Universidad Politécnica
de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Ciencia de Datos e Inteligencia Artificial

Trabajo Fin de Grado

Clasificación de documentos del AD-UPM mediante TDA

Autor: MARIO GARCÍA BERENGUER
Tutor: JUAN ANTONIO FERNANDEZ DEL POZO DE SALAMANCA

Madrid, 04 - 2025

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Ciencia de Datos e Inteligencia Artificial

Título: Clasificación de documentos del AD-UPM mediante TDA

04 - 2025

Autor: MARIO GARCÍA BERENGUER
Tutor: JUAN ANTONIO FERNANDEZ DEL POZO DE SALAMANCA
DEPARTAMENTO DE INTELIGENCIA ARTIFICIAL
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid

Resumen

Este Trabajo de Fin de Grado presenta una metodología para la clasificación automática de documentos académicos, aplicando técnicas combinadas de Procesamiento de Lenguaje Natural (PLN) y Análisis Topológico de Datos (TDA). El caso de estudio se centra en el Archivo Digital de la Universidad Politécnica de Madrid (AD-UPM), donde se busca asignar etiquetas institucionales, como la Institución o el Departamento de origen, a partir de la información contenida en el título, el resumen y las conclusiones de cada documento.

El trabajo se estructura como un pipeline modular compuesto por varias etapas. En primer lugar, se lleva a cabo la extracción y estructuración de textos a partir de documentos en formato HTML y PDF. A continuación, se aplican técnicas de representación del lenguaje como TF-IDF, seguidas de procesos de reducción de dimensionalidad y selección de características. Sobre estos vectores se realiza un análisis exploratorio mediante proyecciones semánticas y algoritmos de agrupamiento no supervisado. Posteriormente, se extraen descriptores topológicos mediante complejos de Vietoris-Rips, y se analizan las distancias intra e interclase utilizando métricas como Wasserstein o los paisajes de persistencia. Finalmente, se construye un clasificador supervisado que se basa esta información topológica para mejorar realizar predicciones, en este caso de pertenencia a una Institución concreta, en base a probabilidades generadas por distancias entre Diagramas de Vietoris-Rips.

Más allá del contexto concreto del AD-UPM, la propuesta metodológica desarrollada tiene vocación de generalidad. El pipeline diseñado es totalmente replicable y puede ser adaptado a otros entornos de análisis textual que presenten estructuras similares, como repositorios científicos, sistemas de gestión documental o colecciones especializadas con etiquetas temáticas o institucionales. La combinación de técnicas clásicas de PLN con herramientas de TDA permite capturar tanto patrones locales como relaciones estructurales globales, haciendo del enfoque una alternativa versátil, eficiente e interpretable.

Los resultados obtenidos permiten evaluar el potencial de los descriptores topológicos como complemento a las representaciones textuales tradicionales, y evidencian su utilidad en tareas de clasificación documental en dominios complejos.

Palabras Clave: Procesamiento de Lenguaje Natural, Análisis Topológico de Datos, homología persistente, clasificación documental, Vietoris–Rips, giotto-tda, reducción de dimensionalidad, minería de textos, ODS, clasificación institucional, clustering, Mapper, coeficiente Silhouette, TF-IDF, LSA.

Abstract

Abstract

This Bachelor's Thesis presents a methodology for the automatic classification of academic documents, combining techniques from Natural Language Processing (NLP) and Topological Data Analysis (TDA). The case study focuses on the Digital Archive of the Universidad Politécnica de Madrid (AD-UPM), where the aim is to assign institutional labels, such as School or Department, based on the information contained in the title, abstract, and conclusions of each document.

The project is structured as a modular pipeline consisting of several stages. First, text content is extracted and structured from HTML and PDF documents. Then, textual representations such as TF-IDF are applied, followed by dimensionality reduction and feature selection. An exploratory analysis is carried out through semantic projections and unsupervised clustering algorithms. Afterwards, topological descriptors are extracted using Vietoris–Rips complexes, and inter- and intra-class distances are analyzed using metrics such as the Wasserstein distance and persistence landscapes. Finally, a supervised classifier is trained, integrating topological features to improve the accuracy of label assignment.

Beyond the specific context of AD-UPM, the proposed methodology is designed to be generalizable. The developed pipeline is fully replicable and can be adapted to other textual analysis scenarios with similar structure, such as scientific repositories, document management systems, or thematic collections with institutional or categorical labels. By combining classical NLP techniques with TDA tools, the approach captures both local and global structural patterns, offering a versatile, efficient, and interpretable solution.

The results demonstrate the potential of topological descriptors as a complement to traditional textual representations and highlight their usefulness in document classification tasks in semantically complex domains.

Keywords: Natural Language Processing, Topological Data Analysis, persistent homology, document classification, Vietoris–Rips, giotto-tda, dimensionality reduction, text mining, SDGs, institutional classification, clustering, Mapper algorithm, Silhouette score, TF-IDF, Latent Semantic Analysis.

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

La clasificación de documentos desempeña un papel crucial en la gestión del conocimiento, especialmente en contextos donde la organización y recuperación de información deben ser ágiles y precisas. Ante el incremento sostenido de archivos digitales, como ocurre en el Archivo Digital de la Universidad Politécnica de Madrid (AD-UPM), se hace imprescindible el desarrollo de metodologías que optimicen la categorización automática de los contenidos en función de atributos relevantes, como la Escuela o el Departamento de procedencia.

Este Trabajo de Fin de Grado (TFG) plantea una aproximación novedosa a esta tarea, fundamentada en el Análisis Topológico de Datos (TDA, por sus siglas en inglés). Esta disciplina matemática permite identificar estructuras subyacentes en conjuntos de datos, captando patrones complejos que a menudo escapan a las técnicas convencionales de Procesamiento de Lenguaje Natural (PLN) y aprendizaje automático. Mediante herramientas de topología computacional, se busca enriquecer la representación de los documentos, incorporando información estructural que potencie la calidad de los modelos clasificadores.

El proyecto se articula en varias fases: obtención y tratamiento inicial de los datos extraídos del AD-UPM, generación de representaciones textuales y topológicas, desarrollo de modelos de clasificación, y análisis de su rendimiento. Para llevar a cabo estas tareas, se emplearán bibliotecas especializadas como `giotto-tda` para el análisis topológico, `nltk` y `spaCy` para el tratamiento del lenguaje, y `scikit-learn` para la construcción y evaluación de modelos predictivos.

1.1. Motivación del proyecto

La iniciativa surge de la necesidad creciente de automatizar la organización de repositorios académicos de gran envergadura, como el AD-UPM, que contienen documentos generados por múltiples Escuelas y Departamentos. Esta diversidad se traduce en una gran variabilidad estructural y semántica, lo que dificulta el uso de métodos clásicos de clasificación.

Frente a estas limitaciones, el TDA ofrece un marco prometedor para capturar relaciones complejas entre documentos que no son evidentes a través de enfoques meramente estadísticos o lingüísticos. La integración de este enfoque puede contribuir

significativamente a mejorar la accesibilidad, navegabilidad y mantenimiento del repositorio, facilitando su utilización por parte de la comunidad universitaria.

1.2. Contexto del proyecto

El trabajo se inscribe dentro del Grado en Ciencia de Datos de la Universidad Politécnica de Madrid. El objeto de estudio es el AD-UPM, un repositorio institucional que alberga miles de documentos académicos, entre los que se incluyen TFGs, TFMs, tesis doctorales y otros trabajos científicos.

Se analizará una muestra representativa de documentos procedentes de distintas Escuelas y cursos académicos. A partir de variables como el título, resumen y metadatos, se aplicarán técnicas de PLN y TDA para diseñar modelos de clasificación capaces de asignar etiquetas como la Escuela o el Departamento correspondiente. Esta metodología permitirá tanto el análisis exploratorio como el entrenamiento de clasificadores supervisados.

1.3. Objetivos

El objetivo general del trabajo es desarrollar una metodología de clasificación de documentos del AD-UPM que combine técnicas de PLN y TDA, mejorando así la precisión y eficiencia del etiquetado institucional. Los objetivos específicos son:

- Estudiar la estructura del AD-UPM y seleccionar una muestra representativa de documentos.
- Realizar tareas de limpieza, preprocesamiento y selección de variables sobre los datos textuales.
- Generar representaciones topológicas de los documentos.
- Analizar la estructura de los datos mediante métodos no supervisados basados en TDA.
- Construir y evaluar un modelo de clasificación supervisado que integre información topológica, y posiblemente textual.
- Comparar el rendimiento de distintas metodologías y configuraciones de modelo.

1.4. Estructura del Documento

La memoria se organiza en varios capítulos, cada uno de los cuales aborda una fase concreta del proyecto. La estructura propuesta es la siguiente:

- **Capítulo sobre trabajos previos:** Se revisan las técnicas actuales de clasificación de documentos, tanto en el ámbito del PLN como del TDA, y se contextualiza la propuesta dentro de la literatura existente. Pondremos el foco en las técnicas y fundamentos del text-mining y así como en algoritmos de Topological Data Analysis (TDA).

- **Capítulo sobre la fuente de datos:** Se describe la estructura del AD-UPM y el procedimiento seguido para extraer los campos relevantes, así como los fragmentos seleccionados para su análisis posterior.
- **Capítulo sobre la construcción de la base de datos:** Se detalla la transformación del conjunto inicial en una base de datos adaptada al análisis, incluyendo las técnicas de preprocesamiento y reducción de dimensionalidad aplicadas.
- **Capítulo de análisis preliminar y clustering:** Se formula una hipótesis de clasificación y se evalúa mediante algoritmos de agrupamiento, con el objetivo de identificar estructuras latentes en los datos.
- **Capítulo de análisis topológico:** Se examinan las características topológicas extraídas de los documentos, valorando su relevancia y utilidad para la tarea de clasificación.
- **Capítulo de desarrollo del clasificador:** Se entrena un modelo de clasificación supervisado utilizando una variable del dataset como etiqueta. Se abordan aspectos como el balanceo de clases y la evaluación del modelo.
- **Capítulo de conclusiones:** Se resumen los resultados obtenidos, se evalúa la validez de la metodología propuesta y si cumple con los objetivos del TFG presentados en este capítulo. También se plantean posibles líneas de trabajo futuro.

Cada capítulo se desarrollará de forma autónoma, con una estructura interna coherente que facilite la comprensión del proceso seguido y de los resultados alcanzados.

Capítulo 2

Trabajos Previos

El *Análisis Topológico de Datos (TDA)* [1] ha evolucionado en los últimos años como un enfoque prometedor para modelar la estructura de datos de alta dimensionalidad en diversos dominios, incluyendo el *procesamiento de lenguaje natural (PLN)* [2] y la *minería de textos* [3]. A diferencia de los métodos tradicionales de clasificación y análisis textual, TDA permite capturar relaciones y características globales en los datos mediante técnicas de homología persistente y análisis de complejos simpliciales.

En este capítulo se presentan los enfoques y tecnologías más relevantes para la clasificación automática de documentos, abarcando tres líneas principales de estudio: los algoritmos de clasificación empleados en Procesamiento de Lenguaje Natural (PLN), las técnicas para la extracción de características textuales y las técnicas de extracción de características topológicas aplicadas al texto mediante Análisis Topológico de Datos (TDA).

A continuación se muestra la estructura más en detalle:

- **Algoritmos de Clasificación en PLN**: Los algoritmos de clasificación en procesamiento de lenguaje natural (PLN) se han desarrollado desde modelos estadísticos clásicos como Naive Bayes o regresión logística, hasta modelos más avanzados basados en aprendizaje profundo, como redes neuronales recurrentes (RNN) [4], convolucionales (CNN) [5] y transformadores [6]. Estos enfoques permiten clasificar documentos, analizar sentimientos y realizar tareas de minería de texto con alta precisión.

- **Extracción de Características en Corpus Textuales**: La representación de texto es una parte clave en el desempeño de modelos de PLN. Las representaciones clásicas incluyen Bag of Words (BoW) [7] y TF-IDF [8], mientras que las modernas se centran en modelos distribucionales como Word2Vec, GloVe y BERT, los cuales capturan relaciones semánticas y contextuales entre palabras, mejorando la capacidad de los clasificadores.

- **Extracción de Características Topológicas (TDA)**: El Análisis Topológico de Datos (TDA) ofrece una nueva perspectiva para entender la estructura de datos textuales. A través de herramientas como la homología persistente y los complejos simpliciales, es posible capturar la forma global de los datos, revelando patrones que los métodos tradicionales pueden pasar por alto. Esta técnica ha

ganado relevancia recientemente por su capacidad para detectar otras representaciones.

2.1. Algoritmos de Clasificación en PLN

2.1.1. Modelos Estadísticos Clásicos

Los métodos de aprendizaje automático clásicos han sido fundamentales en la clasificación de textos durante décadas. En particular, clasificadores lineales como Naive Bayes [9] y Support Vector Machines (SVM) [10], combinados con representaciones simples (e.g. bolsa de palabras con TF-IDF), demostraron un rendimiento competitivo en tareas de categorización de documentos [11, 12]. Por ejemplo, SVM fue ampliamente utilizado en filtrado de spam y categorización temática, destacando por su capacidad para manejar espacios de características de alta dimensión [10].

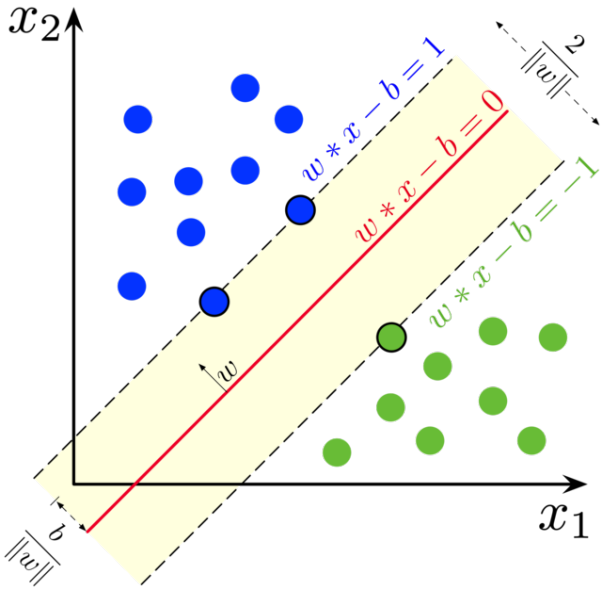


Figura 2.1: Hiperplano de margen máximo y vectores soporte en un clasificador SVM. Fuente: [13].

Aunque los enfoques profundos han ganado protagonismo, estos métodos clásicos siguen siendo relevantes como líneas de base robustas por su rapidez y efectividad con datos limitados [14]. Además, trabajos híbridos han incorporado técnicas modernas a estos esquemas; por ejemplo, la combinación de Word2Vec con modelos como redes neuronales recurrentes (RNN) y Deep Belief Networks (DBN) logró detectar malware en aplicaciones Android con precisiones superiores al 97 % [15, 16].

2.1.2. Modelos basados en Embeddings

La representación distribuida densa de palabras mediante *embeddings* revolucionó el PLN en la última década. Algoritmos como Word2Vec [17], GloVe [18] y FastText [19] aprenden vectores continuos donde la cercanía geométrica refleja similitud semántica. Estas representaciones capturan relaciones léxico-semánticas que las bolsas de

palabras no pueden, mejorando sustancialmente las predicciones en tareas de clasificación de texto.

Word2Vec demostró que palabras en contextos similares ocupan vecindades cercanas en el espacio vectorial, permitiendo a clasificadores distinguir sentidos por proximidad contextual. GloVe integró información global de corpus para producir embeddings robustos a escala de corpus grandes. FastText incorporó subpalabras, capturando estructuras morfológicas útiles para idiomas con mucha flexión. Además de palabras, se introdujeron embeddings de documentos como Doc2Vec [20] que generan vectores para textos completos, preservando la semántica global del documento. Estos embeddings permiten usar modelos lineales o de distancia (p. ej., k-NN) para clasificación con mejor rendimiento que representaciones dispersas tradicionales [21].

En suma, los modelos basados en embeddings proporcionaron un puente entre métodos estadísticos y aprendizaje profundo, al suministrar características continuas de alta calidad aptas para clasificadores variados.

2.1.3. Modelos basados en Deep Learning

Arquitecturas como las Redes de Atención Jerárquica combinaron RNN bidireccionales con mecanismos de atención para clasificar documentos largos destacando frases relevantes [22]. Más recientemente, los Transformers han revolucionado el campo. En particular, BERT [23] y sus variantes (RoBERTa, XLNet, etc.) alcanzan el estado del arte en múltiples tareas de NLP. BERT introdujo un modelo bidireccional preentrenado en grandes corpus, cuyo conocimiento transferido permite clasificar textos con pocos ejemplos específicos, superando por amplio margen a métodos anteriores.

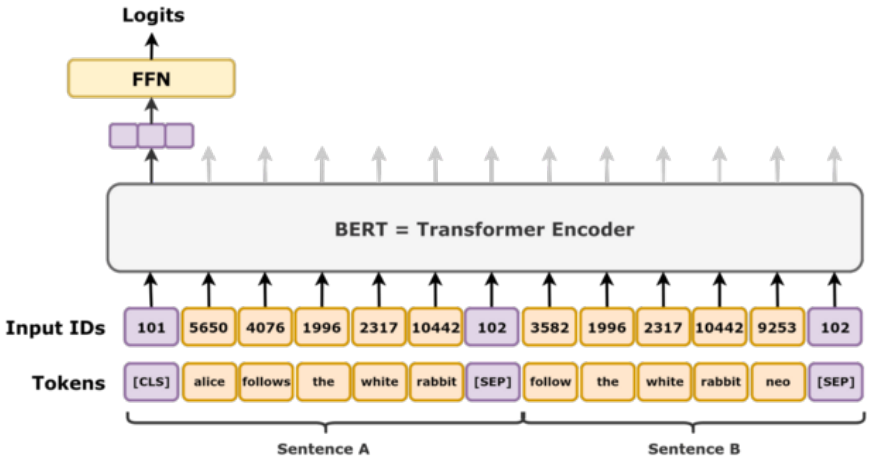


Figura 2.2: Ejemplo del preentrenamiento de BERT para la tarea de predicción de la siguiente oración (Next Sentence Prediction). Fuente: [24].

Durante esta introducción en el estado del arte veremos a esta arquitectura (BERT), aparecer de forma recurrente. Tanto ella como posibles variantes ,p. ej. Topological Bert, resultan muy populares a la hora de trabajar con textos y sus características.

Otros estudios muestran que modelos Transformer superan consistentemente a RNN y CNN en benchmarks de clasificación de textos [25, 26]. En resumen, el panora-

ma actual está dominado por modelos profundos preentrenados, que ofrecen altas precisiones a costa de mayor complejidad computacional, mientras que los enfoques clásicos se mantienen como alternativas interpretables y eficientes en datos escasos.

2.2. Extracción de Características en Corpus Textuales

2.2.1. Representaciones Distribucionales

Una etapa crucial para la clasificación automática es transformar el texto en vectores numéricos. Los enfoques distribucionales clásicos parten del supuesto de que el contexto define el significado (hipótesis distribucional de Harris [7]). Técnicas como Bag of Words (BoW) y TF-IDF representan documentos mediante el conteo de términos, posiblemente ponderados por su frecuencia inversa en el corpus [27]. Estas representaciones, aunque sencillas, han sido ampliamente utilizadas gracias a su efectividad en conjunción con algoritmos lineales. Por ejemplo, en clasificación de noticias y spam, BoW/TF-IDF alimentando a un SVM o regresión logística ha logrado resultados competitivos con un procesamiento mínimo del lenguaje [11].

Su principal limitación es la alta dimensionalidad y la falta de información semántica (ignoran similitudes entre palabras distintas). No obstante, siguen sirviendo como base y a menudo se combinan con métodos más avanzados para captar tanto frecuencia local como significado global.

2.2.2. Word Embeddings

Los embeddings densos de palabras mejoran las representaciones textuales al capturar similitudes semánticas y sintácticas en el espacio vectorial continuo. Modelos como Word2Vec [17], GloVe [18] y FastText [19] aprenden vectores de baja dimensión donde palabras con contextos similares quedan cercanas, resolviendo en parte la escasez de datos al generalizar mejor a vocablos no vistos.

Gracias a estos embeddings, la similaridad semántica puede medirse mediante distancia vectorial, y se ha observado que usar vectores preentrenados como características mejora la precisión de clasificadores supervisados frente a usar solo palabras individuales [28]. Asimismo, es común inicializar redes neuronales con embeddings preentrenados (p. ej. GloVe) para acelerar el aprendizaje y aprovechar conocimiento general del lenguaje [29].

En resumen, los word embeddings han enriquecido la extracción de características textuales al proveer representaciones continuas compactas que capturan las relaciones de significado de forma más efectiva que las representaciones distribucionales tradicionales. Sin embargo, su aplicación en este trabajo es limitada por diversas razones.

La principal limitación es que nosotros buscamos generar embeddings de los documentos, no de sus palabras. Para esto surgen alternativas como Doc2Vec, una extensión del modelo Word2Vec propuesta por Le y Mikolov [20], que permite aprender representaciones vectoriales de longitud fija para secuencias de texto de cualquier tamaño, como frases, párrafos o documentos completos. Esta arquitectura introduce un vector adicional por documento que se entrena conjuntamente con los vectores de

palabras, permitiendo capturar tanto el contenido semántico como el contexto global del texto.

Además, estudios posteriores como el de Lau y Baldwin [30] han demostrado la efectividad de Doc2Vec frente a métodos clásicos, y han propuesto buenas prácticas para su entrenamiento en corpus reales, que tal vez puedan servir como pautas en caso de usar en este trabajo este método. Más recientemente, se han propuesto enfoques basados en aprendizaje profundo y aprendizaje contrastivo para mejorar la calidad de los embeddings documentales, como en Saggau et al. [31] o en modelos como SPECTER [32], que emplean arquitecturas Transformer informadas por citas para representar documentos científicos.

2.2.3. Modelado de Tópicos

Más allá del nivel de palabras, es útil extraer temas latentes que describen el contenido semántico de documentos. La técnica más reconocida es Latent Dirichlet Allocation (LDA) [33], que representa cada documento como una distribución sobre tópicos y cada tópico como una distribución sobre palabras. Al inferir estos tópicos, LDA descubre estructuras temáticas ocultas en grandes colecciones de textos.

Por ejemplo, en un corpus de artículos de noticias, LDA puede identificar tópicos como “política”, “deportes” o “tecnología” a partir de patrones de coocurrencia de palabras, permitiendo luego usar las proporciones de tópicos de cada documento como características para su clasificación. Esto ha demostrado mejorar la agrupación y clasificación al nivel de contenido conceptual, especialmente en escenarios de múltiples temas por documento. En la última década han surgido extensiones que integran redes neuronales al modelado de tópicos (p. ej., Neural Topic Models basados en variational autoencoders) para manejar vocabularios más grandes y captar contextos, pero LDA sigue siendo un referente por su sencillez e interpretabilidad [34].

La combinación de LDA con clasificadores supervisados ha mostrado aumentar la precisión en dominios como clasificación de textos cortos o muy ruidosos, al actuar como una forma de reducción de dimensionalidad informada semánticamente. Se trata de otro método popular y extendido para la extracción de características sobre documentos, aunque como ya veremos más adelante, también pueden ser una opción para reducir la dimensionalidad de sus características, usandolo como filtro.

2.2.4. Reducción de Dimensionalidad

Dado que muchas representaciones textuales producen vectores de dimensionalidad elevada (por ejemplo, TF-IDF con decenas de miles de dimensiones para un gran vocabulario), se emplean técnicas de reducción de dimensionalidad para proyectar los datos a espacios más manejables preservando su estructura relevante. Principal Component Analysis (PCA) ha sido utilizado para compresiones lineales, aunque en PLN suele ser más efectivo aplicar métodos no lineales que conservan relaciones complejas.

Dos técnicas muy difundidas son t-SNE (t-Distributed Stochastic Neighbor Embedding) [35] y UMAP (Uniform Manifold Approximation and Projection) [36]. t-SNE es especialmente útil para visualizar en 2D/3D clusters de documentos o palabras, pues agrupa puntos con vecindades similares enfatizando estructuras locales. UMAP es un método más reciente que preserva mejor las estructuras globales de la variedad

de datos, logrando proyecciones de alta fidelidad incluso con datos densos como embeddings contextuales.

En tareas de clasificación, estas técnicas se emplean a veces para preprocesar características (reduciendo ruido y colinearidad) o para visualización y análisis exploratorio de la distribución de clases en el espacio semántico. Por ejemplo, proyectar embeddings de oraciones con UMAP ha permitido identificar grupos correspondientes a diferentes intenciones en un clasificador de intención en sistemas de diálogo. En concluisón, la reducción de dimensionalidad facilita manejar datos textuales complejos al simplificar su estructura interna, sea para mejorar la eficiencia de los clasificadores o para interpretar mejor los patrones subyacentes.

El **Análisis Semántico Latente** (Latent Semantic Analysis, LSA) es una técnica de reducción de dimensionalidad que permite identificar patrones semánticos en grandes colecciones de texto. Esta técnica se basa en el principio de que las palabras que se usan en contextos similares tienden a tener significados similares. Para capturar estas relaciones, LSA construye una matriz de términos y documentos (por ejemplo, ponderada por TF-IDF) y aplica posteriormente una descomposición en valores singulares (SVD) [37].

A diferencia de métodos más recientes como Word2Vec o BERT, LSA no requiere entrenamiento sobre grandes corpus con modelos complejos, lo que lo convierte en una herramienta eficiente en contextos con recursos computacionales limitados. Sin embargo, su principal limitación es que no capta adecuadamente la polisemia (múltiples significados de una palabra) ni el orden de las palabras, aspectos mejor tratados por modelos basados en redes neuronales [38].

Por contraparte, encontramos el método **Latent Dirichlet Allocation (LDA)**, una técnica de modelado probabilístico que hemos visto que resulta de gran utilidad en el modelado de tópicos. El modelado de tópicos está altamente relacionado con la reducción de dimensionalidad, ya que este es capaz de disminuir las características de cada documento llevándolas a nivel de tópico. Su principal ventaja es que esta reducción se hace en el contexto semántico de las características, no en el estadístico. LDA asume que cada documento está compuesto por una mezcla de temas latentes, y que cada tema es una distribución sobre un conjunto fijo de palabras. Esta formulación permite representar documentos en un espacio temático de baja dimensionalidad, facilitando tareas como la clasificación, la recuperación de información y la exploración semántica de corpus extensos [33].

Desde su introducción, LDA ha sido objeto de múltiples mejoras y extensiones. Por ejemplo, se han propuesto variantes supervisadas, modelos jerárquicos y enfoques dinámicos que permiten capturar la evolución temporal de los temas. Además, estudios como el de Srivastava y Sutton [34] han explorado el uso de autoencoders variacionales para modelar temas de manera más eficiente, integrando el aprendizaje profundo con el enfoque probabilístico clásico de LDA. Estos avances han reforzado el papel de LDA como herramienta fundamental en la extracción de características semánticas en PLN. Un Autoencoder Variacional (VAE) codifica los datos en una distribución gaussiana latente y los reconstruye mediante un decodificador, optimizando la reconstrucción junto con una regularización mediante la divergencia de Kullback-Leibler [39].

2.3. Extracción de Características Topológicas (TDA)

2.3.1. Fundamentos del TDA

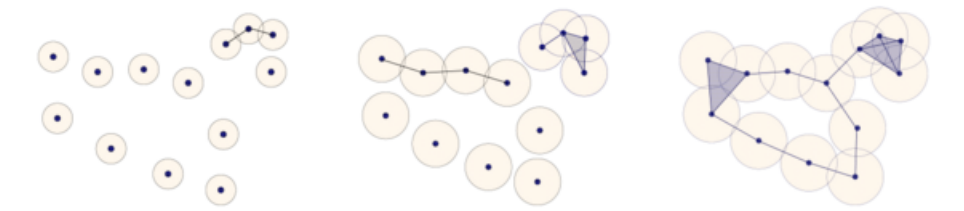
El Análisis Topológico de Datos (TDA) es un conjunto de técnicas que utilizan la topología algebraica para describir la "forma"de los datos [40]. A diferencia de los enfoques tradicionales que dependen fuertemente de la métrica o de un espacio de características fijo, TDA se centra en propiedades invariantes frente a deformaciones continuas, como la conectividad y los agujeros de distintas dimensiones en la nube de puntos que representa los datos [41].

La herramienta central es la *homología persistente*, que computa invariantes topológicos (componentes conexas, túneles, cavidades, etc.) a través de múltiples escalas de similitud [42]. Las estructuras topológicas que persisten en un amplio rango de escalas se asumen como rasgos robustos de la geometría de los datos, tanto frente al muestreo como con respecto a las métricas del espacio topológico ambiente [43].

La salida típica es un *diagrama de persistencia* o un *código de barras*, que resume la aparición y desaparición de características topológicas en función de la escala. Estas representaciones son invariantes a isometrías, robustas al ruido y capturan relaciones globales no evidentes. Además, existen resultados teóricos sobre su estabilidad [44].

2.3.2. Aplicación del TDA a Datos Textuales

Aunque TDA se ha aplicado principalmente en imágenes y datos biomédicos, recientemente ha comenzado a usarse en PLN. Generalmente se transforma el texto a embeddings (por ejemplo, Word2Vec) y se construyen complejos simpliciales sobre estos puntos en $\mathbb{R}^n$ [45]. A través de filtraciones de Vietoris-Rips se obtienen características como bucles y cavidades semánticas persistentes; estos complejos se utilizan por su simplicidad computacional y porque permiten construir aproximaciones topológicas a partir de distancias entre puntos. Otras alternativas incluyen los complejos de Čech, Alpha y Witness, que ofrecen diferentes compromisos entre precisión y eficiencia según la métrica y la distribución de los datos.



(a) Rips complex at some scale (b) ..at a larger scale. (c) ...and at an even larger scale.

Figura 2.3: Ejemplo de una filtración de Vietoris–Rips sobre un conjunto de puntos en el plano euclídeo. La Figura muestra cómo el complejo crece a medida que aumenta el parámetro de escala. Fuente: [46].

Estas características topológicas capturan relaciones semánticas globales que no son fácilmente capturadas por técnicas locales o secuenciales. Por ejemplo, Gholizadeh et al. [47] aplicaron homología persistente sobre representaciones TF-IDF y embeddings,

y encontraron que las descripciones topológicas complementaban a las representaciones tradicionales.

2.3.3. Aplicaciones de los Descriptores Topológicos

El uso de descriptores topológicos en el contexto del procesamiento de lenguaje natural (PLN) ha ganado atención en años recientes por su capacidad de capturar estructuras semánticas globales más allá de las relaciones locales representadas por métodos clásicos. Estos descriptores, derivados de técnicas como la homología persistente, han demostrado utilidad en diversas tareas aplicadas al análisis textual y multimodal.

En el trabajo de Gholizadeh et al. [47], se utilizaron diagramas de persistencia generados a partir de embeddings y representaciones TF-IDF, mostrando que dichos invariantes topológicos contienen información semántica complementaria que mejora el rendimiento de clasificadores convencionales. De forma similar, el modelo Topological BERT propuesto por Perez y Reinauer [48] aplica homología persistente sobre los mapas de atención de BERT, alcanzando resultados competitivos en tareas como análisis de sentimiento y detección de spam.

En el ámbito de procesamiento de señales, Bonafos et al. [49] combinaron descriptores topológicos con coeficientes cepstrales (MFCCs) para mejorar la clasificación de locutores y fonemas, demostrando que TDA también puede extraer patrones relevantes en señales acústicas. Asimismo, en tareas centradas exclusivamente en texto, se ha comprobado la utilidad de los invariantes topológicos para la detección de contradicciones [50] y noticias falsas [51], reforzando la idea de que la forma de los datos contiene información valiosa para problemas complejos de clasificación.

Estas aplicaciones subrayan el potencial del TDA como complemento a los enfoques tradicionales, aportando una nueva dimensión analítica basada en la geometría y topología de las representaciones textuales.

Una línea especialmente afín a la propuesta de este trabajo es la desarrollada por Ferrà et al. [52], donde se introduce un clasificador topológico basado en la comparación del descriptor de persistencia de una clase antes y después de incorporar una nueva muestra. Su hipótesis central ,también adoptada en este proyecto, es que una muestra que perturba poco la estructura topológica del conjunto al que se añade es compatible con él. Aunque utilizan *silhouettes* en lugar de diagramas de persistencia brutos, la lógica es análoga: se interpreta la distancia topológica antes/después como medida de pertenencia. Sus experimentos, realizados sobre datos de señales EEG, demuestran que este enfoque es competitivo frente a métodos clásicos como k-NN, destacando su capacidad para operar con pocas dimensiones (3–5) y su independencia respecto a la varianza explicada. Estas observaciones respaldan la validez del enfoque aquí adoptado, que se apoya directamente en la comparación de diagramas de persistencia mediante distintas métricas para evaluar la compatibilidad de una muestra con una clase dada.

Otro enfoque relevante lo encontramos en el trabajo de Kindelán et al. [53], donde se propone un clasificador basado exclusivamente en herramientas de Análisis Topológico de Datos (TDA), sin necesidad de combinarlo con modelos de aprendizaje automático convencionales. Su propuesta, denominada TDABC, utiliza complejos simpliciales filtrados y operadores como *star* y *link* para propagar etiquetas entre

puntos etiquetados y no etiquetados, ponderando las contribuciones a través de los valores de filtración. Si bien su objetivo principal se orienta hacia la clasificación de conjuntos multiclase desequilibrados, comparten con este trabajo la idea de que las estructuras topológicas ,y su evolución durante la filtración, pueden ser informativas para el problema de clasificación. Sus resultados muestran que el método supera a clasificadores tradicionales como KNN, SVM y Random Forest en escenarios con alta desbalanceo o solapamiento entre clases, lo que refuerza el valor de considerar representaciones topológicas completas para tareas de clasificación.

2.3.4. Herramientas Actuales para Análisis Topológico de Datos

El creciente interés por el Análisis Topológico de Datos (TDA) ha impulsado el desarrollo de diversas herramientas computacionales que permiten su implementación práctica en contextos reales. Estas herramientas proporcionan interfaces para construir complejos simpliciales, calcular homología persistente y representar resultados mediante diagramas de persistencia y códigos de barras. Entre las más utilizadas se encuentran `GUDHI`, `Ripser`, `Dionysus`, `scikit-tda`, y recientemente, `giotto-tda`.

`giotto-tda` [54] es una biblioteca de Python construida sobre una arquitectura modular compatible con `scikit-learn`, lo cual facilita su integración en flujos de trabajo de aprendizaje automático. Ofrece herramientas eficientes para el preprocesamiento, cálculo de homología persistente y vectorización de características topológicas, lo cual permite su uso directo en tareas supervisadas o no supervisadas. Además, incluye soporte para diagramas de persistencia, paisajes de persistencia y otras transformaciones vectoriales.

Una de sus principales ventajas es su accesibilidad para usuarios de PLN, al permitir aplicar TDA sobre espacios de embedding derivados de textos (por ejemplo, Word2Vec, BERT), generando complejos sobre estos datos y extrayendo características topológicas relevantes para clasificación o análisis exploratorio. La documentación de `giotto-tda` incluye ejemplos detallados y casos de uso que van desde el análisis de series temporales hasta PLN, lo que la convierte en una herramienta versátil tanto para la investigación como para aplicaciones industriales [55].

Otras bibliotecas relevantes incluyen `GUDHI` [56], conocida por su flexibilidad y rendimiento al construir filtraciones como Rips, Alpha y Witness complexes, y `Ripser` [57], que destaca por su velocidad en el cálculo de homología persistente de alto orden en grandes conjuntos de puntos. Estas herramientas también se integran con entornos de visualización y exploración de datos como Jupyter y permiten análisis exploratorios potentes en entornos interactivos.

La combinación de estas herramientas con bibliotecas estándar de aprendizaje automático y PLN abre una vía poderosa para la extracción y uso de descriptores topológicos en tareas de clasificación, detección de anomalías y visualización de datos complejos.

2.4. Discusión Crítica

Aunque la clasificación de documentos académicos ha sido ampliamente investigada, la mayoría de los enfoques existentes se centran únicamente en el análisis de

texto convencional utilizando características como palabras clave, frecuencia de términos o embeddings preentrenados. Sin embargo, existe una brecha significativa en la aplicación combinada de análisis textual tradicional con métodos avanzados de análisis topológico, lo que puede ofrecer una visión más profunda sobre la estructura semántica y contextual de los documentos.

Uno de los principales vacíos en la literatura es el uso de apartados específicos del documento, como el título, el resumen y las conclusiones, como fuentes clave para la extracción de características. Tradicionalmente, los enfoques de clasificación tienden a analizar el contenido completo de los textos, pero no se ha explorado suficientemente el potencial de estos apartados específicos, que a menudo contienen la esencia del documento y su contribución principal. Este trabajo aborda este vacío al extraer características de estos apartados utilizando el enfoque TF-IDF, que permite identificar las palabras y frases más representativas de cada sección.

Además, la incorporación de características topológicas mediante el uso de herramientas como la homología persistente y complejos simpliciales es un enfoque innovador que no ha sido ampliamente aplicado en la clasificación de documentos académicos. Este enfoque no solo busca mejorar la clasificación tradicional basada en el análisis textual, sino también capturar relaciones globales y patrones no lineales en los datos, que no siempre son evidentes mediante métodos estadísticos convencionales. Al aplicar TDA a las características extraídas de los apartados clave (título, resumen y conclusiones), buscamos identificar estructuras más complejas en los textos que puedan mejorar la precisión en la clasificación.

Finalmente, este trabajo también apunta a un vacío en la clasificación de documentos en el ámbito académico, específicamente en la clasificación de categorías como institución, departamento o incluso Objetivos de Desarrollo Sostenible (ODS). Mientras que muchos estudios previos se han centrado en clasificaciones más generales, como temas o disciplinas, poco se ha hecho en términos de asignar etiquetas más específicas y detalladas, lo cual podría tener aplicaciones importantes en la organización de bases de datos académicas o el análisis de tendencias en la investigación.

Por lo tanto, este trabajo no solo aborda vacíos metodológicos en la clasificación de documentos académicos, sino que también establece un marco para explorar cómo la combinación de análisis textual y topológico puede mejorar la precisión de la clasificación y proporcionar nuevas perspectivas sobre la organización y el análisis de textos académicos.

Capítulo 3

Fuente de Datos

3.1. Contexto

El archivo digital de la Universidad Politécnica de Madrid (UPM) es un repositorio institucional que alberga una extensa colección de trabajos académicos y científicos producidos en el ámbito de la universidad. Este archivo constituye una herramienta fundamental para la comunidad académica y científica, proporcionando acceso abierto a investigaciones, tesis, proyectos fin de grado, artículos científicos y otros documentos relevantes en diversas disciplinas, especialmente en el ámbito de la ingeniería y las ciencias aplicadas [58].

En el contexto de este Trabajo Fin de Grado (TFG), el archivo digital de la UPM se utiliza como fuente de datos mediante la técnica de *web scraping*. Esta metodología permite extraer automáticamente información estructurada de las entradas del repositorio, incluyendo campos como título, autores, resumen o conclusiones de cada documento. Este enfoque garantiza la recopilación eficiente y sistemática de información relevante para el análisis posterior.

La capacidad de acceder automáticamente a datos de múltiples documentos es fundamental para el presente trabajo, ya que permite construir una base de datos que alimenta los modelos de análisis y clasificación propuestos. Además, el uso del archivo digital de la UPM como fuente de datos garantiza que los contenidos utilizados en el proyecto provienen de investigaciones académicas verificadas, asegurando la calidad y fiabilidad de la información procesada.

De esta manera, el archivo digital de la UPM no solo aporta el contenido fundamental para el desarrollo del TFG, sino que también permite mantener un enfoque metodológico sólido basado en la recolección sistemática de datos relevantes.

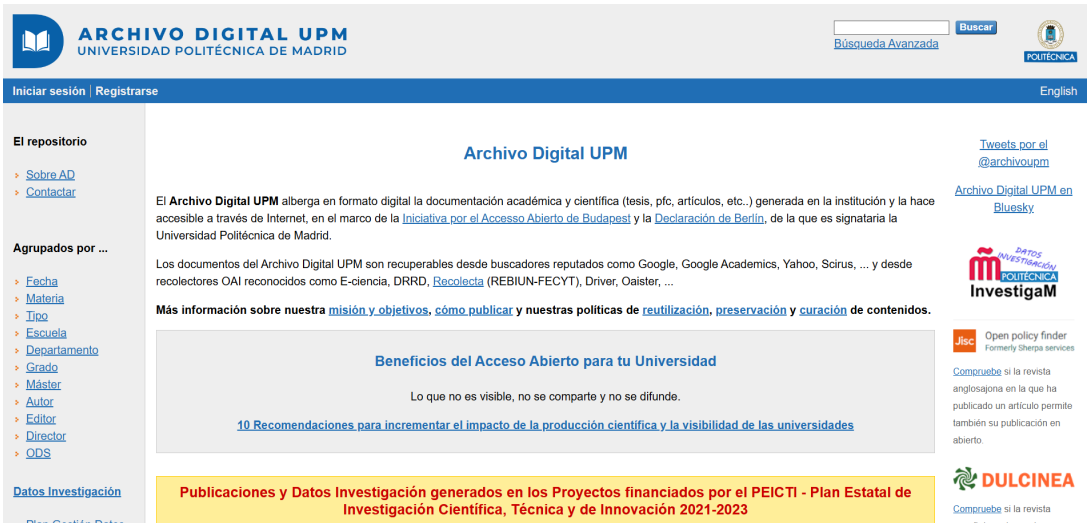


Figura 3.1: Página de Inicio del Archivo Digital

3.2. Estructura del Archivo Digital

El archivo digital de la Universidad Politécnica de Madrid (UPM) está organizado principalmente en torno a áreas temáticas que corresponden a las distintas disciplinas académicas de la universidad. La página principal del archivo muestra un listado de estos temas, permitiendo a los usuarios navegar directamente hacia los documentos relacionados con cada área. Ejemplos de categorías principales incluyen informática, ingeniería industrial, telecomunicaciones, arquitectura, entre otras.

Cada categoría se subdivide en colecciones específicas que agrupan los documentos por tipo o por evento académico. Los tipos de documentos más comunes incluyen tesis doctorales, trabajos fin de grado (TFG), artículos científicos, comunicaciones en congresos y otros materiales académicos [58]. Dentro de cada colección, los documentos están organizados cronológicamente y pueden ser filtrados por título, autor o fecha.

Cada registro individual en el archivo digital presenta una página dedicada con metadatos completos que incluyen el título del documento, los autores, el año de publicación, el resumen, palabras clave y, en muchos casos, el texto completo en formato PDF. Además, se proporciona información sobre el departamento o centro académico responsable del trabajo y el evento académico (si aplica).

Para facilitar el acceso, el archivo digital cuenta con un motor de búsqueda avanzado que permite realizar consultas específicas filtrando por campos como título, autor, palabras clave o tipo de documento. Esta estructura modular y bien organizada garantiza que los usuarios puedan localizar eficientemente los recursos académicos más relevantes para sus proyectos de investigación. Sin embargo, no resulta muy efectiva para recopilar grandes cantidades de datos de forma efectiva, por lo que deberemos recurrir a técnicas como el Web Scrapping para obtener esta información.

3.2.1. Estructura del HTML

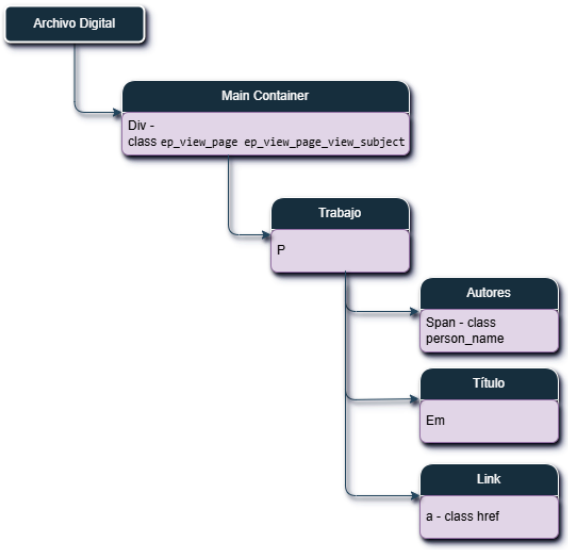
El archivo digital de la Universidad Politécnica de Madrid (UPM) está compuesto por páginas web estructuradas en formato HTML que presentan la información académica de manera organizada y accesible. Cada página dentro del archivo corresponde a una lista de documentos pertenecientes a una categoría específica, como puede ser el área de informática.

Cada documento dentro de una página de resultados está contenido en un bloque HTML que generalmente utiliza etiquetas como `<div>` o `<li>` para encapsular los metadatos principales. Dentro de estos bloques, se encuentran elementos como el título del documento, los autores, el enlace al texto completo y otros datos relevantes. Estos elementos están envueltos en etiquetas como `<a>` para los enlaces, `<span>` para detalles específicos y `<p>` para descripciones breves o resúmenes.

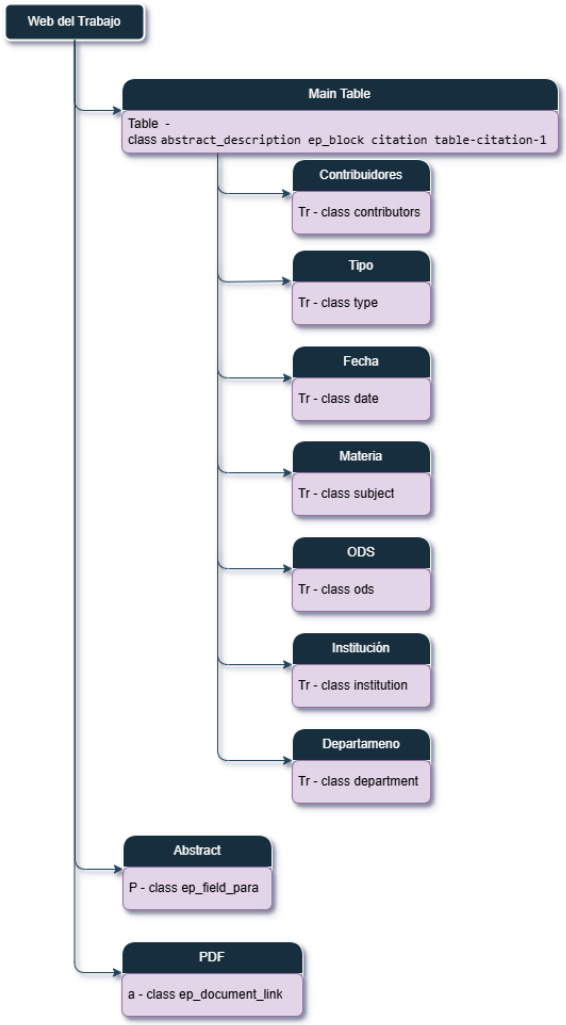
Además, la estructura HTML de cada página de detalle de un documento incluye secciones dedicadas a los metadatos completos, donde se presentan el título, el autor, el año de publicación, el departamento académico, el resumen, las palabras clave y el enlace directo al PDF. Estos datos suelen estar contenidos en etiquetas como `<meta>` con atributos específicos (como `name` o `content`), lo que permite extraerlos de manera eficiente durante el proceso de scraping.

Gracias a esta estructura HTML bien definida, es posible acceder automáticamente a la información relevante de manera sistemática, garantizando así la recopilación de datos de múltiples documentos de forma eficiente.

En la Figura 3.2 se presentan dos diagramas que resumen visualmente la estructura del HTML de las páginas del archivo digital de la UPM. El primero representa cómo se organiza el listado de trabajos dentro de una categoría temática, mientras que el segundo describe el árbol de componentes en la vista individual de un documento. Estos diagramas fueron fundamentales para diseñar el scrapper y comprender las rutas de extracción de la información.



Estructura del listado de trabajos en una categoría



Estructura de la página de detalle del documento

Figura 3.2: Diagramas de estructura HTML del archivo digital

Estas son algunas de las etiquetas más relevantes que podemos encontrar en la estructura HTML del repositorio. Aunque existen muchas más presentes en el código fuente, aquí se recogen únicamente aquellas que han sido utilizadas específicamente para el desarrollo del scrapper implementado en este trabajo.

3.2.2. Estructura del PDF

Los documentos descargados desde el archivo digital de la Universidad Politécnica de Madrid (UPM) suelen estar en formato PDF y corresponden a trabajos académicos como Trabajos Fin de Grado (TFG), tesis doctorales, ponencias, comunicaciones en congresos y otros tipos de publicaciones científicas. Estos documentos pueden estar redactados tanto en español como en inglés, dependiendo del contexto académico y del área de estudio.

La estructura interna de los PDFs no sigue un estándar común, ya que la heteroge-

neidad en cuanto al ámbito de estudio, estilo de redacción, contenido y formato es considerable. Aunque muchos documentos mantienen un esquema académico convencional, existen variaciones significativas según el departamento, el tipo de trabajo y el año de publicación.

Por lo general, estos documentos contienen una portada que incluye el título del trabajo, los autores, el departamento o escuela de la UPM a la que pertenece el proyecto y el año de publicación. A continuación, suele aparecer un índice o tabla de contenidos que facilita la navegación por el documento. Sin embargo, algunos trabajos pueden omitir estos elementos o estructurarlos de manera distinta, dependiendo de la normativa del centro o de los requisitos específicos del proyecto.

En términos de formato, los documentos están estructurados en páginas de texto continuo, a veces intercaladas con figuras, tablas y ecuaciones. El texto generalmente está organizado en columnas simples o dobles, dependiendo del formato de publicación. Además, los encabezados y pies de página pueden contener información adicional, como el nombre del proyecto, el autor o el departamento académico.

En los trabajos en español, el contenido típico incluye secciones como Resumen, Introducción, Metodología, Resultados, Discusión y Conclusiones. Sin embargo, algunas ponencias y comunicaciones pueden optar por formatos más breves y esquemáticos, como resúmenes extendidos o actas de conferencia. Además, muchos trabajos incluyen una sección de agradecimientos y referencias bibliográficas al final del documento.

Dado que en este trabajo nos centraremos exclusivamente en los documentos redactados en español, se ha decidido filtrar previamente los archivos recopilados para evitar análisis en documentos en inglés. Esto permite centrar el proceso de scraping y el posterior análisis en aquellos textos que comparten un mismo idioma, garantizando así la coherencia lingüística en el procesamiento de la información.

3.2.3. Objetivos del Scrapper

Una vez descrita la estructura del Archivo Digital de la UPM y justificado su uso como fuente de datos, resulta necesario definir los objetivos específicos del scrapper desarrollado para este proyecto. En concreto, debemos establecer qué elementos del repositorio son de interés y qué criterios se han seguido para seleccionar los documentos a extraer.

El objetivo principal del scrapper es recopilar información textual relevante únicamente de trabajos redactados en castellano. Esta decisión responde a la necesidad de mantener la coherencia lingüística en el tratamiento de los datos, dado que el repositorio contiene documentos tanto en español como en inglés. Incorporar ambos idiomas aumentaría significativamente la complejidad del preprocesamiento textual, así como del diseño del clasificador y del análisis posterior. Por ello, se opta por filtrar previamente los documentos y centrarse exclusivamente en aquellos escritos en lengua española.

De cada uno de los trabajos seleccionados se extraen los siguientes elementos clave:

- **Título del trabajo**: proporciona una descripción breve y directa del contenido del documento.

- **Resumen (Abstract)**: resume los objetivos, metodología y conclusiones principales del trabajo.
- **Conclusiones**: sección que permite identificar las aportaciones finales y reflexiones del autor.

Mientras que el título y el resumen suelen estar disponibles directamente en los metadatos del repositorio, la sección de conclusiones debe extraerse de forma específica a partir del contenido interno del archivo PDF. Esto requiere una búsqueda más detallada dentro del documento, lo cual añade una capa adicional de complejidad al scraping.

Adicionalmente, se recopilan metadatos contextuales del documento, como la fecha de publicación, la institución o departamento al que está asociado, y las etiquetas temáticas como los Objetivos de Desarrollo Sostenible (ODS). Estos atributos permiten enriquecer el dataset, facilitando tareas de segmentación, agrupación o análisis comparativo entre grupos de trabajos con características comunes. En esta línea, estudios recientes como *MotifClass* demuestran que el uso de metadatos de orden superior mejora significativamente el rendimiento de los clasificadores textuales en entornos de baja supervisión, lo que refuerza el valor de su incorporación en este tipo de análisis [59].

3.3. Desarrollo del Scrapper

Para la recolección automatizada de los datos del Archivo Digital de la UPM se ha desarrollado un scrapper en Python que combina técnicas de análisis de HTML y procesamiento de documentos PDF. Las dos librerías principales utilizadas son la librería `BeautifulSoup` para el análisis de HTML [60] y `PyMuPDF` para la extracción del contenido de los PDF [61].

El proceso completo de scraping se ha estructurado en cuatro fases diferenciadas:

- **Obtención del listado de trabajos en la página principal:** Se accede al índice del área temática (en este caso, informática) del archivo digital. A través del análisis del código HTML se localizan los elementos correspondientes a cada entrada académica publicada.
- **Iteración sobre cada entrada del índice:** Para cada elemento del listado se extraen metadatos básicos como el título, los autores y, fundamentalmente, el enlace a la página específica del documento dentro del repositorio.
- **Acceso a la página individual de cada documento:** En esta fase se recolectan metadatos más detallados como los colaboradores, el tipo de documento, la fecha de publicación, el departamento, la institución, los ODS (cuando están disponibles), entre otros campos relevantes presentes en el HTML. También extraemos el abstract de esa página, evitando consultar el propio PDF.
- **Descarga temporal del archivo PDF y extracción de conclusiones:** Finalmente, se descarga de forma temporal el documento completo en formato PDF, con el objetivo de extraer la sección de "Conclusiones". Para ello, se analiza el contenido textual del archivo a partir de la página 15, asumiendo que las conclusiones suelen ubicarse hacia el final de los trabajos académicos, y se emplea una expresión regular que detecta dicha sección bajo el patrón `r'(?i)\bconclusiones\b'`,

el cual ignora mayúsculas y minúsculas y localiza coincidencias exactas con la palabra clave.

Una vez identificada la posición inicial de las conclusiones, se determina su final buscando el comienzo de una nueva sección habitual como "Bibliografía", "Referencias" o "Anexos". Este método se basa en una estructura académica comúnmente seguida en trabajos de grado y posgrado, lo que permite una extracción fiable sin necesidad de una segmentación más compleja del documento.

Además, esta fase cumple una doble función: en los casos en los que no se encuentra coincidencia con la expresión regular, es probable que el documento esté redactado en inglés, por lo que esta ausencia puede utilizarse como un filtro indirecto para descartar trabajos no redactados en español. En el caso de que no se encuentre el apartado de conclusiones, simplemente añadimos un NaN a ese apartado, para más adelante eliminar estos documentos.

Toda la información recopilada durante el proceso de scraping se encapsula en instancias de una clase denominada `Documento`, diseñada para almacenar de forma estructurada todos los campos de interés. Esta clase incluye un método `to_dict()` que permite transformar los datos recogidos en un diccionario de Python, facilitando su posterior conversión a un `DataFrame` de `pandas` y su análisis posterior.

Este diseño modular y orientado a objetos permite mantener el código organizado, facilitar la depuración de errores y habilitar futuras extensiones, como el uso de otros campos adicionales o la adaptación a nuevas secciones del repositorio.

3.4. Dataset Final

Tras completar el proceso de scraping, se ha generado un conjunto de datos en formato `CSV`, donde cada fila representa un trabajo académico individual del Archivo Digital de la UPM. La información extraída se ha organizado en torno a las siguientes variables:

- **Título**: Título oficial del trabajo.
- **Autores**: Nombre de los autores principales que figuran en el repositorio.
- **Link**: Enlace directo a la página del documento en el archivo digital.
- **Contributors**: Otros colaboradores o coautores asociados al trabajo.
- **Tipo**: Clasificación del documento (TFG, tesis, comunicación, etc.).
- **Fecha**: Fecha de publicación o presentación.
- **Subject**: Materia o categoría temática del trabajo.
- **ODS**: Objetivos de Desarrollo Sostenible vinculados al contenido del documento.
- **Institución**: Centro, escuela o facultad a la que pertenece el trabajo.
- **Departamento**: Departamento académico responsable del documento.
- **Abstract**: Resumen textual, extraído del campo de metadatos del repositorio.
- **Conclusiones**: Sección de conclusiones localizada dentro del PDF del documento.

Estas variables permiten representar tanto contenido textual como metadatos estructurales y contextuales, lo que abre la puerta a distintos tipos de análisis.

Cabe señalar que, aunque el conjunto de datos se presenta en forma de tabla, no puede considerarse completamente estructurado. Esto se debe a que varios de sus campos contienen texto libre y de longitud variable, como el título, el resumen o las conclusiones. Estos elementos, al no estar normalizados ni convertidos aún en características numéricas, requieren de un tratamiento adicional para poder ser utilizados en modelos analíticos o de clasificación.

Por este motivo, en el siguiente capítulo se llevará a cabo un proceso de preprocesamiento textual que permitirá transformar dicho contenido en un formato estructurado. Este paso es esencial para extraer características significativas del texto y comenzar a trabajar con representaciones formales que combinen tanto el contenido semántico como los metadatos asociados a cada documento.

Capítulo 4

Extracción de Datos

4.1. Introducción

En el capítulo anterior se llevó a cabo el proceso de recopilación de información proveniente del Archivo Digital de la Universidad Politécnica de Madrid (UPM), centrado específicamente en trabajos del ámbito de la informática como Trabajos Fin de Grado (TFG), tesis y otras contribuciones académicas. El resultado de este scraping fue un conjunto de datos compuesto por un total de 11 056 entradas, cada una representando un documento académico.

Este dataset incluye tanto contenido textual, como el título, el resumen (abstract) y las conclusiones, como metadatos relevantes, tales como los autores, los Objetivos de Desarrollo Sostenible (ODS) vinculados, el departamento académico y la institución correspondiente. Si bien esta recopilación constituye una base amplia y rica en información, muchos de estos campos contienen texto libre y no estructurado, lo que hace necesario un procesamiento adicional para su posterior análisis.

Además, en este capítulo se aplicarán los primeros criterios de filtrado que permitirán depurar el conjunto inicial y centrarse exclusivamente en aquellos documentos que cumplen con los requisitos definidos para este estudio, como el idioma o la presencia de secciones clave. Este proceso supondrá una reducción significativa del número de documentos, sentando así las bases para el preprocesamiento y extracción de características textuales en los capítulos siguientes.

4.2. Preprocesamiento del corpus

4.2.1. Filtrado inicial

En primera instancia tras cargar nuestros datos, realizamos una limpieza sobre las conclusiones. Para los documentos en inglés, o aquellos donde no hemos podido localizar las conclusiones, hemos insertado el valor NaN en este apartado. Por tanto, comenzaremos eliminando esos NaN.

Debernos darnos cuenta que muchos de los documentos donde no hemos podido localizar conclusiones son documentos en inglés, ya que el nombre de la sección es distinto. Es por esto que de esta forma también eliminamos aquellos trabajos en otro

lenguaje (aunque debemos darnos cuenta que no eliminamos las palabras en inglés de documentos en español).

En el resto de columnas no encontramos valores NaN, por lo que seguiremos con nuestro pipeline de preprocesado.

4.2.2. División de columnas categóricas

Las columnas categóricas son las posibles labels que tienen nuestros documentos. Me gustaría destacar 3 categorías en concreto, que son las mas potenciales para ser analizadas: Institución, Departamento y ODS. Estas categorías son relevantes porque nos permiten clasificar los documentos según su origen institucional, el departamento académico al que pertenecen y los Objetivos de Desarrollo Sostenible (ODS) que abordan.

El análisis posterior se realizará en parte sobre estas tres posibles etiquetas. Sin embargo, en numerosas ocasiones nos centraremos únicamente en las Instituciones y ODS, ya que considero que son las etiquetas más relevantes e interesantes para estudiar.

Para estas categorías realizaremos un paso a un tipado correcto, ya que al cargarlas desde un dataframe venían como formato object. En el caso de Institución y Departamento pasaran a ser formato string. Los ODS serán guardados por ahora como una lista en el Dataframe, aunque más adelante serán tratados con una estrategia de *MultiLabelEncoder*.

En la Figura 4.1 se puede observar la distribución de las apariciones de cada Institución en el dataset. Como se puede ver, la mayoría de los documentos pertenecen a la E.T.S. de Ingenieros Informáticos (UPM), seguida por la E.T.S.I. de Sistemas Informáticos (UPM) y la E.T.S.I. y Sistemas de Telecomunicación (UPM). Existe un gran abismo entre la clase más numerosa y las siguientes, algo que nos dará problemas más adelante en el análisis. El gráfico de la derecha representa la agregación de todas las intituciones consideradas como *Antigüa Denominación*. Estas han sido agrupadas en su Institución actual correspondiente. De la misma forma, solo mostramos las 6 clases más numerosas. Más adelante, en el capítulo 7, profundizaremos más en este aspecto.

En la tabla 4.1 se muestra más en detalle las Instuticiones que aparecen en nuestro dataset, y su frecuencia de aparición.

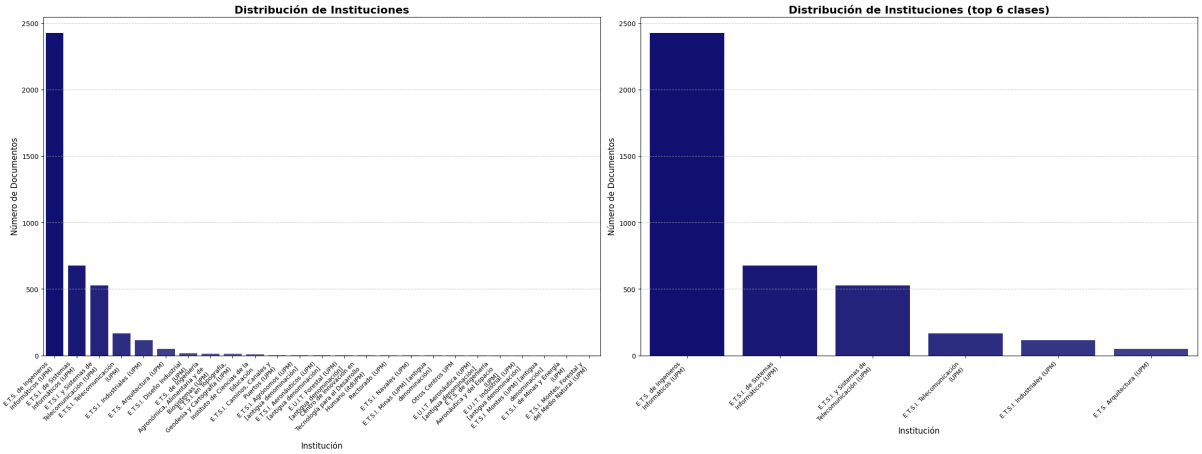


Figura 4.1: Distribución de las apariciones de cada Institución tras la agregación en Others

Institución	**Cantidad**
E.T.S. de Ingenieros Informáticos (UPM)	2090
E.T.S.I. de Sistemas Informáticos (UPM)	666
E.T.S.I. y Sistemas de Telecomunicación (UPM)	528
Facultad de Informática (UPM) [antigua denominación]	335
E.T.S.I. Industriales (UPM)	117
E.U.I.T. Telecomunicación (UPM) [antigua denominación]	92
E.T.S.I. Telecomunicación (UPM)	76
E.T.S. Arquitectura (UPM)	48
E.T.S.I. Diseño Industrial (UPM)	18
E.T.S. de Ingeniería Agronómica, Alimentaria y de Biosistemas (UPM)	15
E.T.S.I. en Topografía, Geodesia y Cartografía (UPM)	14
E.U. de Informática (UPM) [antigua denominación]	12
Instituto de Ciencias de la Educación	8
E.T.S.I. Caminos, Canales y Puertos (UPM)	4
E.T.S.I. Agrónomos (UPM) [antigua denominación]	3
Centro de Innovación en Tecnología para el Desarrollo Humano (itdUPM)	3
E.U.I.T. Forestal (UPM) [antigua denominación]	3
E.T.S.I. Aeronáuticos (UPM) [antigua denominación]	3
E.T.S.I. Minas (UPM) [antigua denominación]	2
Rectorado (UPM)	2
E.T.S.I. Navales (UPM)	2
E.U.I.T. Industrial (UPM) [antigua denominación]	1
E.T.S.I. Montes (UPM) [antigua denominación]	1
E.T.S. de Ingeniería Aeronáutica y del Espacio (UPM)	1
E.U.I.T. Aeronáutica (UPM) [antigua denominación]	1
E.T.S.I. de Minas y Energía (UPM)	1
Otros Centros UPM	1
E.T.S.I. Montes, Forestal y del Medio Natural (UPM)	1

Cuadro 4.1: Distribución de Instituciones

4.2.3. División de las columnas de texto

Para poder transformar nuestras secciones de los documentos en datos estructurados debemos comenzar elaborando el propio corpus. Para hacerlo, la estrategia seguida ha sido la siguiente.

En primer lugar, hemos concatenado los tres elementos directamente relacionados con el documento: el Título, el Abstract y las Conclusiones.

Tras eso hemos obtenido una cadena muy larga de texto. Esta ha sido procesada por una función encargada de limpiar el texto. Más concretamente seguía estos pasos:

- Eliminar saltos de línea y espacios duplicados para homogeneizar las entradas del corpus.
- Eliminar caracteres no imprimibles o raros, como ñ's, tildes o ç. De esta forma obtenemos todo el texto más similar y legible.
- Tokenizar el texto. Separamos el texto por los tokens que correspondan (mayoritariamente palabras)
- Eliminar las *stopwords* en castellano e inglés y dejar el texto en minúscula. Es importante eliminar las palabras de ambos idiomas, ya que es probable que aún en textos en castellano haya algunas partes de ciertos párrafos, como citas o títulos, que esten en otro idioma.
- Eliminar tokens no alfabéticos, como números. De este tipo de tokens no podremos sacar ningún valor.

Para todo este trabajo me he apoyado en la librería *NLTK* de Python [62], que me ha permitido trabajar con el corpus de forma muy cómoda e intuitiva, realizando tareas como la tokenización o la eliminación de stopwords.

Tras este proceso, hemos transformado nuestros campos desestructurados en una lista de palabras.

4.3. Extracción de Características del Corpus

El siguiente paso es extraer información del corpus que acabamos de obtener. Para ello existen una gran cantidad de técnicas, desde métodos más clásicos hasta modelos generadores de embeddings. Tras la investigación realizada, se decidió que la mejor estrategia sería una herramienta simple pero robusta y popular, la matriz TF-IDF.

4.3.1. TF-IDF

TF-IDF (Term Frequency - Inverse Document Frequency) es una técnica ampliamente utilizada en el procesamiento del lenguaje natural para evaluar la importancia de una palabra dentro de un documento, en relación con un corpus de documentos. Se basa en dos componentes principales: la frecuencia de término (*TF*) y la frecuencia inversa de documento (*IDF*). La frecuencia de término mide cuántas veces aparece una palabra en un documento, mientras que la frecuencia inversa de documento penaliza las palabras que son comunes en muchos documentos, disminuyendo así su peso en la representación final.

Formalmente, para un término t en un documento d perteneciente a un conjunto de documentos D, el valor TF-IDF se calcula como:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D)$$

donde

- $\text{TF}(t, d)$ es la frecuencia del término t en el documento d
- $\text{IDF}(t, D) = \log\left(\frac{N}{1+|\{d \in D : t \in d\}|}\right)$, siendo N el número total de documentos y el denominador el número de documentos que contienen el término t

Esta técnica permite representar los documentos en forma de vectores ponderados, mejorando significativamente el rendimiento en tareas como clasificación de textos, recuperación de información y clustering.

El resultado de aplicar TF-IDF a un corpus es una matriz numérica donde las filas representan documentos y las columnas representan términos del vocabulario. Dado que en la mayoría de los textos solo aparece una fracción pequeña del vocabulario total, esta matriz contiene principalmente ceros. Por ello, se representa como una *sparse matrix* o matriz dispersa, lo cual permite un uso eficiente de memoria y recursos computacionales al almacenar únicamente los valores distintos de cero y sus posiciones.

Para este desarrollo me apoyé en la implementación de esta técnica existente en la librería de python *scikit-learn*.

Como ya se ha comentado, el método de TF-IDF genera las conocidas como Sparse Matrix (Matrices Dispersas). Estas no son más que matrices con valores 0 en gran parte de sus celdas. Más en concreto, el valor de densidad de la matriz es de 0.0045, lo que significa que tan solo el 0.45 % de los elementos de la misma son distintos de 0.Esto implica que la matriz está relativamente vacía para su gran tamaño, y debemos de conseguir reducirlo quedándonos únicamente con las características más relevantes. Es por ello que a continuación comentaremos cuales son estos filtros.

4.3.2. Bag of Words (BoW)

El modelo *Bag of Words* (Bolsa de Palabras) es una representación textual ampliamente utilizada en procesamiento de lenguaje natural. Su principio fundamental es transformar un conjunto de documentos en vectores numéricos, basándose únicamente en la ocurrencia de palabras, sin tener en cuenta la gramática ni el orden de las mismas. En este enfoque, se construye un vocabulario a partir de todas las palabras únicas presentes en el corpus y, posteriormente, cada documento se representa como un vector cuya dimensión corresponde al tamaño del vocabulario. Cada componente del vector indica la frecuencia con la que aparece una palabra específica en el documento.

Una de las principales ventajas del modelo BoW es su simplicidad y facilidad de implementación. Sin embargo, esta técnica genera vectores de alta dimensionalidad, generalmente muy dispersos (sparse), y no captura relaciones semánticas ni contextuales entre palabras. A pesar de estas limitaciones, BoW ha demostrado ser eficaz como base para tareas de clasificación, análisis de sentimientos y recuperación de información, especialmente cuando se combina con técnicas de reducción de dimensionalidad o modelos más complejos como TF-IDF.

En nuestro caso, esto último es justo lo que vamos a hacer. Vamos a combinar TF-IDF con el output de BoW, para poder aplicar métodos de reducción de dimensionalidad.

4.4. Reducción de Dimensionalidad

Tras la aplicación de la técnica de TF-IDF a nuestro dataset hemos obtenido una matrix de tamaño 4048 x 153078. Lógicamente el hecho de tener tantas "features"hace inviable realizar cualquier tipo análisis sobre nuestros datos, y mucho menos desarrollar un clasificador basado en la topología de los mismos.

Es por ello que el proceso de reducción de dimensionalidad de nuestros datos es tan relevante. Debemos ser capaces de eliminar dimensiones sin desprendernos de características representativas de los documentos. Para ello podemos aplicar distintos tipos de filtro, como los que veremos a continuación.

4.4.1. Filtros Lineales sobre la Frecuencia

Dado que los propios valores de la matriz de TF-IDF se pueden interpretar como el peso de una palabra en el documento, la primera idea que se ocurre es filtrar en esa misma estructura. La forma más directa de hacer esto es a través de un umbral. Si la suma de las relevancias de esa palabra en todos los documentos no lo supera, esta palabra se elimina.

Debemos darnos cuenta que el hecho de que una palabra sume menos que cierta cantidad está directamente relacionados con la importancia de esa palabra en el corpus. Es posible que la relevancia de esa palabra sea mucha en 1 o 2 documentos, pero dado que estamos extrayendo características globales del corpus, lo que realmente buscamos son palabras que sean relevantes en un conjunto de documentos, ya que no nos podemos permitir aumentar la dimensión con palabras muy específicas.

Tras un conjunto de pruebas, decidí fijar el valor del umbral en 0.5, ya que me reducía el conjunto de palabras en mi vocabulario de 153000 a 16039 palabras, lo que supone cerca de un 90 % menos de dimensiones. Con esto también conseguía aumentar la densidad de mi matriz de datos, pasando de un 0.45 % de celdas distintas de 0 a un 3.68 %. Este valor seguía significando que casi toda la matriz seguía estando vacía, pero eso me asegura que las palabras que quedan no reparten su relevancia de forma homogénea entre todos los documentos, sino que pertenecen a un grupo de documentos solo. Nos asegura en cierta medida que sigue habiendo discriminancia entre documentos.

4.4.2. Entropía de Documentos y Palabras

La **entropía** es un concepto fundamental en teoría de la información, introducido por Claude Shannon, [63] que mide la cantidad de incertidumbre o imprevisibilidad asociada a una distribución de probabilidad. En el contexto del procesamiento de lenguaje natural, se puede aplicar para analizar la distribución de los pesos TF-IDF dentro de un documento o entre documentos. La entropía proporciona una forma de cuantificar cuán concentrada o dispersa está la información contenida en los textos.

En este trabajo se ha calculado la entropía de dos formas: por documento (filas de la matriz TF-IDF) y por palabra (columnas de la matriz). Para cada documento, se toma

el vector de pesos TF-IDF normalizado como una distribución de probabilidad sobre las palabras presentes en ese documento. La entropía se calcula usando la fórmula de Shannon:

$$H(p) = -\sum_i p_i \log p_i$$

donde p_i representa la probabilidad asociada a cada palabra del documento, en este caso estimada como el peso TF-IDF de esa palabra dividido por la suma total de pesos del documento. Una entropía alta indica que los pesos están repartidos entre muchas palabras, mientras que una entropía baja implica que pocas palabras concentran la mayor parte de la información del documento.

Por otro lado, al calcular la entropía por columna, se está midiendo la dispersión de una palabra concreta a lo largo de todos los documentos. En este caso, cada columna se normaliza como una distribución de probabilidad sobre los documentos, y se evalúa si una palabra se distribuye uniformemente o está muy concentrada en pocos documentos. Una entropía alta por palabra indica un término común en muchos documentos (menos discriminativo), mientras que una entropía baja sugiere una palabra más específica, útil para diferenciar documentos. Esta técnica ya ha sido utilizada en otros trabajos como en el de Jerry Watkins [64], donde se porponía un selector de características basado en esta misma entropía.

Estas métricas permiten obtener una visión más profunda de la estructura informativa del corpus: la entropía por documento puede ayudar a detectar textos redundantes o genéricos, mientras que la entropía por palabra permite identificar términos más relevantes para tareas de clasificación o filtrado de características.

Una vez introducido este nuevo concepto, podemos usar este valor de entropía para estudiar cómomo se distribuye la información en nuestra matriz.

En la Tabla 4.2, se resumen los valores obtenidos para ambos enfoques:

Métrica	**Valor**	**Valor Máximo Posible**
Entropía media por documento	0.1646	≈ 14
Entropía media por palabra	3.6229	≈ 12

Cuadro 4.2: Valores medios de entropía sobre la matriz TF-IDF filtrada

La entropía media por documento, con un valor de 0.1646, indica que la distribución de los pesos TF-IDF dentro de cada documento está altamente concentrada. Es decir, cada documento utiliza de forma significativa solo un pequeño subconjunto de palabras del vocabulario total, lo cual es coherente con la naturaleza dispersa del lenguaje: la mayoría de los textos hacen uso de un número limitado de términos relevantes.

En contraste, la entropía media por palabra tiene un valor más elevado, 3.6229, lo que refleja que muchas palabras están distribuidas a lo largo de múltiples documentos. Sin embargo este aún está muy lejos de su posible máximo. Esto sugiere que el corpus incluye términos frecuentes o de uso general, los cuales aparecen en diversos contextos. Sin embargo, este valor también puede indicar que ciertas palabras aportan poca capacidad discriminativa si aparecen de forma homogénea en todo el conjunto.

Estos resultados pueden explicarse en gran parte por la baja densidad de la matriz TF-IDF, que presenta un valor de 0.03. Esta elevada dispersión implica que, en promedio, cada documento está representado por muy pocos términos relevantes, lo que naturalmente reduce la entropía media por documento. A su vez, muchas de las palabras que aparecen lo hacen repartidas entre múltiples documentos, dando lugar a una mayor entropía por palabra. Por tanto, la densidad refuerza cuantitativamente la interpretación de la distribución desigual de los pesos en las representaciones TF-IDF a nivel tanto de documentos como de términos.

4.4.3. Información Mutua

La **información mutua** (*mutual information*) es una medida de dependencia estadística entre dos variables aleatorias. Desde el punto de vista de la teoría de la información, cuantifica cuánta información del valor de una variable se puede obtener observando el valor de otra. A diferencia de otras medidas como la correlación lineal, la información mutua captura tanto relaciones lineales como no lineales, lo que la convierte en una herramienta muy útil para problemas de selección de características o análisis de relevancia en modelos de aprendizaje automático.

Formalmente, para dos variables discretas X e Y, su información mutua se define como:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right)$$

donde $p(x,y)$ es la probabilidad conjunta de X e Y, y $p(x)$, $p(y)$ son sus probabilidades marginales. Si X e Y son completamente independientes, entonces $I(X;Y) = 0$. Por el contrario, valores altos indican una fuerte dependencia entre ambas variables.

La idea es calcular la información mutua de nuestro vocabulario con respecto a las variables Institución o Tipo para detectar palabras poco discriminativas y así eliminarlas.

A continuación mostramos un gráfico en la Figura 4.2 que representa las 50 palabras más discriminantes para las categorías de Institución y Departamento, calculando la información mutua entre cada palabra y cada una de estas variables. Para ello, se ha utilizado la función *mutual_info_classification* de la librería *sklearn*, que permite calcular la información mutua entre las palabras y las categorías institucionales y departamentales.

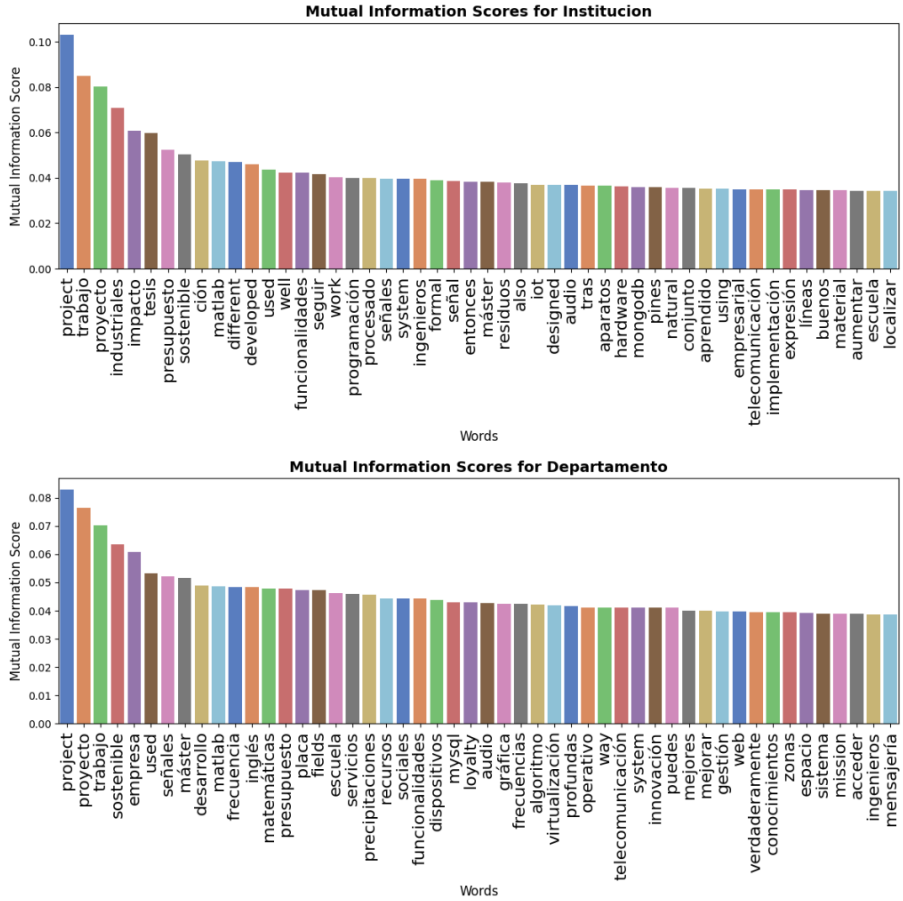


Figura 4.2: Top 50 palabras con mayor información Mutua

En el gráfico superior, correspondiente a la variable *Institución*, destacan términos como *project*, *trabajo*, *proyecto*, *industriales* y *empresa*. Estos resultados sugieren que ciertos términos técnicos o contextuales están asociados con instituciones específicas, probablemente reflejando su enfoque académico o profesional predominante. Por ejemplo, la palabra *industriales* puede tener mayor frecuencia en documentos provenientes de la Escuela de Ingenieros Industriales, mientras que *project* o *developed* podrían estar más presentes en escuelas con un enfoque tecnológico o internacional.

En el gráfico inferior, correspondiente a la variable *Departamento*, se observa una distribución distinta. Palabras como *project*, *trabajo*, *sostenible*, *empresa*, *señales* o *desarrollo* presentan los valores más altos de información mutua, indicando que son especialmente informativas para discriminar entre diferentes departamentos. También se observa la aparición de términos técnicos más específicos como *frecuencia*, *audio*, *sql*, *virtualización* o *operativas*, lo cual sugiere una mayor especialización temática a nivel departamental.

Un aspecto destacable de los resultados es la propia distribución de los valores de información mutua. En ambos gráficos se observa que el valor máximo no supera los 0.15, y que los valores descienden de forma relativamente homogénea y continua. Esta característica indica que, si bien existen palabras que son algo más informativas que otras, no hay ningún término que por sí solo sea extremadamente determinante para predecir la categoría de institución o departamento. La baja magnitud de los valores

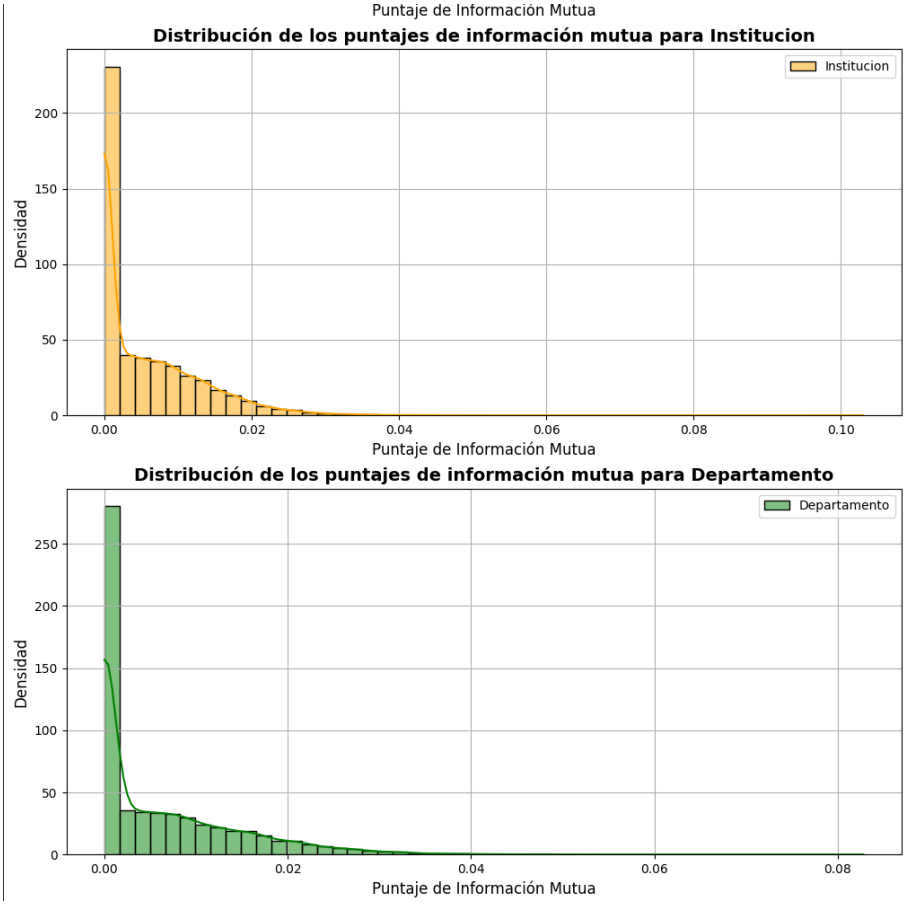


Figura 4.3: Distribución de los valores de información mutua

de información mutua es esperable en tareas de clasificación multiclase con muchas categorías y vocabulario amplio, ya que las palabras tienden a estar distribuidas entre múltiples clases con diferentes niveles de ambigüedad o solapamiento semántico. Además, la caída suave sugiere una distribución más bien uniforme de la relevancia léxica, sin presencia de términos claramente dominantes.

También podemos estudiar como se distribuye la información mutua en nuestro vocabulario a través de un histograma de la Figura 4.3.

La forma tan sesgada de la distribución de los puntajes de información mutua, concentrada cerca de cero, refuerza la necesidad de realizar una selección de características antes de aplicar técnicas de reducción de dimensionalidad. La mayoría de los términos no aportan información relevante para discriminar entre las categorías institucionales y departamentales, lo que podría introducir ruido en representaciones posteriores sin contribuir significativamente a la estructura semántica del espacio.

Con el objetivo de preservar únicamente las palabras más informativas, se procederá a eliminar aquellas cuyo valor medio de información mutua (promediando entre las variables *Institución* y *Departamento*) sea inferior a un umbral de 0.001. Esta decisión se sostiene sobre la distribución observada. Al reducir el espacio a un subconjunto más relevante de términos, se facilita una posterior aplicación de técnicas de reducción lineal como *Latent Semantic Analysis* (LSA), que permitirán proyectar los documentos en un espacio semántico denso de menor dimensión.

Tras la aplicación del filtro, se ha obtenido una representación TF-IDF más compacta y relevante, reduciendo el vocabulario a 6233 términos. Para evaluar el impacto de este filtrado y comprobar si se ha logrado una mejor estructuración del corpus, se han recalculado las entropías medias por documento y por palabra. Los resultados se muestran en la Tabla 4.3.

Métrica	**Valor**	**Valor Máximo Posible**
Entropía media por documento	5.1367	≈ 12.6
Entropía media por palabra	3.9757	≈ 12.0

Cuadro 4.3: Valores medios de entropía tras el filtrado de información mutua

La entropía media por documento ha aumentado de forma considerable respecto a la matriz original, reflejando una mayor diversidad léxica en los textos, mientras que la entropía por palabra se mantiene moderada, lo que indica que muchas palabras conservan un valor discriminativo razonable. Esta combinación sugiere que el filtrado por información mutua ha eliminado términos redundantes o poco relevantes, conservando aquellos que aportan diferenciación entre clases y enriquecen semánticamente la representación.

Estas conclusiones también se ven reflejadas en la Figura 4.4, donde se muestran las distribuciones de las entropías por documento y por palabra tras el filtrado. En ambas se observa una forma aproximadamente normal, centrada en los valores medios indicados, sin colas excesivas ni concentraciones en valores extremos. Esto indica que el filtrado ha estabilizado el comportamiento informativo tanto a nivel de documentos como de términos, lo que constituye una buena base para aplicar técnicas de reducción de dimensionalidad como *Latent Semantic Analysis* (LSA).

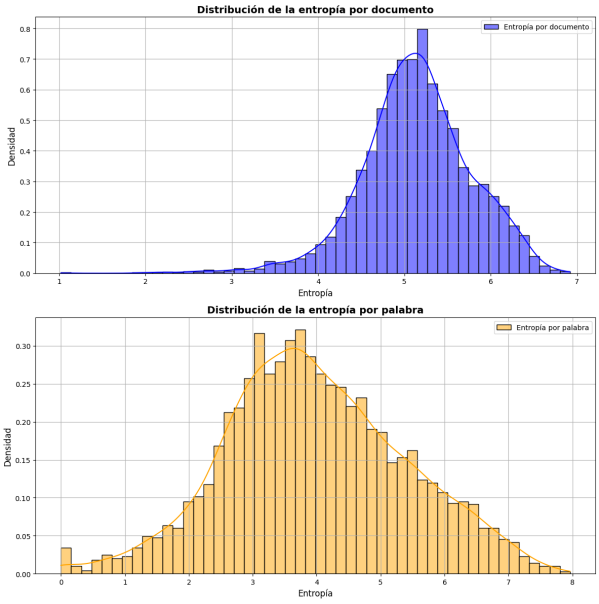


Figura 4.4: Distribución de la entropía por documento (arriba) y por palabra (abajo) tras el filtrado por información mutua

4.4.4. Latent Semantic Analysis (LSA)

Latent Semantic Analysis (LSA), introducida en el paper de Deerwester [37] es una técnica estadística ampliamente utilizada en el campo del procesamiento de lenguaje natural para capturar relaciones semánticas latentes entre palabras y documentos. Su objetivo principal es proyectar una matriz de alta dimensionalidad, típicamente una representación TF-IDF, a un espacio de menor dimensión donde se preserven las relaciones semánticas más importantes. Este proceso permite reducir el ruido, mejorar la eficiencia computacional y descubrir estructuras profundas en los datos textuales, incluso cuando las palabras no aparecen explícitamente en los mismos contextos. Numerosos estudios, como el realizado por Singh et al [65], han usado este método como técnica de reducción de dimensionalidad en el contexto de corpus de texto.

Desde el punto de vista matemático, LSA se basa en la **descomposición en valores singulares** (*Singular Value Decomposition*, SVD) de la matriz TF-IDF. Sea $X \in \mathbb{R}^{m \times n}$ la matriz TF-IDF, con m documentos y n palabras, la descomposición se expresa como:

$$X = U\Sigma V^\top$$

donde $U \in \mathbb{R}^{m \times r}$ y $V \in \mathbb{R}^{n \times r}$ son matrices ortogonales que contienen los vectores singulares izquierdos y derechos, respectivamente, y $\Sigma \in \mathbb{R}^{r \times r}$ es una matriz diagonal con los valores singulares ordenados de mayor a menor. El rango r representa el número de componentes no nulas de la descomposición.

LSA consiste en truncar esta descomposición a los primeros k componentes principales:

$$X_k = U_k \Sigma_k V_k^\top$$

donde $k < r$ y X_k es una aproximación de rango reducido de la matriz original. Esta reducción dimensional permite conservar las estructuras semánticas globales mientras se eliminan las variaciones menos significativas, que a menudo corresponden a ruido o relaciones superficiales. Las columnas de $U_k\Sigma_k$ representan los documentos en el nuevo espacio semántico latente, mientras que las filas de $V_k^\top$ representan las palabras en ese mismo espacio.

Una propiedad fundamental del LSA es la **ortogonalidad de las dimensiones latentes**, lo cual implica que las nuevas variables extraídas (componentes) están decorrelacionadas entre sí. Esto resulta ventajoso para análisis posteriores como visualización, clustering o clasificación, ya que reduce la redundancia entre características. Además, la descomposición SVD tiene una interpretación geométrica clara: cada documento se proyecta sobre una base ortonormal que captura las direcciones de máxima varianza, similar al análisis de componentes principales (PCA), pero aplicada a datos textuales.

Numerosos estudios han demostrado la eficacia de LSA en la representación y análisis de datos textuales. Por ejemplo, en el trabajo de Deerwester et al. [37], se introdujo formalmente LSA como una herramienta para mejorar la recuperación de información, mostrando cómo la reducción dimensional puede capturar relaciones semánticas implícitas en grandes colecciones de documentos. Posteriormente, Dumais [66] revisó múltiples aplicaciones exitosas de LSA en clasificación, agrupamiento y análisis semántico, destacando su capacidad para manejar sinónimos y polisemia. Más recientemente, investigaciones como la de Zhang et al. [67] han aplicado LSA sobre

representaciones TF-IDF en contextos multilingües y corpus técnicos, demostrando mejoras en tareas de categorización y recuperación.

Además de reducir la dimensionalidad, LSA genera una representación latente en la que cada documento puede interpretarse como una combinación lineal de los componentes semánticos extraídos. Los valores que toma cada documento en cada componente, positivos o negativos, reflejan el grado de asociación con esa dimensión semántica concreta. Un valor positivo elevado en una componente indica que el documento comparte significativamente el contenido temático que define dicha dimensión, mientras que un valor negativo puede interpretarse como oposición o ausencia de los patrones semánticos representados por esa componente. Esta simetría permite no solo identificar agrupaciones temáticas, sino también contrastes entre tipos de documentos, aportando una base sólida para el estudio topológico posterior y la clasificación supervisada basada en estructura semántica.

Para ejecutar esta técnica en Python, usaremos las clases TruncatedSVD y Normalizer de la librería de scikit-learn. De esta forma, eligiendo un número de componentes, obtendremos un dataframe cuyas columnas son ese número de compponentes, y cuyas filas son los documentos.

La Figura 4.5 muestra el valor absoluto medio de cada una de las 100 componentes latentes extraídas mediante *Latent Semantic Analysis* (LSA). Este valor medio refleja la magnitud promedio de la contribución de cada componente a la representación de los documentos, independientemente del signo. Se observa claramente que las primeras componentes tienen un valor significativamente más alto en comparación con las posteriores, lo que indica que concentran una mayor proporción de la variabilidad semántica del corpus.

Este patrón es coherente con la naturaleza matemática de la descomposición en valores singulares (SVD), sobre la que se fundamenta el LSA. Al igual que en el Análisis de Componentes Principales (PCA), los valores singulares están ordenados de mayor a menor, de forma que las primeras componentes corresponden a las direcciones del espacio vectorial donde se concentra la mayor varianza de los datos. En el contexto textual, esto significa que los primeros componentes latentes representan los temas o patrones semánticos más dominantes y globales, mientras que los siguientes capturan matices más específicos o menos frecuentes.

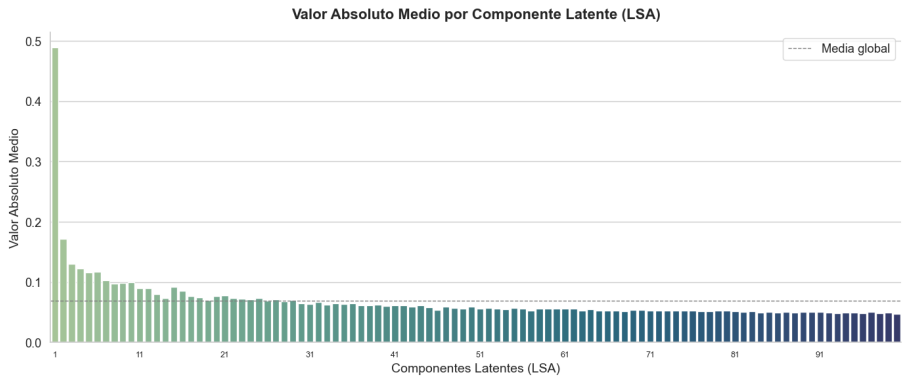


Figura 4.5: Valor absoluto medio de cada componente latente (LSA). La línea discontinua indica la media global de todas las componentes.

Para analizar estas componentes e intentar descifrar su semántica podemos intentar observar cuales son las palabras que tienen más peso en cada una de las componentes. Ya que no podemos analizar las 100 componentes, nos quedaremos con las 10 más importantes y relevantes.

En esta gráfica podemos ver cuales son las 10 palabras con mayor peso en cada una de las componentes latentes:

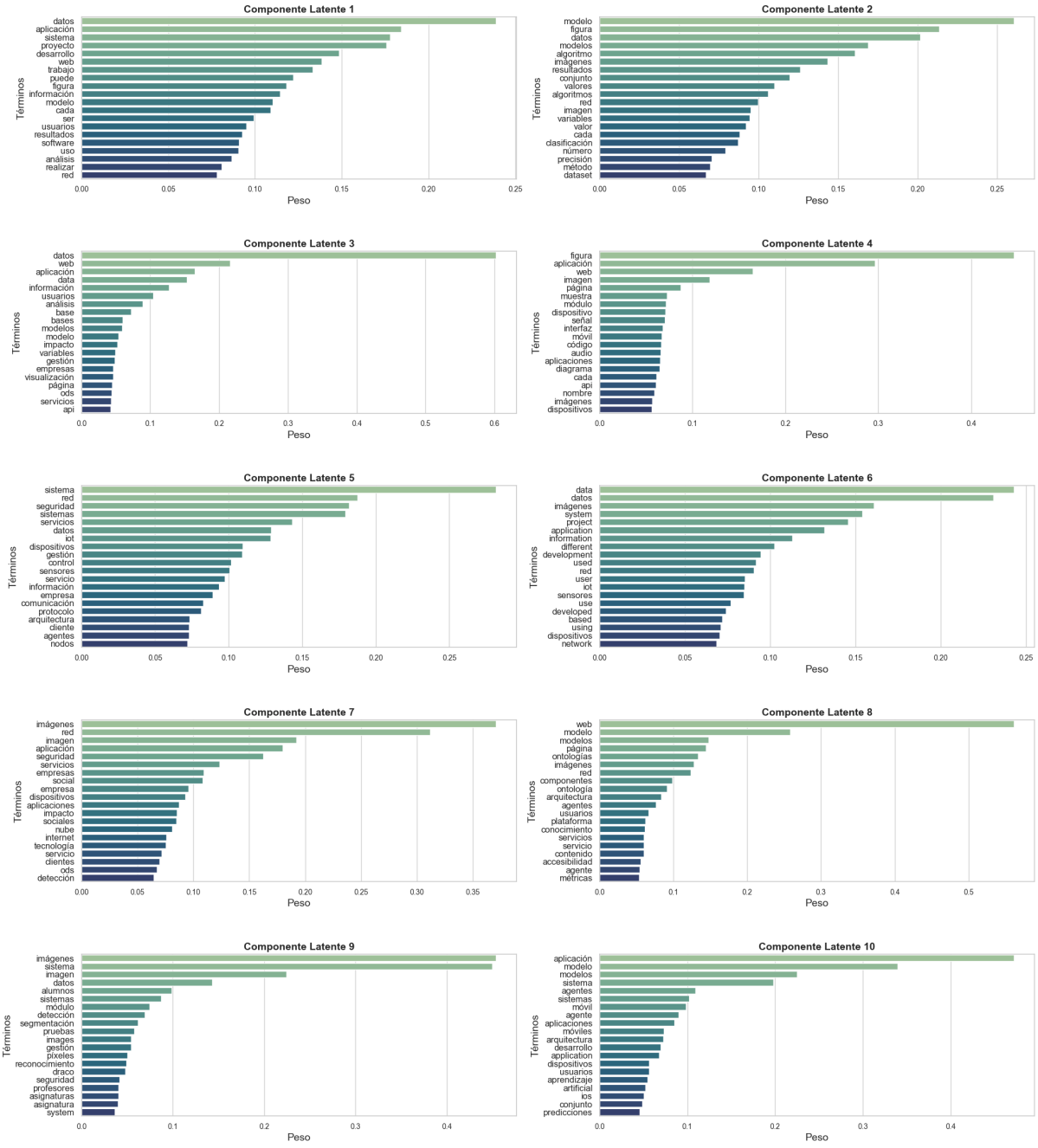


Figura 4.6: Top 10 términos más importantes en las 10 primeras componentes semánticas

A partir de los términos con mayor peso en cada una de las primeras componentes

latentes obtenidas mediante *Latent Semantic Analysis* (LSA), es posible asignar un significado temático aproximado a cada dimensión. La siguiente tabla resume la interpretación semántica de los 10 primeros componentes, basada en el análisis del vocabulario dominante en cada uno de ellos:

Componente	Tema principal	Palabras clave representativas
1	Ciencia de Datos y Visualización	datos, aplicación, procesamiento, información, visualización, software, analizar, modelo
2	Visión por Computador	modelo, imagen, clasificación, detección, segmentación, visión, métricas, dataset
3	Bases de Datos y Gestión de Información	base, datos, almacenamiento, gestión, sistemas, consultas, estructura, entorno
4	Interfaces de Usuario y Aplicaciones Móviles	figura, pantalla, interfaz, usuario, aplicación, dispositivo, móvil, experiencia
5	Ciberseguridad y Redes	sistema, seguridad, comunicación, nodos, dispositivos, protocolos, autenticación, firewall
6	Desarrollo de Software y Arquitecturas Web	data, software, services, architecture, desarrollo, aplicación, backend, cliente
7	Imágenes Médicas y Análisis Visual	imágenes, segmentación, tumores, estructuras, regiones, análisis, detección
8	Opinión y Web Semántica	web, opiniones, recomendación, usuarios, reviews, satisfacción, comentarios, métrica
9	Reconocimiento de Imágenes	imagen, reconocimiento, entrenamiento, clasificación, deep, red, modelo
10	Predicción y Simulación de Sistemas	predicción, modelo, simulación, parámetros, valores, estimación, sistema

Cuadro 4.4: Interpretación semántica de los 10 primeros componentes latentes del modelo LSA

Se puede apreciar que hay cierto solapamiento semántico entre distintas componentes. Para apreciarlo mejor podemos comentar la siguiente gráfica:

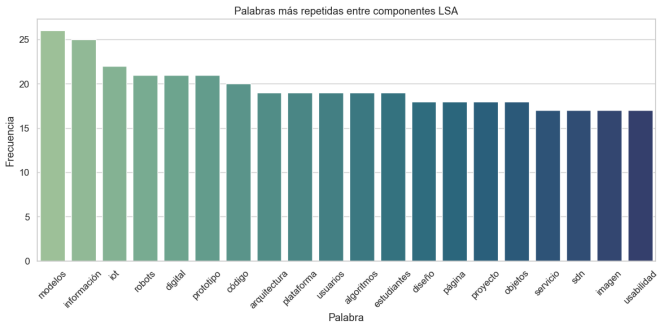


Figura 4.7: Conteo de las palabras más populares entre los componentes semánticos

Como se observa en la Figura 4.7, varios términos clave se repiten en múltiples componentes. Por ejemplo, palabras como *datos*, *modelo*, *imagen*, *aplicación* o *información* aparecen en distintas dimensiones latentes con pesos elevados. Este fenómeno indica que ciertas áreas temáticas, como el procesamiento de imágenes, el análisis de datos o el desarrollo de software, no están claramente separadas en componentes independientes, sino que están fragmentadas en varias dimensiones que capturan aspectos similares del contenido.

Este solapamiento semántico puede explicarse por la naturaleza del corpus, donde múltiples documentos abordan conceptos interrelacionados o reutilizan terminología común. Sin embargo, también plantea la posibilidad de que el número de componentes seleccionadas sea excesivo, generando una división artificial del espacio semántico. Por ello, esta observación refuerza la conveniencia de aplicar un criterio adicional para reducir el número de dimensiones, como la varianza acumulada o el rendimiento en tareas de clasificación, con el fin de preservar únicamente las componentes más informativas y diferenciadoras.

Para poder decidir el número de componentes vamos a basarnos en la varianza explicada acumulada hasta la componente K. Para mi caso donde $K = 100$ tenemos un valor cercano al 30 % de la varianza explicada. Debo hacer un inciso para aclarar que puede parecer poco, pero realmente hemos reducido la dimensión de 6233 componentes a tan solo 100, lo que significa que hemos reducido la dimensión en un **98.4 %**, conservando al mismo tiempo un 30 % de la parte significativa de la estructura semántica del corpus. En representaciones textuales de alta dimensionalidad, es habitual que la varianza explicada por los primeros componentes sea baja, ya que la información se encuentra fragmentada en múltiples direcciones. Por ello, es más importante que las componentes retenidas capturen relaciones semánticas relevantes y diferenciadoras entre documentos, más allá de su peso estadístico global.

Buscando un K más óptimo, hemos dibujado la siguiente gráfica, que relaciona el número de componentes con la varianza explicada por cada uno.

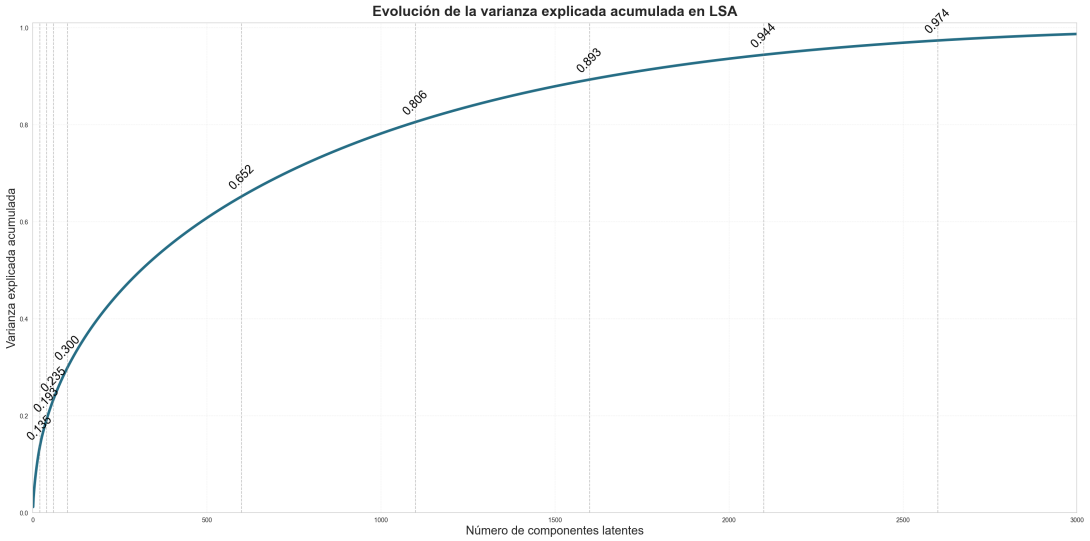


Figura 4.8: Evolución de la varianza explicada acumulada en LSA

En este sentido, la curva permite identificar regiones donde el crecimiento de la va-

rianza explicada se estabiliza, lo que puede ser indicativo de un punto de saturación a partir del cual añadir más componentes aporta información marginal. Aunque no utilizaremos este criterio como decisión final, sí nos proporciona un marco cuantitativo inicial para delimitar el rango de valores de K que exploraremos en los análisis topológicos posteriores. La elección definitiva se fundamentará en la evaluación empírica de la diferenciación estructural entre clases, basada en distancias persistentes y estructuras geométricas obtenidas mediante los diagramas de Vietoris–Rips. Viendo la gráfica, podríamos probar a usar desde 20 componentes, para intentar eliminar la redundancia semántica de las componentes, hasta incluso 600 componentes en caso de que queramos maximizar la varianza explicada.

La decisión final tomada será explicada en el capítulo 5

Capítulo 5

Clustering y Redacción de Hipótesis

El capítulo anterior finalizó con la aplicación de la técnica de LSA para reducir la dimensionalidad de nuestro dataset. Sin embargo, como vimos en la Figura 4.8, si escogíamos un número bajo de componentes, la varianza acumulada explicada era bastante baja, y llegaba a un 60 % a partir de las 600 componentes.

Con este primer análisis se busca aportar claridad sobre la mejor dimensión para nuestro dataset de LSA. Una vez hayamos conseguido una respuesta aplicaremos UMAP, una técnica de reducción de la dimensionalidad que será introducida más adelante. Gracias a la proyección en 2 dimensiones podremos visualizar tanto los datos como los cluster, lo que nos aportará un gran apoyo en el análisis. Siguiendo con la línea semántica, buscaremos una jerarquía entre las componentes de nuestro recién definido dataset. Gracias a ello podremos obtener una línea de análisis sobre las posibles dimensiones de nuestros datos. Por último, entraremos ligeramente en materia de la topología con el algoritmo Mapper, estudiando la forma que tiene nuestro conjunto.

5.1. Análisis semántico de los Datasets

En primer lugar, comenzaremos con un análisis semántico de los datasets. Ya sabemos que las dimensiones del LSA se corresponden de alguna manera con direcciones semánticas de los documentos. Dado el alto coste computacional de las operaciones topológicas en alta dimensionalidad, nos interesa que este número sea el más bajo posible sin dejar eliminar componentes semánticos de los documentos.

Vamos a comenzar estudiando los datasets con mayor dimensión, desde las 600 hasta las 100 dimensiones. Buscamos altos índices de repeticiones de palabras importantes. Esto querrá decir que se tratan de componentes semánticas que son bastante específicas.

Podemos seguir con el análisis de las palabras más populares entre componentes. Comenzando con el dataset de 600 componentes, nos encontramos la gráfica 5.1:

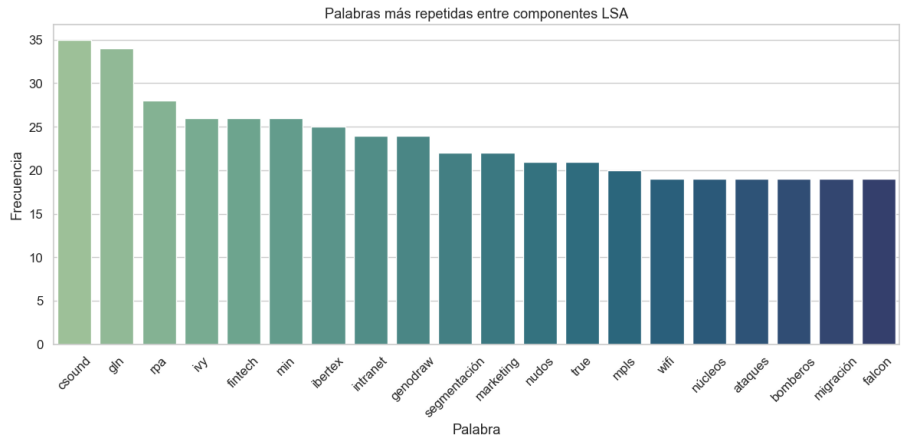


Figura 5.1: Conteo de las palabras más populares entre los componentes semánticos con 600 componentes

Un hallazgo destacable es la alta frecuencia con la que ciertas palabras de dominio altamente específico, como *csound*, *rpa* o *fintech*, aparecen entre las más representativas de múltiples componentes latentes. Este patrón sugiere la existencia de solapamiento semántico entre distintos ejes latentes, ya que en un modelo ideal estas palabras, por su naturaleza técnica y acotada, deberían asociarse de forma preferente a temáticas muy concretas. Su aparición transversal podría deberse a la influencia estadística que ejercen sobre el espacio latente, al número elevado de componentes empleados o a la falta de ortogonalidad semántica del modelo.

A medida que reducimos el número de dimensiones, las palabras mas técnicas dejan de estar en el ranking, y aparecen términos que se podrían usar en varios campos, como *segmentación*, *solución*, *móvil*. Sin embargo, sigue habiéndo palabras técnicas que no deberían de estar aquí.

Con el objetivo de analizar el grado de solapamiento semántico entre componentes latentes, se ha examinado la presencia de términos altamente específicos ,como *csound*, *rpa* o *fintech*, en las diez palabras más relevantes de cada componente generada por el modelo LSA. El cuadro 5.1 resume cuántas veces aparece cada término técnico en el top de cada componente, permitiendo identificar términos cuya influencia está distribuida a lo largo del espacio latente.

Componente	**csound**	**rpa**	**fintech**	**mpls**	**wifi**	**ivy**	**gln**
$Comp_{600}$	35	28	26	20	19	26	34
$Comp_{500}$	31	26	27	11	18	11	19
$Comp_{400}$	16	21	22	2	16	2	11
$Comp_{300}$	3	20	14	0	11	0	3
$Comp_{200}$	0	2	0	0	0	0	0
$Comp_{100}$	0	0	0	0	0	0	0

Cuadro 5.1: Presencia de palabras técnicas entre las 10 más relevantes de cada componente LSA

La Tabla 5.1 muestra la evolución en la presencia de ciertos términos altamente específicos entre las diez palabras más relevantes de cada componente latente del modelo LSA, a medida que se reduce el número total de componentes utilizadas. Este

análisis permite observar cómo cambia la distribución semántica del espacio latente en función del grado de compresión aplicado.

En el caso de $K = 600$, las palabras seleccionadas ,como *csound*, *rpa*, *fintech* o *gln*, aparecen con una frecuencia muy elevada, llegando a formar parte de las top 10 en más de 30 componentes. Esta situación sugiere un fuerte solapamiento semántico entre dimensiones, y puede interpretarse como una falta de especialización en la representación, donde los mismos términos dominan múltiples temas latentes. Esta redundancia puede ser problemática, ya que afecta tanto a la interpretabilidad como a la capacidad del modelo para representar temáticas diversas de forma diferenciada.

Sin embargo, conforme se reduce el número de componentes a 500, 400 e incluso 300, se observa una disminución notable en la presencia de estos términos. Esto indica que la compresión del espacio comienza a forzar una selección más diferenciada de vocabulario, y que los temas más generales tienden a ocupar un lugar más relevante frente a términos excesivamente técnicos o específicos. A partir de $K = 200$, la mayoría de estas palabras desaparecen completamente del top 10, y con $K = 100$, ninguna de ellas figura entre los términos más representativos de ninguna componente.

Este último caso, sin embargo, plantea una limitación importante: si bien se logra eliminar la redundancia semántica, también puede ocurrir que ciertas temáticas específicas ,precisamente aquellas ligadas a los términos eliminados, desaparezcan por completo del modelo. Esto sugiere que una compresión excesiva puede sacrificar la representación de dominios especializados, lo que debe tenerse en cuenta especialmente en tareas que requieren una cobertura temática diversa. Por tanto, la elección del número óptimo de componentes debe equilibrar la eliminación de solapamiento semántico con la preservación de la riqueza temática del corpus.

Parece que la opción mas coherente en este caso sería elegir el corpus con 100 o 200 componentes. Si que es cierto que la palabra *rpa* tiene cierta importancia en 2 de las componentes. Para asegurarnos de que esas 2 acepciones de *rpa* no se solapan semánticamente podemos comprobar de que trata cada una. Viendo las palabras en la Figura 5.2, los títulos que les podemos asignar son:

- Componente 180: Frameworks y sistemas inteligentes aplicados a la automatización
- Componente 193: Estudios técnicos y experimentales en entornos automatizados y simulados

Claramente la semántica de estas componentes no se solapa, por lo que podríamos dar como válido este test realizado sobre vocabulario específico y su solapamiento en componentes.

Gracias a esta idea del solapamiento semántico, podemos idear una métrica que se extienda más allá de las propias palabras técnicas. La idea es usar el top-n palabras con mayor importancia en la componente, y ver que solapamientos de palabras hay en ese top. Véase en la Figura 5.3 una curva típica de evolución de la importancia de las palabras.

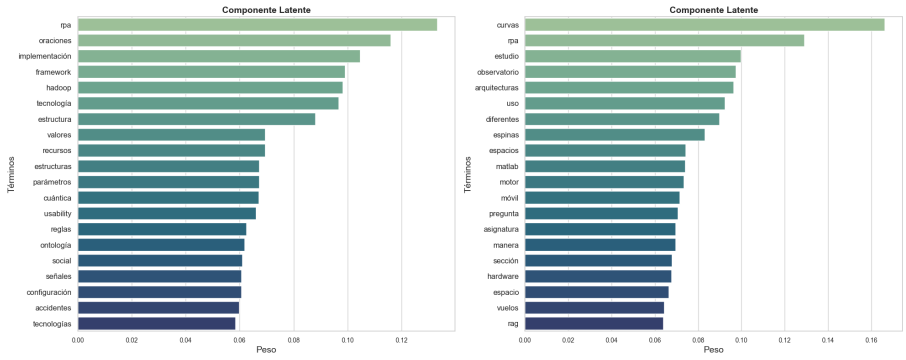


Figura 5.2: Comparación entre las palabras más relevantes en las componentes semánticas 180 (izq.) y 192 (der.)

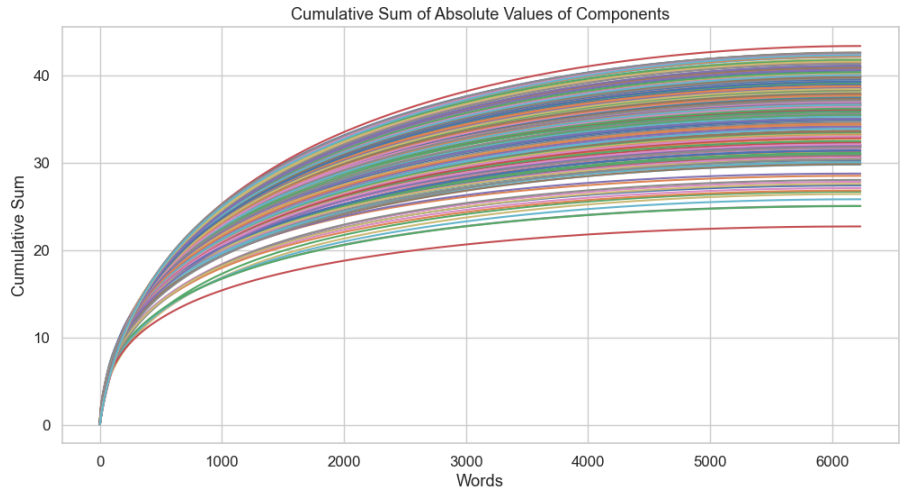


Figura 5.3: Descripción de la evolución de la importancia de las palabras en forma de suma acumulativa por componente

De esta forma observamos que con tener en cuenta entre las 500 y las 1000 palabras más importantes en cada componente, deberíamos de cubrir la parte semántica más importante del mismo. Esto quiere decir que entre las 500 y las 1000 palabras más relevantes de la componente serán las que guíen el significado del mismo.

Métrica	**200 comp.**	**100 comp.**	**80 comp.**	**60 comp.**
Palabras únicas en top-N	2540	2309	2237	2125
Palabras que aparecen en más de 1 componente	2249	2004	1913	1782
Proporción solapada (%)	88	86	85	83
Media de apariciones por palabra	39	21	17	14
Máxima aparición	168	92	78	57

Cuadro 5.2: Estadísticas del solapamiento semántico de palabras en top-N términos por componente para distintos valores de K

Tras comparar distintas configuraciones del modelo LSA, se concluye que la elección de $K = 100$ componentes podría represnetar un compromiso óptimo entre la riqueza dimensional necesaria para capturar patrones topológicos complejos y la necesidad

de mantener un vocabulario distribuido de forma semánticamente diferenciada entre componentes.

Tal como se muestra en la Tabla 5.2, al reducir de 200 a 100 componentes se consigue una disminución significativa en la media de apariciones por palabra (de 39 a 21) y en el número máximo de componentes en las que una palabra aparece (de 168 a 92), lo que indica una mejora clara en la especialización temática de las componentes. Al mismo tiempo, se conserva una cantidad elevada de términos únicos (2309 palabras), lo cual sugiere que no se está perdiendo excesiva diversidad léxica.

Para complementar este análisis, en la Figura 5.4 se muestra la distribución del número de componentes en las que aparece cada palabra entre las top-500 más relevantes. Se observa una clara concentración de palabras que solo aparecen en un pequeño número de componentes, lo cual es indicativo de un reparto semántico saludable. Si bien existe una cola larga que refleja la presencia de algunas palabras más generales o recurrentes, su presencia no es dominante, por lo que no compromete la diferenciación léxica global del modelo.

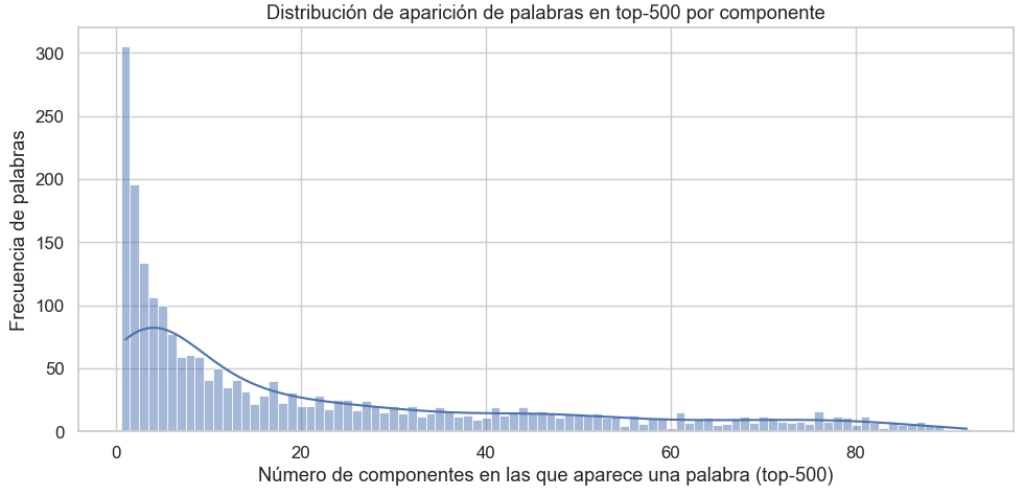


Figura 5.4: Distribución de aparición de palabras en top-500 por componente para $K = 100$

5.2. UMAP y Clustering

Para explorar la estructura latente de los documentos en el espacio generado por LSA, se ha empleado la técnica *UMAP* (Uniform Manifold Approximation and Projection), un método de reducción de dimensionalidad no lineal que permite preservar tanto la estructura local como la global de los datos en espacios de baja dimensión. Su aplicación en este contexto está justificada por la necesidad de representar de forma visual la distribución semántica de los documentos proyectados en el espacio latente, así como detectar agrupamientos potenciales o estructuras topológicas relevantes.

UMAP fue introducido por McInnes, Healy y Melville [36] como una alternativa más eficiente y flexible que técnicas previas como t-SNE. A diferencia de estas, UMAP está basado en una formulación matemática desde la perspectiva de la teoría de variedades y la topología algebraica. El algoritmo parte de la suposición de que los datos están distribuidos sobre una variedad de dimensión baja embebida en un espacio de

dimensión mayor, y busca construir un grafo que represente las relaciones de vecindad entre puntos en el espacio original. Posteriormente, este grafo es optimizado en un espacio de menor dimensión preservando sus propiedades estructurales mediante técnicas de aprendizaje aproximado.

Formalmente, UMAP construye una aproximación *fuzzy* simplicial a la estructura de la variedad utilizando distancias locales basadas en el parámetro *n_neighbors*, y posteriormente minimiza una función de pérdida cruzada entre distribuciones de probabilidad de grafos de alta y baja dimensión. Este enfoque permite mantener la coherencia en las relaciones entre puntos cercanos (estructura local), al tiempo que conserva relaciones globales a mayor escala, lo cual es particularmente útil en tareas de exploración y visualización de datos textuales.

Numerosos trabajos recientes han adoptado UMAP como herramienta para la exploración de espacios semánticos derivados de técnicas como LSA, LDA o embeddings basados en BERT. Por ejemplo, [68] utilizaron UMAP para visualizar clústeres semánticos de textos científicos, mientras que [69] lo emplean para analizar relaciones entre conceptos en modelos generativos. Su eficiencia computacional, interpretabilidad visual y estabilidad lo convierten en una opción idónea para el análisis exploratorio de datos textuales de alta dimensión.

Como ya se ha comentado, UMAP parece una herramiente perfectamente útil en nuestra temática, ya que mantiene la estructura topológica presente en los datos incluso al bajarlos de dimensión. Esto provoca que el resultado que visualizamos no esté tan lejos de lo que realmente es la proyección de nuestro dataset.

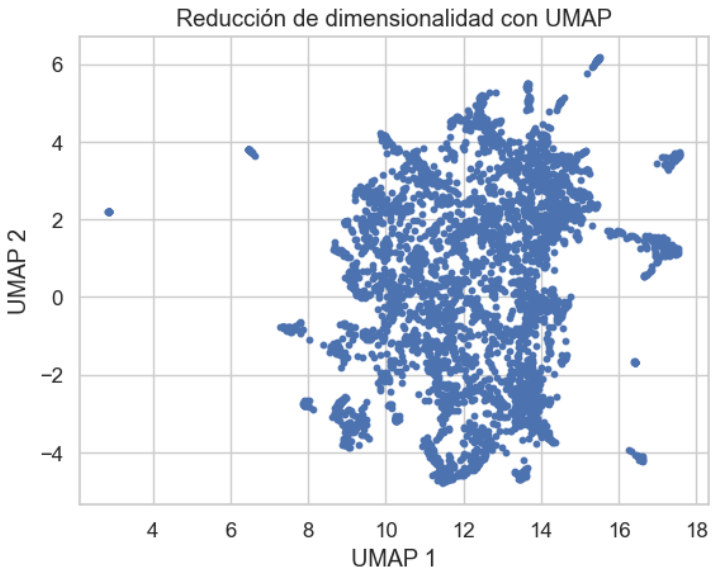


Figura 5.5: Proyección bidimensional del espacio LSA mediante UMAP

En la Figura 5.5 se muestra la reducción bidimensional de los documentos proyectados en el espacio latente de LSA ($K = 100$), realizada con la técnica UMAP. A nivel global, la nube presenta una estructura claramente no aleatoria, en la que se distingue una **zona central de alta densidad** rodeada de **agrupaciones periféricas parcialmente diferenciadas**, lo cual sugiere que existen ciertas regularidades semánticas que UMAP ha logrado preservar.

Este tipo de morfología es indicativa de un *espacio semántico con una estructura central común a la mayoría de los documentos*, pero también con subconjuntos que se desvían hacia regiones diferenciadas, posiblemente por pertenecer a contextos más específicos o temáticas más especializadas. Aunque la proyección no revela clústeres totalmente separados, sí se pueden identificar *grupos de documentos más cohesionados* alrededor del núcleo, lo que abre la posibilidad de que los algoritmos de agrupamiento puedan descubrir particiones con sentido.

Además, la relativa continuidad del espacio proyectado sugiere que existe una **gradualidad semántica** en lugar de límites estrictos entre clases, lo cual es común en datos textuales reales. Debido a este comportamiento, seguiramente debamos de buscar algoritmos de clustering que no requieran de particiones rígidas ni circulares. Un ejemplo del mismo es HDBSCAN, muy utilizado en contextos de proyecciones con UMAP.

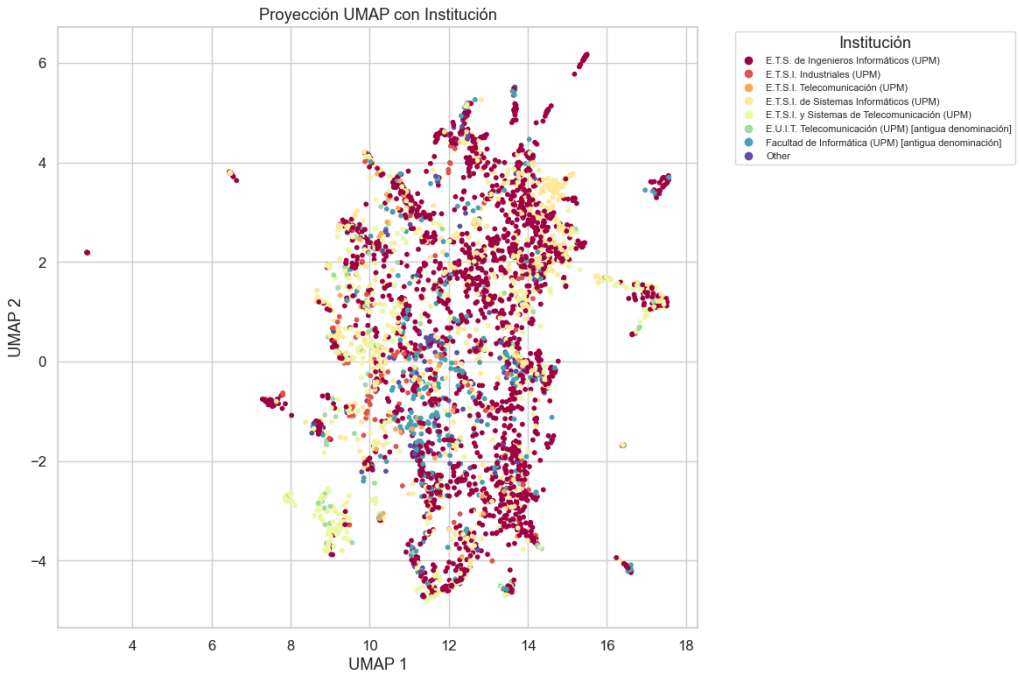


Figura 5.6: Proyección UMAP de los documentos, coloreada por institución

La Figura 5.6 representa la proyección UMAP de los documentos, coloreados según la institución a la que pertenecen. Para evitar clases con una representación residual, aquellas instituciones con menos de 50 documentos han sido agrupadas bajo la categoría *Other*.

A primera vista, se aprecia que **no existe una separación clara entre las diferentes instituciones** en el plano proyectado. Las distintas categorías se solapan ampliamente en la mayoría de regiones, lo cual sugiere que las representaciones semánticas generadas por LSA no son completamente discriminativas respecto a la institución. Este fenómeno podría estar relacionado con la diversidad temática compartida entre centros o con la existencia de enfoques similares en distintas titulaciones, especialmente en áreas afines como ingeniería, informática o telecomunicaciones. Además, el *marco temporal* también puede influir significativamente, ya que el vocabulario académico tiende a difuminarse entre clases a lo largo del tiempo, ya sea por modas

terminológicas, cambios en los significados o la adopción común de ciertos términos, lo que añade una dimensión evolutiva y compleja al análisis semántico.

Sin embargo, es destacable que algunas categorías ,como *E.T.S. de Ingenieros Informáticos (UPM)* o *E.T.S.I. Industriales (UPM)*, muestran cierta tendencia a agruparse en regiones específicas del espacio latente, particularmente en las zonas periféricas del gráfico. Esto sugiere que, si bien el modelo no consigue una separación tajante entre clases, sí existen áreas donde ciertas instituciones dominan de forma relativa, lo que podría ser aprovechado por modelos más sensibles a estructuras locales, como HDBSCAN o clasificadores topológicos basados en distancias.

En conjunto, la visualización confirma que la proyección UMAP no revela fronteras nítidas entre las instituciones, aunque sí ofrece indicios de patrones organizativos más sutiles que podrían ser explotados mediante técnicas de agrupamiento no supervisadas o análisis topológicos posteriores.

Figura 5.7: Proyección UMAP de documentos separados por ODS (Objetivos de Desarrollo Sostenible)

En la Figura 5.7 se representa la proyección UMAP de los documentos, dividida según el ODS al que cada uno ha sido asignado. Esta visualización permite analizar de forma intuitiva si existen estructuras semánticas diferenciadas asociadas a cada uno de los objetivos sostenibles.

De forma general, se aprecia que ciertos ODS presentan una clara **agrupación y densidad** dentro de la proyección, lo cual sugiere una fuerte cohesión temática entre los documentos que los componen. Este fenómeno es particularmente evidente en los ODS **03 (Salud y bienestar)**, **04 (Educación de calidad)** y **09 (Industria, in-**

novación e infraestructura), donde se observa una alta concentración de puntos en regiones específicas del espacio UMAP. Esto podría deberse a que los textos relacionados con estos objetivos utilizan un vocabulario técnico o temático más especializado y consistente, facilitando así su separación en el espacio latente.

En contraste, otros ODS como el **01 (Fin de la pobreza)**, **02 (Hambre cero)** o **13 (Acción por el clima)** presentan una distribución mucho más dispersa, lo que sugiere una mayor heterogeneidad léxica o conceptual entre los documentos asignados. Esto podría estar relacionado con la naturaleza más transversal o abstracta de estos objetivos, que abordan problemáticas complejas desde enfoques variados.

Esta visualización resulta especialmente útil para explorar la viabilidad de tareas de clasificación o agrupamiento en función de los ODS, y también proporciona una primera validación visual del modelo semántico construido a partir de LSA y UMAP.

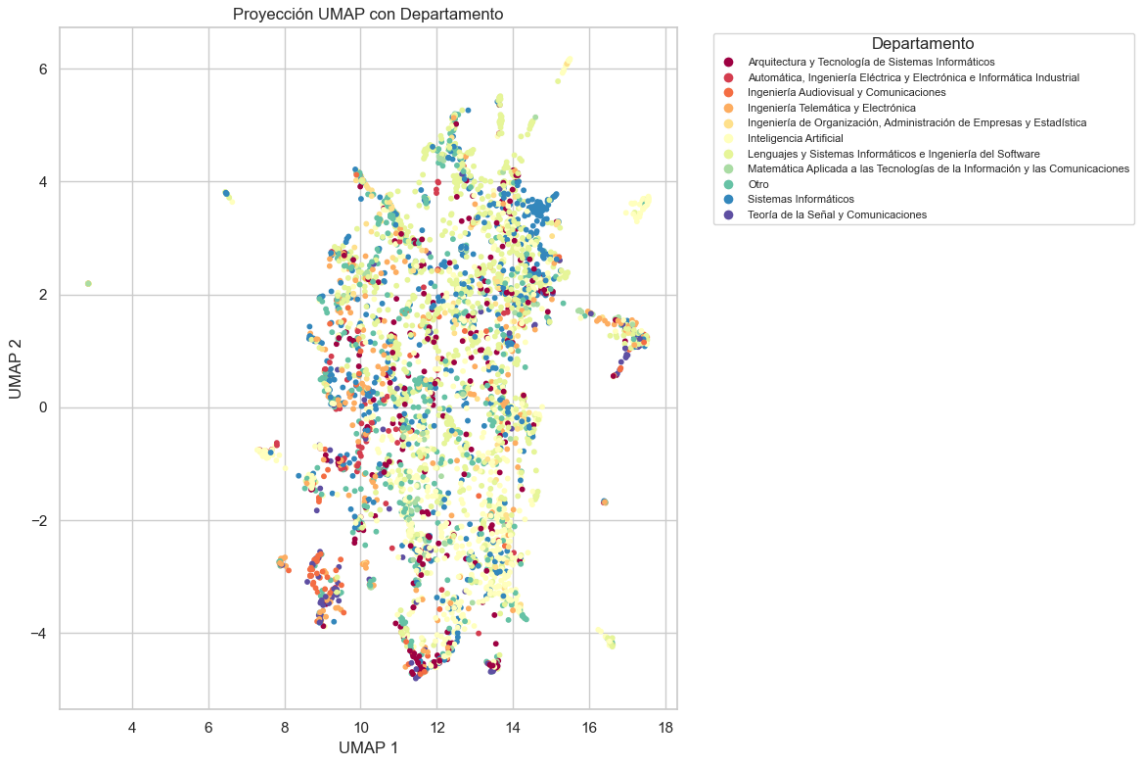


Figura 5.8: Proyección UMAP de los documentos coloreada por departamento académico

La Figura 5.8 muestra la proyección UMAP de los documentos, esta vez coloreada según el departamento académico asociado a cada uno. Al igual que en el caso anterior con la variable *Institución*, no se observan **fronteras nítidas** entre categorías, aunque sí se aprecian ciertas **zonas de predominancia relativa** por parte de algunos departamentos.

Por ejemplo, el departamento de *Inteligencia Artificial* se manifiesta con cierta concentración visible en regiones concretas del espacio, en especial en los extremos laterales del mapa. Esto puede deberse a la especificidad del vocabulario utilizado en este ámbito. De forma similar, otras categorías como *Ingeniería Audiovisual y Comunicaciones* o *Lenguajes y Sistemas Informáticos* tienden a distribuirse de forma más agrupada,

en comparación con departamentos como *Matemática Aplicada* o *Organización y Estadística*, cuyos documentos aparecen más dispersos a lo largo del espacio latente.

El solapamiento generalizado entre muchas clases también puede deberse a que algunos departamentos comparten temáticas o líneas transversales de investigación, lo cual se traduce en representaciones semánticas similares en el espacio LSA. A esto se suma el solapamiento temporal, ya que el vocabulario tiende a difundirse y evolucionar entre promociones a lo largo del tiempo, lo que dificulta aún más la delimitación semántica entre departamentos. No obstante, la existencia de pequeñas agrupaciones visuales sugiere que sí hay **estructura semántica latente vinculada al departamento**, aunque probablemente no sea lineal ni fácilmente separable sin técnicas más especializadas.

5.2.1. HDBSCAN

Para la tarea de agrupamiento no supervisado en espacios latentes complejos, uno de los algoritmos más robustos y adecuados es **HDBSCAN** (Hierarchical Density-Based Spatial Clustering of Applications with Noise) [70]. Este método es una extensión del conocido algoritmo DBSCAN, del cual hereda la capacidad para detectar clústeres de forma arbitraria sin necesidad de especificar un número de grupos *a priori*, y lo mejora mediante una formulación jerárquica más flexible y estable.

A diferencia de DBSCAN, que utiliza un umbral de densidad fijo para identificar regiones densas, HDBSCAN construye un árbol jerárquico de densidades basado en una transformación del espacio original mediante una métrica de densidad relativa. Esta jerarquía de clústeres se genera al analizar cómo se agrupan los puntos a diferentes escalas de densidad. Posteriormente, esta estructura se condensa utilizando criterios de estabilidad para seleccionar automáticamente los clústeres más persistentes. El resultado es una partición más refinada, que permite capturar grupos con densidades heterogéneas y además identificar puntos atípicos o ruido.

Una de las grandes ventajas de HDBSCAN es que sólo requiere un parámetro principal: `min_cluster_size`, que controla el tamaño mínimo de los clústeres a detectar. Además, a diferencia de DBSCAN, este algoritmo es más robusto ante diferencias de densidad interna entre clústeres y evita la necesidad de ajustar de forma manual el parámetro `eps`, que suele ser problemático y sensible en entornos de alta dimensionalidad.

En el contexto de este trabajo, el uso de HDBSCAN se justifica especialmente al combinarse con la proyección generada por **UMAP**. Dado que UMAP tiende a preservar relaciones de vecindad locales en una dimensión reducida, el espacio resultante es particularmente apto para aplicar técnicas de agrupamiento basadas en densidad. En este nuevo espacio 2D o 3D, los documentos que comparten patrones semánticos similares suelen agruparse de manera compacta, lo que facilita la detección de clústeres significativos sin asumir una geometría específica o número fijo de grupos.

Por tanto, la aplicación de HDBSCAN sobre el espacio UMAP representa una solución eficiente y coherente con los objetivos del análisis topológico y semántico de documentos, permitiendo identificar agrupaciones naturales sin supervisión y proporcionando una interpretación más rica de la estructura interna del corpus.

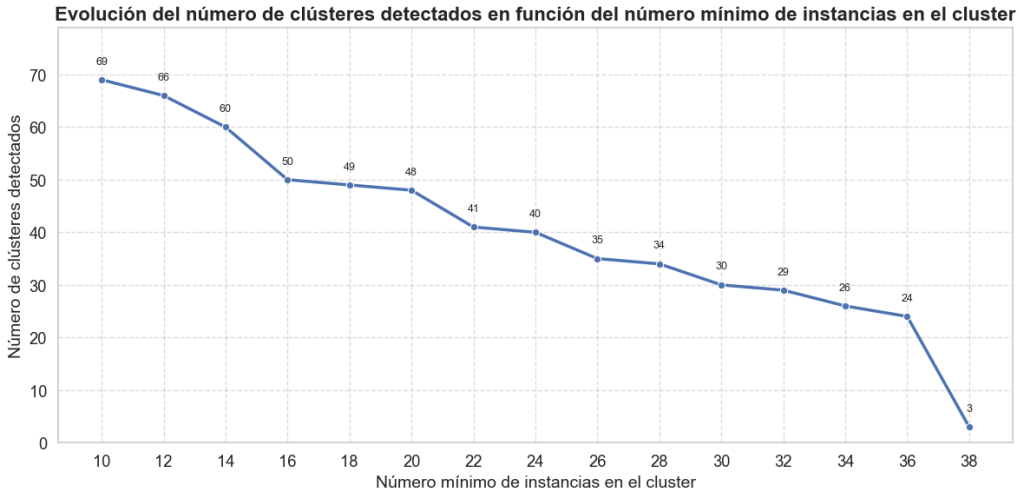


Figura 5.9: Evolución del número de clústeres detectados en función del parámetro `min_cluster_size`

En la Figura 5.9 se muestra cómo varía el número de clústeres detectados por HDBSCAN en función del parámetro `min_cluster_size`, utilizando distancia euclídea. En general, se observa una tendencia decreciente: al aumentar este parámetro, se exige mayor densidad para formar agrupaciones, lo que reduce el número de clústeres y clasifica muchos grupos pequeños como ruido. Hasta un valor cercano a 18, la reducción es progresiva, pero a partir de ese umbral se produce una caída abrupta en la cantidad de grupos detectados. Resultados similares se obtienen con otras métricas, por lo que no se presentan en este apartado.

En la Figura 5.10, se comparan los resultados del algoritmo HDBSCAN aplicando diferentes valores del parámetro `min_cluster_size` (18, 28 y 38) sobre la proyección bidimensional obtenida mediante UMAP. Este parámetro controla el tamaño mínimo permitido para considerar un conjunto de puntos como un clúster denso. Las diferencias entre las tres configuraciones ilustran la sensibilidad del algoritmo a esta elección.

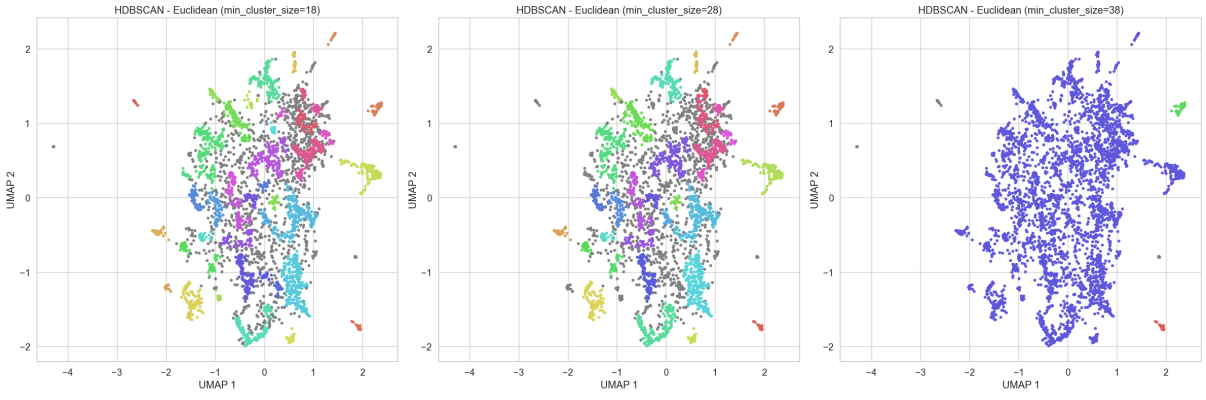


Figura 5.10: Comparativa de HDBSCAN sobre UMAP con distintos valores de `min_cluster_size`. Los puntos en gris representan ruido no asignado.

En la primera configuración (`min_cluster_size` = 18), el modelo genera un gran

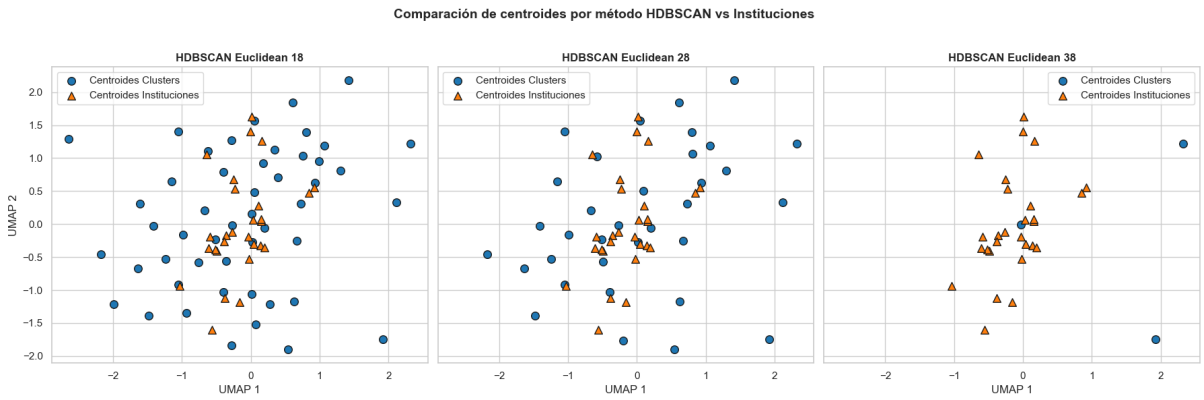


Figura 5.11: Comparación de centroides generados por HDBSCAN (distintos valores de `min_cluster_size`) frente a los centroides de las instituciones.

número de agrupaciones, muchas de ellas pequeñas y potencialmente inestables. Este enfoque ofrece una segmentación fina, aunque también más sensible al ruido.

En el segundo gráfico (`min_cluster_size` = 28), el número de grupos disminuye de forma notable. Las estructuras pequeñas se han descartado como ruido, pero los patrones principales ,correspondientes a regiones de mayor densidad, se mantienen prácticamente invariantes, mostrando una mayor robustez estructural.

Por el contrario, en la tercera visualización (`min_cluster_size` = 38), la segmentación es excesivamente agresiva. El algoritmo apenas identifica clústeres, absorbiendo la mayoría de los puntos como un único conjunto. El resultado es una estructura poco informativa y donde la interpretabilidad erradica en que todos son documentos pertenecientes al dominio informático, excepto algunos más alejados.

A nivel general, se concluye que existe una región de estabilidad topológica en torno a `min_cluster_size` = 28–34, donde se eliminan grupos irrelevantes sin sacrificar los patrones principales. Este equilibrio permite capturar la estructura latente del conjunto de datos sin introducir una fragmentación artificial.

En la Figura 5.11 se muestra la comparación entre los centroides obtenidos a partir del agrupamiento con HDBSCAN (círculos azules) y los centroides de cada institución (triángulos naranjas) en el espacio embebido por UMAP. Esta comparación permite analizar hasta qué punto los clústeres descubiertos por el modelo tienen correspondencia o solapamiento con las agrupaciones naturales que representan las instituciones de origen de los documentos.

En la Figura 5.11, se muestra la comparación entre los centroides calculados por HDBSCAN para diferentes valores de `min_cluster_size` y los centroides medios de cada institución. A lo largo de las tres configuraciones, se observa una pauta clara: los centroides institucionales se concentran en la zona central del espacio UMAP, mientras que los centroides de los clústeres detectados están significativamente más dispersos.

Esta distribución sugiere que los grupos descubiertos por HDBSCAN representan focos semánticos concretos y localizados. Por el contrario, los documentos pertenecientes a una misma institución no tienden a agruparse en torno a un único foco temático, sino que se reparten entre distintos clústeres semánticos. Esto explicaría

la concentración central de sus centroides: son el resultado de promediar posiciones repartidas por todo el espacio temático.

Algo similar ocurre en la Figura 5.12, donde se presenta una comparación entre los centroides de los clústeres obtenidos mediante HDBSCAN y los centroides asociados a cada ODS (Objetivo de Desarrollo Sostenible). Al igual que en el caso anterior con las instituciones, se observa que los centroides de los clústeres se distribuyen por todo el espacio UMAP, reflejando zonas semánticas específicas y bien definidas.

En contraste, los centroides de los ODS aparecen aún más agrupados en la región central del espacio. Esta concentración sugiere que, si bien los documentos asociados a un ODS pueden relacionarse con múltiples temáticas específicas, su centro de gravedad semántico tiende a coincidir en una región común, más aún que en el caso de las instituciones. Esto podría interpretarse como una mayor transversalidad temática dentro de cada ODS, cuyo contenido no se limita a un único foco semántico sino que abarca varias regiones del espacio temático.

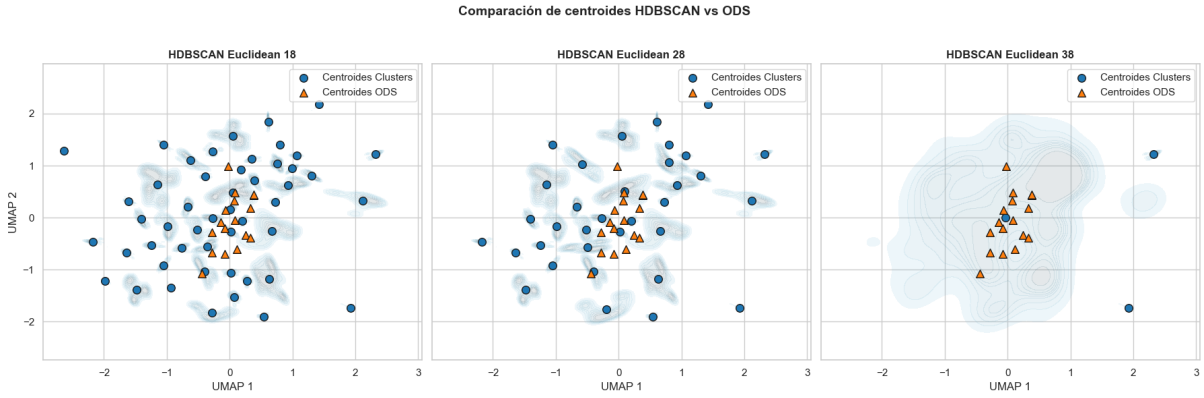


Figura 5.12: Comparación de centroides generados por HDBSCAN (distintos valores de `min_cluster_size`) frente a los centroides de las instituciones.

5.2.2. K-Means

Para contrastar los métodos de agrupamiento basados en densidad, se ha aplicado también *K-Means* sobre las coordenadas obtenidas con UMAP, explorando así una partición más tradicional basada en centroides. Como paso previo, se ha evaluado el número óptimo de clústeres mediante el *Silhouette Score*, una métrica que estima la calidad de la separación entre grupos evaluando la cohesión interna frente a la separación entre clústeres.

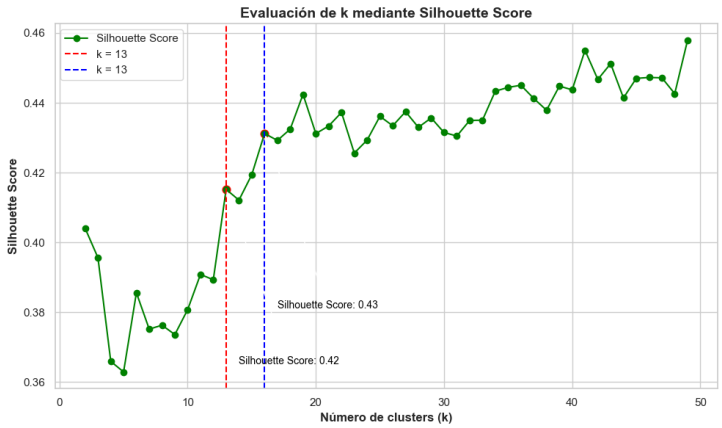


Figura 5.13: Selección del número de clústeres (k) en K-Means mediante *Silhouette Score*

En la Figura 5.13 se muestra la evolución del *Silhouette Score* para valores de k entre 2 y 50 al aplicar *K-Means* sobre las coordenadas reducidas mediante UMAP. Se observa una tendencia ascendente en la métrica a partir de $k = 12$, con mejoras graduales que culminan en un máximo cercano a $k = 49$. Sin embargo, destaca una subida más abrupta entre $k = 13$ y $k = 16$, donde se pasa de un valor de 0.39 a aproximadamente 0.43. Esta región intermedia sugiere una partición más coherente con la estructura latente de los datos, especialmente si tenemos en cuenta que previamente se ha identificado un total de 16 temas semánticos principales (como comentaremos más adelante). Por tanto, valores de k en torno a ese rango resultan particularmente relevantes para el análisis posterior. También podría considerarse fijar k igual al número de clases conocidas (cerca de 26), pero dicha configuración presenta un *Silhouette Score* muy similar al de otros valores intermedios, sin mejoras destacables en términos de cohesión o separación.

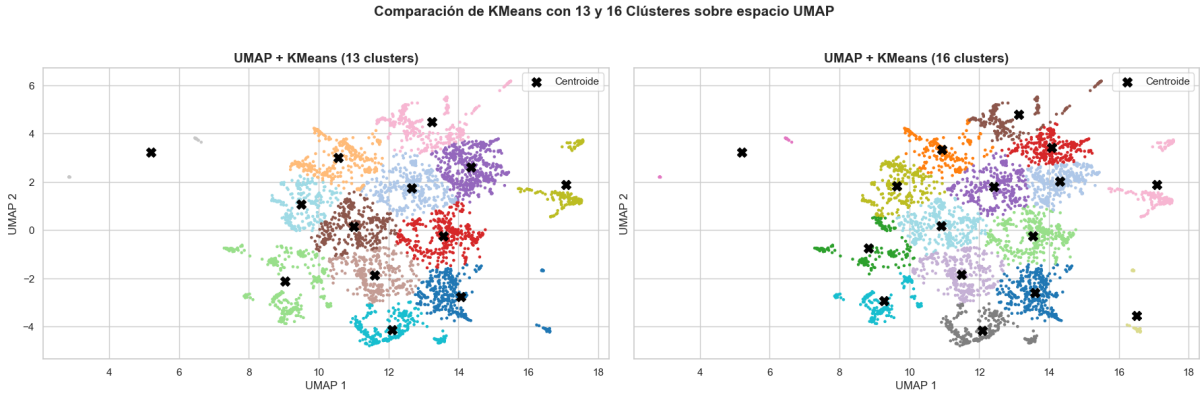


Figura 5.14: Comparación visual de *K-Means* con $k = 13$ y $k = 16$ sobre la proyección UMAP. Se muestran los clústeres identificados y sus centroides.

En la Figura 5.14 se comparan los resultados de aplicar *K-Means* sobre la proyección UMAP con $k = 13$ y $k = 16$. Se observa que la mayoría de los centroides se mantienen en posiciones similares en ambos casos, lo cual indica cierta estabilidad en las agrupaciones dominantes del espacio. El incremento de k provoca la división de algunos

clústeres ya existentes, como es el caso del clúster verde ubicado en la zona inferior izquierda, que se subdivide de forma natural en dos regiones más densas. Este tipo de divisiones sugiere que, al aumentar la granularidad, se capturan estructuras internas adicionales sin desestabilizar la segmentación general del conjunto.

5.3. Clustering en componentes

Para poder extender nuestro análisis preliminar y la información obtenida a través del clustering con la proyección UMAP, nos gustaría realizar un análisis abordando los posibles solapamientos semánticos en las componentes obtenidas. De esta forma, podremos acompañar el análisis inicial realizado sobre los propios documentos y sus Instituciones y ODS. Este análisis busca extender lo comentado al principio del capítulo. Para este objetivo usaré un algoritmo de clustering jerárquico.

5.3.1. Clustering Jerárquico

Una dificultad inherente a este enfoque radica en la propiedad de ortogonalidad del LSA, que genera componentes mutuamente independientes en el espacio vectorial. Esto provoca que las distancias entre ellas tiendan a ser uniformes, lo que impide que técnicas como el *clustering* jerárquico puedan establecer una jerarquía clara: todas las componentes aparecen a una distancia prácticamente equidistante del resto.

No obstante, en análisis previos se observó que cada componente podría estar caracterizada por un subconjunto reducido de términos de alta relevancia, por ejemplo, las 500 o 1000 palabras con mayor peso. Esta observación permite abordar el problema desde una perspectiva alternativa: calcular la similitud entre componentes en función de la coincidencia en sus términos más representativos.

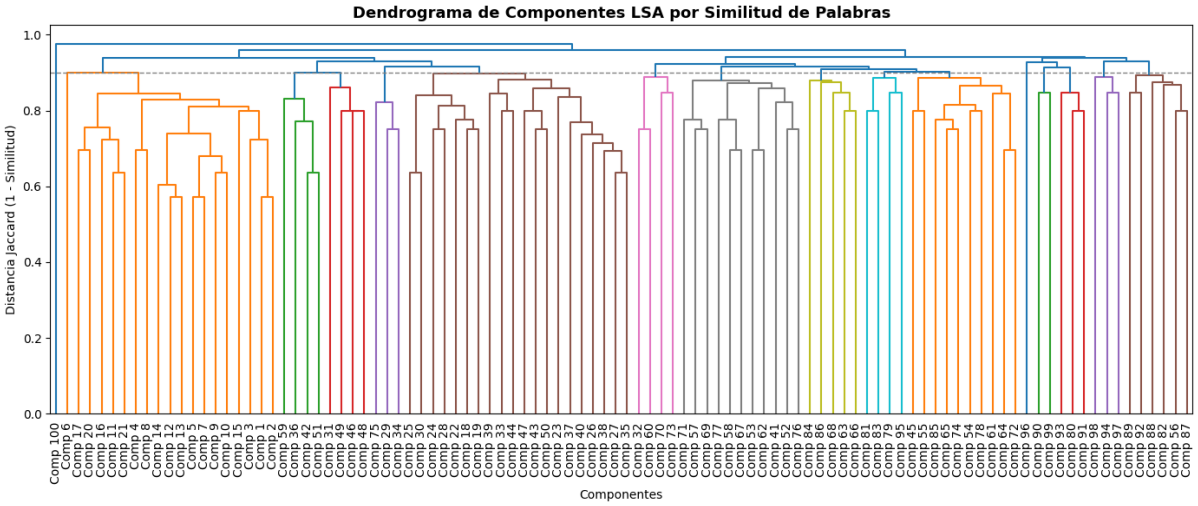


Figura 5.15: Dendrograma jerárquico de las componentes LSA basado en la similitud de términos relevantes

La Figura 5.15 muestra el dendrograma resultante de aplicar *clustering* jerárquico a las componentes latentes generadas por LSA, utilizando como criterio de similitud la intersección de sus *top-500* términos más relevantes. La métrica empleada ha sido la

distancia de Jaccard, que cuantifica la disimilitud entre conjuntos: valores cercanos a 0 indican conjuntos muy similares (alta coincidencia de términos), mientras que valores próximos a 1 representan conjuntos disjuntos.

Como puede observarse, las componentes comienzan a agruparse a partir de una distancia Jaccard de aproximadamente 0.6. Esto implica que hay subconjuntos de componentes que comparten al menos un 40 % de sus términos principales, lo que refuerza la idea de la existencia de cierto solapamiento semántico. A medida que se reduce el umbral de similitud (mayor altura en el dendrograma), se van unificando agrupaciones cada vez más dispares.

Si vamos más allá y buscamos dados los centroides de los componentes cuales son sus posibles temáticas nos encontramos lo siguiente:

Cluster	Tema estimado	Palabras clave indicativas
0	Web semántica y ontologías	datos, ontologías, usuarios, imágenes, información, semántico, ontología, agregación
1	IoT y sensores	nodos, sensores, imagen, arduino, detección, online, smart
2	Infraestructura y UPM	campus, telecomunicaciones, upm, plataforma, entorno, pruebas
3	Prototipado y diseño	prototipo, diseño, desarrollo, comunicación, interfaz, dispositivo, sensor
4	IA aplicada a proyectos	proyectos, simulación, artificial, inteligencia, análisis, predicción
5	Robótica y visión artificial	calidad, robots, visión, sensores, cámara, imagen, estructuras
6	Ciberseguridad y malware	red, malware, amenazas, ataques, firewall, ransomware
7	Sistemas de información	datos, usuarios, gestión, arquitectura, modelos, análisis
8	Computación distribuida y cloud	digital, edge, proyectos, blockchain, docker, infraestructura
9	Educación y aprendizaje	aprendizaje, educación, evaluación, formación, estudiantes, recursos
10	Economía digital y plataformas	economía, servicios, comercio, plataformas, consumo, herramienta
11	Ciencia de datos y predicción	clasificación, predicción, machine, datos, aprendizaje, modelo
12	Seguridad y protocolos	autenticación, protocolo, seguridad, comunicación, cifrado
13	Deep learning y visión por computador	clasificación, imagen, redes, real, entrenamiento, convolucional
14	PLN y traducción automática	traducción, frases, texto, procesamiento, python, análisis
15	Energía y sostenibilidad	energía, recursos, sostenibilidad, cambio, residuos, clima

Cuadro 5.3: Temas estimados de los clusters obtenidos a partir de los centroides semánticos

El análisis de los términos más representativos de cada clúster permite una interpretación temática bastante clara. Como puede observarse, los grupos descubiertos reflejan áreas técnicas específicas (como IoT, visión artificial o ciberseguridad) y aplicaciones concretas (como educación, energía o economía digital).

A la luz de los resultados obtenidos, podríamos concluir que las componentes semánticas extraídas mediante LSA tienden a organizarse como ramas más específicas

de un conjunto reducido de grandes temáticas, representadas aquí por los 16 clústeres identificados. Cada clúster refleja una agrupación coherente de componentes que comparten vocabulario dominante y orientación temática.

Cabe destacar que ciertos clústeres, como el **Cluster 4** (IA aplicada a proyectos) o el **Cluster 0** (datos, ontologías, imágenes), agrupan un mayor número de componentes (20 y 19 respectivamente), lo que sugiere que corresponden a temáticas más generales o transversales dentro del corpus analizado. Por el contrario, otros clústeres como el **Cluster 12** o el **Cluster 15**, con una sola componente cada uno, representan áreas semánticas mucho más específicas o menos frecuentes.

Esta distribución apoya la idea de que, aunque el modelo LSA genera componentes ortogonales, la estructura semántica subyacente no es completamente uniforme, y tiene cabida a un análisis más profundo.

5.4. Algoritmo Mapper

El algoritmo *Mapper*, introducido por Singh et al. en 2007 [71], es una herramienta fundamental dentro del campo del Análisis Topológico de Datos. Mapper permite construir una representación simplificada de la estructura topológica de un conjunto de datos de alta dimensión, capturando tanto su forma global como la presencia de estructuras locales significativas.

Formalmente, Mapper opera sobre un espacio métrico (X, d) y una función de filtro $f : X \to \mathbb{R}$ (o $\mathbb{R}^k$), que proyecta los datos en un espacio de menor dimensión preservando alguna propiedad de interés (densidad, proyección lineal, etc.). A partir de esta proyección, se construye una cubierta $\{U_i\}_{i \in I}$ sobre la imagen $f(X)$, usualmente mediante solapamiento de intervalos. Para cada U_i, se considera la fibra inversa $f^{-1}(U_i)$ y se aplica un algoritmo de clustering (e.g., DBSCAN o k-means) para obtener componentes conexas locales.

El complejo de Mapper se construye como un grafo cuyos nodos representan los clústeres obtenidos en cada fibra, y se conectan mediante aristas si los correspondientes subconjuntos de datos comparten puntos. Este grafo sirve como una aproximación a la estructura del espacio original, preservando información sobre componentes conexas, ciclos y agrupaciones jerárquicas.

Mapper puede considerarse una herramienta intermedia entre la reducción de dimensionalidad y el estudio de invariantes topológicos, como la homología. En particular, permite capturar características relevantes como la conectividad o la presencia de bucles, que son centrales en el cálculo de grupos de homología $H_n(X)$ del espacio de datos. Esta representación se ha utilizado con éxito en diversos contextos donde la estructura geométrica latente es crucial, incluyendo genética, imágenes médicas, y semántica de textos.

En esta sección se emplea el algoritmo *Mapper* para construir una representación topológica simplificada del conjunto de datos, con el objetivo de identificar la estructura global y las relaciones entre subgrupos semánticos a través de la construcción de un grafo interpretativo de la forma de los datos.

En nuestro caso, el filtro utilizado es una proyección bidimensional mediante *UMAP*, seguida de un escalado estándar para garantizar una cobertura equitativa del espacio

proyectado. La cobertura del espacio filtrado se realiza mediante una *CubicalCover*, con 10 intervalos y un solapamiento del 20 %, lo cual permite capturar la continuidad de las regiones densas manteniendo interconexiones entre regiones semánticas próximas. Posteriormente, se aplica *DBSCAN* como algoritmo de clustering dentro de cada intervalo, con parámetros ($\varepsilon = 0.5$, min_samples $= 3$), dada su capacidad para detectar agrupaciones de forma arbitraria y manejar ruido.

Además, se explorarán diferentes combinaciones de hiperparámetros del cubrimiento y del algoritmo de clustering. En particular, se considerará también el uso de *KMeans* con $k = 16$ clústeres, en consonancia con los resultados obtenidos previamente en la identificación de niveles semánticos óptimos. Esta variación permite contrastar diferentes perspectivas topológicas sobre la organización latente de los documentos.

NOTA: La escala de colores representa el tamaño de cada nodo del grafo, que corresponde con el número de puntos asociados a ese cluster. Recordemos que cada uno de los nodos del grafo representa a un cluster de puntos del espacio filtrado y superpuesto. Normalmente, debido a la superposición de intervalos, un mismo punto está asociado a varios nodos. De hecho, esta última regla es la que define que dos nodos del grafo simplicial estén conectados por una arista.

5.4.1. Configuración con DBSCAN

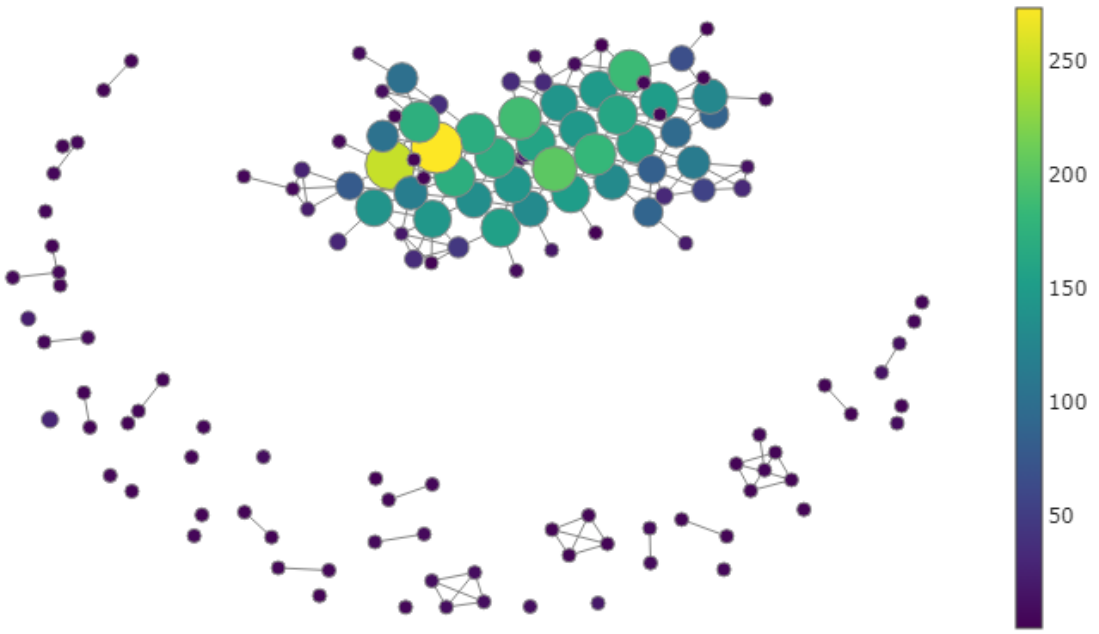


Figura 5.16: Resultado del análisis topológico mediante *Mapper* aplicado sobre la proyección UMAP de los datos originales. El grafo representa la estructura global del corpus: un núcleo denso central indica regiones semánticamente coherentes, mientras que las ramas y nodos periféricos reflejan transiciones, trayectorias latentes y posibles outliers.

En la Figura 5.16 se presenta la estructura topológica obtenida mediante el algoritmo *Mapper*, aplicado sobre la proyección UMAP del espacio semántico previamente descrito. Este grafo sintetiza la forma global de los datos agrupando subconjuntos locales mediante *DBSCAN* y enlazándolos según su solapamiento en el espacio filtrado.

La estructura resultante revela un núcleo central densamente conectado, que actúa como una región de alta coherencia semántica donde numerosos documentos comparten patrones similares. Además, se observa una rama curvada que se extiende hacia la parte inferior, con menor densidad de conexiones. Este patrón sugiere una transición continua entre subgrupos, posiblemente reflejando una trayectoria latente en el espacio semántico ,como puede ocurrir en la evolución de temas, estilos o categorías híbridas.

La periferia del grafo incluye múltiples nodos con escasa conectividad, que representan regiones poco densas del espacio de datos. Estos nodos periféricos pueden corresponder a documentos atípicos o especializados.

La estructura resultante guarda una clara correspondencia con las distribuciones observadas previamente en la proyección UMAP (5.5). En aquellas representaciones ya se apreciaba una concentración central de alta densidad, rodeada por regiones intermedias algo más dispersas pero aún conectadas, y finalmente un conjunto de puntos periféricos aislados. De forma análoga, el grafo Mapper presenta un núcleo compacto de nodos fuertemente conectados, rodeado por ramas más laxas que capturan transiciones locales, y una periferia compuesta por pequeños conglomerados desconectados, reflejo directo de las regiones menos densas del espacio proyectado.

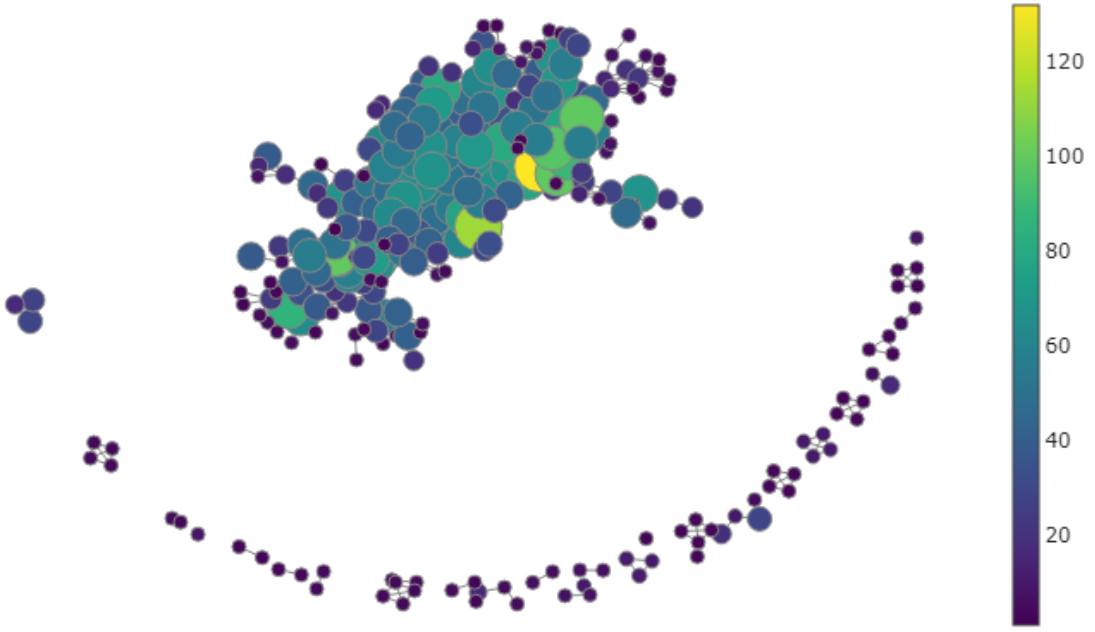


Figura 5.17: Representación del Mapper con una configuración más granular: 20 intervalos y 30 % de solapamiento. Se observa una mayor densidad en el núcleo central y una mejor resolución de la trayectoria curva inferior, permitiendo identificar microgrupos y transiciones sutiles en la estructura de los datos.

En la figura 5.17 observamos el resultado con otra configuración. Con un mayor número de intervalos y un mayor solapamiento en la cubierta (20 intervalos y un 30 % de solapamiento), el Mapper revela una estructura topológica más rica y detallada. Se aprecia una mayor densidad de nodos en el núcleo principal, así como una resolución más precisa de la trayectoria curva que se extiende en la parte inferior. Este aumento en la granularidad permite identificar microgrupos y ramificaciones sutiles que en la configuración anterior quedaban ocultas, lo que sugiere la presencia de estructuras internas latentes en los datos.

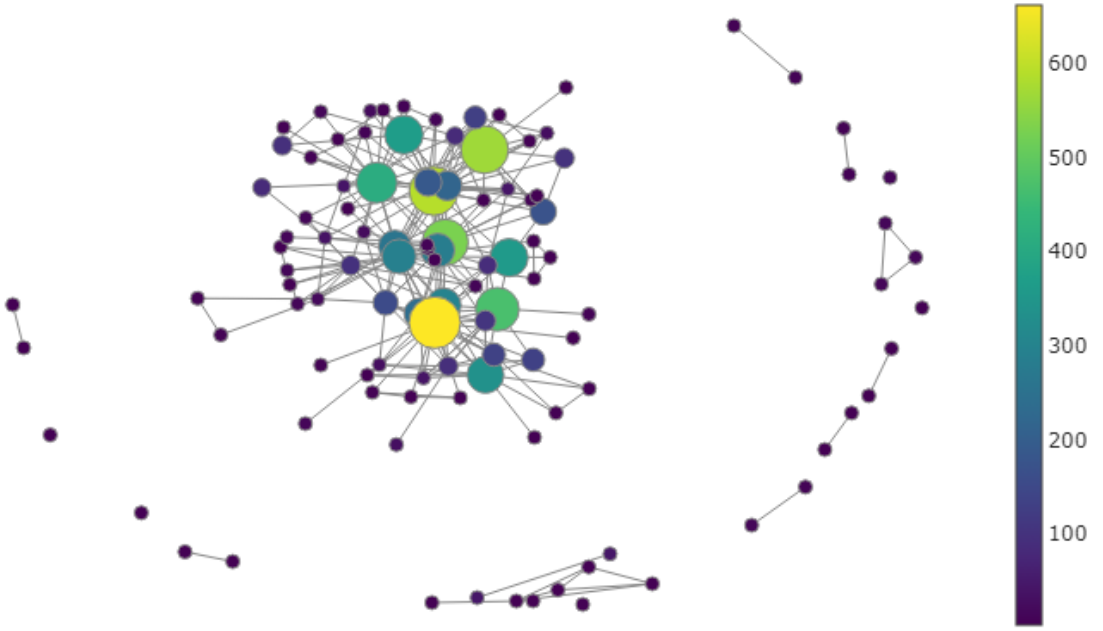


Figura 5.18: Grafo Mapper generado con una configuración de baja resolución (5 intervalos y $\varepsilon = 0.75$). La estructura se simplifica, con un núcleo central más compacto que refleja los patrones dominantes del conjunto. A pesar de la reducción, persiste una rama inferior curvada, indicativa de una trayectoria semántica latente robusta.

Con una configuración de baja resolución (5 intervalos) y un valor más permisivo de `eps` en el clustering ($\varepsilon = 0.75$) obtenemos como resultado la Figura 5.18, el grafo Mapper ofrece una representación simplificada de la estructura global del conjunto de datos (5.18). En esta configuración, el núcleo central agrupa una mayor cantidad de puntos, lo que atenúa la microestructura observada en configuraciones anteriores y pone de relieve los patrones dominantes del corpus. A pesar de esta reducción, se mantiene visible una rama curvada en la parte inferior del grafo, lo que sugiere que dicha trayectoria representa una estructura latente robusta..

5.4.2. Configuración con K-Means

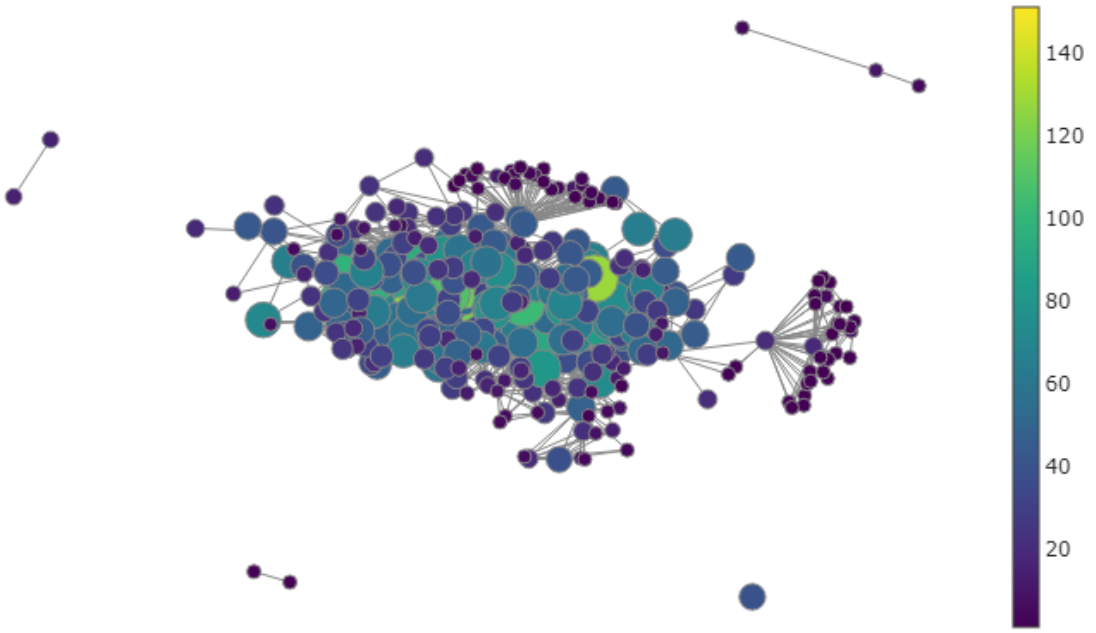


Figura 5.19: Grafo Mapper generado con 6 intervalos y 50 % de solapamiento, utilizando *KMeans* con $k = 16$. La estructura topológica alargada y conectada sugiere una progresión semántica continua a lo largo del espacio de temas, con focos especializados en los extremos y nodos periféricos que podrían representar documentos atípicos.

En esta nueva configuración, véase la Figura 5.19, se ha aplicado el algoritmo Mapper con una cubierta de baja resolución (6 intervalos) pero alto solapamiento (50 %), combinada con *KMeans* como método de agrupamiento local, utilizando $k = 16$, en concordancia con las 16 temáticas semánticas previamente detectadas en el corpus. El grafo resultante presenta una estructura alargada y altamente conectada, que sugiere una evolución progresiva y continua de los datos a lo largo de uno o varios ejes semánticos latentes.

A diferencia de representaciones más fragmentadas, esta forma alargada se asemeja a un gradiente estructural, lo que refuerza la hipótesis de que las categorías semánticas identificadas no son totalmente disjuntas, sino que pueden organizarse a lo largo de trayectorias intermedias o temáticas puente. Además, la aparición de pequeños grupos laterales, parcialmente conectados al cuerpo principal, sugiere la existencia de focos especializados dentro del dominio, mientras que algunos nodos aislados podrían estar relacionados con documentos atípicos o de contenido excepcionalmente singular.

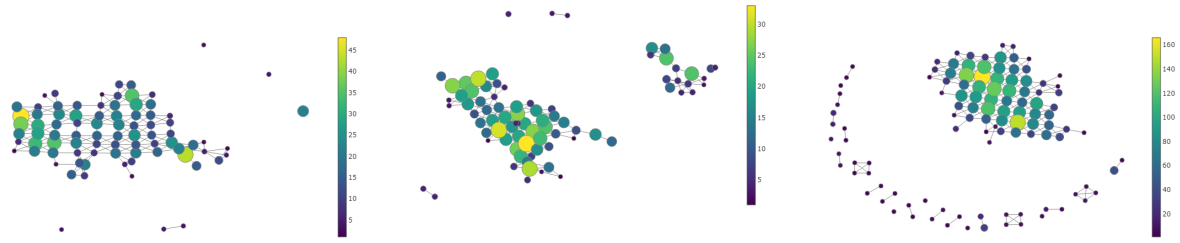
5.4.3. Mapper por subconjuntos temáticos

Con el objetivo de obtener representaciones topológicas más precisas y significativas, se ha decidido restringir el análisis a las entidades con mayor número de documentos en el corpus. Dado que el algoritmo Mapper depende críticamente de la densidad local y del solapamiento de datos para generar una estructura interpretable, aplicar este método sobre subconjuntos poco representados puede producir grafos escasamente conectados o sin valor analítico.

Por ello, en esta sección se seleccionan las tres instituciones con mayor número de muestras, así como los tres ODS (Objetivos de Desarrollo Sostenible) más frecuentes. El análisis topológico se realiza de forma independiente para cada uno de estos subconjuntos, permitiendo observar si las estructuras latentes detectadas en el conjunto completo se mantienen o evolucionan cuando se analizan dominios semánticos más acotados.

Este enfoque también permite explorar si ciertas instituciones o temáticas generan por sí solas estructuras internas ricas ,lo que podría reflejar especialización, coherencia temática o evolución semántica interna, o si, por el contrario, su representación topológica resulta más homogénea o compacta.

En la Figura 5.20 mostramos el resultado de aplicar el algoritmo mapper a los documentos pertenecientes a las instituciones de *E.T.S.I. de Sistemas Informáticos (UPM)*, *E.T.S.I. y Sistemas de Telecomunicación (UPM)* y *E.T.S. de Ingenieros Informáticos (UPM)*



(a) Mapper de la E.T.S.I. de Sistemas Informáticos (UPM)

(b) Mapper de la E.T.S.I. y Sistemas de Telecomunicación (UPM)

(c) Mapper de la E.T.S. de Ingenieros Informáticos (UPM)

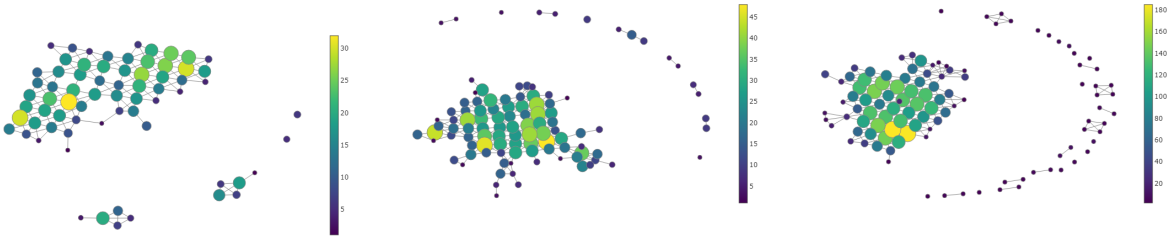
Figura 5.20: Comparación de la estructura topológica mediante Mapper en las tres instituciones con mayor número de muestras. La E.T.S. de Ingenieros Informáticos (UPM) presenta una estructura más rica y detallada, coherente con su mayor volumen de datos y complejidad.

La aplicación del algoritmo Mapper sobre subconjuntos definidos por institución revela diferencias estructurales que complementan el análisis semántico general. En el caso de la E.T.S. de Ingenieros Informáticos (UPM), la mayor cantidad de documentos, diversidad y temáticas multidisciplinares permite detectar una topología rica y organizada, con una estructura central densa y transiciones periféricas que sugieren una diversidad de focos temáticos interconectados.

Por otro lado, la E.T.S.I. de Sistemas Informáticos (UPM) presenta una forma alargada y lineal, con menos bifurcaciones, lo que podría asociarse a una producción más homogénea en términos de temática o estilo.

Por último, en la E.T.S.I. y Sistemas de Telecomunicación (UPM), la menor cantidad de datos se traduce en una red más fragmentada, con grupos poco conectados entre sí, lo que podría reflejar tanto una mayor dispersión semántica como limitaciones en la densidad de datos para construir una topología robusta. Aún así, de nuevo se aprecia a ver una nube densa altamente conexa, manteniendo completamente la estructura hasta ahora estudiada.

Finalmente, estudiaremos la misma comparación para los 3 ODS con más apariciones, en este caso son *03. Salud y bienestar*, *04. Educación de calidad* y *09. Industria, innovación e infraestructura*, véase en la Figura 5.21.



(a) ODS 03: Salud y Bienestar

(b) ODS 04: Educación de Calidad

(c) ODS 09: Industria, Innovación e Infraestructura

Figura 5.21: Comparación de las estructuras topológicas resultantes del algoritmo Mapper aplicadas a los documentos de los tres ODS más representados en el conjunto de datos.

La estructura topológica de los subconjuntos correspondientes a los ODS 3, 4 y 9 muestra diferencias relevantes. **ODS 3 (Salud y Bienestar)** presenta una forma alargada y continua, muy similar a las ya observadas en clases homogéneas, lo que refuerza la idea de baja diversidad semántica. **ODS 4 (Educación de Calidad)** mantiene el patrón general de un núcleo compacto con una curva periférica, aunque con mayor resolución interna, indicando una clase más diversa discursivamente. Finalmente, **ODS 9 (Industria, Innovación e Infraestructura)** reproduce de forma más marcada el esquema de núcleo y ramificación disgregada, reflejando su alta heterogeneidad. De hecho, este último mapper deja en entrever perfectamente el núcleo denso con una periferia prácticamente conexa, lo que implica la existencia de una región semántica común fuertemente compartida, en torno a conceptos industriales y tecnológicos, acompañada de múltiples subespacios temáticos especializados que se ramifican progresivamente y mantienen cierta relación estructural con el centro.

5.4.4. Conclusiones

A partir de los distintos experimentos realizados con el algoritmo Mapper, puede extraerse una descripción coherente de la forma subyacente del espacio semántico construido a partir del corpus. En general, los datos presentan una estructura con un núcleo central denso y bien conectado, que agrupa los documentos más representativos o comunes en términos semánticos. Este cuerpo principal se mantiene consistente a lo largo de múltiples configuraciones, lo que sugiere la existencia de una base temática compartida entre buena parte del conjunto.

Más allá de este núcleo, los Mapper generados revelan la presencia de trayectorias

curvadas o alargadas que se extienden desde la región central. Estas trayectorias muestran una evolución suave en el espacio semántico, apuntando a la existencia de dimensiones latentes que conectan distintas temáticas de forma continua. Esta interpretación encaja con el análisis previo donde se identificaron 16 grandes temas, que lejos de ser totalmente disjuntos, parecen formar un continuo estructurado con transiciones progresivas entre ellos.

Además, en casi todas las configuraciones han aparecido nodos periféricos o agrupaciones laterales menos conectadas, lo que refuerza la presencia de focos temáticos más especializados, así como documentos que podrían funcionar como outliers o casos fronterizos entre categorías. En conjunto, la forma del espacio semántico revelada por Mapper combina cohesión central, gradientes temáticos y especialización periférica, componiendo una topología rica y expresiva que justifica el uso de herramientas de análisis topológico para su exploración.

Por otro lado, el análisis centrado en subconjuntos específicos ,como determinadas instituciones o etiquetas ODS, ha mostrado que los grupos más numerosos tienden a conservar esta morfología global: una región principal compacta y conectada, rodeada de una periferia articulada. Esta forma indica una región semántica común fuertemente compartida, con alta densidad conceptual ligada a términos más generales de la Institución o la temática, desde la cual emergen especializaciones que se ramifican sin desconectarse entre ellas, lo cual aunque las diferencia también muestra características comunes. En cambio, al reducirse el número de muestras, la estructura se fragmenta: tanto el cuerpo central como las ramas pierden cohesión, evidenciando una mayor sensibilidad a la dispersión temática y menor capacidad de generar continuidad semántica.

Capítulo 6

Análisis Topológico de los Datos

En este capítulo analizaremos la topología de nuestro dataset, base fundamental para la construcción del clasificador. En el capítulo anterior se ofreció una primera aproximación a la forma del corpus y de algunos subconjuntos relevantes a través del algoritmo mapper; aquí profundizaremos en ese análisis.

Comenzaremos con una introducción a los conceptos topológicos necesarios, como las homologías H_0 y ciertas invariantes asociadas. A continuación, abordaremos estructuras fundamentales como los complejos de Vietoris-Rips y los complejos alpha. Finalmente, aplicaremos estas herramientas para extraer características topológicas segmentadas por variables como la institución de procedencia o los Objetivos de Desarrollo Sostenible (ODS).

Este análisis resulta clave para validar la viabilidad del clasificador. En particular, nos aseguraremos de que: (1) las clases estén suficientemente separadas en el espacio topológico, evitando solapamientos que dificulten la clasificación; y (2) los subconjuntos pertenecientes a una misma clase mantengan una coherencia interna que no induzca a errores.

6.1. Conceptos Topológicos Básicos

La homología es una herramienta fundamental en topología algebraica que permite cuantificar y clasificar las características topológicas de un espacio, como componentes conexas, agujeros y cavidades. Formalmente, la k-ésima homología de un espacio topológico X se denota como $H_k(X)$ y describe las k-dimensionales ägujeros"presentes en X [72].

6.1.1. Grupos de Homología

Los grupos de homología se construyen a partir de complejos de cadenas, que son secuencias de grupos abelianos conectados por operadores de frontera. Para un complejo simplicial K, se define una secuencia de grupos de cadenas $C_k(K)$ y operadores de frontera $\partial_k : C_k(K) \to C_{k-1}(K)$ que satisfacen $\partial_k \circ \partial_{k+1} = 0$. El grupo de homología $H_k(K)$ se define como el espacio vectorial cociente [43]:

$$H_k(K) = \frac{\ker(\partial_k)}{\text{im}(\partial_{k+1})}$$

Donde $\ker(\partial_k)$ son los k-ciclos (cadenas sin frontera) y $\text{im}(\partial_{k+1})$ son los k-bordes (fronteras de $(k+1)$-cadenas).

La interpretación de las homologías depende de la dimensión de la misma, pero en esencia, las principales interpretaciones son:

- H_0: Cuenta el número de componentes conexas del espacio
- H_1: Detecta ciclos o .ªgujerosünidimensionales, como los presentes en un toro o una circunferencia.
- H_2: Identifica cavidades tridimensionales, como las del interior de una esfera hueca.
- ...

Estas estructuras se repiten análogamente para dimensiones superiores. Sus interpretaciones permiten una comprensión intuitiva de las características topológicas del espacio analizado.

6.1.2. Cálculo de la Homología mediante Complejos de Vietoris-Rips

Para calcular la homología de un conjunto de datos discretos (nube de puntos), se utiliza el complejo de Vietoris-Rips, que es especialmente adecuado para espacios métricos finitos. Dado un conjunto de puntos P en un espacio métrico y un parámetro de escala $r > 0$, el complejo de Vietoris-Rips $VR(P, r)$ se define como el complejo simplicial donde un k-simplejo está presente si y solo si la distancia entre cualquier par de sus vértices es menor o igual a $2r$ [41].

Este enfoque permite construir una filtración de complejos:

$$VR(P, r_0) \subseteq VR(P, r_1) \subseteq \cdots \subseteq VR(P, r_n)$$

A través de esta filtración, se puede analizar cómo evolucionan las características topológicas a diferentes escalas, lo que conduce al concepto de homología persistente.

La homología persistente estudia la aparición y desaparición de características topológicas a lo largo de una filtración. Se representa mediante diagramas de persistencia o códigos de barras, donde cada barra indica la "vida"de una característica topológica desde su aparición hasta su desaparición en la filtración [73].

Este enfoque es robusto frente al ruido y permite identificar las características topológicas más significativas del conjunto de datos.

6.2. Extracción de Características Topológicas

En esta sección aplicaremos los conceptos topológicos introducidos previamente para extraer información estructural de nuestro dataset. Utilizaremos complejos simpliciales, en particular el complejo de Vietoris-Rips, para calcular la homología persistente

y obtener descriptores topológicos como las longitudes de persistencia y los diagramas de persistencia. Estos se usarán posteriormente como características discriminativas en nuestro modelo de clasificación.

6.2.1. Introducción a Vietoris-Rips

Como se ha mencionado anteriormente, el complejo de Vietoris-Rips permite observar la evolución de los complejos simpliciales construidos a partir de nuestro conjunto de datos, revelando las homologías que lo caracterizan.

Para su implementación se ha utilizado la librería *giotto-tda* en Python [54]. El resultado de este análisis es un conjunto de tripletas, donde el primer valor representa el parámetro ϵ en el que la homología aparece, el segundo indica el valor de ϵ en el que desaparece, y el tercero corresponde a la dimensión de la homología detectada.

Cabe destacar que el complejo de Vietoris-Rips se fundamenta en la noción de distancia, por lo que la elección de la métrica puede influir significativamente en los resultados obtenidos. En nuestro caso, se han probado tres métricas distintas:

- **Distancia euclídea**: adecuada para espacios de alta dimensionalidad cuyas componentes son ortogonales (como ocurre tras aplicar LSA). En este caso, las bolas crecen de forma uniforme en todas las direcciones, manteniendo su forma esférica.
- **Distancia Manhattan**: alternativa a la euclídea, en la que las bolas topológicas se convierten en hipercubos. Esto puede alterar la forma del complejo resultante y resaltar diferentes relaciones entre puntos.
- **Distancia de Chebyshev**: mide la máxima diferencia entre coordenadas correspondientes. En este caso, las bolas se transforman en cubos bajo la norma infinita (*max-norm*), dando lugar a agrupamientos más angulares.

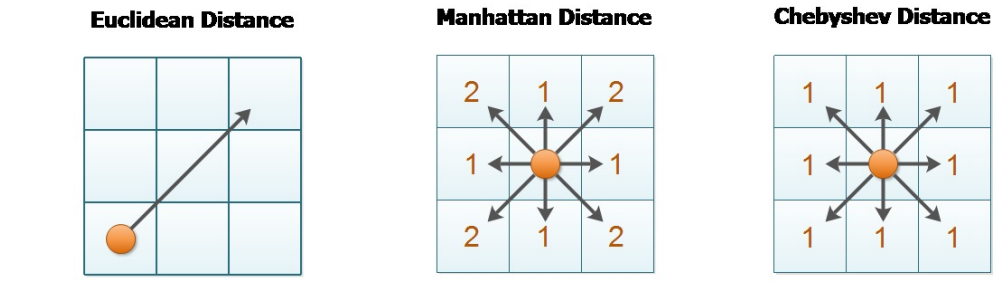
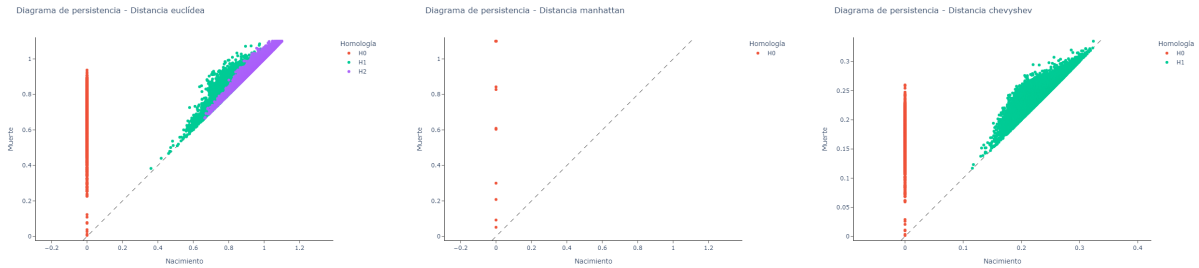


Figura 6.1: Comparación visual de las métricas de distancia Euclídea, Manhattan y Chebyshev en un entorno bidimensional. Imagen obtenida de [74].

Hemos calculado los diagramas de Vietoris Rips de todo nuestro conjunto usando cada una de estas distancias. A continuación se muestran los resultados para las 3 primeras homologías, excepto para la distancia Chebyshev.



(a) Distancia euclídea (b) Distancia Manhattan (c) Distancia Chebyshev

Figura 6.2: Comparativa de diagramas de persistencia generados con distintas métricas de distancia. Cada una resalta estructuras topológicas distintas en función de la forma en que se define la proximidad entre puntos.

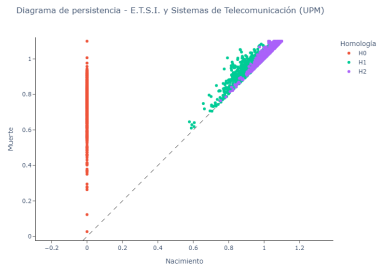
Me gustaría añadir el siguiente comentario acerca de los gráficos:

- **Distancia Euclídea (gráfico 6.2a)**: Se visualizan clases H_0, H_1 y H_2. Las H_0 siguen un patrón similar al observado en el resto de métricas. Las H_1 presentan algunos ciclos alejados de la diagonal, lo que podría revelar la presencia de ciclos estables y relevantes. Además, se detectan clases H_2 (cavidades) con cierta persistencia, lo que sugiere la existencia de regiones vacías o *estructuras volumétricas* dentro del espacio de representación.
- **Distancia Manhattan (gráfico 6.2b)**: Solo se detectan clases H_0 (componentes conexas). Se observa una gran concentración de nacimientos en $\epsilon = 0$, con muertes dispersas que alcanzan valores superiores a 1, lo que sugiere la existencia de grupos bien separados o posibles *outliers* que tardan en conectarse al resto del conjunto. El conjunto presenta una baja conectividad inicial y unas pocas componentes dominantes que persisten largo tiempo antes de fusionarse.
- **Distancia Chebyshev (gráfico 6.2c)**: Aparecen clases H_0 y H_1. Las H_0 se agrupan y mueren rápidamente (alrededor de $\epsilon = 0.25$), lo que indica una conectividad densa local. Las clases H_1, sin embargo, se sitúan muy cerca de la diagonal (el más lejano a 0.05 unidades de distancia), lo que refleja ciclos de corta vida, interpretables como *ruido topológico*. No se observan ciclos persistentes significativos.

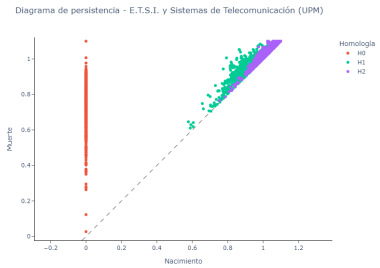
De entre las 3, la distancia euclídea parece la más fiable, proporcionandonos una gran cantidad de información relevante en un corto periodo de tiempo. Es por esto que, aunque el resto de distancias nos proporcionen una comparativa trasnversal, esta será la métrica estándar para el resto del análisis.

6.2.2. Características Topológicas por Institución

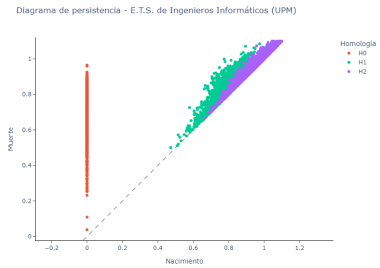
El siguiente paso es estudiar las características topológicas por Institución. Al igual que hicimos en el capítulo 5 trabajando con mappers, no tiene sentido estudiar la topología de clases sin apenas muestras, por lo que mostraremos únicamente las 6 clases con más muestras. También vamos a juntar las escuelas con la antigua denominación con la nueva denominación, para unificar un poco las etiquetas y eliminar aquellas con menor representación.



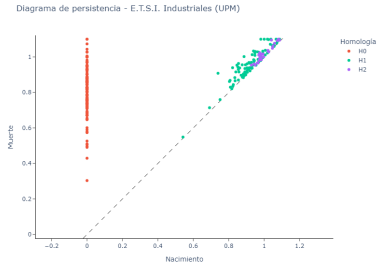
(a) Sistemas Informáticos (UPM)



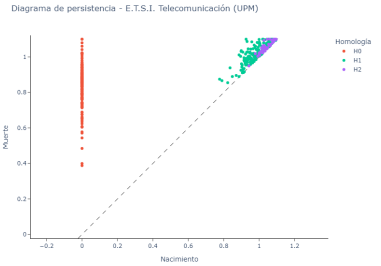
(b) Sistemas y Telecomunicación (UPM)



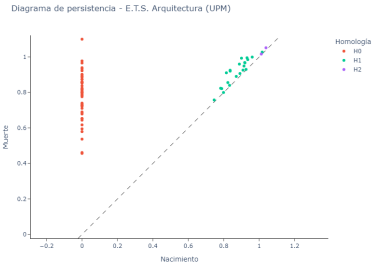
(c) Ingenieros Informáticos (UPM)



(d) Industriales (UPM)



(e) Telecomunicación (UPM)



(f) Arquitectura (UPM)

Figura 6.3: Diagramas de persistencia segmentados por institución. Cada gráfico refleja las estructuras topológicas extraídas del conjunto de documentos asociados a una escuela concreta de la UPM.

En la Figura 6.3, a primera vista puede observarse que las clases menos representadas no presentan una estructura topológica claramente definida, probablemente debido al reducido número de muestras. Esta baja densidad de datos dificulta la identificación de patrones consistentes y, por tanto, complica su detección por parte del clasificador. Por este motivo, el análisis más relevante debe centrarse en los tres gráficos superiores, correspondientes a las instituciones de *Sistemas Informáticos (UPM)*, *Sistemas y Telecomunicación (UPM)* e *Ingenieros Informáticos (UPM)*.

En estos tres casos se aprecia un patrón general común, con ciertas variaciones particulares.

En cuanto a las homologías H_0, se observan componentes que persisten hasta valores elevados de ϵ, lo que indica la existencia de grupos de documentos inicialmente desconectados que se fusionan más tarde. En la institución de *Ingenieros Informáticos (UPM)*, este proceso ocurre a un valor de ϵ inferior a 1, lo que sugiere una mayor cohesión interna del conjunto. En cambio, en *Sistemas Informáticos (UPM)* son dos los componentes que perduran, lo que refleja una mayor dispersión. Por su parte, *Sistemas y Telecomunicación (UPM)* presenta un único componente persistente.

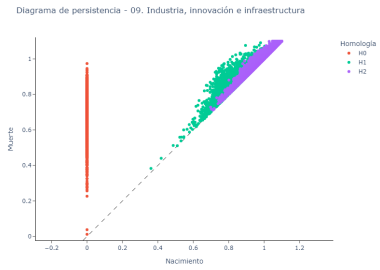
Respecto a las clases H_1, en los tres diagramas se identifican ciclos alejados de la diagonal, indicio de la presencia de estructuras cíclicas relevantes. En el caso de *Ingenieros Informáticos (UPM)*, estos ciclos aparecen en valores bajos de ϵ, lo que sugiere una estructura interna densa y radial. En las otras dos instituciones, los ciclos emergen más tarde; en particular, en *Sistemas y Telecomunicación (UPM)* aparece un pequeño grupo de ciclos que desaparecen rápidamente.

Finalmente, también se detectan homologías H_2 en las tres instituciones, lo que podría señalar la presencia de cavidades o regiones volumétricas dentro del espacio de representación. No obstante, dada la posible influencia del número de muestras en estas estructuras, no es posible extraer conclusiones sólidas al respecto.

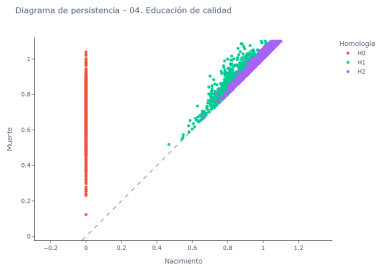
6.2.3. Características Topológicas por ODS

Del mismo modo que hemos analizado la estructura topológica del dataset segmentando por institución, en esta sección realizamos un análisis equivalente agrupando los documentos según los Objetivos de Desarrollo Sostenible (ODS) a los que están asociados. A diferencia del caso anterior, un mismo documento puede estar vinculado a varios ODS, lo que introduce una mayor superposición entre grupos.

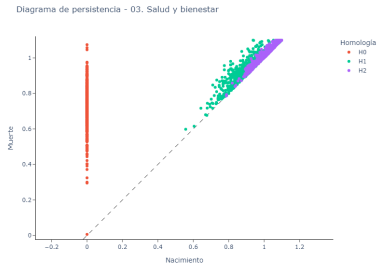
Para facilitar la interpretación de los resultados y garantizar una cantidad suficiente de muestras por grupo, se han seleccionado los seis ODS con mayor número de documentos. A partir de estos subconjuntos se construyen los diagramas de persistencia correspondientes, sobre los que se analizarán las homologías más relevantes.



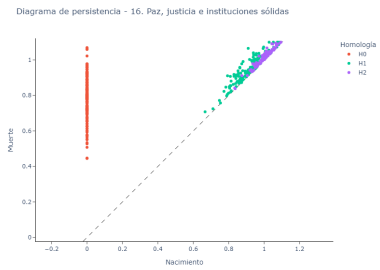
(a) ODS 9: Industria, innovación e infraestructura



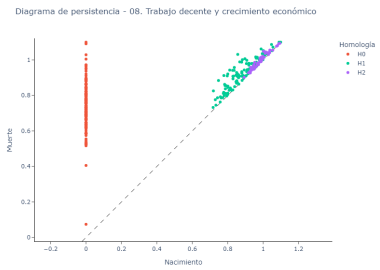
(b) ODS 4: Educación de calidad



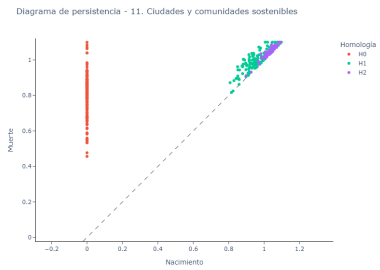
(c) ODS 3: Salud y bienestar



(d) ODS 16: Paz, justicia e instituciones sólidas



(e) ODS 8: Trabajo decente y crecimiento económico



(f) ODS 11: Ciudades y comunidades sostenibles

Figura 6.4: Diagramas de persistencia topológica segmentados por Objetivo de Desarrollo Sostenible (ODS). Se representan las homologías H_0, H_1 y H_2 obtenidas mediante complejos de Vietoris-Rips.

En la Figura 6.4, se analizan los diagramas de persistencia correspondientes a los tres ODS con mayor volumen de muestras: *Industria, innovación e infraestructura (ODS 9)*, *Salud y bienestar (ODS 3)* y *Educación de calidad (ODS 4)*. Esta segmentación permite explorar si existen estructuras topológicas características en función del ODS dominante.

En los tres casos se observa una gran cantidad de componentes conexas H_0 con nacimiento en torno a $\epsilon = 0$, lo cual es esperable en conjuntos densos. Sin embargo, la persistencia de estas componentes varía ligeramente: en el ODS 9 la mayoría se conectan rápidamente, mientras que en los ODS 3 y 4 se aprecian componentes que persisten hasta valores más elevados, lo que sugiere cierta dispersión interna.

Respecto a las homologías H_1, todos los diagramas presentan ciclos alejados de la diagonal, especialmente en el caso del ODS 9, donde se agrupan desde más temprano y con mayor rango de persistencia. Esto sugiere una estructura más rica en ciclos y, posiblemente, una mayor complejidad en la relación entre documentos. En contraste, ODS 3 y ODS 4 presentan patrones más dispersos, algunos siendo ciclos bastante persistentes.

Por último, las homologías H_2 aparecen en las tres clases con una distribución similar, formando una banda próxima al borde superior derecho. Aunque su persistencia no es elevada, su presencia consistente podría indicar ciertas cavidades topológicas compartidas entre documentos del mismo ODS.

Tras todo este análisis hemos podido dislumbrar ligeras diferencias entre los diagramas de persistencia. Sin embargo, aún no podemos decidir si estas diferencias son lo suficientemente significativas como para que el clasificador pueda aprender de ellas. Para ello, es necesario realizar un análisis más profundo comparando a través de distintas distancias estos diagramas.

6.3. Distancias entre Clases

Para evaluar la separabilidad de las clases, se han calculado las distancias entre los diagramas de persistencia obtenidos para cada clase. Estas distancias se han medido utilizando la métrica de wasserstein, que es adecuada para comparar distribuciones de probabilidad y, en este caso, diagramas de persistencia [75]. También se ha calculado la distancia entre los diagramas de persistencia a través de sus paisajes de persistencia y las representaciones de los *Shilhouette Persistence*, tal y como se probó en este artículo [52]. En el se usaban la distancia entre diagramas de Shilhouette para clasificar ondas cerebrales.

Las *Persistence Landscapes* se utilizan como representación funcional de los diagramas de persistencia. Estas funciones codifican la información de los intervalos de persistencia como una secuencia de capas continuas por partes, capturando tanto la altura como la ubicación de las clases topológicas de forma más estable ante perturbaciones.

Cada paisaje se construye seleccionando un número de capas (*layers*) y evaluando las funciones triangulares asociadas a cada punto del diagrama sobre un soporte discretizado. En nuestro caso se emplean cinco capas y 100 divisiones para una alta resolución, permitiendo detectar tanto ciclos persistentes como estructuras más finas.

Para comparar dos paisajes, aplicamos la distancia de Chebyshev entre sus representaciones vectorizadas:

$$d = \|\Lambda^{(1)} - \Lambda^{(2)}\|_\infty$$

donde $\Lambda^{(i)}$ representa el paisaje de persistencia del conjunto i. Esta métrica es sensible a diferencias locales máximas entre las formas topológicas de los datos.

En esta Figura 6.5 se muestra un ejemplo de un paisaje de persistencia para el ODS 09.

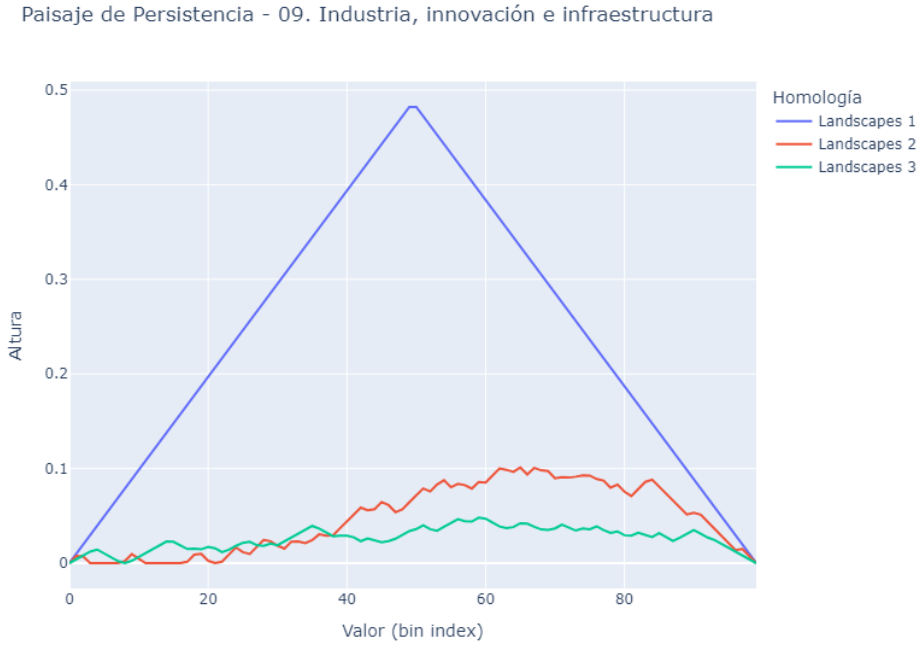


Figura 6.5: Curvas de Betti para la institución de Sistemas Informáticos (UPM). Se observa la evolución del número de clases de homología H_0, H_1 y H_2 en función del parámetro de filtración ϵ.

Además del paisaje de persistencia, empleamos las *Persistence Silhouettes* como representación funcional alternativa de los diagramas de persistencia. Estas suavizan la información topológica codificando los intervalos en una única función continua por partes, lo que aporta mayor robustez frente al ruido y a pequeñas variaciones locales.

Cada silhouette se construye a partir de funciones triangulares $\Lambda(b_i, d_i)(t)$, definidas entre los valores de nacimiento b_i y muerte d_i de cada clase topológica, ponderadas por su persistencia $w_i = d_i - b_i$. La expresión general de la silhouette es:

$$\phi_w(t) = \frac{\sum_{i=1}^{m} w_i \Lambda(b_i, d_i)(t)}{\sum_{i=1}^{m} w_i}$$

Para su implementación práctica, se discretiza el dominio en 1000 puntos, obteniendo un vector que resume de forma densa la estructura topológica del conjunto.

La comparación entre dos silhouettes se realiza mediante distancia euclídea entre sus versiones vectorizadas:

$$d = \|\phi^{(1)} - \phi^{(2)}\|_2$$

donde $\phi^{(i)}$ representa la silhouette asociada al conjunto i. Esta métrica permite detectar diferencias tanto locales como globales en la distribución topológica.

La Figura 6.6 muestra un ejemplo de las *Persistence Silhouettes* para el ODS 09.

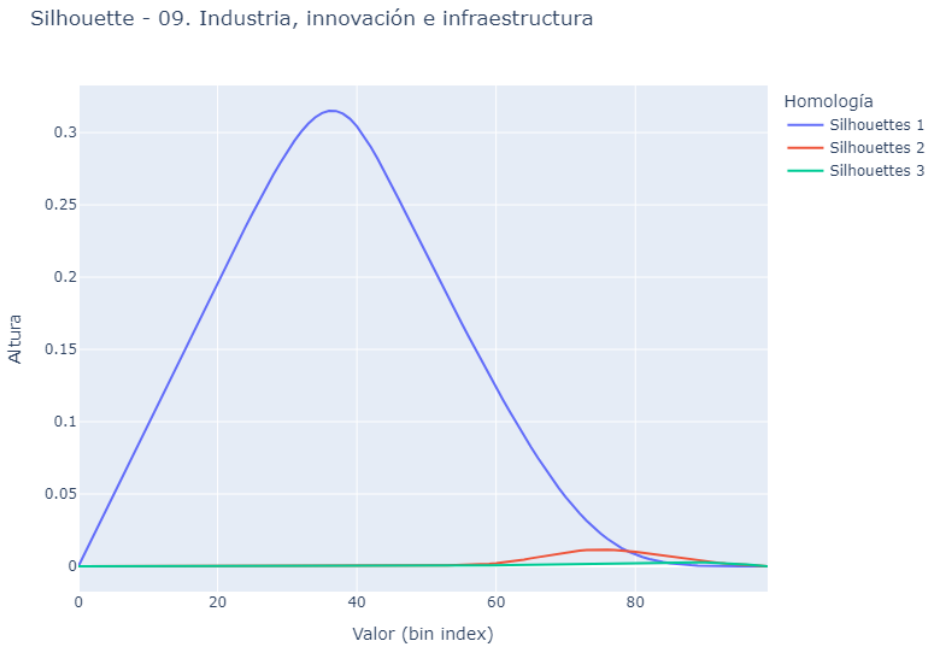


Figura 6.6: Silhouette de persistencia para el ODS 09 (Industria, innovación e infraestructura). La curva representa una media ponderada de las clases de homología H_0, evaluada sobre el parámetro de filtración ϵ.

El siguiente paso es comprobar la separabilidad de las clases a través de estas distancias.

6.3.1. Distancias entre Instituciones

A continuación se muestran las distancias entre las top 6 instituciones más numerosas, teniendo en cuenta que no podemos analizar la topología en todas ellas debido a la escasez de muestras. Además, para evitar sobrecarga computacional y dados los diagramas de Vietoris-Rips, se ha decidido calcular únicamente las homologías H_0 y H_1. Estas serán las utilizadas por el clasificador.

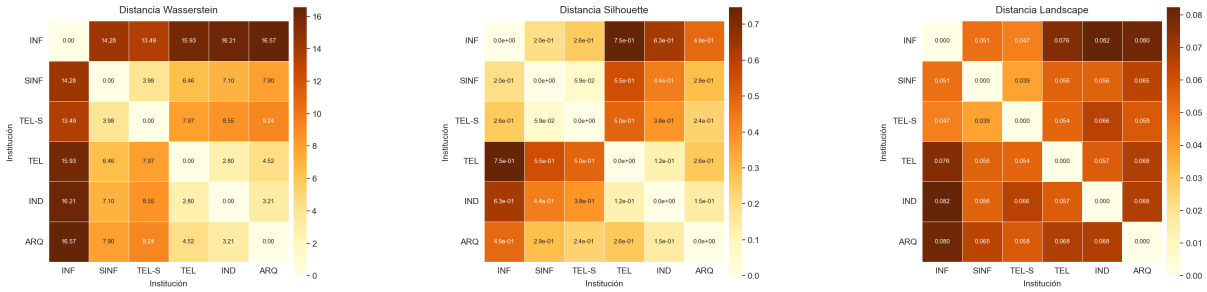


Figura 6.7: Comparación entre instituciones académicas según tres representaciones topológicas: distancia de Wasserstein (izquierda), distancia entre curvas de Betti (centro) y distancia entre paisajes de persistencia (derecha). Se observan agrupaciones coherentes en los tres casos, con la E.T.S. de Ingenieros Informáticos claramente diferenciada del resto.

La Figura 6.7 muestra las distancias entre las distintas instituciones académicas según tres representaciones topológicas: la **distancia de Wasserstein** entre diagramas de persistencia, la **distancia euclídea entre Silhouettes** y la **distancia de Chebyshev sobre paisajes de persistencia**. Estas medidas permiten evaluar el grado de disimilitud entre las estructuras topológicas globales de los documentos agrupados por institución.

En las tres gráficas se observan patrones distintos, lo que sugiere que cada métrica resalta diferentes aspectos. Destacaría que la métrica de Wassertein parece ser muy sensible al número de muestras, ya que la distancia entre la clase más representada (E.T.S. de Ingenieros Informáticos) y el resto es considerablemente mayor que la distancia entre las otras clases. Este suceso no aparece con las otras distancias.

Más en concreto, la distancia entre Silhouettes parece generar 2 grupos de instituciones. Es curioso porque estos grupos diferencian entre la institucion *Sistema de Telecomunicación* y *Telecomunicación*, algo que consideraría dificil al ser tan parecidas a simple vista.

Finalmente, este análisis debe completarse con el estudio de las **distancias intra-clase**, que permitirá discernir si las separaciones inter-clase reflejan una buena cohesión interna o simplemente una alta dispersión general. Una clase bien separada pero internamente dispersa podría seguir resultando difícil de clasificar. En este sentido, la combinación de múltiples métricas topológicas aporta una visión más robusta y complementaria de la estructura latente de los datos.

6.3.2. Distancias entre ODS

A continuación se presentan las distancias entre los seis Objetivos de Desarrollo Sostenible (ODS) con mayor representación en el conjunto de datos. Se emplean las mismas homologías H_0 y H_1, así como las métricas topológicas ya descritas: la **distancia de Wasserstein**, la **distancia euclídea entre Silhouettes** y la **distancia de Chebyshev sobre paisajes de persistencia**. Esta comparación busca evaluar el grado de disimilitud entre las estructuras topológicas asociadas a los documentos agrupados por ODS.

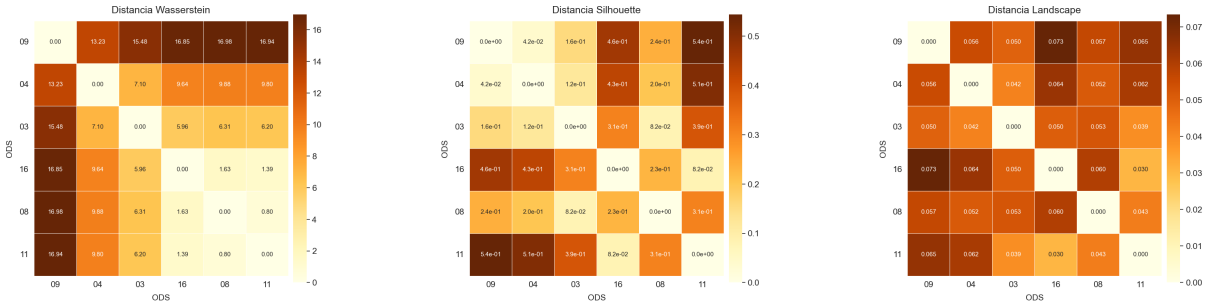


Figura 6.8: Comparación entre ODS según tres representaciones topológicas: distancia de Wasserstein (izquierda), distancia entre curvas de Betti (centro) y distancia entre paisajes de persistencia (derecha). Se observan diferencias estructurales entre los grupos de documentos según el ODS predominante.

La Figura 6.8 muestra las matrices de distancia entre los ODS más representados en el dataset.

De nuevo se aprecia el efecto antes comentado sobre la distancia de Wasserstein, que parece ser muy sensible al número de muestras. En este caso, la distancia entre el ODS 9 (Industria, innovación e infraestructura) y el resto es considerablemente mayor que la distancia entre los otros ODS.

La distancia de Silhouette parece revela patrones curiosos. Parece agrupar bien las 3 primeras clases, es decir, estas deberían de tener una estructura similar. Por otro lado, la distancia entre paisajes de persistencia genera separación entre todos los ODS, sin ningún patrón claro.

A priori, parece que estos experimentos revelan al landscape como una métrica capaz de detectar mayores diferencias entre clases. Sin embargo, hasta que no estudiemos la cohesión interna de cada clase, no podremos extraer conclusiones definitivas.

6.4. Distancias Intra-Clase

Para complementar el análisis anterior, es necesario estudiar la cohesión interna de cada clase. Este paso resulta clave para valorar si las distancias observadas entre instituciones u ODS responden a estructuras realmente diferenciadas o si, por el contrario, se deben a una alta dispersión general en los datos.

El procedimiento seguido consiste en segmentar el dataset por clase (institución u ODS), y a continuación, dividir cada grupo en múltiples subconjuntos (*sub-datasets*) de menor tamaño. Posteriormente, se calcula la distancia topológica entre todos los subconjuntos de una misma clase, generando así una matriz de distancias interna para cada métrica.

Este análisis nos permitirá observar qué clases presentan mayor estabilidad topológica interna y cuáles muestran una mayor variabilidad estructural, algo especialmente relevante a la hora de diseñar y evaluar un clasificador robusto.

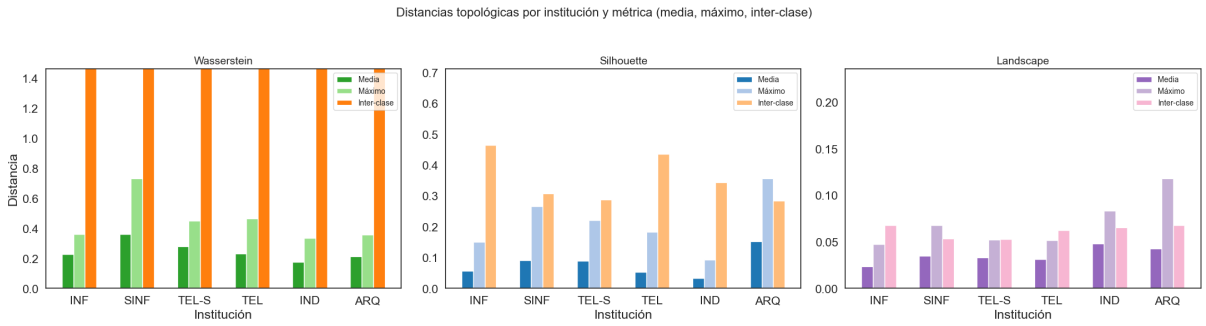


Figura 6.9: Rango de las métricas de distancia entre clases. Se observa que la métrica de Wasserstein presenta un rango más amplio, lo que sugiere una mayor variabilidad en las distancias inter-clase.

6.4.1. Distancias Intra-Clase por Institución

A continuación, en el cuadro 6.1 se presentan las distancias intra-clase para las instituciones académicas más representadas junto con la distancia mínima frente a otras clases De esta forma podremos comparar mejor los resultados.

Cuadro 6.1: Resumen de distancias intra-clase (media y máxima) y mínima inter-clase para cada métrica topológica.

Clase	**Wasserstein**			**Silhouette**			**Landscape**			
	Mean	Max	Inter	Mean	Max	Inter	Mean	Max	Inter	Nº Muestras
INF	0.24	0.51	13.49	0.05	0.15	0.46	0.03	0.07	0.04	2425
SINF	0.36	0.73	3.98	0.09	0.26	0.30	0.03	0.06	0.03	678
TEL-S	0.28	0.57	3.98	0.08	0.22	0.28	0.03	0.05	0.03	528
TEL	0.24	0.49	2.80	0.05	0.18	0.43	0.00	0.00	0.05	168
IND	0.18	0.34	2.80	0.03	0.09	0.34	0.04	0.08	0.05	117
ARQ	0.21	0.36	3.21	0.15	0.35	0.28	0.04	0.11	0.05	48

Los datos en formato tabular nos permiten comparar las diferencias, pero con este diagrama, que muestra la misma información en un gráfico de barras será más fácil interpretarlo.

En este gráfico 6.9 podemos apreciar algunos fenónemos comentados anteriormente.

Lo primero que vemos es que efectivamente la distancia de Wasserstein es más alta en las clases con mayor número de muestras. Además, la diferencia entre la distancia intra-clase y la inter-clase parecen encontrarse en escalas muy diferentes, algo que seguramente sea así si esta distancia fuera sensible a las muestras.

En cuanto a la distancian de Silhouettes, parece una buena opción. No se aprecia esta sensibilidad, pero si que deja ver diferencias entre las distancias intra-clase e inter-clase. En la mayoría de los casos la máxima distancia intra-clase no supera a la mínima entre las instituciones, lo que da cierta confianza.

Finalmente, la distancia entre paisajes de persistencia no parece tan fiable. Es cierto que cumple algunas características como la distancia de Silhouette, pero no deja ver

claramente diferencias que nos aseguren una estabilidad interna de las Instituciones.

6.4.2. Distancias Intra-Clase por ODS

A continuación, en el cuadro 6.2 se presentan las distancias intra-clase para los Objetivos de Desarrollo Sostenible (ODS) más representados. De nuevo adjuntamos la tabla con los resultados de las distintas distancias.

Cuadro 6.2: Resumen de distancias intra-clase (media y máxima) y mínima inter-clase para cada métrica topológica entre ODS.

ODS	**Wasserstein**			**Silhouette**			**Landscape**			
	Mean	Max	Inter	Mean	Max	Inter	Mean	Max	Inter	Nº Muestras
ODS 09	0.23	0.44	13.23	0.05	0.13	0.28	0.02	0.04	0.05	2425
ODS 04	0.33	0.77	7.10	0.11	0.35	0.26	0.03	0.05	0.04	678
ODS 03	0.31	0.68	5.96	0.08	0.28	0.21	0.03	0.05	0.03	528
ODS 16	0.21	0.48	1.38	0.05	0.14	0.30	0.02	0.05	0.03	168
ODS 08	0.24	0.37	0.80	0.05	0.13	0.21	0.03	0.05	0.04	117
ODS 11	0.20	0.41	0.80	0.06	0.16	0.36	0.03	0.05	0.03	48

Y también tenemos el gráfico asociado a la tabla anterior.

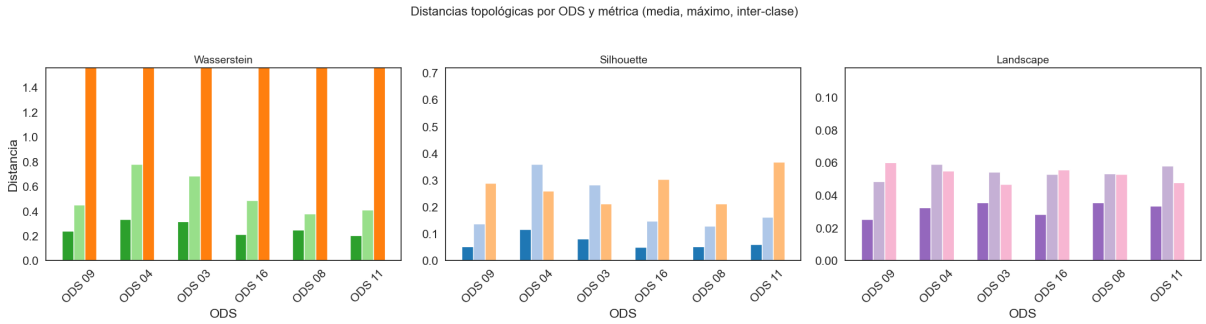


Figura 6.10: Rango de las métricas de distancia entre clases. Se observa que la métrica de Wasserstein presenta un rango más amplio, lo que sugiere una mayor variabilidad en las distancias inter-clase.

En la Figura 6.10 seguimos apreciando el sesgo de la distancia de Wasserstein, lo que no nos permite confiar en su eficiencia en la discriminación.

A diferencia que en el caso de las instituciones, la distancia de Silhouette esta vez no parece permitir diferenciar de forma tan clara. De hecho, aunque para algunos ODS la distancia máxima entre la misma clase es menor que la mínima con respecto a otros ODS, para los ODS 3 y 4 no es así.

No obstante, si la distancia de Silhouette es considerada como poco discriminante, la distancia entre paisajes funciona pero para este caso. En realidad, si usaramos esta métrica para intentar separar las clases, no seríamos capaces de diferenciarlas realmente, ya que los valores intra-clase e inter-clase son bastante parejos.

Capítulo 7

Desarrollo del Clasificador

7.1. Base del funcionamiento del clasificador

El clasificador desarrollado se trata de un modelo, que dado un conjunto de elementos pertenecientes a un grupo etiquetado y una nueva muestra, sepa generar una probabilidad de que ese elemento pertenezca a ese conjunto o no.

La forma que tendrá el clasificador para determinar si un elemento pertenece a una clase o no, será a través de la distancia existente entre un primer diagrama de persistencia, formado por los elementos de la clase sobre la que se quiere clasificar el documento, y un segundo diagrama de persistencia, formado por el conjunto anteriormente mencionado y el nuevo elemento. La idea es que si la distancia entre ambos diagramas de persistencia es pequeña, el nuevo elemento pertenece a la clase, y si la distancia es grande, el nuevo elemento no pertenece a la clase.

Para que esto sea posible necesitamos 2 requisitos, ya estudiados en el capítulo anterior. El primero de ellos es encontrar diferencias sustanciales entre las homologías de las distintas clases. Esto se traduce en que los diagramas de persistencia de las distintas clases deben de ser diferentes entre sí, ser distantes. El segundo requisito es que la topología de cada clase sea estable, es decir, que no varíe en exceso dado un subconjunto de esa misma clase. Esto es importante porque en caso de que la topología de la clase cambiase en función de los elementos que la componen, no podríamos establecer una frontera entre lo que pertenece a la clase y lo que no.

El enfoque adoptado en este trabajo comparte con Ferrà et al. [52] la idea de evaluar la compatibilidad de una muestra con una clase a través del impacto topológico que produce al incorporarse a ella. Sin embargo, el clasificador propuesto aquí difiere en varios aspectos clave. En primer lugar, se trabaja directamente con diagramas de persistencia sin transformarlos en representaciones intermedias como *silhouettes*, lo que permite conservar de forma explícita la información sobre el nacimiento y muerte de las clases topológicas. En segundo lugar, se utilizan múltiples métricas de comparación (Wasserstein, curvas de Betti, paisajes de persistencia), lo que aporta una perspectiva más rica y permite evaluar distintas nociones de proximidad estructural. Además, mientras que Ferrà et al. centran su análisis en señales fisiológicas (EEG), este clasificador se aplica sobre representaciones topológicas de documentos textuales, lo que introduce nuevas características en la variabilidad interna de los datos y en la forma en que se manifiestan sus estructuras persistentes.

El clasificador se va a evaluar sobre las etiquetas de Institución. He considerado que este análisis es el más adecuado e interesante, ya que estas instituciones pueden ser representativas de distintas áreas de conocimiento. Por contraparte, no se veía tan interesante realizar el experimento sobre el Departamento, ya que no se veían fronteras tan claras entre ellos. El análisis sobre ODS podría haber resultado fructífero, pero dado que un documento podía pertenecer a varios ODS, no se ha podido abordar un procedimiento óptimo para evaluar estas casuísticas.

7.2. Desarrollo del clasificador

Una vez presentada la idea base del clasificador, vamos a comentar como funciona de la mano de las librerías de scikit-learn y giotto-tda.

El primer paso es la carga de los datasets de entrenamiento. Por un lado se le entregarán las componentes del LSA de cada documento, y por otro lado, las etiquetas. Una vez tengas creada la base del clasificador, podrás llamar a un método para evaluar en tu conjunto de test. Una ventaja de este clasificador es que la fase de `fit` es muy rápida, ya que solo necesita calcular los diagramas de persistencia de cada una de las clases y almacenarlos. Por contraparte, la fase de `predict` es más lenta, ya que necesita calcular el diagrama de persistencia del nuevo elemento y compararlo con los diagramas de persistencia de cada una de las clases.

El segundo paso es la evaluación del conjunto de test. En este momento por cada elemento de test se realizarán los siguientes pasos:

- Se calculará para cada una de las clases el diagrama de persistencia del subconjunto de entrenamiento que pertenece a esa clase junto con ese punto de test. Para esto nos basaremos en la implementación de `giotto-tda` para calcular las dos primeras dimensiones de homologías persistentes. Para evitar problemas en tiempos de ejecución, limitaremos el tamaño máximo que puede tomar el parámetro *alpha* durante la construcción del complejo a $\alpha = 1.2$.
- Se evaluará a través de distintas métricas la distancia entre el diagrama de persistencia original de cada clase y el diagrama de persistencia del conjunto de entrenamiento más el punto de test. Estas distancias han sido estudiadas en el capítulo de análisis topológico 6.3 y son: la distancia de Wasserstein, la distancia eucídea entre diagramas de Silhouette H_0 y la distancia chebyshev entre los paisajes de persistencia H_0. Todas sus implementaciones están disponibles en la librería `giotto-tda`, y se pueden calcular de forma sencilla.
- Con estas distancias se generará un vector de probabilidades a través de una función softmax inversa, donde cada distancia se transformará en una probabilidad de pertenencia a la clase.
- Se calculará la media de las probabilidades obtenidas en el paso anterior. La etiqueta con mayor probabilidad será la clase predicha.

7.2.1. Limitaciones

A primera vista, el funcionamiento del clasificador parece sencillo y coherente. Sin embargo, existen ciertas limitaciones que se deben de resolver para que este funcione correctamente.

La primera de ellas está relacionada con el desbalanceo de clases. Si una clase tiene muy pocos elementos, el diagrama de persistencia no va a poder capturar ninguna estructura topológica interesante o propia de la clase, ya que estás muy probablemente aún no hayan aparecido. Esto es una limitación que se venía arrastrando desde el inicio del proyecto, pero que hasta ahora no ha generado mucho problema. Sin embargo, este es un punto crítico para el clasificador, que va a perjudicar su capacidad en gran medida.

En la Figura 4.1 podemos ver el número de documentos por institución. Como se puede observar, a partir de la cuarta institución el número de documentos por institución se reduce drásticamente a prácticamente 1 o 2 documetos por etiqueta. Lo primero que podemos hacer para solucionar esto es juntar las instituciones que realmente son las mismas, aunque sean antiguas denominaciones. Por ejemplo, los trabajos con etiqueta *Facultad de Informática (UPM) [antigua denominación]* se podrían juntar con los de *E.T.S. de Ingenieros Informáticos (UPM)*. De esta forma no solo conseguimos agrupar categorías, sino que además aumentamos el número de muestras en algunas. No obstante, esto no es suficiente, y tenemos que reducir el número de etiquetas a las 6 más representativas para poder tener clases con una mínima estructura. Estas clases que comento son las utilizadas para el análisis del capítulo 6.

Otra limitación de este modelo de clasificador es su sensibilidad a la agregación del punto al conjunto de datos. Por regla general, las características topológicas son muy poco sensibles al ruido, y no suelen ser afectadas por este tipo de pequeñas variaciones. Es por esto que agregar únicamente un punto es posible que no afecte a la topología del conjunto de datos, devolviendonos diagramas de persistencia poco distantes entre ellos. La solución que se ha ideado pasa por no integrar el subconjunto de la etiqueta con un solo punto de test, sino añadir una nube de puntos muy cercanos al punto de test. De esta forma, podemos garantizar que la topología de los datos será afectada en mayor grado, al no estar añadiendo únicamente un punto ruidoso. La nube de puntos se constuirá como un ruido gaussiano alrededor del punto original. Esta solución nos genera 2 nuevos hiperparámetros: el número de puntos con el que construir la nube y la distancia con respecto al punto de test.

Por último, aparece una nueva limitación relacionada con el desbalanceo de clases. Por regla general, la distancia para las etiquetas con pocas muestras va a ser mayor, ya que sus diagramas son menos estables y por tanto tienden a presentar mayores diferencias. Realmente no somos capaces de solucionar este problema completamente, pero si que se ha podido mitigar parte de su efecto. Para ello, en lugar de comparar la topología global de las clases, se ha optado por aplicar una técnica de estudio de la topología local. Esto implica que en lugar de calcular el diagramde persistencia sobre el conjunto de la clase con todos sus puntos, elegimos los k puntos más cercanos al punto de test, y ese será nuestro conjunto sobre el que calcular las distancias. De esta forma, podemos comparar la distancia entre etiquetas con un número de puntos mucho más similar, y por tanto ese sesgo hacia las etiquetas con menos muestras debería de reducirse. Esta técnica ya se ha introducido en otras investigaciones, como la realizada por Wang et al. [76] o por Chowdhury et al. [77], aplicando la topología local en tareas de clasificación. De nuevo, esto nos genera un nuevo hiperparámetro, que es el número de puntos k a considerar.

7.2.2. Hiperparámetros

El clasificador cuenta con una serie de hiperparámetros que se pueden ajustar para mejorar su rendimiento. Sin embargo, muchos de ellos estarán fijos durante los experimentos, ya que no tiene sentido modificarlos si conocemos por teoría su valor óptimo.

Estos hiperparámetros son los correspondientesa la creación del Silhouette, donde podemos identificar el número de intervalos a considerar, y la ponderación de la persistencia en la creación del diagrama. Dado que en nuestro caso las diferencias entre los diagrama de Silhouette van a ser muy pequeñas, nos interesa que las homologías poco persistentes ponderen practicamente igual que las más persistentes. Esto hace que el parámetro *Power* deba ser prácticamente 0. Por otro lado, el número de intervalos a considerar deberá de ser lo más alto posible, ya que nos permitirá construir el Silhouette con una mayor resolución.

Los hiperparámetros propios del clasificador sobre los que vamos a actuar son los siguientes:

- **Número de puntos de la nube:** Este parámetro indica el número de puntos que se van a generar alrededor del punto de test para construir la nube de puntos.
- **Ruido de la nube:** Este parámetro indica el ruido que se va a usar para generar la nube de puntos alrededor del punto de test.
- **Número de puntos del vecindario:** Este parámetro indica el número de puntos más cercanos a considerar para calcular el diagrama de persistencia.

7.3. Generación de probabilidades

Una vez tenemos las distancias entre el diagrama de persistencia del conjunto de entrenamiento y el diagrama de persistencia del conjunto de entrenamiento más el punto de test, podemos generar un vector de probabilidades. Este vector se genera a través de una función softmax inversa, que transforma las distancias, que actuan como *logits* en probabilidades. Además, para hacer nuestra función softmax inversa más sensible a pequeñas variaciones en las distancias, se ha añadido un nuevo parámetro, que es el *temperature* de la función softmax. Este parámetro se ajustará automáticamente en función de los *logits* obtenidos. Todo este proceso provoca que las distancias más pequeñas tengan una mayor probabilidad de pertenecer a la clase.

Durante los experimentos hemos comprobado que la distancia de Silhouette H_0 funciona mejor que la distancia de Wasserstein en ciertos contextos, y viceversa. En el caso del Silhouette, esta métrica no ha resultado muy sensible, lo que significa que necesita generar una nube de puntos densa para que la diferencia entre los diagramas sea notable. También esto provoca que no sea necesario escoger un vecindario muy pequeño.

El caso de la distancia de Wasserstein es completamente opuesto. Esta métrica resulta mucho más sensible, lo que requiere que no haga falta introducir una gran nube de puntos, sino que con un par se podría ver esta diferencia. Sin embargo, esta alta sensiblidad se ve traducida en la necesidad de escoger un vecindario más pequeño para estudiar la topología local de los datos, para asegurar que todas las etiquetas tengan un vecindario con una estructura prácticamente definida.

Es por esto que se ha optado por realizar un clasificador híbrido, que combine ambos enfoques. De esta forma, realmente se entrenarán 2 modelos de clasicación con distintos hiperparámetros. Tras esto, se calcularán las propabilidades de pertenecia a cada clase a través de las respectivas distancias, y se elegirá la etiqueta con mayor probabilidad media entre ambas métricas.

7.4. Exploración de Hiperparámetros

A continuación se presentan los resultados obtenidos en función de los distintos hiperparámetros explorados del clasificador. En la tabla 7.1 se muestran tan solo algunos de los resultados obtenidos que se han considerado más relevantes. En todos estos casos, el ruido gaussiano usado para generar la nube está fijado a $\gamma = 0.0001$.

Vec1	Pts1	Vec2	Pts2	Arquitectura	Informáticos	Industriales	Telecom	Sistemas Info	Sist. Telecom
70	10	25	10	0.0000	0.7654	0.0909	0.0588	0.4412	0.3774
100	10	50	10	0.0000	0.6749	0.0909	0.0588	0.4265	0.3396
1000	20	50	10	0.0000	0.7078	0.0000	0.0000	0.5000	0.3585
1000	50	25	10	0.0000	0.7901	0.0000	0.0000	0.3382	0.3585
100	50	100	10	0.0000	0.6996	0.0909	0.0000	0.4265	0.2261
200	30	200	30	0.0000	0.6667	0.0000	0.0000	0.3676	0.4151
200	30	100	10	0.0000	0.6543	0.0000	0.0000	0.4559	0.3585
1000	50	100	10	0.0000	0.8189	0.0000	0.0000	0.3529	0.1887
100	1	20	1	0.0000	0.6502	0.0000	0.0588	0.2794	0.3396
50	10	20	10	0.0000	0.6872	0.0000	0.0000	0.3235	0.3396

Cuadro 7.1: Accuracy por institución según los hiperparámetros de ambos modelos.

Las columnas **Vec1** y **Pts1** hacen referencia a los hiperparámetros del primer modelo, que calcula la distancia de Silhouette H_0. Las columnas **Vec2** y **Pts2** hacen referencia a los hiperparámetros del segundo modelo, que calcula la distancia de Wasserstein.

Vemos que el modelo consigue un accuracy de hasta un 81.89% en el caso de la etiqueta *Informáticos*, y un 41.51% en el caso de la etiqueta *Sistemas Telecom*. Sin embargo, para las clases con menos muestras, como *Industriales* o *Telecom*, el modelo no consigue una precisión aceptable. También es cierto que apenas hay muestras en el conjunto de test de estas clases, por lo que para estas el resultado no puede ser concluyente.

Lo que sí puede afirmarse es que parece existir una relación directa entre el número de muestras por clase y el *accuracy* obtenido. Al aplicar el test estadístico de **Spearman** para comprobar esta hipótesis, se obtiene un *p-valor* de 0.0048, lo que indica una correlación significativa. Esto sugiere que el modelo podría beneficiarse de un mayor número de muestras por clase para mejorar su rendimiento.

A continuación se muestra una gráfica para ilustrar los resultados de la tabla anterior 7.1.

En la Figura 7.1 podemos ver mejor este fenómeno, en el que la Institución de *Informáticos* es la que mejor resultado obtiene, seguida de *Sistemas Info* y *Telecom*. Por otro lado, las Instituciones con menos muestras, como *Industriales* o *Telecom*, obtienen resultados muy bajos. Tal vez en este caso el mejor modelo sea la primera combinación mostrada, donde al menos no todas las clases menos representadas desaparecen, sino que las llegamos a clasificar correctamente en algunos momen-

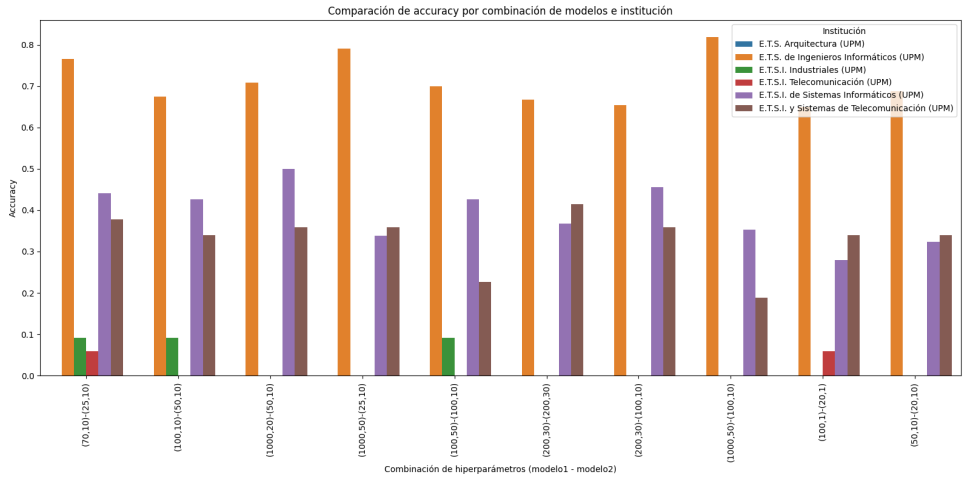


Figura 7.1: Accuracy del clasificador según los hiperparámetros y la Institución.

tos. Sin embargo, con este estudi0 y fijandonos en la Institución de *Informáticos*, nos damos cuenta del potencial que tienen estos tipos de clasificadores.

7.5. Comparación con otros clasificadores

Para evaluar el rendimiento del clasificador propuesto, se ha comparado con otros modelos de clasificación clásicos, como el *Random Forest*,*Support Vector Machine (SVM)* y *K-Nearest Neighbors*. Estos modelos se han entrenado utilizando las mismas características de entrada (componentes del LSA) y etiquetas de Institución.

En esta tabla 7.2 condensamos la comparación de los resultados obtenidos.

Institución	**Clas Topológico**	**SVM**	**Random Forest**	**KNN**
E.T.S. Arquitectura	0.000	1.000	0.000	0.400
E.T.S. de Ingenieros Informáticos	0.765	0.767	0.891	0.834
E.T.S. de Ingenieros Industriales	0.090	0.272	0.000	0.181
E.T.S. de Ingenieros de Telecomunicación	0.058	0.000	0.000	0.117
E.T.S.I. de Sistemas Informáticos	0.441	0.308	0.220	0.235
E.T.S.I. y Sistemas de Telecomunicación	0.377	0.566	0.320	0.452

Cuadro 7.2: Comparativa de accuracy entre distintos modelos por institución.

Los resultados muestran que el clasificador topológico no se encuentra lejos del accuracy de los modelos clásicos, e incluso en algunas etiquetas como *E.T.S.I. de Sistemas Informáticos (UPM)* supera los resultados de los mismos. Esta idea muestra el potencial de los modelos topológicos. Estos resultados, junto con lo que parecía una relación directa entre el número de muestras y el accuracy del modelo, sugieren que el clasificador topológico podría ser una alternativa interesante a los modelos clásicos en tareas de clasificación de documentos textuales.

Como conclusiones, podemos afirmar que el clasificador topológico es capaz de obtener un rendimiento similar a modelos clásicos como SVM o Random Forest. Tal vez con un dataset sintético mejor balanceado y equispaciado, el rendimiento del clasi-

ficador propuesto podría amumentar. Otro apartado que se puede explorar es el de la optimización de los hiperparámetros y distancias, que podría repercutir en una mejora en la tarea de clasificación.

Capítulo 8

Resultados y conclusiones

En este capítulo se presentarán los resultados tanto objetivos como subjetivos obtenidos a lo largo del desarrollo del TFG. En la sección de 8.1 buscaremos mostrar conclusiones obtenidas de forma objetiva, es decir, resultados obtenidos a partir de los experimentos realizados y de los datos obtenidos en el desarrollo del TFG. En la sección de 8.2 se presentarán las conclusiones personales del estudiante sobre el trabajo realizado, así como la opinión personal sobre los temas tratados en el mismo. Por último, en la sección de 8.3 se presentarán las posibles líneas de trabajo futuro que se podrían realizar, pensando en que la base del clasificador desarrollado se trata de una idea novedosa y poco explorada, que viendo resultados de otros experimentos, podría suponer un potente método de clasificación.

8.1. Resultados

En esta sección analizaremos los resultados de estre proyecto en base a los objetivos planteados en la introducción del TFG. 1.3.

El primer objetivo se trataba de analizar la estructura del AD-UPM y extraer una muestra significativa de documentos. Creo que este se ha completado con éxito, ya que no solo se ha extraido una muestra de más de **4000 documentos** que cumplen con los requisitos establecios para el análisis, sino que también se ha documentado la estructura del *Archivo Digital de la UPM* y se ha desarrollado un *Scrapper* funcional que permite obtener estos documentos de forma automática y rápida. Este *Scrapper* se ha desarrollado en Python, utilizando la librería `BeautifulSoup` [60] para el análisis de los documentos HTML y la extracción de los campos necesarios.

El segundo objetivo consistía en diseñar un pipeline de preprocesado para este tipo de documentos, que buscara aplicar técnicas de limpieza, preprocesamiento y selección de variables sobre los datos textuales. Este objetivo también se ha completado, principalmente desarrollado en el capítulo 5 4, donde no solo se ha presentado el pipeline, sino que también se ha buscado justificar cada una de las decisiones tomadas y procesos ejecutados. El resultado es un proceso muy completo, que parte de la base de un corpus de textos formados por los campos de título, resumen y conclusiones, y finaliza en un dataset completamente listo y estructurado para realizar cualquier tipo de análisis posterior. Para ello, el proceso definido comienza con la limpieza de los textos, eliminando stopwords, signos de puntuación, números y caracteres especia-

les. A continuación, se realiza un proceso de extracción de características a través de la técnica de *TF-IDF*. Dado la alta dimensionalidad del dataset resultante (que ya es estructurado) se aplica una serie de filtros, justificando siempre el proceso a través de métricas como *la entropía de Shannon* [63]. Tras este filtrado, el proceso continua con la aplicación de **LSA**, presentado en el artículo de Deerwester et al. [37], que permite reducir la dimensionalidad del dataset a un número específico de componentes. Este proceso acaba con la selección y estudio de las componentes del `Latent Semantic Analysis`, realizado al comienzo del capítulo 5.

El tercer objetivo buscaba explorar la estructura de los datos a través de distintos enfoques. Esto se ha desarrollado en los capítulos de análisis exploratorio 5 y análisis topológico 6. El enfoque usado es muy variado. Gracias a la proyección `UMAP` hemos podido visualizar el resultado de distintos algoritmos de clustering, como `K-Means` o `HDBSCAN`. También se han buscado agrupaciones entre los propios componentes del `LSA`, a través de la técnica de *clustering jerárquico*. Sin embargo, el enfoque más novedoso y que más resultados ha dado ha sido el análisis topológico, donde se ha aplicado la técnica de Mapper [71] para dar una primera visión de la estructura de distintos subconjuntos. También se han explorado las homologías de los documentos de cada una de las instituciones, así como de los ODS, y se ha explorado la distancia entre dichos diagramas de persistencia desde distintas perspectivas. Por todo esto, considero que este objetivo se ha desarrollado lo suficientemente bien.

El cuarto objetivo consistía en diseñar y evaluar un modelo de clasificación supervisado que utilice información topológica y puede que textual. Este objetivo se ha desarrollado en el capítulo 7, donde se ha diseñado un clasificador que utiliza la topología de los datos para clasificar documentos. El clasificador se basa en la distancia entre diagramas de persistencia, y tiene un enfoque similar al desarrollado por Ferrà et al. [52]. También se han introducido ideas de otros clasificadores, como el propuesto en el artículo de Kindelan el al. [53]. Este clasificador se basa totalmente en información topológica.

Por último, en ese mismo capítulo comparamos el rendimiento de nuestro modelo con otros clasificadores más tradicionales, como `KNN` o `SVM`, utilizando las mismas características de entrada para asegurar una comparación justa. Los resultados obtenidos muestran que, si bien el clasificador topológico es competitivo en clases con un número suficiente de muestras, como *Informáticos* o *Sistemas Informáticos*, su rendimiento se ve claramente afectado en aquellas clases menos representadas. Esta observación, respaldada por un análisis estadístico de correlación, sugiere que el rendimiento del modelo depende en gran medida del tamaño de la clase, lo que refuerza la necesidad de disponer de datos equilibrados para explotar todo su potencial. A pesar de esta limitación, el modelo ha demostrado ser una alternativa válida a los métodos tradicionales, cumpliendo así con el objetivo propuesto.

Como conclusión final, me gustaría especificar cual es el pipeline propuesto en este TFG para el desarrollo del clasificador. Este está representado a través de la Figura8.1.

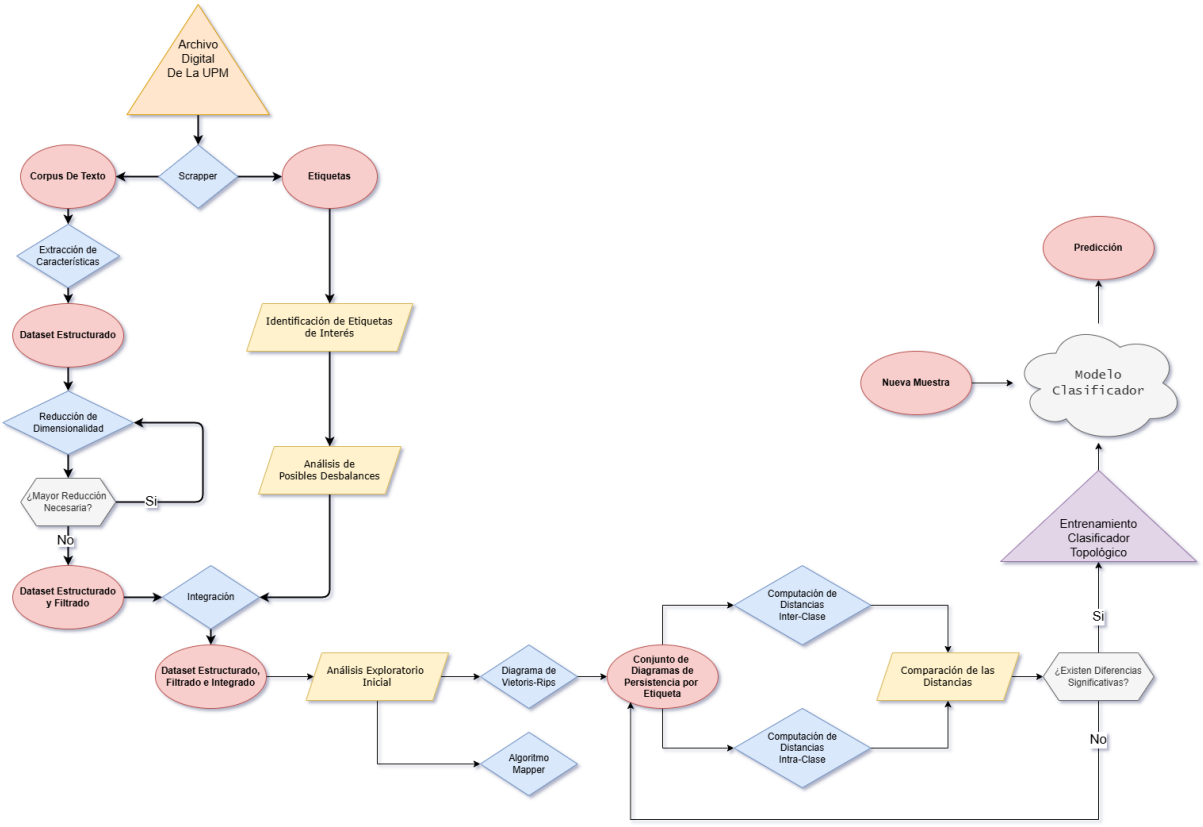


Figura 8.1: Diagrama del pipeline de análisis y clasificación propuesto en este TFG. El proceso comienza con la extracción de los documentos desde el *Archivo Digital de la UPM*, y finaliza con la clasificación de nuevos documentos a través del clasificador desarrollado.

8.2. Conclusiones personales

En primer lugar, me gustaría comenzar diciendo que el tema de partida del TFG me parecía muy interesante en lo personal, ya que supone una línea de investigación muy poco explorada, lo que me ha dado la libertad de poder reunir técnicas de otros ámbitos y aplicarlas a este nuevo concepto. No exiten apenas trabajos que sigan este enfoque fecha de hoy.

Por otro lado, esto también supone que los resultados no estaban asegurados, y no podría asegurar en un primer momento que esta idea fuera a dar sus frutos.

Una de las mayores complicaciones que he encontrado a lo largo del análisis y desarrollo del clasificador ha sido la delgada línea que separaba los trabajos entre sí. Tanto por la etiqueta de institución como por la de ODS, al venir todos los trabajos desde la misma témática base (la informática), todos ellos era muy similares entre sí. Esto hacía complicado encontrar fronteras durante el análisis. También dificultaba la tarea de encontrar diferencias entre la topología de los datos, ya que las diferencias que encontré eran muy sutiles al estar en un espacio tan relacionado. Creo que si la temática entre documentos hubiera estado más diversificada, habría resultado más sencillo el desarrollo del clasificador.

Otra gran limitación asociada a la clasifiación ha sido el desbalanceo de clases. Esta limitación se reflejada sobre todo al intentar entender la topología de los datos, ya que no podemos definir la forma de un dataset con apenas 2, 3 o 20 elementos. Esos subconjuntos no tiene una topología definida, son tan solo una nube de puntos en el espacio donde no encontramos una estructura clara. Este desbalanceo ha hecho que muchas clases poco representadas no funcionen correctamente en el clasificador, pero en especial nos ha hecho eliminar algunas etiquetas por no llegar apenas a las 50 muestras.

No obstante, creo que la metodología usada para partir desde el conjunto de documentos hasta el clasificador supone el desarrollo de un enfoque que podría ser novedoso, partiendo desde un conjunto de datos desestructurados y llegando hasta un dataset útil e informativo a través de un pipeline de análisis y extracción de información basado en un proceso totalmente justificado y documentado. Al final, la idea de este TFG era desarrollar no solo el clasificador, sino todo el proceso que acompaña al tratamiento de los datos, y considero que esa parte ha sido un éxito.

Creo que el clasificador desarrollado es un buen punto de partida para la creación de un modelo de clasificación basado en la topología de los datos, e incluso se podrían buscar otros enfoques que partan desde la misma idea, el uso de la topología en lugar de técnicas de machine learning más tradicionales, como SVM o KNN, para la clasificación de documentos. Es una idea con un gran potencial que puede ser explotada desde muchso ángulos y bajo muchos enfoques distintos. Tal vez este no fuera el contexto ideal, pero los resultados demuestran que este pipeline de análisis y clasificación tiene gran potencial.

8.3. Trabajo futuro

Unas de las líneas de trabajo más naturales que se podrían seguir a partir de este TFG es el análisis y la clasificación de los **ODS** asignados a los documentos. Para esto se requerirían una serie de modificaciones tanto en el análisis como en el clasificador, en vista de la asignación de distintos ODS a un mismo documento. Esto podría suponer un enfoque mucho más generalista y que podría dar resultados más interesantes. De la misma forma, se podrían buscar otras etiquetas, como el Departamento de procedencia del documento, o la materia del documento.

Otra línea de trabajo bastante clara plantea el rendimiento de este clasificador bajo un conjunto de datos sintético, que permita simular una situación con clases más separadas y distantes entre sí, ademas de con un mayor de muestras por clase. Esto permitiría comprobar la hipótesis de que realmente el clasificador mejora su rendimiento con un mayor número de muestras por clase. Además, de esta forma se podría experimentar con otro tipo de hiperparámetros y distancias, dando lugar así a nuevos modelos.

Otra línea de trabajo podría pasar por la mejora del clasificador, ya sea en busca de nuevas métricas para determinar la pertenecia a la clase incluso en casos donde la topología del subconjunto no esté desarrollada completamente. Esto podría llevar a no comparar únicamente diagramas de persistencia, sino también otro tipo de representaciones topológicas, en busca de nuevas formas de visión de estas características. Una opción podría ser buscar formas de calcular distancias entre resultados de Mapper, aunque puede resultar un análisis complicado. También se podría bus-

car la mejora del pipeline de análisis, buscando nuevas formas de preprocesar los documentos, o incluso nuevas formas de extraer características de los documentos. Algunas de las opciones para estas etapas del pipeline podrían ser la extracción de *embeddings* a través de modelos como `BERT` [23], o la aplicación de técnicas de *Topic Modeling* como `LDA` o `NMF`. Estas técnicas podrían necesitar el desarrollo de un nuevo pipeline de análisis y de clasificación.

Un último enfoqque podría ser la integración de este clasificador en un modelo de *Machine Learning*, que permita suplir la falta de estructuras de algunas clases, pero que a su vez sea capaz de aprender de la topología de los datos. Una posible idea sería un modelo capaz de, dado una etiqueta y un subconjunto de documentos con un bajo número de muestras, generar un subconjunto de documentos mayor que simule la topología "real" que deberían de tener esos documentos. Gracias a este modelo, las limitaciones por falta de estructura del clasificador quedarían suplidas y se podría trabajar mejor con estas etiquetas poco representadas en el *Archivo Digital de la UPM*.

Capítulo 9

Impacto del trabajo

9.1. Impacto general

Los resultados obtenidos en este Trabajo de Fin de Grado tienen un impacto potencial en varios contextos vinculados a la clasificación documental y la gestión del conocimiento académico. En primer lugar, en el ámbito universitario, la metodología propuesta mejora la organización del Archivo Digital de la UPM al permitir una categorización más precisa de los trabajos académicos según atributos como la institución. Esto facilita en primer lugar la automatización de procesos de inserción y análisis de documentos, y en segundo lugar, la localización y recuperación de información relevante para investigadores, estudiantes y personal administrativo.

En un contexto más amplio, la combinación de Procesamiento de Lenguaje Natural (PLN) y Análisis Topológico de Datos (TDA) abre nuevas posibilidades para la clasificación automática de documentos en repositorios digitales, bibliotecas científicas o sistemas de apoyo a la toma de decisiones. La metodología, al centrarse en representaciones compactas y robustas, resulta especialmente útil en entornos con recursos computacionales limitados o en dominios con estructuras semánticas complejas.

Durante el desarrollo del trabajo se han tomado decisiones fundamentadas en la consideración del impacto. Por ejemplo, la selección de secciones clave del documento (título, resumen y conclusiones) responde al objetivo de capturar la esencia semántica del texto de forma eficiente, reduciendo el coste computacional asociado al análisis topológico. Asimismo, la elección de técnicas topológicas como la homología persistente, frente a modelos de *Deep Learning*, refleja la intención de maximizar la interpretabilidad y la aplicabilidad en contextos prácticos con pocos recursos disponibles.

Además, se ha incluido un análisis de distancias intra e inter-clase utilizando métricas topológicas con el fin de entender mejor las relaciones estructurales entre categorías. Este enfoque no solo mejora la clasificación, sino que aporta valor para futuros estudios exploratorios o auditorías del repositorio. Por tanto, el impacto del trabajo no se limita a los resultados clasificatorios, sino que también proporciona herramientas y enfoques replicables para el análisis de colecciones textuales en múltiples dominios.

9.2. Objetivos de Desarrollo Sostenible

El presente trabajo se alinea con varios Objetivos de Desarrollo Sostenible (ODS) establecidos por la ONU en el marco de la Agenda 2030. En concreto, los resultados obtenidos y las decisiones metodológicas adoptadas contribuyen de forma significativa a los ODS 4, 9 y 16, tal y como se justifica a continuación.

ODS 4: Educación de calidad. Este objetivo busca garantizar una educación inclusiva, equitativa y de calidad, y promover oportunidades de aprendizaje durante toda la vida para todos. Entre sus metas se encuentra mejorar el acceso a materiales educativos relevantes y aumentar la disponibilidad de recursos para la investigación. El trabajo contribuye directamente a esta meta al desarrollar una herramienta que mejora la organización, clasificación y accesibilidad del Archivo Digital de la UPM. Al facilitar la recuperación de trabajos académicos por institución, departamento u objetivo temático, se optimiza el uso del conocimiento generado, haciendo más accesibles los contenidos para estudiantes, docentes e investigadores.

ODS 9: Industria, innovación e infraestructura. Este objetivo promueve la construcción de infraestructuras resilientes, la industrialización sostenible y el fomento de la innovación. El TFG aplica técnicas innovadoras, como el Análisis Topológico de Datos combinado con PLN, para resolver un problema práctico en la gestión documental académica. Este enfoque no solo representa una mejora técnica frente a los métodos clásicos, sino que también puede trasladarse a otros contextos institucionales, promoviendo una digitalización más inteligente y estructurada de contenidos. Además, el uso de herramientas abiertas y eficientes contribuye a construir soluciones replicables y sostenibles.

ODS 16: Paz, justicia e instituciones sólidas. Este objetivo incluye metas como desarrollar instituciones eficaces, responsables y transparentes, así como garantizar el acceso público a la información. Al estructurar, clasificar y analizar documentos institucionales de forma automatizada y objetiva, el trabajo fortalece los mecanismos de gestión documental de una universidad pública, fomentando así la transparencia, la trazabilidad del conocimiento y la eficiencia institucional. El enfoque reproducible, basado en código abierto, refuerza la equidad y la apertura de las soluciones propuestas.

Bibliografía

[1] Frédéric Chazal y Bertrand Michel. «An introduction to Topological Data Analysis: fundamental and practical aspects for data scientists». En: *arXiv preprint arXiv:1710.04019* (2017). URL: https://arxiv.org/abs/1710.04019.

[2] Noam Chomsky. *Syntactic Structures*. Mouton, 1957. URL: https://www.worldcat.org/title/syntactic-structures/oclc/185719.

[3] Hans Peter Luhn. «The automatic creation of literature abstracts». En: *IBM Journal of Research and Development* 2.2 (1958), págs. 159-165. URL: https://ieeexplore.ieee.org/document/5392820.

[4] David E Rumelhart, Geoffrey E Hinton y Ronald J Williams. «Learning representations by back-propagating errors». En: *Nature* 323 (1986), págs. 533-536. URL: https://www.nature.com/articles/323533a0.

[5] Yann LeCun et al. «Gradient-based learning applied to document recognition». En: *Proceedings of the IEEE* 86.11 (1998), págs. 2278-2324. URL: https://ieeexplore.ieee.org/document/726791.

[6] Ashish Vaswani et al. «Attention is all you need». En: *Advances in Neural Information Processing Systems*. Vol. 30. 2017. URL: https://arxiv.org/abs/1706.03762.

[7] Zellig S Harris. «Distributional structure». En: *Word* 10.2-3 (1954), págs. 146-162. URL: https://www.tandfonline.com/doi/abs/10.1080/00437956.1954.11659520.

[8] Karen Sparck Jones. «A statistical interpretation of term specificity and its application in retrieval». En: *Journal of Documentation* 28.1 (1972), págs. 11-21. URL: https://www.emerald.com/insight/content/doi/10.1108/eb026526/full/html.

[9] Andrew McCallum y Kamal Nigam. «A comparison of event models for naive bayes text classification». En: *AAAI-98 workshop on learning for text categorization* 752 (1998), págs. 41-48. URL: https://www.aaai.org/Papers/Workshops/1998/WS-98-05/WS98-05-007.pdf.

[10] Corinna Cortes y Vladimir Vapnik. «Support-vector networks». En: *Machine Learning* 20.3 (1995), págs. 273-297. DOI: 10.1007/BF00994018. URL: https://link.springer.com/article/10.1007/BF00994018.

[11] Thorsten Joachims. «Text categorization with support vector machines: Learning with many relevant features». En: *European conference on machine learning*. Springer. 1998, págs. 137-142. URL: https://link.springer.com/chapter/10.1007/BFb0026683.

[12] Fabrizio Sebastiani. «Machine learning in automated text categorization». En: *ACM Computing Surveys (CSUR)*. Vol. 34. 1. ACM New York, NY, USA, 2002,

págs. 1-47. DOI: 10.1145/505282.505283. URL: https://dl.acm.org/doi/10.1145/505282.505283.

[13] Larhmam. *SVM margin.png*. https://commons.wikimedia.org/wiki/File:SVM_margin.png. Licencia CC BY-SA 4.0. 2018.

[14] Xiang Zhang, Junbo Zhao y Yann LeCun. «Character-level convolutional networks for text classification». En: *Advances in Neural Information Processing Systems*. Vol. 28. 2015. URL: https://papers.nips.cc/paper/5782-character-level-convolutional-networks-for-text-classification.

[15] Ziyuan Yuan, Yong Lu y Zhaoguo Xue. «DroidDetector: Android malware characterization and detection using deep learning». En: *Tsinghua Science and Technology* 21.1 (2016), págs. 114-123. DOI: 10.1109/TST.2016.7399282. URL: https://ieeexplore.ieee.org/document/7399282.

[16] Wen Liu et al. «Deep learning based Android malware detection». En: *Proceedings of the 31st International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems* (2016). DOI: 10.1007/978-3-319-42007-3_40. URL: https://link.springer.com/chapter/10.1007/978-3-319-42007-3_40.

[17] Tomas Mikolov et al. «Efficient estimation of word representations in vector space». En: *arXiv preprint arXiv:1301.3781* (2013). URL: https://arxiv.org/abs/1301.3781.

[18] Jeffrey Pennington, Richard Socher y Christopher D Manning. «GloVe: Global Vectors for Word Representation». En: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, págs. 1532-1543. URL: https://aclanthology.org/D14-1162.

[19] Piotr Bojanowski et al. «Enriching word vectors with subword information». En: *Transactions of the Association for Computational Linguistics* 5 (2017), págs. 135-146. URL: https://aclanthology.org/Q17-1010.

[20] Quoc Le y Tomas Mikolov. «Distributed Representations of Sentences and Documents». En: *International Conference on Machine Learning*. PMLR. 2014, págs. 1188-1196. URL: https://proceedings.mlr.press/v32/le14.html.

[21] Armand Joulin et al. «Bag of tricks for efficient text classification». En: *arXiv preprint arXiv:1607.01759* (2016). URL: https://arxiv.org/abs/1607.01759.

[22] Zichao Yang et al. «Hierarchical attention networks for document classification». En: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2016, págs. 1480-1489. DOI: 10.18653/v1/N16-1174. URL: https://aclanthology.org/N16-1174/.

[23] Jacob Devlin et al. «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». En: *NAACL-HLT* (2019). DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423/.

[24] Daniel Voigt Godoy. *BERT next sequence prediction task.png*. https://commons.wikimedia.org/wiki/File:BERT_next_sequence_prediction_task.png. Licencia CC BY 4.0. 2021.

[25] Yinhan Liu et al. «RoBERTa: A Robustly Optimized BERT Pretraining Approach». En: *arXiv preprint arXiv:1907.11692* (2019). URL: https://arxiv.org/abs/1907.11692.

[26] Zhilin Yang et al. «XLNet: Generalized Autoregressive Pretraining for Language Understanding». En: *Advances in Neural Information Processing Systems*.

Vol. 32. 2019. URL: https://papers.nips.cc/paper/8812-xlnet-generalized-autoregressive-pretraining-for-language-understanding.

[27] Gerard Salton y Christopher Buckley. «Term-weighting approaches in automatic text retrieval». En: *Information processing & management* 24.5 (1988), págs. 513-523. URL: https://doi.org/10.1016/0306-4573(88)90021-0.

[28] Tomas Mikolov et al. «Distributed representations of words and phrases and their compositionality». En: *Advances in neural information processing systems*. Vol. 26. 2013. URL: https://papers.nips.cc/paper_files/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html.

[29] Yoav Goldberg y Omer Levy. «word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method». En: *arXiv preprint arXiv:1402.3722* (2014). URL: https://arxiv.org/abs/1402.3722.

[30] Jey Han Lau y Timothy Baldwin. «An empirical evaluation of doc2vec with practical insights into document embedding generation». En: *Proceedings of the 1st Workshop on Representation Learning for NLP*. 2016, págs. 78-86. DOI: 10.18653/v1/W16-1609. URL: https://aclanthology.org/W16-1609/.

[31] Daniel Saggau et al. «Efficient Document Embeddings via Self-Contrastive Bregman Divergence Learning». En: *arXiv preprint arXiv:2305.16031* (2023). URL: https://arxiv.org/abs/2305.16031.

[32] Arman Cohan et al. «Specter: Document-level representation learning using citation-informed transformers». En: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, págs. 2270-2282. DOI: 10.18653/v1/2020.acl-main.207. URL: https://aclanthology.org/2020.acl-main.207/.

[33] David M Blei, Andrew Y Ng y Michael I Jordan. «Latent dirichlet allocation». En: *Journal of machine Learning research* 3.Jan (2003), págs. 993-1022. URL: http://www.jmlr.org/papers/v3/blei03a.html.

[34] Akash Srivastava y Charles Sutton. «Autoencoding variational inference for topic models». En: *arXiv preprint arXiv:1703.01488* (2017). URL: https://arxiv.org/abs/1703.01488.

[35] Laurens van der Maaten y Geoffrey Hinton. «Visualizing data using t-SNE». En: *Journal of Machine Learning Research* 9.Nov (2008), págs. 2579-2605. URL: https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf.

[36] Leland McInnes, John Healy y James Melville. «UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction». En: *arXiv preprint arXiv:1802.03426* (2018). URL: https://arxiv.org/abs/1802.03426.

[37] Scott Deerwester et al. «Indexing by Latent Semantic Analysis». En: *Journal of the American Society for Information Science* 41.6 (1990), págs. 391-407. DOI: 10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9. URL: https://doi.org/10.1002/(SICI)1097-4571(199009)41:6%3C391::AID-ASI1%3E3.0.CO;2-9.

[38] Thomas K Landauer, Peter W Foltz y Darrell Laham. «An introduction to latent semantic analysis». En: *Discourse Processes* 25.2-3 (1998), págs. 259-284. DOI: 10.1080/01638539809545028. URL: https://www.tandfonline.com/doi/abs/10.1080/01638539809545028.

[39] Diederik P. Kingma y Max Welling. «Auto-Encoding Variational Bayes». En: *arXiv preprint arXiv:1312.6114* (2013). URL: https://arxiv.org/abs/1312.6114.

[40] Gunnar Carlsson. «Topology and data». En: *Bulletin of the American Mathematical Society* 46.2 (2009), págs. 255-308. DOI: 10.1090/S0273-0979-09-01249-X. URL: https://www.ams.org/bull/2009-46-02/S0273-0979-09-01249-X/.

[41] Robert Ghrist. «Barcodes: The persistent topology of data». En: *Bulletin of the American Mathematical Society* 45.1 (2008), págs. 61-75. DOI: 10.1090/S0273-0979-07-01191-3. URL: https://www.ams.org/bull/2008-45-01/S0273-0979-07-01191-3/.

[42] Herbert Edelsbrunner, David Letscher y Afra Zomorodian. «Topological persistence and simplification». En: *Proceedings 41st Annual Symposium on Foundations of Computer Science*. IEEE. 2000, págs. 454-463. URL: https://pub.ista.ac.at/~edels/Papers/2002-04-TopologicalPersistence.pdf.

[43] Nina Otter et al. «A roadmap for the computation of persistent homology». En: *EPJ Data Science* 6.1 (2017), pág. 17. DOI: 10.1140/epjds/s13688-017-0109-5. URL: https://epjdatascience.springeropen.com/articles/10.1140/epjds/s13688-017-0109-5.

[44] David Cohen-Steiner, Herbert Edelsbrunner y John Harer. «Stability of persistence diagrams». En: *Discrete & Computational Geometry* 37.1 (2007), págs. 103-120. DOI: 10.1007/s00454-006-1276-5. URL: https://link.springer.com/article/10.1007/s00454-006-1276-5.

[45] Roberto Rizzo y Martina Scolamiero. «A topological data analysis approach for natural language processing». En: *arXiv preprint arXiv:2009.13247* (2020). URL: https://arxiv.org/abs/2009.13247.

[46] Aruni Choudhary. *Example of a Vietoris-Rips filtration on point cloud data.png*. https://commons.wikimedia.org/wiki/File:Example_of_a_Vietoris-Rips_filtration_on_point_cloud_data.png. Licencia CC BY 3.0. 2017.

[47] Saber Gholizadeh, Bianca Zadrozny y Boleslaw K Szymanski. «Topological signatures of data: A view from the applied side». En: *Information Sciences* 530 (2020), págs. 112-130. DOI: 10.1016/j.ins.2020.05.015. URL: https://doi.org/10.1016/j.ins.2020.05.015.

[48] Ilan Perez y Raphael Reinauer. «The Topological BERT: Transforming Attention into Topology for Natural Language Processing». En: *arXiv preprint arXiv:2206.15195* (2022). URL: https://arxiv.org/abs/2206.15195.

[49] Eduard Tulchinskii et al. «Topological Data Analysis for Speech Processing». En: *arXiv preprint arXiv:2211.17223* (2023). URL: https://arxiv.org/abs/2211.17223.

[50] Saber Gholizadeh et al. «Topological Analysis of Contradictions in Text». En: *Proceedings of the 2021 Workshop on Deep Learning and Formal Languages*. 2021. DOI: 10.1145/3477495.3531881. URL: https://dl.acm.org/doi/10.1145/3477495.3531881.

[51] Xiaoyu Deng y Sergey Duzhin. «Fake news detection using topological and contextual embeddings». En: *Mathematics* 10.5 (2022), pág. 760. DOI: 10.3390/math10050760. URL: https://www.mdpi.com/2227-7390/10/5/760.

[52] Aina Ferrà et al. «A topological classifier to characterize brain states: When shape matters more than variance». En: *PLOS ONE* 18.10 (2023), e0292049. DOI: 10.1371/journal.pone.0292049. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0292049.

[53] Iker Kindelán et al. «A topological data analysis-based classifier». En: *Applied Network Science* 7.1 (2022), págs. 1-23. DOI: 10.1007/s41109-022-00489-4.

URL: https://link.springer.com/article/10.1007/s41109-022-00489-4.

[54] Guillaume Tauzin et al. «giotto-tda: A topological data analysis toolkit for machine learning and data exploration». En: *Journal of Machine Learning Research* 22.39 (2021), págs. 1-6. URL: https://www.jmlr.org/papers/volume22/20-1342/20-1342.pdf.

[55] Yasuhiro Umeda. «Topological Data Analysis in Python: A modern overview with practical examples using giotto-tda». En: *Software Impacts* 17 (2023), pág. 100308. DOI: 10.1016/j.simpa.2023.100308. URL: https://www.sciencedirect.com/science/article/pii/S2665963823000334.

[56] Clément Maria et al. «The GUDHI library: Simplicial complexes and persistent homology». En: *International Congress on Mathematical Software*. Springer. 2014, págs. 167-174. DOI: 10.1007/978-3-662-44199-2_28. URL: https://link.springer.com/chapter/10.1007/978-3-662-44199-2_28.

[57] Ulrich Bauer. «Ripser: efficient computation of Vietoris–Rips persistence barcodes». En: *Journal of Applied and Computational Topology* 5.3 (2021), págs. 391-423. DOI: 10.1007/s41468-021-00071-5. URL: https://link.springer.com/article/10.1007/s41468-021-00071-5.

[58] Universidad Politécnica de Madrid. *Archivo Digital UPM - Informática*. https://oa.upm.es/view/subjects/informatica.html. Accedido: 2025-04-03. 2025.

[59] Xiangyu Zhang et al. «MotifClass: Weakly Supervised Text Classification with Higher-order Metadata Information». En: *arXiv preprint arXiv:2111.04022* (2021). URL: https://arxiv.org/abs/2111.04022.

[60] Leonard Richardson. *Beautiful Soup Documentation*. https://www.crummy.com/software/BeautifulSoup/. Accedido: 2025-04-03. 2023.

[61] Artifex Software Inc. *PyMuPDF (fitz) Documentation*. https://pymupdf.readthedocs.io. Accedido: 2025-04-03. 2023.

[62] Steven Bird, Ewan Klein y Edward Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Inc., 2009. ISBN: 9780596516499. URL: https://www.nltk.org/book/.

[63] Claude E. Shannon. «A Mathematical Theory of Communication». En: *Bell System Technical Journal* 27.3 (1948), págs. 379-423. DOI: 10.1002/j.1538-7305.1948.tb01338.x. URL: https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf.

[64] Jerry Watkins et al. «Entropy-based feature selection for capturing impacts in Earth system models with extreme forcing». En: *arXiv preprint arXiv:2409.18011* (2024). URL: https://arxiv.org/abs/2409.18011.

[65] Sandeep Singh y Rajesh Sharma. «Reducing Dimensionality of Text Documents Using Latent Semantic Analysis». En: *International Journal of Computer Applications* 112.5 (2015), págs. 1-4. URL: https://www.ijcaonline.org/archives/volume112/number5/19660-1078/.

[66] Susan T. Dumais. «Latent Semantic Analysis». En: *Annual Review of Information Science and Technology* 38.1 (2004), págs. 188-230. DOI: 10.1002/aris.1440380105. URL: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/aris.1440380105.

[67] Yimin Zhang, Rong Jin y Zhi-Hua Zhou. «Understanding bag-of-words model: A statistical framework». En: *International Journal of Machine Learning and Cy-*

bernetics 1.1–4 (2011), págs. 43-52. DOI: 10.1007/s13042-010-0001-0. URL: https://link.springer.com/article/10.1007/s13042-010-0001-0.

[68] Francisco Díaz, Rubén González y Víctor Prieto. «UMAPTopic: Interactive Visual Topic Modeling using UMAP and HDBSCAN». En: *Journal of Machine Learning Research* 23.25 (2022), págs. 1-27. URL: https://www.jmlr.org/papers/volume23/22-014/22-014.pdf.

[69] Jiwei Li, Will Monroe y Dan Jurafsky. «Visualizing and Understanding Neural Models in NLP». En: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. 2021. DOI: 10.18653/v1/N16-1082. URL: https://aclanthology.org/N16-1082/.

[70] 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 (PAKDD 2013)*. Vol. 7819. Lecture Notes in Computer Science. Springer, 2013, págs. 160-172. DOI: 10.1007/978-3-642-37456-2_14. URL: https://doi.org/10.1007/978-3-642-37456-2_14.

[71] Gurjeet Singh, Facundo Mémoli y Gunnar Carlsson. «Topological methods for the analysis of high dimensional data sets and 3D object recognition». En: *Eurographics Symposium on Point-Based Graphics*. The Eurographics Association, 2007, págs. 91-100. URL: https://research.math.osu.edu/tgda/mapperPBG.pdf.

[72] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002. URL: https://pi.math.cornell.edu/~hatcher/AT/AT.pdf.

[73] Herbert Edelsbrunner y John Harer. *Computational Topology: An Introduction*. American Mathematical Society, 2010. URL: https://bookstore.ams.org/view?ProductCode=MBK%2F69.

[74] OpenGenus Foundation. *Euclidean vs Manhattan vs Chebyshev distance*. https://iq.opengenus.org/euclidean-vs-manhattan-vs-chebyshev-distance/. 2018.

[75] Primoz Skraba y Katharine Turner. «Improved stability for the p-Wasserstein distance between persistence diagrams». En: *arXiv preprint arXiv:2006.16824* (2023). URL: https://arxiv.org/abs/2006.16824.

[76] Bei Wang, Hernando Ombao y Moo K. Chung. «Exploring persistent local homology in topological data analysis». En: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2016, págs. 6435-6439. DOI: 10.1109/ICASSP.2016.7472915. URL: https://www.sci.utah.edu/~beiwang/publications/PLH_ICASSP_BeiWang_2016.pdf.

[77] Shubhra Sankar Ray Chowdhury, Facundo Mémoli y Katharine Turner. «Topology applied to machine learning: From global to local». En: *Frontiers in Artificial Intelligence* 4 (2021), pág. 668302. DOI: 10.3389/frai.2021.668302. URL: https://www.frontiersin.org/articles/10.3389/frai.2021.668302/full.

Apéndice A

Anexo

A.1. Recibo de plagio de Turnitin

En esta sección incluiré las primeras 6 páginas del informe de plagio de Turnitin, que se genera al subir el trabajo a la plataforma. Este informe es un documento oficial que certifica que el trabajo es original y no contiene plagio.

MARIO GARCIA BERENGUER

tfg_etsiinf_NombreApellidos.pdf

Turnitin Memoria Final

TFG ETSIINF (Moodle PP)

Universidad Politecnica de Madrid

Detalles del documento

Identificador de la entrega
trn:oid:::1:3267072626

Fecha de entrega
2 jun 2025, 7:55 p.m. GMT+2

Fecha de descarga
2 jun 2025, 8:03 p.m. GMT+2

Nombre de archivo
23601_MARIO_GARCIA_BERENGUER_tfg_etsiinf_NombreApellidos_83714_1098837085.pdf

Tamaño de archivo
5.3 MB

111 Páginas

37.291 Palabras

210.251 Caracteres

4% Genel Benzerlik

Her veri tabanı için çakışan kaynaklar da dâhil tüm eşleşmelerin kombine toplamı.

Rapordan Filtrelenen

- Bibliyografya
- Alıntılanan Metin

Hariç tutulacaklar

- 1 Çıkarılan Kaynak

Ön Sıradaki Kaynaklar

0% İnternet kaynakları
0% Yayınlar
4% Gönderilen çalışmalar (Öğrenci Makaleleri)

Ön Sıradaki Kaynaklar

0% İnternet kaynakları
0% Yayınlar
4% Gönderilen çalışmalar (Öğrenci Makaleleri)

Ön Sıradaki Kaynaklar

Gönderi içinde en yüksek eşleşme sayısına sahip kaynaklar. Çakışan kaynaklar görüntülenmeyecektir.

1 Öğrenci makaleleri
Universidad Politécnica de Madrid <1%

2 Öğrenci makaleleri
Unviersidad de Granada <1%

3 Öğrenci makaleleri
Universitat Politècnica de València <1%

4 Öğrenci makaleleri
Universidad Internacional de la Rioja <1%

5 Öğrenci makaleleri
Imperial College of Science, Technology and Medicine <1%

6 Öğrenci makaleleri
TU Delft <1%

7 Öğrenci makaleleri
Universidad Rey Juan Carlos <1%

8 Öğrenci makaleleri
University of Pavol Jozef Safarik <1%

9 Öğrenci makaleleri
The University of Manchester <1%

10 Öğrenci makaleleri
Universidad Francisco de Vitoria <1%

11 Öğrenci makaleleri
University College London <1%

12 Öğrenci makaleleri
Cankaya University <1%

13 Öğrenci makaleleri
Universidad de Burgos UBUCEV <1%

14 Öğrenci makaleleri
University Politehnica of Bucharest <1%

15 Öğrenci makaleleri
Universidad Internacional Isabel I de Castilla <1%

16 Öğrenci makaleleri
Universidad de Salamanca <1%

17 Öğrenci makaleleri
Universidad del Istmo de Panamá <1%

18 Öğrenci makaleleri
Universidad Católica San Pablo <1%

19 Öğrenci makaleleri
University of Warwick <1%

20 Öğrenci makaleleri
National University of Ireland, Galway <1%

21 Öğrenci makaleleri
Vrije Universiteit Amsterdam <1%

22 Öğrenci makaleleri
th-koeln <1%

23 Öğrenci makaleleri
Coventry University <1%

24 Öğrenci makaleleri
Universidad de León <1%

25 Öğrenci makaleleri
University of Glamorgan <1%

26 Öğrenci makaleleri
University of Sunderland <1%

27 Öğrenci makaleleri
Üsküdar Üniversitesi <1%

28 Öğrenci makaleleri
Universidad Fernando Pessoa Canarias <1%

29 Öğrenci makaleleri
University of Southampton <1%

30 Öğrenci makaleleri
American University in the Emirates <1%

31 Öğrenci makaleleri
Jacobs University, Bremen <1%

32 Öğrenci makaleleri
Universidad Autonoma de Chile <1%

33 Öğrenci makaleleri
parsurnd <1%

34 Öğrenci makaleleri
Pontificia Universidad Catolica del Ecuador - PUCE <1%

35 Öğrenci makaleleri
Universidad del Rosario <1%

36 Öğrenci makaleleri
University of Sannio <1%

37 Öğrenci makaleleri
University of Stirling <1%

38 Öğrenci makaleleri
University of Surrey <1%

39 Öğrenci makaleleri
institutoeuropeodeposgrado <1%

40 Öğrenci makaleleri
BENEMERITA UNIVERSIDAD AUTONOMA DE PUEBLA BIBLIOTECA <1%

41 Öğrenci makaleleri
Pontificia Universidad Catolica Madre y Maestra PUCMM <1%

42 Öğrenci makaleleri
Swinburne University of Technology <1%

43 Öğrenci makaleleri
Universidad de Murcia <1%

44 Öğrenci makaleleri
Whitireia Community Polytechnic <1%

45 Öğrenci makaleleri
La Trobe University <1%

46 Öğrenci makaleleri
UCL <1%

47 Öğrenci makaleleri
UNICAF <1%

48 Öğrenci makaleleri
Universidad Europea de Madrid <1%

49 Öğrenci makaleleri
University of Oxford <1%

50 Öğrenci makaleleri
University of Southern California <1%

51 Öğrenci makaleleri
University of Sydney <1%

52 Öğrenci makaleleri
Queen's University of Belfast <1%

53 Öğrenci makaleleri
Universidad Autónoma de Nuevo León <1%

54 Öğrenci makaleleri
Universidad de Las Palmas de Gran Canaria <1%

55 Öğrenci makaleleri
Universiteit van Amsterdam <1%

56 Öğrenci makaleleri
University of Edinburgh <1%

57 Öğrenci makaleleri
uazuay <1%

58 Öğrenci makaleleri
CITY College, Affiliated Institute of the University of Sheffield <1%

59 Öğrenci makaleleri
UNIBA <1%

60 Öğrenci makaleleri
University of Birmingham <1%

61 Öğrenci makaleleri
Babes-Bolyai University <1%

62 Öğrenci makaleleri
Innopolis University <1%

63 Öğrenci makaleleri
Harrisburg University of Science and Technology <1%

Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Ciencia de Datos e Inteligencia Artificial

Trabajo Fin de Grado

Clasificación de documentos del AD-UPM mediante TDA

Autor: MARIO GARCÍA BERENGUER
Tutor: JUAN ANTONIO FERNANDEZ DEL POZO DE SALAMANCA

Madrid, 04 - 2025

A.2. Código del Clasificador

A continuación mostramos el código del clasificador desarrollado en el trabajo. Además, este código se encuentra disponible en un repositorio público de Google Drive, donde se puede consultar y descargar todo el proceso seguido durante el desarrollo del trabajo.

```
import numpy as np
import pandas as pd
from sklearn.neighbors import NearestNeighbors
from sklearn.metrics import pairwise_distances
from scipy.spatial.distance import pdist, squareform
from gtda.homology import VietorisRipsPersistence
from gtda.diagrams import PersistenceLandscape
from gtda.diagrams import Silhouette
from scipy.stats import wasserstein_distance
from tqdm import tqdm


COMPONENTS_COLUMNS = [f"Component_{i}" for i in range(1, 101)]
LABELS = ['E.T.S. de Ingenieros Informaticos (UPM)',
 'E.T.S.I. de Sistemas Informaticos (UPM)',
 'E.T.S.I. y Sistemas de Telecomunicacion (UPM)',
 'E.T.S.I. Telecomunicacion (UPM)',
 'E.T.S.I. Industriales (UPM)',
 'E.T.S. Arquitectura (UPM)']

class LabelClasifier():
    def __init__(self, train, test, k, components, local_topology, adding,
        n_add=20, ruido=0.001):
        self.X_train = train[[COMPONENTS_COLUMNS]]
        self.Y_train = train[['Institucion']]
        self.X_test = test[[COMPONENTS_COLUMNS]]
        self.Y_test = test[['Institucion']]
        self.local_topology = local_topology # If True, use local topology
            to select neighbors
        self.k_neighbors = k # Number of neighbors to consider
        self.components = components # Number of components to consider
            (100)
        self.adding = adding # If True, add points to create a point cloud
        self.n_add = n_add # Number of points to add
        self.ruido = ruido # Noise to add to the points

    def select_k_nearest(self, element, x_train_subset):
        # select the k nearest elements on x_train_subset for the element
        neigh = NearestNeighbors(n_neighbors=min(self.k_neighbors, len(
            x_train_subset)))
        neigh.fit(x_train_subset[[f"Component_{i}" for i in range(1, self.
            components + 1)]])
        distances, indices = neigh.kneighbors(element)
        return distances, indices

    def create_neighbors(self, item, n, ruido = 0.0005):
        vecinos = item + np.random.normal(loc=0, scale=ruido, size=(n, len(
            item)))
```

```
43          return vecinos
44
45      def calculate_diagram(self, x_train):
46          # Calculate the diagram for the x_train
47          df_componentes = x_train[[f"Component_{i}" for i in range(1, self.
                components + 1)]]
48          X = df_componentes.values.astype(np.float64)
49          D = squareform(pdist(X, metric='euclidean'))
50          D = np.expand_dims(D, axis=2)
51          D = D.reshape(1, X.shape[0], X.shape[0])
52          vr = VietorisRipsPersistence(metric='precomputed',
                homology_dimensions=[0, 1], n_jobs=4, max_edge_length=1.2)
53          diagrams = vr.fit_transform(D)
54          return diagrams
55
56      def fit_instance(self, item, label):
57          x_train = self.X_train.copy()
58          x_train = x_train[self.Y_train['Institucion'] == label]
59          # Use the filtered x_train_copy for NearestNeighbors
60          if self.local_topology:
61              comp_item = item[COMPONENTS_COLUMNS]
62              distances, indices = self.select_k_nearest(comp_item.values.
                    reshape(1, -1), x_train)
63              index = indices[0]
64              # Select the k nearest elements from x_train_copy
65              x_train= x_train.iloc[index]
66          dmg = self.calculate_diagram(x_train)
67          if self.adding:
68              # Create neighbors for the item
69              vecinos = self.create_neighbors(item[COMPONENTS_COLUMNS].values
                    , n=self.n_add, ruido=self.ruido)
70              # Convert to DataFrame
71              vecinos_df = pd.DataFrame(vecinos, columns=COMPONENTS_COLUMNS)
72              # Concatenate the item with the neighbors
73              item = pd.concat([item.to_frame().T, vecinos_df], ignore_index=
                    True)
74          if type(item) == pd.Series:
75              item = item.to_frame().T
76          x_train_aug = pd.concat([x_train, item], ignore_index=True)
77          # x_train_aug = x_train_aug[components_columns]
78          dmg_aug = self.calculate_diagram(x_train_aug)
79          return dmg, dmg_aug
80
81      def wasserstein_distance(self, dgm1, dgm2, homology_dim=1):
82          # Extract (birth, death) pairs for the specified homology dimension
83          dgm1_pairs = [tuple(pt[:2]) for pt in dgm1 if int(pt[2]) ==
                homology_dim]
84          dgm2_pairs = [tuple(pt[:2]) for pt in dgm2 if int(pt[2]) ==
                homology_dim]
85          distance = wasserstein_distance(dgm1_pairs, dgm2_pairs)
86          return distance
87
88      def pers_landscape_distance(self, dgm1, dgm2):
89          landscape_1 = PersistenceLandscape(
90                  n_layers=50,
91                  n_bins=500,
```

```
92          )
93          landscape_2 = PersistenceLandscape(
94                  n_layers=50,
95                  n_bins=500,
96          )
97          landscape_1 = landscape_1.fit_transform(dgm1).reshape(1, -1)
98          landscape_2 = landscape_2.fit_transform(dgm2).reshape(1, -1)
99          dist = pairwise_distances(landscape_1, landscape_2, metric='
               chebyshev')[0, 0]
100         return dist
101
102     def silhouette_distance(self, dgm1, dgm2, homology_dim=0):
103
104         sil = Silhouette(power=0.00001, n_bins=1000)
105
106         d1 = np.expand_dims(dgm1[dgm1[:, 2] == homology_dim], axis=0)
107         d2 = np.expand_dims(dgm2[dgm2[:, 2] == homology_dim], axis=0)
108
109         s1 = sil.fit_transform(d1).squeeze()
110         s2 = sil.fit_transform(d2).squeeze()
111
112         return np.linalg.norm(s1 - s2)
113
114     @staticmethod
115     def inverse_softmax_probabilities(distances, temperature=1.0):
116         distances = np.array(distances)
117         inverse_distances = -distances / temperature  # Inverse distances
               for softmax
118         exp_scores = np.exp(inverse_distances - np.max(inverse_distances))
                # Warning for overflow
119         probabilities = exp_scores / np.sum(exp_scores)
120         return probabilities
121
122     @staticmethod
123     def choose_temperature(distances):
124         std = np.std(distances)
125         if std < 0.1:
126             return 0.05
127         elif std < 0.5:
128             return 0.3
129         else:
130             return 0.8
131
132     def fit(self, metric):
133         result = []
134         for idx, instance in self.X_test.iterrows():
135             item = instance[COMPONENTS_COLUMNS]
136             label = self.Y_test.loc[idx, 'Institucion']
137             # Fit the instance for each label
138             distances = []
139             for l in LABELS:
140                 dgm, dgm_aug = self.fit_instance(item, l)
141                 if metric == 'wasserstein':
142                     dist = self.wasserstein_distance(dgm, dgm_aug)
143                 elif metric == 'pers_landscape':
144                     dist = self.pers_landscape_distance(dgm, dgm_aug)
```

```
                elif metric == 'silhouette':
                    dist = self.silhouette_distance(dgm, dgm_aug)
                else:
                    raise ValueError("Unknown metric. Choose 'wasserstein',
                        'pers_landscape', or 'silhouette'.")
                distances.append(dist)
            distances = np.array(distances)
            temperature = choose_temperature(distances)
            probabilities = inverse_softmax_probabilities(distances,
               temperature)
            result.append(probabilities)
            df_result = pd.DataFrame(index=LABELS, columns=[f'Prob_{metric}
               ', 'label'])
            df_result[f'Prob_{metric}'] = probabilities
            df_result['label'] = label
        return df_result


def fit_predict(clf1, clf2):
    #clf1 fits the Wasserstein distance
    df_wasserstein = clf1.fit(metric='wasserstein')
    #clf2 fits the Silhouette distance
    df_silhouette = clf2.fit(metric='silhouette')

    df_full = pd.DataFrame(index=LABELS, columns=['Prob_Wasserstein', '
       Prob_Silhouette', 'Prob_Mean', 'label'])
    df_full['Prob_Wasserstein'] = df_wasserstein['Prob_wasserstein']
    df_full['Prob_Silhouette'] = df_silhouette['Prob_silhouette']
    df_full['Prob_Mean'] = (df_full['Prob_Wasserstein'] + df_full['
       Prob_Silhouette']) / 2
    df_full['label'] = df_wasserstein['label']
    df_full['predicted_label'] = df_full['Prob_Mean'].idxmax(axis=1)

    df_full['correct'] = df_full['predicted_label'] == df_full['label']
    return df_full
```

El resto del código se puede encontar en este Repositorio de Google Drive.

A.3. Archivo Digital de la UPM

En la Figura 3.1 se muestra como es la página inicial del Archivo Digital de la UPM, donde se pueden ver los distintos tipos de documentos que se pueden encontrar en el repositorio. Mas en concreto, en la Figura A.1 se muestra la estructura de la sección de la que hemos extraido los documentos para el trabajo. Para complementar esto, tenemos un esquema en la Figura 3.2 que muestra como esta estructurada la información en el repositorio.

(ONDM)", 15-18 May 2017, Budapest, Hungary. pp. 1-6. https://doi.org/10.23919/ONDM.2017.7958519.

Aguado Moreno, Eduardo (2017). *Mensajería instantánea bajo sistemas SAACs para dispositivos Android "CHATAACSDROID"*. Trabajo Fin de Grado / Proyecto Fin de Carrera, E.T.S.I. de Sistemas Informáticos (UPM), Madrid.

Aguado Peña, Juan (2016). *Diseño y desarrollo de una plataforma web para la gestión y tracking de preguntas médicas y sus respuesta a través de Twitter para un análisis de inteligencia colectiva*. Trabajo Fin de Grado / Proyecto Fin de Carrera, E.T.S. de Ingenieros Informáticos (UPM), Madrid, España.

Aguado de Cea, Guadalupe (2007). *A multiperspective approach to specialized phraseology: Internet as a reference corpus for phraseology*. En: "The Texture of Internet: Netlinguistics in Progress". Cambridge Scholars Publishing, Cambridge. ISBN 9781847181732.

Aguado de Cea, Guadalupe (1994). *Algunos ejemplos de polisemia y sinonimia en la terminología informática*. Monografía (Informe Técnico). Facultad de Informática (UPM) [antigua denominación], Luxembourg.

Aguado de Cea, Guadalupe (2012). *Back to the Future: LSP fifty years later ESP in Spain*. En: "XI CONGRESSO INTERNACIONAL AELFE 2012. El papel de las lenguas aplicadas en el escenario pos-Boloña: ¿fomento de la autonomía y movilidad en un mundo globalizado?", 20 a 22 de septiembre de 2012, La Escola Superior de Estudos Industriais e de Gestão (ESEIG), Vila do Conde, Porto, Portugal.

Aguado de Cea, Guadalupe (1990). *Comando, instrucción, sentencia: ¿Sinónimos en el campo informático?*. En: "III Encuentros Complutenses en torno a la Traducción", 2-6 April 1990. ISBN 84-7491-468-X.

Aguado de Cea, Guadalupe (2006). *De bits y bugs a blogs y webs: aspectos interdisciplinares ,socioculturales y lingüísticos de la terminología informática*. En: "CORCILLVM: Estudios de traducción, lingüística y filología dedicados a Valentín García Yebra". Arco Libros, Madrid, Spain, pp. 693-720. ISBN 84-7635-648-X.

Aguado de Cea, Guadalupe (1986). *El lexema ware en el campo informático*. En: "Congreso Nacional de AEDEAN", 16-19 December 1986.

Aguado de Cea, Guadalupe (1998). *Homo ludens-designing tomorrow's games*. "The European Network for Intelligent Information Interfaces en la revista i3magazine" (n. 3); pp. 8-10. ISSN 1397-906-X.

Aguado de Cea, Guadalupe (1996). *La formación de formadores en Terminología*. "Terminómetro", v. La Ter ; pp. 63-64.

Aguado de Cea, Guadalupe (2007). *La fraseología en las lenguas de especialidad*. En: "Las lenguas profesionales y académicas". Ariel, Barcelona,

Figura A.1: Página de inicio del Archivo Digital de la UPM.

Para más información sobre el Archivo Digital de la UPM, se puede consultar su página oficial en oa.upm.es. Si se quiere consultar más en detalle el código HTML del archivo, podeis consultar este enlace para ver el ejemplo de un trabajo en concreto o este otro enlace para la sección de Informática del repositorio.

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
	Fecha/Hora	Wed Jun 04 10:11:33 CEST 2025
	Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
	Numero de Serie	561
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)