forma coherente los métodos propuestos en este trabajo. Esta tarea se llevó a cabo utilizando los repositorios de código descritos en el Apéndice A.2, a los que este TFM contribuye de manera significativa. Estos repositorios permiten construir mapas semánticos en un formato unificado a partir del *ground truth* de cada conjunto de datos. Además, ofrecen herramientas para visualizar las escenas completas, incluyendo los objetos segmentados, sus nubes de puntos asociadas y las imágenes RGB-D de las secuencias originales. También hacen uso de las poses de cámara para proyectar cada objeto sobre todos los *frames* disponibles, lo que resulta fundamental para algunas de las operaciones desarrolladas, como la selección de la imagen óptima para inferir la relación entre dos objetos. Asimismo, integran otras utilidades requeridas por los métodos implementados, como filtrado, transformación o gestión estructurada de los datos.

Con el objetivo de evaluar cuantitativamente la segmentación de lugares, se ha creado un conjunto de segmentaciones y categorizaciones de referencia para cada mapa semántico $\mathbf{m}^{(j)}$ de los conjuntos de datos. Este conjunto de referencia creado se denota como $\left\{\mathcal{P}_{\text{GT}}^{(j,\ell)}\right\}_{\ell=1}^{s_j}$, donde cada elemento $\mathcal{P}_{\text{GT}}^{(j,\ell)}$ representa una posible segmentación y categorización válida del mapa j y se define como:

$$\mathcal{P}_{\text{GT}}^{(j,\ell)} = \left\{(P_k, t_k, d_k) \mid k \in K^{(j,\ell)}\right\}. \tag{29}$$

En esta expresión, P_k es el conjunto de objetos agrupados en un mismo lugar (es decir, la segmentación), t_k es una etiqueta semántica breve que resume la función o identidad del lugar, y d_k es una descripción en lenguaje natural que aporta contexto funcional adicional (categorización). El subíndice j identifica el mapa semántico considerado dentro del conjunto total, mientras que el índice ℓ distingue entre diferentes segmentaciones de referencia posibles para un mismo mapa, ya que una misma escena puede admitir múltiples particiones coherentes.

La existencia de múltiples segmentaciones para un mismo mapa semántico refleja que un entorno puede ser interpretado de distintas maneras desde una perspectiva funcional. Este fenómeno es especialmente común en espacios reducidos, donde numerosos objetos se encuentran agrupados en una misma zona, dando lugar a diferentes posibilidades de agrupación coherentes. Por ejemplo, supongamos que el mapa semántico $\mathbf{m}^{(j)}$ contiene dos puestos de trabajo, cada uno compuesto por los objetos “mesa”, “pantalla”, “teclado”, “ratón” y “taza” (véase la Figura 16). Si ambos puestos se encuentran físicamente próximos en la escena, pueden surgir al menos dos segmentaciones plausibles de $\mathbf{m}^{(j)}$, representadas en el conjunto $\left\{\mathcal{P}_{\text{GT}}^{(j,\ell)}\right\}_{\ell=1}^{2}$:

- $\mathcal{P}_{\text{GT}}^{(j,1)}$, en la que cada puesto de trabajo se considera un lugar independiente, reflejando una interpretación basada en unidades funcionales individuales.

- $\mathcal{P}_{\text{GT}}^{(j,2)}$, en la que ambos puestos se agrupan como un único lugar, interpretado como un espacio compartido de trabajo y estudio, donde los objetos se entienden como parte de una misma actividad global.

Ambas segmentaciones son igualmente válidas, ya que representan diferentes criterios de agrupación sobre el mismo mapa semántico, destacando la naturaleza contextual de esta tarea.

Figura 16: Escena de ejemplo con dos puestos de trabajo próximos, cada uno compuesto por una mesa, una silla, una pantalla, un teclado, un ratón y una taza. Este entorno permite al menos dos segmentaciones funcionales distintas: una que considera cada puesto como un lugar independiente, y otra que agrupa ambos como un único espacio de trabajo colaborativo.

5.1.2. Detalles de implementación

Los modelos de *embedding* de palabras probados fueron BERT [12] y RoBERTa [27]. Para obtener *embeddings* a nivel de frases se utilizó el modelo `all-mpnet-base-v2`[6], perteneciente a la familia de Sentence Transformers [47], que combina la arquitectura Transformer [60] con técnicas de aprendizaje por pares para representar eficazmente el significado de frases. En cuanto a los LLMs, se han evaluado tanto modelos de código abierto ejecutados localmente —como Qwen 2.5 14B[7], con 14 mil millones de parámetros— como modelos propietarios accesibles mediante API, como Gemini 2.0 Flash[8]. El LVLM utilizado en todos los experimentos también fue Gemini 2.0 Flash, ya que cuenta con capacidades multimodales. Todos los modelos de código abierto probados han sido utilizados a través de la librería Hugging Face.

Para la segmentación basada en *clustering* se empleó HDBSCAN (*Hierarchical Density-Based Spatial Clustering of Applications with Noise*) [6], una versión jerárquica de DBSCAN

[6] https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[7] https://huggingface.co/Qwen/Qwen2.5-14B-Instruct
[8] https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-0-flash

que construye una representación de la densidad de los datos para identificar agrupamientos de manera automática, sin necesidad de establecer manualmente parámetros cruciales como `epsilon` o `min_samples`. En la re-aplicación del algoritmo de DBSCAN en el proceso de división de clústeres del refinamiento de la agrupación inicial, los parámetros fueron `epsilon = 1.0` y `min_samples = 2`. La dimensión de los descriptores semánticos se redujo a 3 utilizando PCA. La reducción de la dimensión de los descriptores y el *clustering* mediante DBSCAN y HDBSCAN se implementaron utilizando funciones de la librería scikit-learn[9].

Los distintos parámetros y umbrales de las técnicas propuestas fueron fijados de manera empírica tras la realización de algunas pruebas:

- El peso de la información semántica en la distancia entre descriptores al aplicar *clustering* se fijó a $\alpha_s = 0{,}55$ (véase la Sección 3.2.1).
- El compromiso entre variabilidad semántica y dispersión geométrica a la hora de calcular el valor de división de un clúster en el refinamiento se fijó a $\lambda = 0{,}6$. El umbral sobre dicho valor de división de clústeres se fijó a $\theta_{\text{split}} = 2{,}5$ (véase la Sección 3.2.2).
- En la unión de clústeres del refinamiento, el umbral sobre la distancia euclídea se fijó a $\theta_{\text{merge,g}} = 2{,}0$, y el umbral sobre la similitud de los descriptores semánticos de clústeres a $\theta_{\text{merge,s}} = 0{,}85$ (véase la Sección 3.2.2).
- El umbral de distancia máxima entre objetos para evaluar relaciones semánticas se fijó a $\delta_{\text{max}} = 0{,}5$ (véase la Sección 4.2).

5.1.3. Métricas para evaluación de segmentaciones

Para evaluar la calidad de las segmentaciones de lugares frente a las segmentaciones de referencia propuestas, se emplearon tres métricas estándar en el ámbito de *clustering*, cada una capturando diferentes aspectos de la calidad de los resultados:

- *Adjusted Rand Index* (ARI) [22]: cuantifica la coincidencia entre pares de objetos en la segmentación predicha y la de referencia, ajustando el resultado para corregir las coincidencias esperadas por azar. Sus valores oscilan entre −1 y 1, donde 1 indica segmenta-

[9] https://scikit-learn.org/stable/

ciones idénticas, 0 equivale a una asignación aleatoria y valores negativos reflejan una concordancia peor que la aleatoria.

- *Normalized Mutual Information* (NMI): mide la información compartida entre la segmentación predicha y la de referencia, normalizando la información mutua para que el resultado esté en el rango [0, 1]. En este trabajo se utiliza una versión que emplea la media geométrica de las entropías de ambas segmentaciones como factor de normalización, lo cual penaliza más severamente las discrepancias entre particiones. Un valor de 1 indica identidad perfecta entre las segmentaciones y 0 que no comparten ninguna información.

- *V-Measure* [49]: calcula la media armónica de dos criterios: homogeneidad —que verifica que cada clúster predicho contenga únicamente objetos de un mismo clúster de referencia— y completitud —que comprueba que todos los objetos de un clúster de referencia se asignen al mismo clúster predicho—. El resultado varía entre 0 y 1, penalizando tanto la fragmentación excesiva como la fusión inapropiada de clústeres.

En los mapas semánticos con múltiples segmentaciones de referencia, los resultados de cada método se evalúan frente a todas las segmentaciones, y se conserva la mejor puntuación alcanzada.

5.2. Evaluación de la segmentación de lugares

La Tabla 1 presenta una comparación cuantitativa de los métodos propuestos y varios estudios de ablación, evaluando su precisión respecto a las segmentaciones de referencia. Los métodos denotados por "LLM [...]" se refieren a la segmentación y categorización del mapa de forma conjunta basada puramente en LLMs (véase la Sección 3.3). En concreto, se comparan dos variantes: una que utiliza el modelo comercial del estado del arte Gemini 2.0 Flash (método "LLM (Gemini 2.0 Flash)" y otra que recurre al LLM abierto Qwen 2.5 14B con 14 mil millones de parámetros (método "LLM (Qwen 2.5 14B)"). Los métodos acabados en "[...] *Clustering* + Refinamiento" combinan el *clustering* geométrico-semántico con un paso adicional de refinamiento sobre los clústeres obtenidos (véase la Sección 3.2.2). En particular, uno de ellos emplea descriptores contextualizados generados por un LLM para cada objeto y procesados con Sentence-BERT (método "LLM + S-BERT + *Clustering* + Refinamiento"), mientras que

Método	ARI	NMI	V-Measure
LLM (Gemini 2.0 Flash)	**0,460**	0,628	0,625
LLM + S-BERT + *Clustering* + Refinamiento	0,423	**0,681**	**0,670**
LLM + S-BERT + *Clustering*	0,419	**0,681**	0,667
Geométrico + *Clustering*	0,384	0,630	0,621
BERT + *Clustering* + Refinamiento	0,376	0,671	0,651
BERT + *Clustering*	0,372	0,655	0,638
RoBERTa + *Clustering*	0,322	0,644	0,631
LLM (Qwen 2.5 14B)	0,267	0,434	0,427

Tabla 1: Evaluación de los métodos propuestos para la segmentación de lugares de un mapa semántico, ordenados por resultado en la métrica ARI. Se resaltan en negrita las mejores puntuaciones obtenidas para cada métrica.

el otro se basa únicamente en los descriptores semánticos de palabras producidos por BERT (método "BERT + *Clustering* + Refinamiento"). Los métodos acabados en "[...] *Clustering*" implementan una segmentación geométrico-semántica directa sin refinamiento posterior (véase la Sección 3.2.1), utilizando los descriptores generados por el modelo BERT (método "BERT + *Clustering*") o por el modelo RoBERTa (método "RoBERTa + *Clustering*"). Además, se incluye un enfoque puramente geométrico que agrupa los objetos en el espacio sin recurrir a información semántica (método "Geométrico + *Clustering*").

El análisis comparativo revela que el método que realiza la segmentación y categorización conjunta basado en LLMs, en concreto utilizando Gemini 2.0 Flash (ARI = 0,460), junto con los métodos basados en *clustering* descriptores semánticos contextualizados y refinamiento (0,423 y 0,419), superan con holgura a la segmentación de lugares puramente geométrica (0,384). De esta forma, se demuestra que el uso de la semántica beneficia la tarea de segmentación de lugares en un mapa semántico previamente construido con información acerca de los objetos. Aún así, los resultados del método puramente geométrico son competitivos, alcanzando un rendimiento intermedio en comparación con el resto de alternativas. Por debajo quedan los enfoques basados únicamente en descriptores obtenidos por modelos de *embedding* de palabras sin refinamiento, como BERT (0,372) y RoBERTa (0,322), evidenciando que,

al no capturarse adecuadamente la relación semántica funcional entre los elementos, presentan limitaciones para representar las estructuras del mapa. Los peores resultados de todos los ofrece el método basado puramente en LLMs implementado en el modelo abierto Qwen 2.5 14B (0,267).

Como se ha señalado, el mejor resultado global proviene del método de segmentación y categorización basado en LLMs, implementado con un modelo de última generación como Gemini 2.0 Flash. Este modelo, de escala masiva y mantenido por una infraestructura computacional de alto nivel como la que proporciona Google, no solo lidera en ARI, sino que ofrece un equilibrio sólido en todas las métricas analizadas. Sin embargo, esta excelencia conlleva un coste significativo, no solo en términos computacionales, sino también económicos, al tratarse de un modelo comercial al que se accede mediante una API de pago. Dicha API calcula el coste en función del número de *tokens* procesados, tanto de entrada como de salida, lo que implica que, al introducir la representación completa de un mapa semántico como entrada, el coste por consulta puede incrementarse notablemente. Cuando este LLM se sustituye por uno abierto de menor tamaño y menores requerimientos computacionales, como Qwen 2.5 14B, el rendimiento cae drásticamente, alcanzando resultados muy bajos que podrían perjudicar la utilización de las segmentaciones generadas. Por este motivo, el segundo mejor método — el clustering geométrico-semántico basado en descriptores contextualizados y seguido de un refinamiento— representa una alternativa especialmente atractiva: requiere menos recursos y, al mismo tiempo, domina las métricas de NMI (0,681) y V-Measure (0,670), lo que es reflejo de una notable coherencia y completitud en las segmentaciones.

Una forma de mejorar la precisión del método basado en LLMs con modelos de menor envergadura, como Qwen 2.5 14B, sería recurrir al uso de flujos de trabajo basados en agentes (en inglés, *agentic workflows*). Estos flujos constituyen procesos estructurados que guían la generación de respuestas de los LLMs mediante la descomposición de la tarea en pasos intermedios [61], el uso de memoria externa y mecanismos de retroalimentación [29], con el fin de compensar las limitaciones del modelo subyacente. En el contexto de la segmentación y categorización de mapas semánticos, esto podría implicar dividir la escena en regiones, razonar sobre cada una de forma secuencial y combinar las decisiones de forma informada, logrando así un comportamiento más robusto sin necesidad de recurrir a modelos de escala masiva.

La comparación directa entre descriptores semánticos confirma la superioridad de los des-

criptores semánticos contextualizados, empleando un LLM para generar una frase relacionada con la funcionalidad del objeto y posteriormente un modelo de *embedding* de frases. Tanto con refinamiento (ARI = $0{,}423$) como sin él ($0{,}419$), los vectores generados a partir de frases contextualizadas capturan relaciones semánticas que los modelos de *embedding* de palabras como BERT o RoBERTa no alcanzan. Esta ventaja se reproduce en las métricas NMI y V-Measure, consolidando la idea de que la contextualización es clave para segmentar en espacios basados en la funcionalidad.

Además, la introducción de un paso de refinamiento de los clústeres producidos inicialmente por el método de *clustering* mejora consistentemente los resultados, con independencia del tipo de descriptor empleado. Para los descriptores contextualizados, el ARI sube de $0{,}419$ a $0{,}423$; para los descriptores de BERT, pasa de $0{,}372$ a $0{,}376$. Aunque la mejora pueda parecer modesta, el refinamiento afina la coherencia de los clústeres y los hace más comparables con la referencia geométrica, reforzando su utilidad práctica.

La Tabla 2 presenta los resultados de segmentación ya mostrados en la Tabla 1, pero desglosados según el conjunto de datos de origen. Los resultados evidencian que los métodos propuestos funcionan, en general, con mejor precisión en las escenas de ScanNet que en las de SceneNN. Esto se debe a que las escenas de ScanNet suelen ser más amplias y contener un mayor número de objetos, lo que facilita que los algoritmos de *clustering* encuentren patrones espaciales y funcionales más consistentes. De hecho, el *clustering* puramente geométrico alcanza un ARI de $0{,}407$ en ScanNet, frente al $0{,}360$ obtenido en SceneNN. Por el contrario, las escenas de SceneNN son más pequeñas y presentan una distribución más densa y variada de objetos —mesas, electrodomésticos y otros muebles se mezclan en espacios muy reducidos—, lo que complica la segmentación en lugares funcionales bien diferenciados. Incluso el mejor método global, el basado en LLMs con Gemini 2.0 Flash, obtiene un ARI de solo $0{,}378$ en este conjunto.

El orden de los métodos en ScanNet reproduce fielmente la jerarquía global: la segmentación-categorización directa con Gemini 2.0 Flash lidera con un ARI de $0{,}543$, seguida de los dos enfoques con descriptores contextualizados con refinamiento ($0{,}440$) y sin refinamiento ($0{,}429$). Sin embargo, en SceneNN se observa otro patrón diferente: los mejores resultados los marcan los modelos basados en **embeddings** de palabras —RoBERTa y BERT—, desplazando a Gemini y a los descriptores contextualizados. Una posible explicación es que, al ser escenas pequeñas

Dataset	Método	ARI	NMI	V-Measure
ScanNet [11]	LLM (Gemini 2.0 Flash)	**0,543**	**0,724**	**0,719**
	LLM + S-BERT + *Clustering* + Refinamiento	0,440	0,719	0,707
	LLM + S-BERT + *Clustering*	0,429	0,692	0,679
	Geométrico + *Clustering*	0,407	0,670	0,659
	BERT + *Clustering*	0,333	0,648	0,629
	BERT + *Clustering* + Refinamiento	0,323	0,673	0,649
	LLM (Qwen 2.5 14B)	0,296	0,445	0,437
	RoBERTa + *Clustering*	0,207	0,601	0,586
SceneNN [20]	RoBERTa + *Clustering*	**0,437**	**0,687**	**0,677**
	BERT + *Clustering* + Refinamiento	0,428	0,668	0,654
	BERT + *Clustering*	0,410	0,663	0,646
	LLM + S-BERT + *Clustering*	0,409	0,670	0,655
	LLM + S-BERT + *Clustering* + Refinamiento	0,407	0,643	0,633
	LLM (Gemini 2.0 Flash)	0,378	0,533	0,532
	Geométrico + *Clustering*	0,360	0,591	0,583
	LLM (Qwen 2.5 14B)	0,238	0,422	0,417

Tabla 2: Rendimiento de las alternativas de segmentación de lugares agrupadas por conjunto de datos, ordenadas por *Adjusted Rand Index* (ARI).

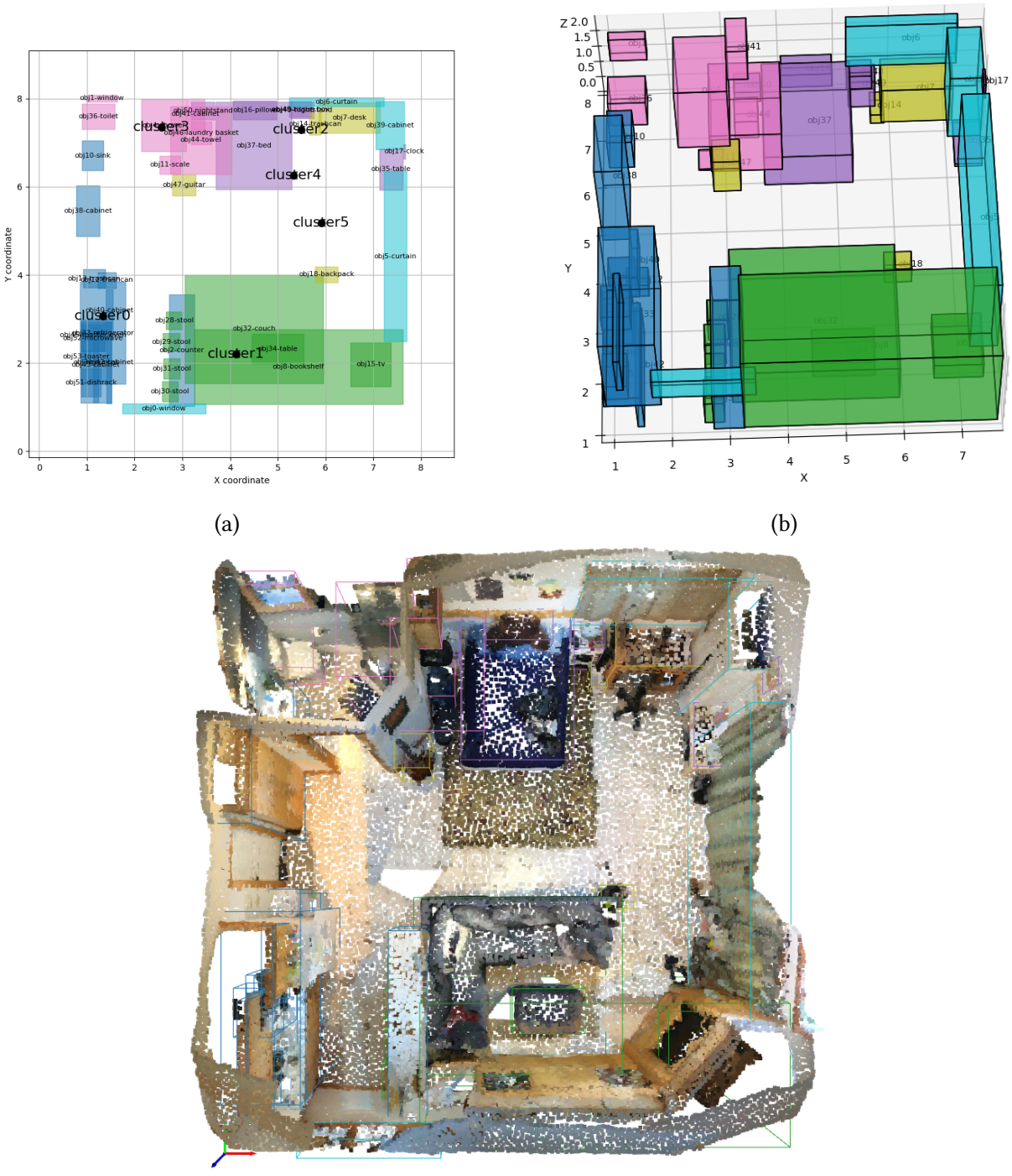
con objetos muy dispares y sin grandes zonas funcionales, las claves semánticas de alto nivel que capta un LLM pierden relevancia. En cambio, los modelos de *embedding* de palabras proporcionan una descripción léxica muy detallada que facilita separar objetos muy distintos entre sí y evita que se mezclen en clústeres poco coherentes. Adicionalmente, es posible que la semántica implícita de los modelos de *embedding* de palabras como BERT o RoBERTa esté casualmente alineada con el tipo de objetos que se presentan en este último conjunto de datos, beneficiando así su capacidad para agrupar objetos funcionalmente relacionados en escenas pequeñas y densas como las de SceneNN.

Pese a estos matices, el peor rendimiento en ambos conjuntos lo sigue ofreciendo la versión puramente basada en LLMs con un modelo abierto de un número de parámetros contenido ("LLM (Qwen 2.5 14B)"), con un ARI de $0{,}296$ en ScanNet y $0{,}258$ en SceneNN. De nuevo se confirma que, cuando el número de parámetros se reduce drásticamente —y con él, la capacidad de razonamiento semántico del modelo—, la segmentación resultante pierde coherencia independientemente de la naturaleza del entorno.

En cualquiera de los casos, los métodos de segmentación de lugares propuestos en este TFM han demostrado ser eficaces para identificar zonas funcionales dentro de un entorno sin depender de la noción tradicional de "habitación", una limitación presente en muchas otras técnicas. Gracias a esta característica, las segmentaciones obtenidas reflejan agrupaciones coherentes desde el punto de vista funcional, incluso en escenas donde no existe una estructura geométrica clara basada en habitaciones, como ocurre en algunos de los mapas utilizados. Un ejemplo ilustrativo es la escena `scannet_scene_0000_00` del conjunto ScanNet (véase la Figura 17), que consiste en un espacio principal abierto que integra varias zonas funcionales —como salón con sofá y televisión, cocina, cama y área de estudio, entre otras— junto con un baño ubicado en una habitación separada. Como se observa, los métodos propuestos logran identificar agrupaciones de objetos con funciones afines, incluso cuando estos comparten un mismo espacio físico sin divisiones arquitectónicas claras.

5.3. Evaluación de la categorización de lugares

Los experimentos realizados confirman que la categorización de lugares basada en LLMs (véase la Sección 3.2.3 y la Sección 3.3) es una estrategia eficaz para asignar información semántica relevante a los lugares previamente segmentados del mapa. Esta categorización per-



(a) (b)

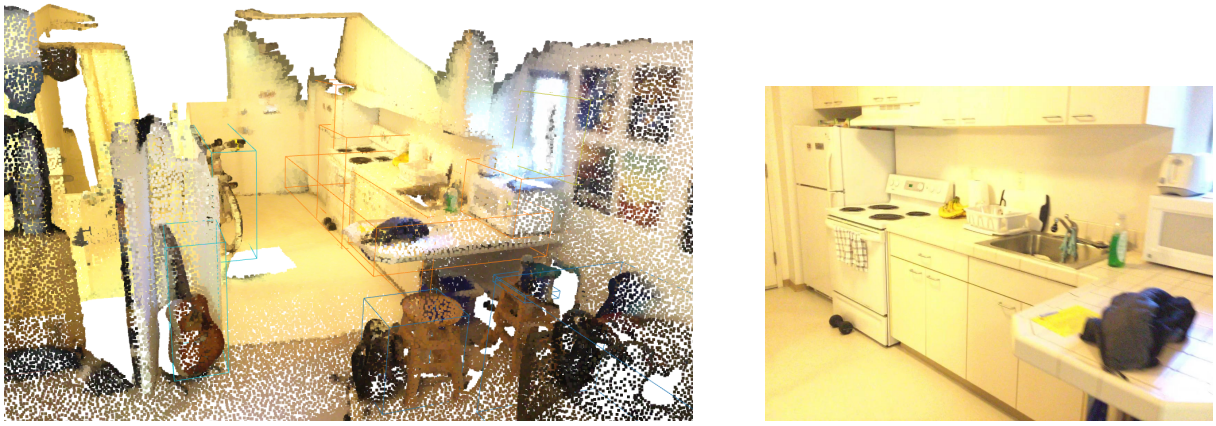
(c)

Figura 17: Visualización de la segmentación de la escena `scannet_scene_0000_00` generada mediante el método basado en LLMs con Gemini 2.0 Flash, mostrada en tres formatos: (a) vista superior en 2D donde cada objeto aparece como una *bounding box* coloreada según su clúster, (b) vista superior en 3D con las *bounding boxes* igualmente coloreadas por clúster, y (c) nube de puntos de la escena con las *bounding boxes* de los objetos coloreadas según el tamaño del clúster al que pertenecen.

mite generar etiquetas breves o descripciones funcionales sencillas en función de los objetos presentes en cada lugar.

Un ejemplo ilustrativo se muestra en la escena `scannet_scene_0101`, procesada con el método basado en LLMs con Gemini 2.0 Flash. La Figura 18 presenta dos vistas complementarias de uno de los lugares segmentados. En la Figura 18a, se muestra la nube de puntos anotada, donde cada objeto aparece rodeado por su *bounding box* coloreada. Los objetos asignados al mismo lugar comparten el mismo color en sus cajas delimitadoras. El grupo naranja agrupa elementos característicos de una cocina, como "horno" (*oven*), "encimera" (*countertop*), "microondas" (*microwave*), "refrigerador" (*refrigerator*) y "mueble de cocina" (*kitchen cabinet*). Otros objetos, como "bicicleta" (*bicycle*), "guitarra" (*guitar*) o "banco" (*stool*), aparecen con cajas de colores distintos al pertenecer a lugares diferentes. En la Figura 18b, se muestra una imagen de la escena real procedente de la grabación proporcionada en el conjunto de datos, centrada en los objetos del lugar. Para este grupo de objetos, la fase de categorización basada en LLMs generó la etiqueta "*kitchen_zone*" y la descripción funcional: "*A food preparation and storage area featuring essential kitchen appliances and surfaces, supporting activities like cooking, reheating meals, and organizing ingredients*". Como se puede comprobar, la etiqueta y la descripción generadas por el modelo capturan correctamente la finalidad del conjunto de objetos agrupados, reflejando una interpretación coherente del entorno basada en su funcionalidad y no únicamente en la geometría o disposición espacial de los elementos.

Otro ejemplo ilustrativo se encuentra en la escena `scannet_scene_0000_00`, también procesada con el método de segmentación basado en LLMs con Gemini 2.0 Flash. La Figura 19 presenta dos vistas complementarias de uno de los lugares segmentados. En la Figura 19a, se muestra la nube de puntos anotada, donde las *bounding boxes* de color verde agrupan objetos característicos de una zona de salón, como "banco" (*stool*), "televisión" (*tv*), "sofá" (*couch*) y "estantería" (*bookshelf*), entre otros. En la Figura 19b, se muestra una imagen capturada durante la grabación de la escena, en la que pueden observarse estos objetos de una forma clara. Para este grupo, el modelo generó la etiqueta "*media_lounge*" y la descripción funcional: "*A casual entertainment space centered around a TV, with a couch and extra stools arranged around a coffee table for group viewing and socializing, and a bookshelf for storing reading materials*". De nuevo, la categorización generada por el modelo va más allá de una mera enumeración de objetos, integrando su función y disposición para deducir el propósito del espacio como un

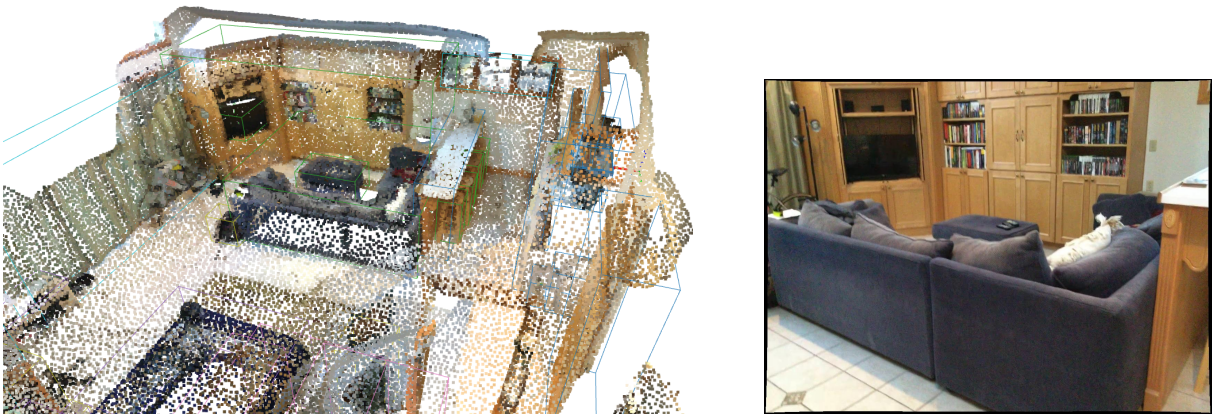


(a) (b)

Figura 18: Visualización del lugar etiquetado como cocina en la escena `scannet_scene_0101`, donde se muestra su segmentación en la nube de puntos con *bounding boxes* de color naranja (a) y una imagen real de la escena capturada en el *frame* 605 de la grabación, centrada en los objetos agrupados en ese lugar (b).

entorno de ocio compartido.

Por otro lado, los experimentos revelan que cuando las segmentaciones producen lugares sin sentido práctico, las categorizaciones resultantes pierden utilidad para la operación de un robot móvil. Un ejemplo de esta casuística se encuentra en el mapa semántico `scannet_scene_0392_01` de ScanNet, segmentado con el método basado en LLMs utilizando Qwen 2.5 14B. En la Figura 20 se muestran algunas imágenes extraídas de la grabación de la escena proporcionada en el conjunto de datos, cada una capturada desde un punto de vista diferente. Como se puede observar, hay varios lugares fácilmente identificables, como una cocina, un cuarto de baño y una zona de trabajo, entre otros; sin embargo, en una de las pruebas realizadas, el modelo segmentó la escena creando un único lugar en el mapa, en el que estaban contenidos todos los objetos. La etiqueta semántica obtenida por el proceso de caracterización para ese único lugar fue *living_and_kitchen*, junto con la descripción: *This area serves as a multifunctional space combining living and kitchen functionalities, featuring seating arrangements like a couch and chairs, work areas such as desks and tables, and kitchen appliances including a stove, sink, and cabinets.* Aunque la descripción generada por el modelo es coherente con los objetos presentes en la escena y su disposición, agrupar toda la estancia en un único lugar impide



(a) (b)

Figura 19: Visualización del lugar etiquetado como "*media_lounge*" en la escena `scannet_scene_0000_00`, con la segmentación de la nube de puntos en la que se destacan en verde los objetos asociados al salón (a) y una imagen real de la escena capturada en el *frame* 1943, donde se aprecian dichos objetos en contexto (b).

reflejar su diversidad funcional real. Esta subsegmentación limita la capacidad de la categorización para diferenciar zonas con funciones distintas, dificultando así la identificación de espacios asociados a tareas específicas.

Un caso similar se observa en aquellas segmentaciones donde se generó un lugar compuesto por un único objeto, como en la escena `scannet_scene0515_00`, procesada por el método de descriptores semánticos contextualizados con Sentence-BERT, seguidos de *clustering* y refinamiento. En este caso, la segmentación contenía un lugar compuesto únicamente por el objeto "estantería" (*shelf*). En la Figura 21, se muestra tanto la segmentación en la nube de

(a) (b) (c) (d)

Figura 20: *Frames* de la grabación correspondientes a la escena `scannet_scene_0392_01` del conjunto de datos ScanNet.

(a) (b)

Figura 21: Visualización del lugar identificado como zona de almacenamiento en la escena `scannet_scene0515_00`, donde se muestra la segmentación en la nube de puntos con un único objeto (a) y la imagen real capturada en el *frame* 129, donde se aprecian el objeto en su contexto (b).

puntos (Figura 21a) como una imagen correspondiente al *frame* de la escena capturado desde la grabación original (Figura 21b), donde puede observarse claramente el objeto aislado. El proceso de categorización generó para dicho lugar la etiqueta "*storage*" y la descripción funcional: "*A space primarily intended for storing or organizing items, characterized by a shelf likely used to hold supplies, tools, or personal belongings*". Aunque tanto la etiqueta como la descripción son coherentes con el objeto segmentado, cabe cuestionar si un lugar de estas características resulta útil para la operativa de un robot móvil. En muchos casos, podría ser más conveniente agrupar el objeto con un lugar adyacente funcionalmente más relevante, como una estación de trabajo, para favorecer una interpretación más útil del entorno.

5.4. Evaluación de construcción de relaciones semánticas

Los experimentos realizados confirman que ambos métodos de inferencia de relaciones semánticas —el basado en LLMs (véase la Sección 4.2) y el basado en LVLMs (véase la Sección 4.3)— son eficaces para identificar vínculos significativos de vocabulario abierto entre los objetos de un mapa semántico. Ambos enfoques permiten detectar los distintos tipos de relaciones definidos en la Sección 4.1: espaciales, estructurales, funcionales y causales, si bien estas

últimas se infieren con menor frecuencia en los conjuntos de datos utilizados. El enfoque basado en LVLMs destaca por su capacidad para generar relaciones más precisas y alineadas con la configuración espacial y funcional real del entorno, gracias a la incorporación de información visual además del contexto textual.

Un ejemplo ilustrativo del proceso de inferencia de relaciones semánticas se da en la escena `scannet_scene0000_00` de ScanNet, utilizando el método basado en LLMs. En concreto, se analiza la relación entre los objetos con clase semántica "encimera" (*counter*) y "banco" (*stool*). En la Figura 22, se muestran tanto la nube de puntos segmentada, donde pueden observarse los objetos implicados junto con sus *bounding boxes*, como una imagen de la escena extraída de la grabación correspondiente, centrada en los mismos elementos. Tras proporcionar al LLM la información textual de los objetos, el modelo fue capaz de inferir las siguientes relaciones semánticas expresadas en formato JSON:


```
[ {
    "source_id": "obj2",
    "source": "counter",
    "target_id": "obj30",
    "target": "stool",
    "type": "spatial",
    "predicate": "is next to"
  },
  {
    "source_id": "obj2",
    "source": "counter",
    "target_id": "obj30",
    "target": "stool",
    "type": "functional",
    "predicate": "is used with"
  },
  {
    "source_id": "obj30",
    "source": "stool",
    "target_id": "obj2",
    "target": "counter",
    "type": "spatial",
    "predicate": "is next to"
```

```
  },
  {
    "source_id": "obj30",
    "source": "stool",
    "target_id": "obj2",
    "target": "counter",
    "type": "functional",
    "predicate": "is used at"
  } ]
```

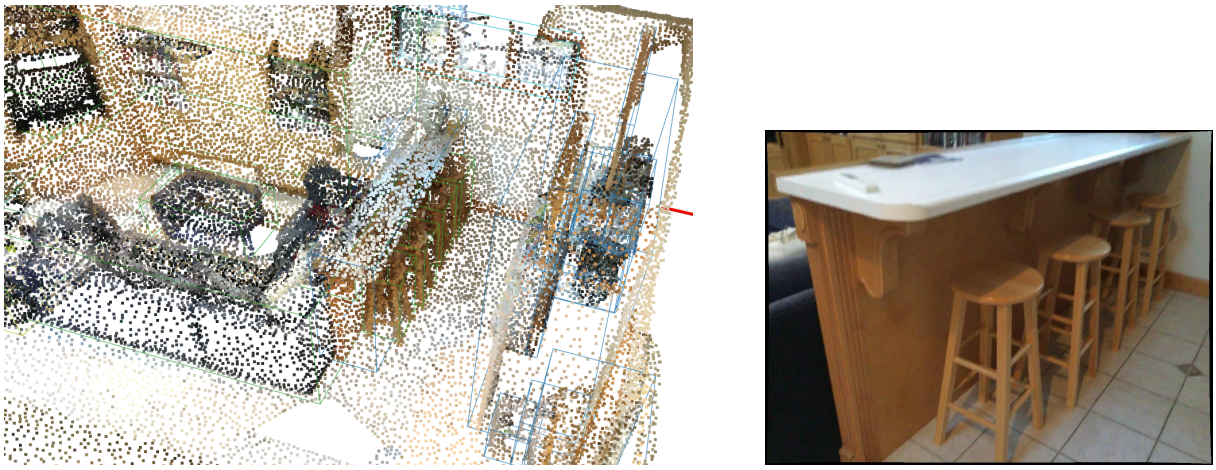
Obsérvese que se identifican cuatro relaciones: dos de tipo espacial ("*is next to*") y dos funcionales ("*is used with*" y "*is used at*"). Aunque correctas, estas relaciones presentan un nivel de detalle limitado. Sería más informativo, por ejemplo, indicar que el banco está "debajo de" o "cubierto por" la encimera, lo cual aporta un contexto espacial más útil para tareas de manipulación o navegación.

Por su parte, el método basado en LVLMs proporciona al modelo, además del texto, una imagen representativa que muestre simultáneamente los objetos implicados. En este caso, se utilizó la imagen mostrada en la Figura 22b, correspondiente al *frame* óptimo de la grabación. La respuesta generada por el LVLM en formato JSON fue la siguiente:

```
[ {
    "source_id": "obj2",
    "source": "counter",
    "target_id": "obj30",
    "target": "stool",
    "type": "spatial",
    "predicate": "is in front of"
  },
  {
    "source_id": "obj2",
    "source": "counter",
    "target_id": "obj30",
    "target": "stool",
    "type": "functional",
    "predicate": "is used with"
  },
  {
```

```
    "source_id": "obj2",
    "source": "counter",
    "target_id": "obj30",
    "target": "stool",
    "type": "functional",
    "predicate": "provides seating area next to"
  },
  {
    "source_id": "obj30",
    "source": "stool",
    "target_id": "obj2",
    "target": "counter",
    "type": "spatial",
    "predicate": "is partially under"
  },
  {
    "source_id": "obj30",
    "source": "stool",
    "target_id": "obj2",
    "target": "counter",
    "type": "functional",
    "predicate": "is used at"
  },
  {
    "source_id": "obj30",
    "source": "stool",
    "target_id": "obj2",
    "target": "counter",
    "type": "functional",
    "predicate": "provides seating for"
  } ]
```


En este caso, la inferencia semántica es más rica y precisa. Las relaciones espaciales no solo indican cercanía, sino que especifican posiciones relativas como que la encimera está "enfrente del" (*in front of*) banco y que el banco está "parcialmente debajo de" (*partially under*) la encimera, lo cual aporta una mejor comprensión tridimensional de la disposición. De forma similar, las relaciones funcionales pasan de simples asociaciones de uso a descripciones más



(a) (b)

Figura 22: Visualización de la escena `scannet_scene0000_00`, donde se muestra la relación entre los objetos “encimera” y “banco” mediante su representación en la nube de puntos (a) y una imagen de la escena real (b).

detalladas, como que el banco “provee asiento para” (*provides seating for*) la encimera o que la encimera “provee un área de asiento con” (*forms a seating area with*) el banco.

Sin embargo, también se han identificado casos en los que el método de inferencia de relaciones semánticas basado en LVLMs no mejora sustancialmente la riqueza informativa con respecto al método textual basado en LLMs. En estas situaciones, la relación entre los objetos es evidente incluso sin necesidad de información visual adicional, ya sea por las propias clases semánticas de los objetos o por su disposición espacial inequívoca. Un ejemplo representativo de esta casuística se encuentra en la escena `scannet_scene0673_04` de ScanNet, donde se analiza la relación entre los objetos “televisión” (*tv*) y “soporte de televisión” (*tv stand*). En la Figura 23 se presentan tanto la nube de puntos segmentada como una imagen extraída de la grabación, donde se puede observar claramente que la televisión está colocada sobre el soporte. En este caso, el método basado en LLMs infirió las siguientes relaciones semánticas:

```
[ {
    "source_id": "obj0",
    "source": "tv",
    "target_id": "obj8",
    "target": "tv stand",
    "type": "spatial",
```

```
    "predicate": "is on"
  },
  {
    "source_id": "obj0",
    "source": "tv",
    "target_id": "obj8",
    "target": "tv stand",
    "type": "functional",
    "predicate": "is supported by"
  },
  {
    "source_id": "obj8",
    "source": "tv stand",
    "target_id": "obj0",
    "target": "tv",
    "type": "structural",
    "predicate": "supports"
  },
  {
    "source_id": "obj8",
    "source": "tv stand",
    "target_id": "obj0",
    "target": "tv",
    "type": "functional",
    "predicate": "is for"
  } ]
```

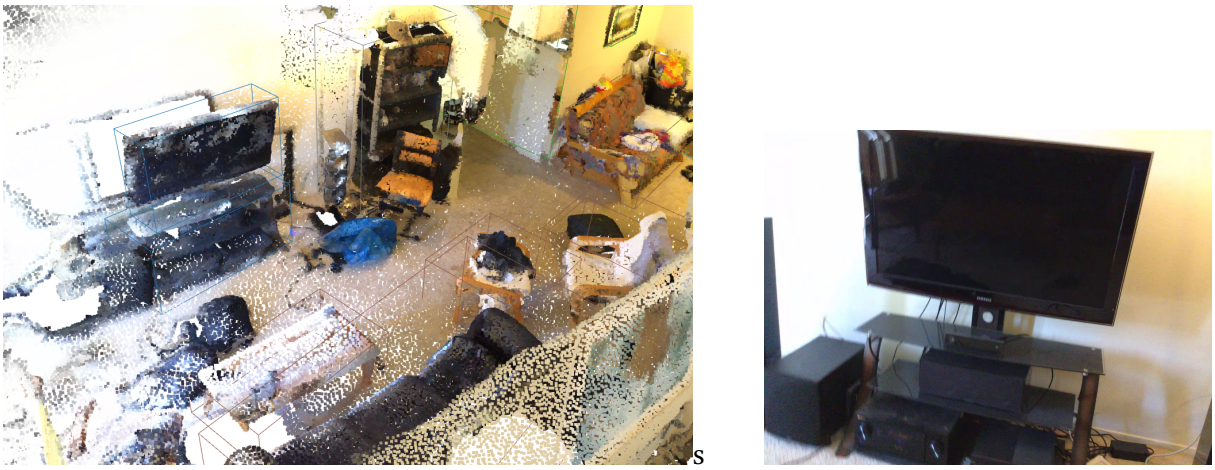
Como se puede observar, el modelo captura correctamente las relaciones fundamentales: que la televisión "está sobre" (*is on*) el soporte y que este "soporta" (*supports*) a la televisión.

Por su parte, el método basado en LVLMs generó la siguiente respuesta:

```
[ {
    "source_id": "obj0",
    "source": "tv",
    "target_id": "obj8",
    "target": "tv stand",
    "type": "spatial",
    "predicate": "is on"
```

```
  },
  {
    "source_id": "obj0",
    "source": "tv",
    "target_id": "obj8",
    "target": "tv stand",
    "type": "structural",
    "predicate": "is supported by"
  },
  {
    "source_id": "obj8",
    "source": "tv stand",
    "target_id": "obj0",
    "target": "tv",
    "type": "spatial",
    "predicate": "is below"
  },
  {
    "source_id": "obj8",
    "source": "tv stand",
    "target_id": "obj0",
    "target": "tv",
    "type": "structural",
    "predicate": "supports"
  },
  {
    "source_id": "obj8",
    "source": "tv stand",
    "target_id": "obj0",
    "target": "tv",
    "type": "functional",
    "predicate": "is designed to hold"
  } ]
```


De nuevo, las relaciones clave están correctamente representadas, aunque se incluyen algunas adicionales que, si bien son coherentes, no aportan un conocimiento significativamente nuevo. Por ejemplo, indicar que el soporte "está debajo" (*is below*) de la televisión o que "está



(a) (b)

Figura 23: Visualización de los objetos “televisión” y “soporte de televisión” en la escena `scannet_scene0673_04`. Se muestra la nube de puntos segmentada (a) y una imagen extraída de la grabación de la escena (b).

diseñado para sostenerla” (*is designed to hold*) puede considerarse redundante, dado que esas conclusiones ya se derivan de la relación estructural principal y de las clases semánticas de los objetos implicados.

5.5. Evaluación del desempeño de un LLM con mapa semántico-topológico

Un robot móvil que emplea un LLM como motor de razonamiento y comprensión puede apoyarse en un mapa semántico como representación interna del entorno, lo que le permite razonar sobre dicho mapa y responder peticiones formuladas por un usuario. Yendo un paso más allá, el LLM puede incluso actuar como motor de planificación para guiar las acciones del robot en ese entorno [46]. En cualquiera de estos casos, es fundamental que el modelo comprenda correctamente el contenido del mapa para garantizar respuestas coherentes y útiles. Aunque algunos trabajos sugieren que la calidad y el nivel de detalle del mapa influyen en el rendimiento de los LLMs en este tipo de tareas, la literatura carece de estudios que lo demuestren de forma concluyente. En general, se asume que una representación más rica del entorno mejora la capacidad de respuesta del modelo, pero esta hipótesis no ha sido validada de manera sistemática.

Con el objetivo de explorar esta cuestión, en este trabajo se ha comparado la calidad de

las respuestas proporcionadas por un LLM ante peticiones de un usuario en dos escenarios distintos:

- Escenario 1: el modelo dispone únicamente de un mapa semántico con información básica de los objetos presentes en la escena (clase semántica, posición y dimensiones). El *prompt* empleado en este escenario se encuentra disponible en el Apéndice B.5.
- Escenario 2: el modelo cuenta con un mapa semántico enriquecido con información topológica obtenida empleando las técnicas propuestas en este trabajo, es decir, segmentaciones y categorizaciones de lugares, además de relaciones semánticas entre objetos. El *prompt* correspondiente a este escenario puede consultarse en el Apéndice B.6.

Los experimentos se han llevado a cabo sobre distintas escenas de los conjuntos de datos utilizados, empleando de nuevo los dos LLMs utilizados en las pruebas anteriores: el modelo propietario Gemini 2.0 Flash, accedido a través de su API, y el modelo abierto Qwen 2.5 14B, ejecutado localmente. A continuación, se presentan distintos tipos de razonamiento evaluados en ambos escenarios con los dos LLMs, junto con las conclusiones obtenidas y ejemplos ilustrativos tomados de la escena `scannet_scene0000_00` del conjunto de datos ScanNet. Esta escena ha sido seleccionada por su gran tamaño y elevada densidad de objetos, lo que permite analizar con mayor profundidad las capacidades de explotación de dicho mapa.

Razonamiento espacial. En tareas que requieren razonamiento espacial, los LLMs obtienen mejores resultados cuando disponen de un mapa semántico-topológico enriquecido (Escenario 2), en lugar de uno que solo contiene información básica de los objetos (Escenario 1). Esta mejora es especialmente notable en el caso del modelo abierto Qwen 2.5 14B, cuyas respuestas en el Escenario 1 suelen ser incorrectas o inconsistentes debido a la falta de contexto espacial adicional. En cambio, con modelos más potentes como Gemini 2.0 Flash, la diferencia entre ambos escenarios es menos pronunciada. No obstante, en el Escenario 1 sigue observándose una mayor variabilidad en las respuestas, ya que el modelo tiende a generar múltiples interpretaciones posibles ante la ausencia de información topológica explícita. Un ejemplo representativo de este tipo de razonamiento en la escena considerada se basa en los bancos alineados bajo una encimera, como se muestra en la Figura 24. Los identificadores de los bancos en la escena, de izquierda a derecha según se visualizan en la imagen, son: `obj28`, `obj29`, `obj31` y `obj30`. Ante la consulta formulada en lenguaje natural:

¿Qué banco se encuentra entre otros dos bancos en la escena?

Which stool is between two other stools in the scene?

las respuestas proporcionadas por los modelos en ambos escenarios se resumen en la Tabla 3. Nótese que las respuestas correctas incluirán los objetos con identificadores `obj29` o `obj31`, ya que son los únicos que están rodeados de otros bancos en la fila de bancos, pero no los objetos con identificadores `obj28` ni `obj30` por estar situados en los extremos de la fila (véase la Figura 24).

Tabla 3: Respuesta de los LLMs ante la petición "*Which stool is between two other stools in the scene?*" en la escena `scannet_scene_0000` de ScanNet.

Modelo	**Escenario 1 (mapa básico)**	**Escenario 2 (mapa enriquecido)**
Gemini 2.0 Flash	(correcta) *The stool obj29 is between obj28 and obj31, as it is spatially next to both of them.*	(correcta) *The stool obj31 is between obj30 and obj29.*
Qwen 2.5 14B	(incorrecta) *Stool obj30 is between stool obj29 and stool obj31 in the scene.*	(correcta) *Stool obj29 is between stool obj28 and stool obj31 in the scene.*

Como se puede observar, el LLM con menor número de parámetros falla cuando únicamente dispone de un mapa semántico básico (Escenario 1), pero acierta cuando se le proporciona un mapa enriquecido con información topológica (Escenario 2). Por su parte, el LLM de mayor tamaño acierta en ambos escenarios, lo que indica una mayor robustez ante la falta de contexto estructural. Este ejemplo ilustra cómo las relaciones espaciales inferidas en el mapa semántico-topológico permiten al modelo identificar correctamente cuál de los bancos se encuentra en una posición intermedia respecto a los demás.

Razonamiento estructural. Otro caso en el que el enriquecimiento del mapa con información topológica resulta especialmente útil es cuando el usuario solicita una lista de objetos que cumplen una determinada relación estructural, como estar sobre otro objeto. Este tipo de razonamiento puede evaluarse mediante la siguiente consulta:

¿Qué objetos están encima de la encimera?

What objects are on the countertop?

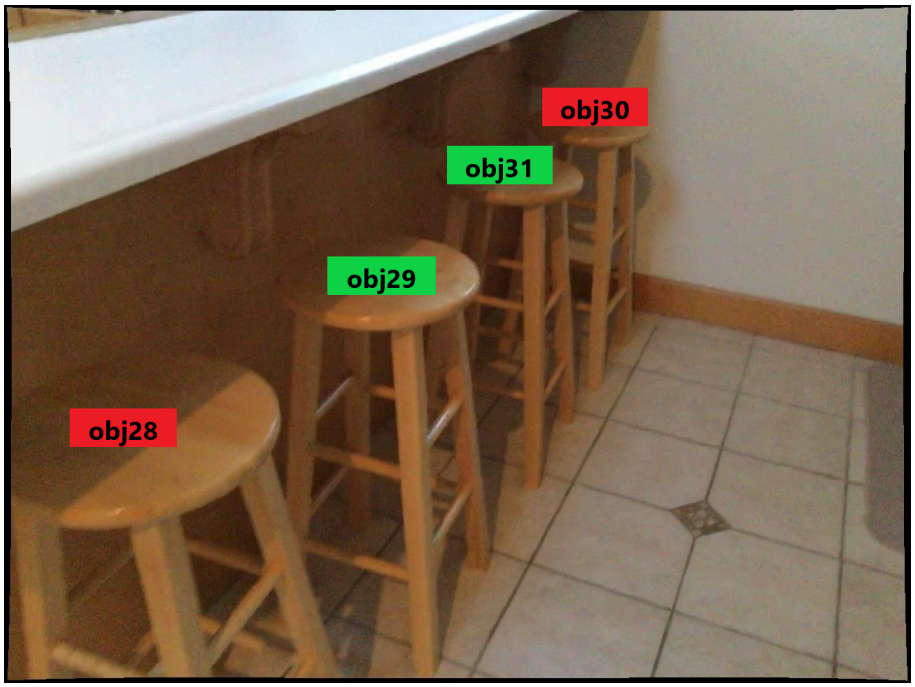


Figura 24: Imagen de la escena `scannet_scene0000_00` de ScanNet, donde se recalcan los objetos correctos (verde) ante la petición "*Which stool is between two other stools in the scene?*" y los objetos incorrectos (rojo).

En el mapa semántico considerado, la respuesta correcta a esta pregunta debe incluir los objetos con identificadores `obj45` ("*toaster oven*"), `obj51` ("*dishrack*") y `obj53` ("*toaster*"), todos ellos ubicados directamente sobre la encimera, como puede verse en la Figura 25. Las respuestas generadas por los LLMs en ambos escenarios se resumen en la Tabla 4.

En este caso, las relaciones semánticas de tipo estructural —como el predicado "soporta" (*supports*)— permiten a los LLMs en el Escenario 2 inferir correctamente qué objetos están sobre la encimera, incluso cuando el mapa no cuenta con una clase explícita "*countertop*" sino su sinónimo "*counter*". Por ejemplo, el objeto `obj42` ("*microwave*") es identificado erróneamente en el Escenario 1 como si estuviera sobre la encimera, cuando en realidad está suspendido en un mueble de cocina. La disponibilidad de relaciones estructurales en el Escenario 2 permite distinguir correctamente este tipo de situaciones, algo más difícil con la información limitada del Escenario 1. En concreto, Qwen 2.5 14B falla completamente en el Escenario 1, mientras que Gemini 2.0 Flash proporciona una respuesta parcialmente correcta. Ambos modelos, sin embargo, aciertan cuando disponen de un mapa semántico enriquecido.

Razonamiento implicando lugares. Ante un razonamiento espacial que implica com-

Tabla 4: Respuesta de los LLMs ante la petición "*What objects are on the countertop?*" en la escena `scannet_scene_0000_00`.

Modelo	**Escenario 1 (mapa básico)**	**Escenario 2 (mapa enriquecido)**
Gemini 2.0 Flash	(incorrecta) *The objects on the countertop are the dishrack (obj51), microwave (obj52), and toaster (obj53).*	(correcta) *The objects on the countertop (obj2 and obj3) include the toaster oven (obj45), dishrack (obj51), and toaster (obj53).*
Qwen 2.5 14B	(incorrecta) *The semantic map does not explicitly define a 'countertop' class, but instances obj2 and obj3 are labeled as 'counter'. No other objects are specified to be directly on these counters based on the given information.*	(correcta) *The objects on the countertop are obj45 (toaster oven), obj51 (dishrack), and obj53 (toaster).*

prensión de los lugares del entorno, se observan resultados similares al ejemplo anterior. Este tipo de razonamiento puede evaluarse mediante una consulta como la siguiente, aplicada sobre el mismo mapa semántico:

> ¿Qué banco está más cerca del cuarto de baño?
> *Which stool is closest to the bathroom?*

Nótese que en este caso, la respuesta correcta mencionaría únicamente el objeto con identificador `obj28` (véase la Figura 26). Las respuestas proporcionadas por los modelos en ambos escenarios se muestran en la Tabla 5.

En este caso, el modelo abierto Qwen 2.5 14B es el único que proporciona una respuesta correcta, y solo cuando dispone de un mapa semántico-topológico enriquecido (Escenario 2). Su éxito en este escenario se debe a que el modelo puede centrarse en razonar sobre la disposición de los elementos, sin tener que inferir simultáneamente la segmentación en lugares, como ocurre en el Escenario 1. Curiosamente, el modelo de mayor tamaño, Gemini 2.0 Flash, no logra identificar correctamente el banco más próximo al baño en ninguno de los escenarios, lo que pone de manifiesto tanto el potencial de los modelos abiertos más recientes como la im-

Figura 25: Imágenes tomadas de la grabación de `scannet_scene_0000_00` con vistas complementarias de los objetos sobre la encimera (verde) y los que no están sobre la encimera (rojo). La petición de usuario realizada fue: "*What objects are on the countertop?*"

portancia de contar con una representación rica del entorno para llevar a cabo razonamientos espaciales complejos.

Otro ejemplo en el que el desempeño del LLM con el mapa enriquecido es superior al del mapa sin enriquecer al considerar lugares es la siguiente consulta:

> ¿Cuántas papeleras hay en la cocina?
> *How many trashcans are in the kitchen?*

En la escena analizada, la respuesta correcta debe identificar dos papeleras, correspondientes a los objetos `obj12` y `obj13`, como se muestra en la Figura 27. Las respuestas proporcionadas por los modelos en ambos escenarios se resumen en la Tabla 6.

En este caso, ambos modelos aciertan cuando disponen del mapa semántico-topológico enriquecido (Escenario 2). Sin embargo, en el Escenario 1, Qwen 2.5 14B comete un error al asumir que el objeto `obj14` también se encuentra en la cocina, cuando en realidad no es así. Este resultado refuerza la idea de que la información topológica sobre los lugares ayuda a los modelos a razonar con mayor precisión y a evitar inferencias incorrectas basadas solo en la proximidad espacial o la clase semántica.

En conjunto, los resultados confirman que la disponibilidad de información topológica mejora significativamente la precisión del razonamiento espacial, permitiendo a los modelos distinguir con mayor fiabilidad la ubicación funcional de los objetos dentro del entorno.

Tabla 5: Respuesta de los LLMs ante la petición "*Which stool is closest to the bathroom?*" en la escena `scannet_scene_0000` de ScanNet.

Modelo	**Escenario 1 (mapa básico)**	**Escenario 2 (mapa enriquecido)**
Gemini 2.0 Flash	(incorrecta) *The closest stool to the bathroom is obj30, located near the counters and closest to the toilet (obj36) and shower (obj4).*	(incorrecta) *The stool closest to the bathroom is obj29.*
Qwen 2.5 14B	(incorrecta) *Obj29, obj30, and obj32 are stools, but obj30 is closest to potential bathroom objects like the toilet (obj36) and shower (obj4).*	(correcta) *The stool closest to the bathroom is obj28, which is located next to counter obj2 in the living room area.*

Tabla 6: Respuesta de los LLMs ante la consulta "*How many trashcans are in the kitchen?*" en la escena analizada.

Modelo	**Escenario 1 (mapa básico)**	**Escenario 2 (mapa enriquecido)**
Gemini 2.0 Flash	(correcta) *There are two garbage cans in the kitchen: obj12 and obj13.*	(correcta) *There are two garbage cans in the kitchen: obj12 and obj13.*
Qwen 2.5 14B	(incorrecta) *There are three trashcans in the kitchen, identified as obj12, obj13, and obj14.*	(correcta) *There are two trashcans in the kitchen area: obj12 and obj13.*

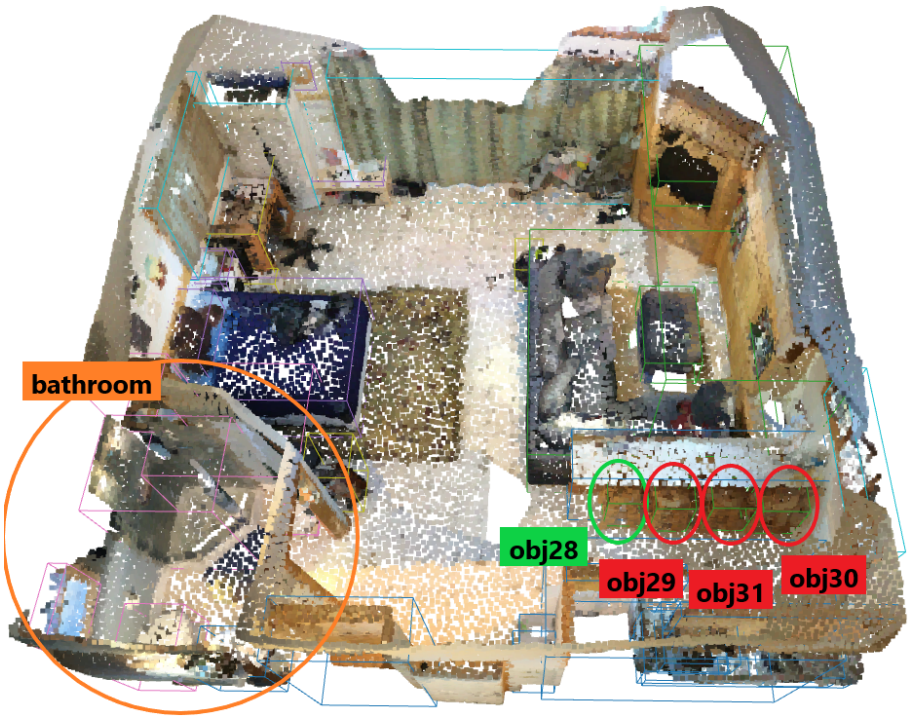


Figura 26: Nube de puntos de la escena `scannet_scene0000_00` de ScanNet, donde se recalcan los objetos correctos (verde) ante la petición "*Which stool is closest to the bathroom?*" (verde), los objetos incorrectos (rojo) y el cuarto de baño (naranja).

Figura 27: Imágenes tomadas de la grabación de `scannet_scene_0000_00` con vistas de las papeleras encontradas en la escena. Ante la petición "*How many trashcans are in the kitchen?*" los objetos correctos aparecen en verde y los incorrectos en rojo.

6

Conclusiones

Este Trabajo Fin de Máster (TFM) ha explorado la construcción automática de mapas semántico-topológicos mediante modelos generativos de inteligencia artificial del estado del arte, como los *Large Language Models* (LLMs) y *Large Vision-Language Models* (LVLMs). Frente a las limitaciones de los enfoques tradicionales —como la necesidad de anotaciones manuales o la restricción de vocabulario cerrado—, la propuesta aprovecha las capacidades de razonamiento y el amplio conocimiento del mundo real de estos modelos para enriquecer los mapas del entorno con información topológica. La incorporación de información topológica se ha abordado desde dos vertientes: por un lado, mediante métodos de segmentación y categorización funcional del espacio basados en los objetos presentes; por otro lado, a través de estrategias para inferir relaciones semánticas abiertas entre objetos.

Para la segmentación y categorización de lugares, se han desarrollado dos estrategias complementarias que amplían el concepto de lugar más allá de las habitaciones, centrándose en grupos de objetos con una funcionalidad compartida. La primera aplica técnicas de *clustering* sobre descriptores geométrico-semánticos, obtenidos mediante LLMs y modelos de *embeddings* que codifican la funcionalidad de los objetos, seguida de una fase de refinamiento para mejorar la coherencia espacial y semántica de las agrupaciones. La segunda plantea un enfoque de extremo a extremo, donde un LLM procesa directamente el mapa semántico completo para generar la segmentación y categorización de los lugares sin pasos intermedios.

Para inferir relaciones semánticas abiertas entre objetos, se han desarrollado dos enfoques complementarios. El primero se basa en utilizar descripciones textuales como entrada a un LLM, que permite inferir relaciones de tipo espacial, estructural, funcional o causal. El segundo combina información visual y textual, de forma que un LVLM pueda razonar multimodalmente sobre estos mismos tipos de relaciones, teniendo en cuenta el contexto visual de los objetos en la escena.

La validación de los métodos desarrollados se ha realizado sobre mapas semánticos obtenidos a partir de los conjuntos de datos ScanNet y SceneNN. La evaluación cuantitativa de los métodos de segmentación de lugares mostró que los mejores resultados se alcanzaron con el enfoque basado exclusivamente en LLMs implementado con el modelo de última generación Gemini 2.0 Flash, así como con el método que combina descriptores contextualizados, *clustering* y una fase de refinamiento. Estos resultados demuestran que la incorporación de información semántica puede mejorar significativamente la tarea de segmentación de lugares en el mapa, y que contextualizar los descriptores semánticos según la funcionalidad de los objetos es una estrategia eficaz para los métodos basados en *clustering*. Por su parte, el método basado en LLMs implementado con un modelo abierto de menor tamaño, como Qwen 2.5 14B, ofreció el peor rendimiento entre todos los evaluados, lo que demuestra que el tamaño de los modelos juega un papel fundamental en su desempeño en esta tarea. Por otro lado, la validación cualitativa de los métodos de categorización de lugares e inferencia de relaciones semánticas confirmó que tanto los LLMs como los LVLMs son herramientas adecuadas para enriquecer el mapa con información semántica de alto nivel.

La evaluación del desempeño de un robot móvil equipado con un LLM como motor de razonamiento en dos escenarios, uno con el mapa semántico de partida, y otro con el mapa semántico enriquecido con información topológica, demostró que la calidad del mapa impacta significativamente en el rendimiento del sistema, especialmente si se emplean modelos de gran escala con menor número de parámetros. Cuando el LLM cuenta con el mapa semántico enriquecido, el sistema es capaz de responder a las preguntas del usuario de una forma más rica y completa, demostrando así la efectividad de los métodos desarrollados para construir mapas semántico-topológicos.

6.1. Futuras líneas de investigación

Aunque los métodos implementados han demostrado ser efectivos y la evaluación ha sido exhaustiva, existen diversas líneas de trabajo que podrían explorarse en el futuro:

1. Ampliar la evaluación realizada en este trabajo mediante un análisis cuantitativo que permita medir cómo la incorporación de información topológica en un mapa semántico influye en el rendimiento de un LLM como motor de razonamiento en tareas de robótica

móvil.

2. A partir de los buenos resultados obtenidos con la inferencia de relaciones semánticas basada en LVLMs, sería interesante explorar métodos de categorización de lugares que incorporen también información visual, además de la textual.

3. Refinar las segmentaciones de lugares teniendo en cuenta elementos inamovibles del entorno, como paredes, techos o suelos, sin perder la generalidad del enfoque propuesto, que no depende del concepto tradicional de habitación.

4. Integrar los métodos propuestos dentro de un flujo de trabajo completo de mapeo semántico, evaluando su rendimiento en escenarios con mapas de entrada con incertidumbres o errores.

5. En la segmentación de lugares, podría incorporarse información adicional sobre los objetos, más allá de su clase, posición y tamaño, como la incertidumbre en su clasificación u otras propiedades que puedan mejorar el proceso.

Referencias

[1] Michael Ahn et al. "Do As I Can, Not As I Say: Grounding Language in Robotic Affordances". En: *6th Conference on Robot Learning (CoRL)*. 2022. DOI: 10.48550/arXiv.2204.01691.

[2] Gregorio Ambrosio-Cestero et al. "Container Based Architecture for Mobile Robotics". En: *XLV Jornadas de Automatica*. 2024.

[3] I. Armeni et al. "3D Scene Graph: A Structure for Unified Semantics, 3D Space, and Camera". En: *IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, págs. 5664-5673. DOI: 10.1109/ICCV.2019.00576.

[4] Frederik Bark, Kevin Daun y Oskar von Stryk. "Affordance-based Actionable Semantic Mapping and Planning for Mobile Rescue Robots". En: *2023 IEEE International Symposium on SSRR*. 2023, págs. 53-60. DOI: 10.1109/SSRR59696.2023.10499938.

[5] Benjamin Bolte et al. *USA-Net: Unified Semantic and Affordance Representations for Robot Memory*. 2023. arXiv: 2304.12164 [cs.RO].

[6] Ricardo J. G. B. Campello, Davoud Moulavi y Joerg Sander. "Density-Based Clustering Based on Hierarchical Density Estimates". En: *Advances in Knowledge Discovery and Data Mining*. Ed. por Jian Pei et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, págs. 160-172. ISBN: 978-3-642-37456-2.

[7] Shivam Chandhok. *SceneGPT: A Language Model for 3D Scene Understanding*. 2024. arXiv: 2408.06926 [cs.CV]. URL: https://arxiv.org/abs/2408.06926.

[8] Kevin Chen et al. "A Behavioral Approach to Visual Navigation with Graph Localization Networks". En: *Proceedings of Robotics: Science and Systems*. FreiburgimBreisgau, Germany, jun. de 2019. DOI: 10.15607/RSS.2019.XV.010.

[9] Lianggangxu Chen et al. "CLIP-Driven Open-Vocabulary 3D Scene Graph Generation via Cross-Modality Contrastive Learning". En: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, págs. 27863-27873. DOI: 10.1109/CVPR52733.2024.02632.

[10] Tomasz Cieslewski et al. “Hydra: A Real-time Spatial Perception System for 3D Scene Graph Construction and Applications to Robot Navigation”. En: *IEEE Transactions on Robotics* 39.1 (2023), págs. 119-138. DOI: 10.1109/TRO.2022.3147152.

[11] Angela Dai et al. “ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes”. En: *CVPR*. 2017, págs. 2432-2443. DOI: 10.1109/CVPR.2017.261.

[12] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. En: *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Vol. 1. 2019, págs. 4171-4186. DOI: 10.18653/v1/N19-1423.

[13] Thanh-Toan Do, Anh Nguyen y Ian Reid. “AffordanceNet: An End-to-End Deep Learning Approach for Object Affordance Detection”. En: *ICRA*. 2018.

[14] Martin Ester et al. “A density-based algorithm for discovering clusters in large spatial databases with noise”. En: *kdd*. Vol. 96. 34. 1996, págs. 226-231.

[15] R. Fermin-Leon, R. Martinez-Cantin y T. Duckett. “Structural Representations for Semantic Mapping”. En: *Robotics and Autonomous Systems* 87 (2017), págs. 162-172.

[16] Cipriano Galindo, Juan-Antonio Fernandez-Madrigal y Javier Gonzalez. “Multihierarchical Interactive Task Planning: Application to Mobile Robotics”. En: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 38.3 (2008), págs. 785-798. DOI: 10.1109/TSMCB.2008.920227.

[17] Cipriano Galindo et al. “Multi-Hierarchical Semantic Maps for Mobile Robotics”. En: *2005 IEEE/RSJ IROS*. 2005, págs. 2278-2283. DOI: 10.1109/IROS.2005.1545511.

[18] Qiao Gu et al. “ConceptGraphs: Open-Vocabulary 3D Scene Graphs for Perception and Planning”. En: *IEEE International Conference on Robotics and Automation (ICRA)*. 2024. DOI: 10.1109/ICRA57147.2024.10610243.

[19] Kaiming He et al. “Mask R-CNN”. En: *IEEE International Conference on Computer Vision (ICCV)*. 2017, págs. 2961-2969. DOI: 10.48550/arXiv.1703.06870.

[20] Binh-Son Hua et al. “SceneNN: A Scene Meshes Dataset with aNNotations”. En: *2016 Fourth International Conference on 3D Vision*. 2016, págs. 92-101. DOI: 10.1109/3DV.2016.18.

[21] Wenlong Huang et al. “Inner Monologue: Embodied Reasoning through Planning with Language Models”. En: *6th Conference on Robot Learning (CoRL)*. 2022. DOI: 10.48550/arXiv.2207.05608.

[22] Lawrence Hubert y Phipps Arabie. “Comparing partitions”. En: *Journal of Classification* 2.1 (1985), págs. 193-218. ISSN: 1432-1343. DOI: 10.1007/BF01908075.

[23] N. Hughes, Y. Chang y L. Carlone. “Hydra: A real-time spatial perception system for 3D scene graph construction and optimization”. En: *arXiv preprint* arXiv:2201.13360 (2022). URL: https://arxiv.org/abs/2201.13360.

[24] Ziwei Ji et al. “Survey of Hallucination in Natural Language Generation”. En: *ACM Computing Surveys* 55.12 (2023), págs. 1-38. DOI: 10.1145/3571730.

[25] Justin Johnson et al. “Image retrieval using scene graphs”. En: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, págs. 3668-3678. DOI: 10.1109/CVPR.2015.7298990.

[26] Bo Liu et al. “LLM+P: Empowering Large Language Models with Optimal Planning Proficiency”. En: *arXiv preprint arXiv:2304.11477* (2023). URL: https://arxiv.org/abs/2304.11477.

[27] Yinhan Liu et al. “Roberta: A robustly optimized bert pretraining approach”. En: (2019). DOI: 10.48550/arXiv.1907.11692.

[28] Andrzej Maćkiewicz y Waldemar Ratajczak. “Principal components analysis (PCA)”. En: *Computers & Geosciences* 19.3 (1993), págs. 303-342. ISSN: 0098-3004. DOI: https://doi.org/10.1016/0098-3004(93)90090-R.

[29] Aman Madaan et al. “Self-Refine: Iterative Refinement with Self-Feedback”. En: *37th International Conference on Neural Information Processing Systems*. 2023. DOI: 10.5555/3666122.3668141.

[30] M. Mancini et al. “Boosting Domain Adaptation by Discovering Latent Domains”. En: *IEEE / CVF CVPR* (2017), págs. 3771-3780.

[31] Jose-Luis Matez-Bandera, Javier Monroy y Javier Gonzalez-Jimenez. "Sigma-FP: Robot Mapping of 3D Floor Plans With an RGB-D Camera Under Uncertainty". En: *IEEE Robotics and Automation Letters* 7.4 (2022), págs. 12539-12546. DOI: 10.1109/LRA.2022.3220156.

[32] Jose-Luis Matez-Bandera et al. "Voxeland: Probabilistic Instance-Aware Semantic Mapping with Evidence-based Uncertainty Quantification". En: *Robotics and Autonomous Systems* (2024). Submitted and under review. DOI: https://doi.org/10.48550/arXiv.2411.08727.

[33] John McCormac et al. "SemanticFusion: Dense 3D Semantic Mapping with Convolutional Neural Networks". En: *IEEE International Conference on Robotics and Automation (ICRA)*. 2017, págs. 4628-4635. DOI: 10.1109/ICRA.2017.7989538.

[34] Jesus Moncada-Ramirez et al. "Agentic Workflows for Improving Large Language Model Reasoning in Robotic Object-Centered Planning". En: *Robotics* 14.3 (2025). ISSN: 2218-6581. DOI: 10.3390/robotics14030024.

[35] David Morilla-Cabello et al. "Robust Fusion for Bayesian Semantic Mapping". En: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2023, págs. 76-81. DOI: 10.1109/IROS55552.2023.10342253. URL: https://ieeexplore.ieee.org/document/10342253/.

[36] O.M. Mozos, C. Stachniss y W. Burgard. "Supervised Learning of Places from Range Data using AdaBoost". En: *IEEE International Conference on Robotics and Automation* (2005), págs. 1730-1735.

[37] OpenAI. "GPT-4 Technical Report". En: *OpenAI Technical Report* (2023). URL: https://openai.com/research/gpt-4.

[38] Amit Kumar Pandey y Rachid Alami. "Affordance graph: A framework to encode perspective taking and effort based affordances for day-to-day human-robot interaction". En: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013, págs. 2180-2187. DOI: 10.1109/IROS.2013.6696661.

[39] Shishir G. Patil et al. "Gorilla: Large Language Model Connected with Massive APIs". En: *arXiv preprint arXiv:2305.15334* (2023). DOI: 10.48550/arXiv.2305.15334.

[40] A. Pronobis y P. Jensfelt. "Large-scale Semantic Mapping and Reasoning with Heterogeneous Modalities". En: *IEEE ICRA* (2012), págs. 3515-3522.

[41] A. Pronobis et al. "Multi-modal semantic place classification". En: *IJRR* 29.2-3 (2010), págs. 298-320.

[42] Jingxing Qian et al. "POCD: Probabilistic Object-Level Change Detection and Volumetric Mapping in Semi-Static Scenes". En: *Proceedings of Robotics: Science and Systems XVIII*. 2022. DOI: 10.15607/RSS.2022.XVIII.013.

[43] Jingxing Qian et al. "POV-SLAM: Probabilistic Object-Aware Variational SLAM in Semi-Static Environments". En: *Proceedings of Robotics: Science and Systems XIX*. 2023. DOI: 10.15607/RSS.2023.XIX.072.

[44] Alec Radford et al. "Improving Language Understanding by Generative Pre-Training". En: *OpenAI Technical Report* (2018). URL: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

[45] Alec Radford et al. "Language Models are Unsupervised Multitask Learners". En: *OpenAI Technical Report* (2019). URL: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

[46] Krishan Rana et al. "SayPlan: Grounding Large Language Models using 3D Scene Graphs for Scalable Robot Task Planning". En: *Conference on Robot Learning (CoRL)*. 2023. DOI: 10.48550/arXiv.2307.06135.

[47] Nils Reimers e Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". En: *2019 Conference on EMNLP and 9th IJCNLP*. 2019, págs. 3982-3992.

[48] Shaoqing Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks". En: *IEEE transactions on pattern analysis and machine intelligence* 39.6 (2016), págs. 1137-1149. DOI: 10.1109/TPAMI.2016.2577031.

[49] Andrew Rosenberg y Julia Hirschberg. "V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure". En: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Ed. por Jason Eisner. Prague, Czech Republic: Association for Computational Linguistics, jun. de 2007, págs. 410-420.

[50] Antoni Rosinol et al. “Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping”. En: *Robotics: Science and Systems*. 2020.

[51] Francisco Rubio, Francisco Valero y Carlos Llopis-Albert. “A review of mobile robots: Concepts, methods, theoretical framework, and applications”. En: *International Journal of Advanced Robotic Systems* 16.2 (2019). DOI: 10.1177/1729881419839596.

[52] Jose-Raul Ruiz-Sarmiento, Cipriano Galindo y Javier Gonzalez-Jimenez. “Building Multiversal Semantic Maps for Mobile Robot Operation”. En: *Knowledge-Based Systems* 119 (2017), págs. 257-272. DOI: 10.1016/j.knosys.2016.12.016.

[53] Martin Rünz, Maud Buffier y Lourdes Agapito. “MaskFusion: Real-Time Recognition, Tracking and Reconstruction of Multiple Moving Objects”. En: *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 2018, págs. 10-20. DOI: 10.1109/ISMAR.2018.00023.

[54] Tom Silver et al. “Generalized Planning in PDDL Domains with Pretrained Large Language Models”. En: *38th AAAI Conference on Artificial Intelligence*. 2024. DOI: 10.1609/aaai.v38i18.30006.

[55] N. Sünderhauf et al. “On the Performance of ConvNets in Place Recognition”. En: *IEEE ICRA* (2016), págs. 4297-4304.

[56] Niko Sünderhauf et al. “Meaningful Maps With Object-Oriented Semantic Mapping”. En: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016. DOI: 10.48550/arXiv.1609.07849.

[57] Moritz Tenorth y Michael Beetz. “KNOWROB — knowledge processing for autonomous personal robots”. En: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2009, págs. 4261-4266. DOI: 10.1109/IROS.2009.5354602.

[58] Mike Uschold y Michael Gruninger. “Ontologies: principles, methods and applications”. En: *The Knowledge Engineering Review* 11.2 (1996), págs. 93-136. DOI: 10.1017/S0269888900007797.

[59] S. Vasudevan y R. Siegwart. “Bayesian space conceptualization and place classification for semantic maps in mobile robotics”. En: *Robotics and Autonomous Systems* 56.6 (2006), págs. 522-537.

[60] Ashish Vaswani et al. “Attention is All You Need”. En: *31st International Conference on Neural Information Processing Systems*. Vol. 30. 2017, págs. 5998-6008. DOI: 10.48550/arXiv.1706.03762.

[61] Jason Wei et al. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”. En: *36th International Conference on Neural Information Processing Systems (NeurIPS)*. 2022, págs. 24824-24837. DOI: 10.5555/3546258.3628351.

[62] Chao Xu et al. *PartAfford: Part-level Affordance Discovery from 3D Objects*. 2022. arXiv: 2202.13519 [cs.CV].

[63] Chao Xu et al. “PartAfford: Part-level Affordance Discovery from 3D Objects”. En: (feb. de 2022). DOI: 10.48550/arXiv.2202.13519.

[64] Rowan Zellers et al. “Neural Motifs: Scene Graph Parsing with Global Context”. En: jun. de 2018, págs. 5831-5840. DOI: 10.1109/CVPR.2018.00611.

[65] Bolei Zhou et al. “Learning Deep Features for Scene Recognition using Places Database”. En: *Advances in Neural Information Processing Systems*. Ed. por Z. Ghahramani et al. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper_files/paper/2014/file/19ea3982b415d7bb3363917eb3d60c4a-Paper.pdf.

Apéndice A
Código del proyecto

A.1. Código de los métodos desarrollados

Todo el código correspondiente a los métodos propuestos para la segmentación y categorización de lugares en mapas semánticos, la inferencia de relaciones entre objetos y la evaluación de resultados está disponible en un repositorio público de GitHub accesible en: https://github.com/MAPIRlab/generative-topological-maps.git.

A.2. Código para el tratamiento de conjuntos de datos

Además del repositorio principal, este TFM contribuye de forma significativa al desarrollo de dos repositorios públicos en GitHub, centrados en el procesamiento de datos de los conjuntos ScanNet (https://github.com/josematez/ScanNet.git) y SceneNN (https://github.com/josematez/SceneNN.git), adaptando el formato de los mapas a los requerimientos específicos del proyecto.

Apéndice B
Prompts

En este apéndice se detallan los distintos *prompts* empleados en los métodos desarrollados en este trabajo para: la segmentación y categorización de un mapa semántico en lugares (véase el Capítulo 3); la inferencia de relaciones semánticas entre objetos del mapa (véase el Capítulo 4); y la explotación de dichos mapas por parte de un LLM ante consultas de un usuario (véase el Capítulo 5). Dado que estos *prompts* son clave para guiar el comportamiento de los modelos de gran escala, su diseño se ha basado en técnicas de ingeniería de *prompts* (*prompt engineering*), descritas en el Apéndice C, con el objetivo de asegurar que las respuestas generadas por los modelos sean precisas, coherentes y relevantes.

B.1. *Prompt* de categorización de lugares

El siguiente *prompt* se utiliza en el método de categorización de lugares (véase la Sección 3.2.3) después de la segmentación generada por el método basado en *clustering* de descriptores semántico-geométricos, tal y como se especifica en la Sección 3.2.

```
<INSTRUCTION>
You are provided with a set of objects that co-occupy a single location
    in a 3D scene.
Each object comes with a semantic class label and a 3D bounding box
    (center and size).

Your goal is to:
1. Assign a **short tag** (1-2 words) that captures the primary
    **function** or **purpose** of this space.
2. Write a **functional description** (1-2 sentences) explaining **what
    activities** or **uses** the area supports, **based on how the
    objects work together**.
   - **Do not** merely list the —objectsyou must infer the space's
    intended use.
   - Only use standard room names (e.g., “kitchen”, “”bedroom) if the
```

```
    function is unmistakably clear; otherwise, describe by function.
</INSTRUCTION>

<INPUT_FORMAT>
Objects are provided in JSON, each entry includes:
- **bbox**: { center: [x, y, z], size: [width, depth, height] }
- **class**: object semantic label
</INPUT_FORMAT>

<OUTPUT_FORMAT>
Return JSON with these two fields:
{
  "tag": "<short_functional_tag>",
  "description": "<functional_description_of_the_space>"
}
</OUTPUT_FORMAT>

<EXAMPLES>
Example 1
Input:
{
  "obj0": { "bbox": { "center": [1,2,0], "size": [0.5,0.5,1] }, "class":
   "chair" },
  "obj1": { "bbox": { "center": [1.5,2,0], "size": [0.5,0.5,1] },
   "class": "chair" },
  "obj2": { "bbox": { "center": [1.25,2.5,0], "size": [1,0.5,0.1] },
   "class": "table" },
  "obj3": { "bbox": { "center": [1.25,1.5,0], "size": [0.8,0.3,0.05] },
   "class": "keyboard" },
  "obj4": { "bbox": { "center": [1.25,1.2,0], "size": [0.4,0.2,0.1] },
   "class": "mouse" },
  "obj5": { "bbox": { "center": [1.75,1.5,0.5], "size": [0.6,0.4,0.3] },
   "class": "monitor" }
}
Output:
{
  "tag": "workstation",
```

```
  "description": "A computer workstation area set up for digital tasks,
    with seating around a desk equipped for typing and screen-based work."
}

Example 2
Input:
{
  "obj0": { "bbox": { "center": [0,0,0], "size": [2,2,0.5] }, "class":
    "bed" },
  "obj1": { "bbox": { "center": [1,0,0], "size": [0.5,0.5,0.5] },
    "class": "nightstand" },
  "obj2": { "bbox": { "center": [1,0.2,0.5], "size": [0.2,0.2,0.2] },
    "class": "lamp" },
  "obj3": { "bbox": { "center": [-0.5,0,-0.2], "size": [1,0.5,1.5] },
    "class": "wardrobe" }
}
Output:
{
  "tag": "rest_area",
  "description": "A restful sleeping zone designed for rest and personal
    storage, with a bed for sleeping and nearby surfaces for lighting and
    clothes organization."
}
</EXAMPLES>

<TASK>
Generate the tag and functional description for a location containing the
    following objects:
{{objects}}
</TASK>
```

B.2. *Prompt* de segmentación y categorización de lugares

El siguiente *prompt* es utilizado en el método de segmentación y categorización de lugares totalmente basado en LLMs, tal y como se especifica en la Sección 3.3.

```
<INSTRUCTION>
```

```
You are given a set of detected objects, each with a semantic class and a
    3D bounding box. Your goal is to group these objects into —
    placesmeaningful clusters that share a common functionality and are
    spatially proximate. A place might be “seating_area”, “workstation”,
    or “storage_zone”, not necessarily a single room.
When grouping:
- Semantic similarity: Objects serving similar functionality (e.g., chair
    + couch → seating area) should be in the same place.
- Spatial proximity: Objects that are near each other in 3D space and
    functionally related reinforce the same grouping.
- Completeness and exclusivity: Every object in the semantic_map must be
    assigned to exactly one place; no object may appear in more than one
    place.
</INSTRUCTION>

<GUIDELINES>
You may perform internal, step-by-step reasoning, but your final response
    must contain only the JSON object in the required structure. Do not
    include any extra text before or after the JSON.
</GUIDELINES>

<INPUT_FORMAT>
You will receive a JSON object named semantic_map with this structure:
{
  "instances": {
    "obj0": {
      "bbox": { "center": [x, y, z], "size": [dx, dy, dz] },
      "class": "object_class"
    },
    "obj1": { … },
    …
  }
}
- bbox.center: [x, y, z] position in meters
- bbox.size: [width, depth, height]
- class: semantic label (e.g., "chair", "table")
</INPUT_FORMAT>
```

```
<OUTPUT_FORMAT>
Return exactly this JSON structure, listing each place with a short tag,
   a human-readable description, and the list of object IDs it contains.
   Do not add comments or extra keys.

{
  "places": [
    {
      "name": "<place_tag>",
      "description": "<brief description of this place>",
      "objects": ["obj1", "obj2", "obj3"]
    },
    {
      "name": "<another_tag>",
      "description": "<brief description>",
      "objects": ["obj4", "obj5"]
    }
    // ... more places ...
  ]
}

- name: short, unique tag for the place (e.g., "seating_area",
   "kitchen_zone")
- description: one or two phrases describing the place and its function
- objects: list of object IDs in that place
</OUTPUT_FORMAT>

<TASK>
Classify the following semantic_map into places using the rules above.
   Only output the final JSON in the specified format.

{{semantic_map}}
</TASK>
```

B.3. *Prompt* de inferencia de relaciones semánticas basada en LLMs

El siguiente *prompt* se utiliza en el método de inferencia de relaciones semánticas basado en LLMs, tal y como se especifica en la Sección 4.2.

```
<MAIN_INSTRUCTION>
Given two objects in a semantic map, identified by both name and unique
    object ID, infer all possible directed relationships in both
    directions:
1) First using object1 as the source and object2 as the target.
2) Then using object2 as the source and object1 as the target.
</MAIN_INSTRUCTION>

<INFORMATION_SOURCES>
Use all available information, including:
- Geometric properties (centroid position, size, relative position,
    overlap, distance).
- Semantic labels (class names, known functional roles).
</INFORMATION_SOURCES>

<RELATIONSHIP_TYPES>
Consider subtle relationships, providing an open vocabulary description
    (predicate):
- Spatial: positional arrangements (e.g., "is on", "is below", "is
    above", "is adjacent to", "is next to", "is connected to", "is
    aligned with", "is inside", "is outside", "is part of a group with").
- Structural: assembly or support connections (e.g., "is part of", "is
    mounted on", "is supported by", "is attached to", "is composed of",
    "contains", "holds", "forms part of").
- Functional: usage or role connections (e.g., "is used with", "is
    required for", "enables", "interacts with", "powers", "provides", "is
    for", "can be operated by").
- Causal: cause-effect interactions (e.g., "activates", "turns on",
    "turns off", "triggers", "initiates", "stops", "changes state of",
    "produces").
</RELATIONSHIP_TYPES>
```

```
<NO_RELATION_RULE>
If no relationships are observed, return an empty array: [].
</NO_RELATION_RULE>

<OUTPUT_FORMAT_DETAILS>
For every relation detected, include:
- source_id: the source object ID
- source: the source object name
- target_id: the target object ID
- target: the target object name
- type: one of "spatial", "structural", "functional", "causal"
- predicate: an open vocabulary description of the relation
</OUTPUT_FORMAT_DETAILS>

<OUTPUT_INSTRUCTION>
Return a JSON array of relation objects. Include as many entries as apply.
</OUTPUT_OUTPUT_INSTRUCTION>

<EXAMPLES>
Example 1: Objects on a Desk (Spatial & Structural)

Input:
- object1_id: "obj1"
- object1_name: "laptop"
- centroid1: [0.5, 0.7, 1.2]
- size1: [0.3, 0.2, 0.03]
- object2_id: "obj2"
- object2_name: "desk"
- centroid2: [0.5, 0.5, 1.0]
- size2: [1.0, 0.6, 0.1]

Expected Output:
[
  {
    "source_id": "obj1",
    "source": "laptop",
    "target_id": "obj2",
```

```
    "target": "desk",
    "type": "spatial",
    "predicate": "is on"
  },
  {
    "source_id": "obj2",
    "source": "desk",
    "target_id": "obj1",
    "target": "laptop",
    "type": "structural",
    "predicate": "supports"
  }
]

Example 2: Light Switch and Lamp (Causal)

Input:
- object1_id: "obj3"
- object1_name: "light switch"
- centroid1: [1.0, 0.8, 1.5]
- size1: [0.05, 0.05, 0.02]
- object2_id: "obj4"
- object2_name: "lamp"
- centroid2: [2.5, 1.0, 1.8]
- size2: [0.3, 0.3, 0.5]

Expected Output:
[
  {
    "source_id": "obj3",
    "source": "light switch",
    "target_id": "obj4",
    "target": "lamp",
    "type": "causal",
    "predicate": "turns on"
  },
  {
```

```
    "source_id": "obj4",
    "source": "lamp",
    "target_id": "obj3",
    "target": "light switch",
    "type": "causal",
    "predicate": "is controlled by"
  }
]

Example 3: Fork and Knife (Functional)

Input:
- object1_id: "obj5"
- object1_name: "fork"
- centroid1: [0.1, 0.2, 0.3]
- size1: [0.02, 0.2, 0.01]
- object2_id: "obj6"
- object2_name: "knife"
- centroid2: [0.15, 0.2, 0.3]
- size2: [0.02, 0.2, 0.01]

Expected Output:
[
  {
    "source_id": "obj5",
    "source": "fork",
    "target_id": "obj6",
    "target": "knife",
    "type": "functional",
    "predicate": "is used with"
  },
  {
    "source_id": "obj6",
    "source": "knife",
    "target_id": "obj5",
    "target": "fork",
    "type": "functional",
```

```
    "predicate": "is used with"
  }
]

Example 4: Two Distant, Unrelated Objects (No Relationships)

Input:
- object1_id: "obj7"
- object1_name: "book"
- centroid1: [10.0, 5.0, 2.0]
- size1: [0.2, 0.15, 0.05]
- object2_id: "obj8"
- object2_name: "mountain"
- centroid2: [100.0, 200.0, 50.0]
- size2: [5000.0, 5000.0, 2000.0]

Expected Output:
[]
</EXAMPLES>

<INPUT_DATA_SECTION>
Now infer relationships for the following objects:
- object1_id: "{{object1_id}}"
- object1_name: "{{object1_name}}"
- centroid1: {{object1_centroid}}
- size1: {{object1_size}}
- object2_id: "{{object2_id}}"
- object2_name: "{{object2_name}}"
- centroid2: {{object2_centroid}}
- object2_size: {{object2_size}}
</INPUT_DATA_SECTION>

<FINAL_INFERENCE_INSTRUCTION>
Think carefully about positional context, size compatibility, and class
   semantics to infer all possible relations.
Return a well-structured JSON array of the detected relationships, or []
   if none apply.
```

```
</FINAL_INFERENCE_INSTRUCTION>
```

B.4. *Prompt* de inferencia de relaciones semánticas basada en LVLMs

El siguiente *prompt* se utiliza en el método de inferencia de relaciones semánticas basado en LVLMs, tal y como se especifica en la Sección 4.3.

```
<MAIN_INSTRUCTION>
Given two objects in a semantic map, identified by both name and unique
    object ID, infer all possible directed relationships in both
    directions:
1) First using object1 as the source and object2 as the target.
2) Then using object2 as the source and object1 as the target.
</MAIN_INSTRUCTION>

<INFORMATION_SOURCES>
Use all available information, including:
- Geometric properties (centroid position, size, relative position,
    overlap, distance).
- Semantic labels (class names, known functional roles).
- Visual information from provided images of the scene where the objects
    appear.
</INFORMATION_SOURCES>

<RELATIONSHIP_TYPES>
Consider subtle relationships, providing an open vocabulary description
    (predicate):
- Spatial: positional arrangements (e.g., "is on", "is below", "is
    above", "is adjacent to", "is next to", "is connected to", "is
    aligned with", "is inside", "is outside", "is part of a group with").
- Structural: assembly or support connections (e.g., "is part of", "is
    mounted on", "is supported by", "is attached to", "is composed of",
    "contains", "holds", "forms part of").
- Functional: usage or role connections (e.g., "is used with", "is
    required for", "enables", "interacts with", "powers", "provides", "is
    for", "can be operated by").
- Causal: cause-effect interactions (e.g., "activates", "turns on",
```

```
   "turns off", "triggers", "initiates", "stops", "changes state of",
   "produces").
</RELATIONSHIP_TYPES>

<NO_RELATION_RULE>
If no relationships are observed, return an empty array: [].
</NO_RELATION_RULE>

<OUTPUT_FORMAT_DETAILS>
For every relation detected, include:
- source_id: the source object ID
- source: the source object name
- target_id: the target object ID
- target: the target object name
- type: one of "spatial", "structural", "functional", "causal"
- predicate: an open vocabulary description of the relation
</OUTPUT_FORMAT_DETAILS>

<OUTPUT_INSTRUCTION>
Return a JSON array of relation objects. Include as many entries as apply.
</OUTPUT_OUTPUT_INSTRUCTION>

<EXAMPLES>
Example 1: Objects on a Desk (Spatial & Structural)

Input:
- object1_id: "obj1"
- object1_name: "laptop"
- centroid1: [0.5, 0.7, 1.2]
- size1: [0.3, 0.2, 0.03]
- object2_id: "obj2"
- object2_name: "desk"
- centroid2: [0.5, 0.5, 1.0]
- size2: [1.0, 0.6, 0.1]
- image_context: (Imagine an image showing a laptop open on a desk)

Expected Output:
```

```
[
  {
    "source_id": "obj1",
    "source": "laptop",
    "target_id": "obj2",
    "target": "desk",
    "type": "spatial",
    "predicate": "is on"
  },
  {
    "source_id": "obj2",
    "source": "desk",
    "target_id": "obj1",
    "target": "laptop",
    "type": "structural",
    "predicate": "supports"
  }
]

Example 2: Light Switch and Lamp (Causal)

Input:
- object1_id: "obj3"
- object1_name: "light switch"
- centroid1: [1.0, 0.8, 1.5]
- size1: [0.05, 0.05, 0.02]
- object2_id: "obj4"
- object2_name: "lamp"
- centroid2: [2.5, 1.0, 1.8]
- size2: [0.3, 0.3, 0.5]
- image_context: (Imagine an image showing a light switch on a wall and a
   lamp nearby)

Expected Output:
[
  {
    "source_id": "obj3",
```

```
    "source": "light switch",
    "target_id": "obj4",
    "target": "lamp",
    "type": "causal",
    "predicate": "turns on"
  },
  {
    "source_id": "obj4",
    "source": "lamp",
    "target_id": "obj3",
    "target": "light switch",
    "type": "causal",
    "predicate": "is controlled by"
  }
]

Example 3: Fork and Knife (Functional)

Input:
- object1_id: "obj5"
- object1_name: "fork"
- centroid1: [0.1, 0.2, 0.3]
- size1: [0.02, 0.2, 0.01]
- object2_id: "obj6"
- object2_name: "knife"
- centroid2: [0.15, 0.2, 0.3]
- size2: [0.02, 0.2, 0.01]
- image_context: (Imagine an image showing a fork and knife placed
   together on a dining table)

Expected Output:
[
  {
    "source_id": "obj5",
    "source": "fork",
    "target_id": "obj6",
    "target": "knife",
```

```
    "type": "functional",
    "predicate": "is used with"
  },
  {
    "source_id": "obj6",
    "source": "knife",
    "target_id": "obj5",
    "target": "fork",
    "type": "functional",
    "predicate": "is used with"
  }
]

Example 4: Two Distant, Unrelated Objects (No Relationships)

Input:
- object1_id: "obj7"
- object1_name: "book"
- centroid1: [10.0, 5.0, 2.0]
- size1: [0.2, 0.15, 0.05]
- object2_id: "obj8"
- object2_name: "mountain"
- centroid2: [100.0, 200.0, 50.0]
- size2: [5000.0, 5000.0, 2000.0]
- image_context: (Imagine two entirely separate images, one of a book,
   another of a mountain range)

Expected Output:
[]
</EXAMPLES>

<INPUT_DATA_SECTION>
Now infer relationships for the following objects, also considering any
   visual information from the scene:
- object1_id: "{{object1_id}}"
- object1_name: "{{object1_name}}"
- centroid1: {{object1_centroid}}
```

```
- size1: {{object1_size}}
- object2_id: "{{object2_id}}"
- object2_name: "{{object2_name}}"
- centroid2: {{object2_centroid}}
- object2_size: {{object2_size}}
- image_context: (This is where the actual image data or a description of
    its content will be provided if available)
</INPUT_DATA_SECTION>

<FINAL_INFERENCE_INSTRUCTION>
Think carefully about positional context, size compatibility, class
    semantics, and **visual cues from the image** to infer all possible
    relations.
Return a well-structured JSON array of the detected relationships, or []
    if none apply.
</FINAL_INFERENCE_INSTRUCTION>
```

B.5. *Prompt* de explotación de mapa semántico

El siguiente *prompt* se utiliza en la evaluación del desempeño de un LLM ante un mapa semántico que no está enriquecido con información topológica, tal y como se especifica en el Escenario 1 de la Sección 5.5.

```
<INSTRUCTION>
You are an LLM assistant specialized in understanding 3D semantic maps.
Your task is to understnad a semantic map and answer questions about it
    with a short sentence.
Include object ids in your answer if relevant.
</INSTRUCTION>

<INPUT_FORMAT>
Objects are provided in JSON, each entry includes:
- **bbox**: { center: [x, y, z], size: [width, depth, height] }
- **class**: object semantic label
</INPUT_FORMAT>
```

```
<INPUT_DATA_SECTION
Consider the following semantic map:
{{semantic_map}}

Now answer this question with a short sentence about the semantic map:
{{question}}
</INPUT_DATA_SECTION>
```

B.6. *Prompt* de explotación de mapa semántico-topológico

El siguiente *prompt* se utiliza en la evaluación del desempeño de un LLM ante un mapa semántico, enriquecido con información topológica, es decir, un mapa semántico-topológico, tal y como se especifica en el Escenario 2 de la Sección 5.5.

```
<INSTRUCTION>
You are provided with a semantic map of a 3D scene. Your goal is to
    reason about the environment using the information provided.
You will receive:
1. A list of objects, each with a semantic label and a 3D bounding box.
2. A segmentation of the scene into functional areas (called "places"),
    each described by:
   - A short **tag** that captures the function of the place (e.g.,
   "kitchen_area").
   - A **description** explaining what kind of objects are present.
   - A list of **object IDs** belonging to that place.
3. A list of **semantic relationships** between pairs of objects, which
    may include spatial, functional, structural, or other types of
    relations. Each relation includes the object IDs, their semantic
    classes, the relation type, and a textual predicate (e.g., "is to the
    left of", "is supported by", "is used with").

You must use this information to answer user queries about the scene with
    a short sentence, leveraging both the spatial organization of objects
    and the higher-level place categories and relations.
Include object ids in your answer if relevant.
</INSTRUCTION>
```

```
<INPUT_FORMAT>
You will receive a JSON object with the following structure:

- "instances": a list of objects present in the scene. Each object
    contains:
     - "id": a unique identifier string for the object.
     - "class": the semantic label or category of the object (e.g.,
    "chair", "table").
     - "bbox": a dictionary representing the 3D bounding box of the
    object, with:
          - "center": a list of three floats indicating the (x, y, z)
    coordinates of the center.
          - "size": a list of three floats indicating the width, depth, and
    height.

- "clusters": a dictionary mapping place identifiers to place
    definitions. Each place includes:
     - "tag": a short string summarizing the function or purpose of the
    place.
     - "description": a textual description of the types of objects or
    activities in the place.
     - "objects": a list of object IDs that belong to this place.

- "relationships": a list of semantic relationships between pairs of
    objects. Each relationship includes:
     - "source_id": the ID of the source object.
     - "source": the semantic class of the source object.
     - "target_id": the ID of the target object.
     - "target": the semantic class of the target object.
     - "type": the category of the relationship (e.g., "spatial",
    "functional", "structural").
     - "predicate": a textual expression describing the relation (e.g.,
    "is next to", "is used with").
</INPUT_FORMAT>

<INPUT_DATA_SECTION
```

```
Consider the following semantic map with places and relationships:
{{semantic_map}}
{{places}}
{{relationships}}

Now answer this question with a short sentence about the semantic map:
{{question}}
</INPUT_DATA_SECTION>
```

Apéndice C
Estrategias de ingeniería de *prompts*

Para aprovechar al máximo el potencial de los modelos de lenguaje de gran escala (LLMs y LVLMs), este trabajo ha adoptado diversas estrategias de ingeniería de *prompts* (*prompt engineering*) orientadas a mejorar la calidad, coherencia y utilidad de las respuestas generadas.

Few-shot prompting. Una de las técnicas clave empleadas ha sido la inclusión de ejemplos explícitos dentro del propio *prompt*, una estrategia conocida como *few-shot prompting*, o aprendizaje en contexto (*in-context learning*). Al mostrar al modelo varios casos resueltos antes de formular la tarea principal, se le proporciona una referencia clara tanto del tipo de razonamiento esperado como del formato de respuesta deseado. Esto contribuye de forma significativa a mejorar su capacidad de generalización frente a nuevas entradas. Por ejemplo, en el *prompt* utilizado para la inferencia de relaciones semánticas (véase el Apéndice B.3), se incluyen varios pares de objetos con información sobre su clase semántica, posición y tamaño, junto con la salida esperada, que consiste en la lista de relaciones identificadas entre ellos. Estos ejemplos cubren diferentes tipos de relaciones —espaciales, estructurales, funcionales y causales—, lo que permite al modelo aprender patrones complejos de interacción y aplicarlos a nuevas escenas.

Chain-of-thought prompting. En tareas que requieren razonamiento no trivial o inferencia espacial y funcional, se ha estimulado el razonamiento paso a paso del modelo mediante la estrategia conocida como *Chain-of-thought prompting*. Aunque la respuesta final debe presentarse en formato estructurado, se permite que el modelo realice internamente un razonamiento progresivo antes de generar la salida definitiva. Esto facilita respuestas más precisas y fundamentadas. Un ejemplo de esta estrategia se encuentra en el *prompt* de segmentación de lugares basado en LLMs (véase el Apéndice B.2), donde se indica explícitamente que el

modelo puede realizar razonamientos internos antes de producir la respuesta final, que debe limitarse a una estructura JSON válida. Esta indicación permite al modelo evaluar con mayor profundidad la coherencia espacial y funcional entre objetos antes de asignarlos a un lugar común.

Guía de buenas prácticas proporcionada por Google para modelos Gemini. Se han seguido recomendaciones recogidas en la guía oficial de estrategias de *prompting* publicada por Google para los modelos Gemini[10]. Entre los principios aplicados destacan: la separación clara de secciones funcionales del *prompt* (por ejemplo, bloques `INSTRUCTION`, `INPUT`, `OUTPUT`), la definición estructurada de formatos de entrada y salida, y la indicación explícita de restricciones o reglas a cumplir por el modelo en su respuesta.

Estructuración de entrada y salida en formato JSON. Con el objetivo de mantener la consistencia y facilitar la integración programática, se ha adoptado de forma sistemática el uso del formato JSON tanto para la entrada de datos como para las respuestas generadas por el modelo. Esta estructura no solo reduce ambigüedades, sino que permite validar automáticamente las salidas y reutilizarlas fácilmente en módulos posteriores del sistema. Por ejemplo, en el *prompt* de categorización de lugares (véase el Apéndice B.2), se especifica que el modelo debe devolver un objeto JSON con dos campos bien definidos: `tag` y `description`. De manera similar, en el *prompt* de inferencia de relaciones semánticas basado en LLMs (véase el Apéndice B.3), tanto la entrada como la salida siguen una estructura jerárquica en JSON, que permite representar relaciones complejas entre objetos de forma clara y procesable por el sistema. Este enfoque estructurado resulta especialmente útil para integrar los resultados generados por el modelo en flujos de trabajo automáticos.

[10] https://ai.google.dev/gemini-api/docs/prompting-strategies

UNIVERSIDAD DE MÁLAGA | uma.es

ESCUELA DE INGENIERÍAS INDUSTRIALES

Escuela de Ingenierías Industriales
Calle Doctor Ortiz Ramos s/n
Campus de Teatinos
29071 Málaga