



# UNIVERSIDAD DE GRANADA

Facultad de Ciencias  
E.T.S. de Ingenierías Informática y de Telecomunicación

## DOBLE GRADO EN INGENIERÍA INFORMÁTICA Y MATEMÁTICAS

TRABAJO DE FIN DE GRADO

# Aplicación de la topología algebraica en redes neuronales

Presentado por:  
Pablo Olivares Martínez

Tutorizado por:  
Miguel Ortega Titos  
Julián Luengo Martín

Curso académico 2023-2024



# Aplicación de la topología algebraica en redes neuronales

Pablo Olivares Martínez

Pablo Olivares Martínez *Aplicación de la topología algebraica en redes neuronales.*  
Trabajo de fin de Grado. Curso académico 2023-2024.

**Responsables de  
tutorización**

Miguel Ortega Titos  
*Departamento de Geometría y Topología*

Doble Grado en Ingeniería  
Informática y Matemáticas

Julián Luengo Martín  
*Departamento de Ciencias de la Computación  
e Inteligencia Artificial*

Facultad de Ciencias  
E.T.S. de Ingenierías  
Informática y de  
Telecomunicación

Universidad de Granada

**DECLARACIÓN DE ORIGINALIDAD**

D./Dña. Pablo Olivares Martínez

Declaro explícitamente que el trabajo presentado como Trabajo de Fin de Grado (TFG), correspondiente al curso académico 2023-2024, es original, entendido esto en el sentido de que no he utilizado para la elaboración del trabajo fuentes sin citarlas debidamente.

En Granada a 26 de julio de 2024

A handwritten signature in black ink, appearing to read "Pablo Olivares Martínez". The signature is somewhat stylized and includes a small circle at the end of the first name.

Fdo: Pablo Olivares Martínez



*A mi familia y amigos,  
y en especial a Inés.*



# Índice general

<b>Agradecimientos</b>	<b>IX</b>
<b>Abstract</b>	<b>XI</b>
<b>Resumen</b>	<b>XIII</b>
<b>Introducción</b>	<b>XV</b>
1. Contexto . . . . .	XV
2. Motivación . . . . .	XVI
3. Estructura del trabajo . . . . .	XVI
4. Objetivos . . . . .	XVII
5. Presupuesto . . . . .	XVIII
6. Planificación . . . . .	XVIII
<b>I. Fundamento matemático</b>	<b>1</b>
<b>1. Fundamentos del álgebra homológica</b>	<b>3</b>
1.1. Módulos . . . . .	3
1.2. Sucesiones exactas . . . . .	7
1.3. Categorías y funtores . . . . .	8
1.4. Módulos diferenciales . . . . .	10
1.5. Complejos de cadenas . . . . .	11
1.6. Subcomplejos y complejos cociente . . . . .	14
<b>2. Simplices y complejos simpliciales</b>	<b>17</b>
2.1. Simplices . . . . .	17
2.2. Complejos simpliciales . . . . .	19
2.3. Celdas y CW-complejos . . . . .	23
2.4. Aplicaciones simpliciales . . . . .	27
2.5. Complejos simpliciales abstractos . . . . .	29
<b>3. Homología simplicial</b>	<b>31</b>
3.1. Homología simplicial orientada . . . . .	31
3.2. Homología del complejo cono . . . . .	35
3.3. Sucesión de Mayer-Vietoris . . . . .	37
3.4. Conexión y el módulo de homología $H_0(K; R)$ . . . . .	41
<b>4. Homología persistente</b>	<b>45</b>
4.1. Complejos de Čech y Vietoris-Rips . . . . .	45
4.2. Módulos de homología persistente . . . . .	47
4.3. Representación de la homología persistente . . . . .	50

<b>II. Aprendizaje profundo</b>	<b>61</b>
5. Aprendizaje automático y visión artificial	63
5.1. Aprendizaje automático . . . . .	63
5.2. Visión artificial . . . . .	65
6. Redes neuronales artificiales	67
6.1. Neuronas biológicas . . . . .	67
6.2. Redes neuronales artificiales . . . . .	68
6.3. Funciones de activación . . . . .	69
6.4. Optimización en redes neuronales . . . . .	70
6.4.1. Descenso del gradiente . . . . .	72
6.4.2. Momentum . . . . .	72
6.4.3. Adagrad . . . . .	73
6.4.4. RMSProp . . . . .	73
6.4.5. Adam . . . . .	74
6.5. Regularización en redes neuronales . . . . .	74
6.5.1. Regularización L1 y L2 . . . . .	74
6.5.2. Dropout . . . . .	75
6.5.3. Early stopping . . . . .	75
6.5.4. Regularización de datos . . . . .	75
7. Redes neuronales convolucionales	77
7.1. La corteza visual y el neocognitrón . . . . .	77
7.2. Arquitectura de una CNN . . . . .	78
7.2.1. Capa convolucional . . . . .	79
7.2.2. Capa de muestreo . . . . .	81
7.3. Modelos y estado del arte en CNNs . . . . .	82
7.3.1. ResNet . . . . .	83
7.3.2. DenseNet . . . . .	84
7.3.3. EfficientNet . . . . .	84
<b>III. Análisis de datos topológico para el estudio de CNNs</b>	<b>89</b>
8. Análisis de Datos Topológico	91
8.1. Análisis de datos topológico . . . . .	91
8.2. TDA en el estudio de CNNs . . . . .	92
8.3. Descriptores topológicos en TDA . . . . .	93
8.4. Propuesta: regularización con persistencia total normalizada . . . . .	94
9. Marco experimental y metodología	97
9.1. Experimentos . . . . .	97
9.2. Entorno de experimentación . . . . .	97
9.3. Conjunto de datos . . . . .	98
9.4. Preprocesamiento de datos . . . . .	100
9.5. Métricas de evaluación . . . . .	103
9.6. Proceso de entrenamiento . . . . .	104
9.7. Visualización de la homología persistente . . . . .	105

<b>10. Resultados experimentales</b>	<b>107</b>
10.1. Rendimiento y selección de modelos . . . . .	107
10.1.1. Selección de hiperparámetros . . . . .	107
10.1.2. Selección de transformaciones de datos . . . . .	111
10.2. Análisis de la homología persistente . . . . .	112
10.2.1. Comparación según la arquitectura . . . . .	113
10.2.2. Comparación según el optimizador . . . . .	114
10.2.3. Comparación según el tamaño de lote . . . . .	116
10.2.4. Comparación en presencia de aumento de datos . . . . .	117
10.2.5. Comparación según la granularidad de las clases . . . . .	119
10.2.6. Comparación según el subconjunto de datos . . . . .	120
10.2.7. Reflexiones sobre la topología en CNNs . . . . .	122
10.3. Propuesta de mejora: regularización topológica . . . . .	123
10.3.1. Mejora en clasificación . . . . .	123
10.3.2. Mejora en transferibilidad . . . . .	129
10.3.3. Impacto de la regularización topológica en CNNs . . . . .	134
<b>IV. Conclusiones y trabajo futuro</b>	<b>137</b>
<b>11. Conclusión</b>	<b>139</b>
<b>12. Trabajo futuro</b>	<b>141</b>
<b>Bibliografía</b>	<b>143</b>



## **Agradecimientos**

Quisiera comenzar agradeciendo a mis tutores, Miguel y Julián, por su gran ayuda y dedicación en todo momento de este trabajo.

A mis padres y hermanos, así como a mis abuelos, por apoyarme de manera incondicional desde la distancia y no dejar que me rindiera.

También quisiera agradecer a todos aquellos compañeros y amigos que, de una forma u otra, me han ayudado con este trabajo tanto con sus ideas y opiniones como con su aliento.

Por último, quisiera agradecérselo a Inés. Por tu paciencia y tu apoyo continuo, por estar a mi lado cuando más lo necesitaba, gracias.



## Abstract

This Bachelor's Thesis explores the integration of Topological Data Analysis (TDA) with convolutional neural networks (CNNs) to clarify and enhance our understanding of how CNNs manipulate data. Given the common perception of CNNs as «black boxes» due to the opacity on their internal decision-making processes, this work adopts an alternative approach through the application of persistent homology techniques, a key tool in TDA. This allows for a detailed analysis of the data structure during CNN processing, facilitating greater transparency and understanding of these networks' internal workings from a topological point of view.

The study focuses on the applicability of persistent homology to provide an additional layer of explainability and optimization to deep learning models. The underlying theory is explored, and implementations are carried out in practical cases using advanced network architectures such as ResNet, EfficientNet and DenseNet. Through controlled experiments, it is demonstrated that topological regulation not only improves the performance of CNNs in image classification and transfer learning tasks, but also offers new insights into the data structure throughout the learning process.

Specific results indicate that different configurations in CNN models significantly influence the topological characteristics of the data. The dynamics of the architectures studied show an initial tendency to simplify the data, possibly to remove noise and irrelevant details. However, as learning progresses, the topological complexity of the data increases, suggesting a deliberate strategy to develop richer and more detailed representations, thereby facilitating better differentiation between classes.

The implementation of a topological regularizer in selected models such as EfficientNet-Bo and DenseNet-121 has proven to be particularly promising. This approach adjusts the topological complexity of the data in a way that reflects a significant improvement in the accuracy and efficacy of classification. Moreover, it is observed that knowledge transfer markedly improves when the data topology is appropriately manipulated, suggesting that topological modifications could be an effective strategy for optimizing CNNs.

In conclusion, this document not only enhances the understanding of CNNs' internal mechanisms through TDA but also marks a step towards more transparent and reliable artificial intelligence (AI) models. The findings shows the utility of TDA in the field of deep learning, proposing a new alternatives for explainability and efficiency that may be interesting to consider in the future evolution of computer vision and data science.

**Keywords:** convolutional neural networks, topological data analysis, persistent homology, AI explainability, optimization of deep learning models, transfer learning.



## Resumen

Este Trabajo de Fin de Grado (TFG) explora la integración del Análisis de Datos Topológico (TDA) con redes neuronales convolucionales (CNNs) para clarificar y mejorar nuestra comprensión de cómo las CNNs manipulan los datos. Dada la percepción común de las CNNs como «cajas negras» debido a la opacidad en sus procesos de toma de decisiones internos, este trabajo adopta un enfoque alternativo mediante la aplicación de técnicas de homología persistente, una herramienta clave en TDA. Esto permite un análisis detallado de la estructura de datos durante el procesamiento en las CNNs, facilitando una mayor transparencia y entendimiento de los mecanismos internos de estas redes desde un punto de vista topológico.

El estudio se centra en la aplicabilidad de la homología persistente para proporcionar una mayor explicabilidad y optimización de los modelos de aprendizaje profundo. Se explora la teoría matemática subyacente, para posteriormente realizar implementaciones en casos prácticos utilizando arquitecturas de CNNs avanzadas como ResNet, EfficientNet y DenseNet. A través de experimentos controlados, se demuestra que la regulación topológica no solo mejora el rendimiento de las CNNs en tareas de clasificación de imágenes y de transferencia de conocimiento, sino que también ofrece nuevas perspectivas sobre la estructura de los datos a lo largo del proceso de aprendizaje.

Los resultados indican que las diferentes configuraciones en los modelos de CNNs influyen significativamente en las características topológicas de los datos. La dinámica de las arquitecturas estudiadas muestra una tendencia inicial a simplificar los datos, posiblemente para eliminar ruido y detalles irrelevantes. Sin embargo, a medida que avanza el aprendizaje, la complejidad topológica de los datos aumenta, sugiriendo una estrategia deliberada para obtener representaciones más ricas y detalladas, facilitando así una mejor diferenciación entre clases.

La implementación de un regularizador topológico en modelos como EfficientNet-Bo y DenseNet-121 ha demostrado ser particularmente prometedora. Este enfoque ajusta la complejidad topológica de los datos de manera que refleja una mejora significativa en la precisión y eficacia de la clasificación. Además, se observa que la transferencia de conocimiento mejora considerablemente cuando la topología de los datos se manipula adecuadamente, sugiriendo que las modificaciones topológicas podrían ser una estrategia efectiva para optimizar las CNNs.

En conclusión, este documento no solo mejora la comprensión de los mecanismos internos de las CNNs a través de TDA, sino que también marca un paso hacia modelos de inteligencia artificial (IA) más transparentes y confiables. Los hallazgos destacan la utilidad del TDA en el campo del aprendizaje profundo, proponiendo nuevas alternativas para la explicabilidad y eficiencia que pueden ser interesantes de considerar en la evolución futura de la visión artificial y la ciencia de datos.

**Palabras clave:** redes neuronales convolucionales, análisis de datos topológico, homología persistente, explicabilidad en IA, optimización de modelos de aprendizaje profundo, transferencia de conocimiento.



# Introducción

## 1. Contexto

El desarrollo humano ha sido constantemente impulsado por avances tecnológicos que han redefinido nuestra comprensión e interacción con mundo. Desde la invención de la imprenta hasta la revolución digital actual, cada era ha estado marcada por innovaciones clave. En particular, la invención de los ordenadores y el avance de las tecnologías de la información han convergido en la capacidad de generar, almacenar y analizar grandes volúmenes de datos. Este volumen de información ha desencadenado lo que ahora conocemos como la Era de la Información, que se caracteriza por el desarrollo de algoritmos avanzados que extraen valor de estos datos de manera automática y eficiente.

Dentro de esta revolución tecnológica, el aprendizaje profundo y, en particular, las redes neuronales convolucionales (CNNs) [8] han emergido como útiles herramientas, especialmente en el ámbito de la visión artificial. Estos modelos son capaces de identificar patrones complejos en datos visuales, superando a menudo el rendimiento humano en tareas de reconocimiento de imágenes. Sin embargo, las decisiones tomadas por las CNN a menudo son opacas y difíciles de interpretar, lo que ha llevado a que se les describa como «cajas negras».

Ante este problema, han surgido diversas técnicas para clarificar cómo estas redes toman decisiones. Una de las más novedosas y prometedoras es el análisis de datos topológico (TDA) [27], que emplea herramientas de la topología algebraica para ofrecer soluciones. El TDA busca comprender la «forma» de los datos, ofreciendo así conocimiento sobre cómo las CNNs estructuran y manipulan la información a nivel global, no solo basándose en instancias individuales.

En este trabajo, aplicaremos técnicas de TDA para desentrañar cómo las CNNs procesan y transforman conjuntos de datos, con el objetivo de comprender los mecanismos subyacentes de estos modelos de aprendizaje profundo y así proporcionar nuevas claves que nos permitan aprovecharlas mejor. Al enfocarnos en las estructuras globales de los datos en lugar de cambios individuales, esperamos ofrecer una comprensión más clara y detallada de cómo trabajan estas redes, contribuyendo así a una mayor comprensión y confianza en los algoritmos de aprendizaje profundo.

El TDA es un campo relativamente reciente. A comienzo de los años 90, bajo la premisa de que los datos tienen una «forma», matemáticos como Patrizio Frosini o Vanessa Robins estudiaron las propiedades que podían extraerse en el estudio de la distancia entre variedades [37] y estructuras relacionadas por el homomorfismo de inclusión [84]. Estas ideas cautivaron a Edelsbrunner, quien les dio forma en lo que hoy se conoce como homología persistente, junto con un algoritmo para calcularla y visualizarla de manera efectiva [33]. La homología persistente es actualmente la piedra angular del TDA, permitiendo el análisis de características topológicas que persisten a través de diferentes escalas. La homología persistente resuelve desafíos en la selección de parámetros al codificar información de todos los valores posibles. En 2008, Gunnar Carlsson [11] dio un paso adelante al reformular la homología persistente dentro del ámbito del álgebra commutativa, proporcionando lo que hoy conocemos como código de barras, facilitando su comprensión y ampliando su aplicación.

bilidad en ciencia de datos y otras áreas tecnológicas. Desde entonces, varios autores como Liwen Zhang, Gregory Naitzat y Lek-Heng Lim [72] han aplicado estas técnicas con el fin de comprender mejor el funcionamiento de las CNN. Su trabajo mostraba que, efectivamente, las CNNs simplificaban la «forma» de los datos. Inspirado por los resultados, German Magai [68] profundizó en estos hallazgos para confirmar dichas afirmaciones.

## **2. Motivación**

La comprensión del funcionamiento de los modelos de aprendizaje profundo ha emergido como una necesidad en el campo de la ciencia de datos. Las CNNs, aunque pesar de ser muy efectivas en muchas tareas de visión artificial, a menudo actúan como «cajas negras». Esto puede ser un problema, ya que nos impide comprender por qué un modelo ha tomado ciertas decisiones que pueden ser erróneas y por tanto ser incapaces de ofrecer solución al problema.

La topología es la rama de las matemáticas que estudia las transformaciones continuas, por lo que nos ofrece una perspectiva única para investigar cómo las CNNs procesan los datos. Aunque la topología es un área bastante abstracta de las matemáticas, su uso en el análisis de datos complejos es relativamente nuevo y prometedor. Dentro de este campo, la homología persistente aplicada en el marco del TDA se presenta como una herramienta innovadora para descifrar la manera en que las CNNs modifican la «forma» de los datos durante su procesamiento.

La aplicación de métodos topológicos a los problemas del aprendizaje profundo no es solo novedosa, sino que también tiene un gran potencial para transformar nuestra comprensión de los modelos complejos. Al explorar cómo el TDA puede mejorar la transparencia y eficacia de las CNNs, este estudio no solo busca aclarar el funcionamiento interno de estos modelos, sino que también se adentra en un campo poco explorado que cruza varias disciplinas con el objetivo final de abrir nuevas vías de investigación y aplicaciones prácticas no solo para mejorar la manera en que interactuamos con estas tecnologías, sino también para entender y confiar en las decisiones que toman.

## **3. Estructura del trabajo**

Con el fin de profundizar en la comprensión de los modelos de CNNs desde la perspectiva de la homología persistente, este trabajo se ha estructurado en cuatro partes.

La primera parte se centra en establecer las bases teóricas de la homología persistente, para lo cual se explora detalladamente la teoría de la homología simplicial. Esta área de la topología algebraica se desarrolló originalmente para estudiar el concepto de «agujero» en diversas dimensiones mediante el uso de simplices, que son estructuras que generalizan el concepto de triángulo a múltiples dimensiones. Los simplices suelen agruparse en lo que se conoce como complejos simpliciales, que son conjuntos de simplices que se combinan de manera que sus intersecciones cumplen ciertas propiedades. Estas estructuras forman parte de una familia más general, conocida como CW-complejos, que ofrecen un marco más flexible para la construcción de espacios topológicos a través de la unión de piezas llamadas celdas. También exploraremos la sucesión de Mayer-Vietoris, una herramienta poderosa en topología algebraica que permite descomponer espacios topológicos complejos en uniones de subespacios más simples, facilitando el cálculo de sus invariantes topológicos como los

módulos de homología y la relación de estos con conceptos como la conexión. Con todo esto podemos finalmente introducir la homología persistente y el Teorema de Correspondencia, el cual nos dará las herramientas necesarias que hoy emplea el TDA.

En la segunda parte de este trabajo, se realizará un estudio detallado sobre los principios del aprendizaje profundo que fundamentan las CNNs, comenzando con un repaso histórico desde las neuronas artificiales básicas hasta los modelos de CNN más avanzados en la actualidad. Esta sección abordará tanto las propiedades fundamentales de las CNNs como sus características específicas, mostrando cómo estas han evolucionado para proporcionar mejores y más eficientes predicciones en el ámbito de la clasificación de imágenes.

Finalmente introduciremos el TDA, explicando sus principios y técnicas, y exploraremos su aplicación en el ámbito de las CNNs para comprender cómo la estructura de los datos afecta el aprendizaje de la red. Esta exploración teórica establecerá la base para los estudios y experimentos realizados en este trabajo. Se expondrán los resultados y la metodología empleada en un exhaustivo estudio de la homología persistente y de las propuestas realizadas.

Para terminar, se presentan las conclusiones obtenidas y propuestas futuras para el estudio de la manera en que las CNNs cambian la «forma» de los datos.

## 4. Objetivos

El presente trabajo propone una serie de objetivos con el fin de profundizar en las bases de la homología persistente y realizar un estudio práctico en el ámbito de las CNNs:

1. En el ámbito de las **matemáticas** se propuso como objetivo principal profundizar en los contenidos de la topología algebraica y una introducción rigurosa a la homología persistente. Con dicho fin, la **Parte I** cumple con los siguientes puntos:
  - Se realiza una descripción de las principales herramientas algebraicas necesarias para el estudio de la homología simplicial y la homología persistente **Capítulo 1**.
  - Se estudian los complejos simpliciales, objetos de estudio de la homología simplicial. Además, se exploran otras estructuras topológicas relevantes como los CW-complejos y las variedades topológicas en el **Capítulo 2**.
  - Se introduce la homología simplicial y una de sus principales herramientas, la sucesión de Mayer-Vietoris, además de su relación con la conexión topológica en el **Capítulo 3**.
  - Se introduce el concepto de homología persistente y su principal resultado, el Teorema de correspondencia, descritos en el **Capítulo 4**.
2. Posteriormente, desde el ámbito del **aprendizaje profundo** se plantea comprender los principales modelos de CNNs en el ámbito de la clasificación de imágenes. Para ello, la **Parte II** explora los siguientes conceptos:
  - Se repasan los principales conceptos de inteligencia artificial, aprendizaje automático y visión artificial en el **Capítulo 5**.
  - Se realiza un repaso histórico y del estado del arte de las CNNs en los Capítulos **6** y **7**.
3. Finalmente, se propone realizar un estudio de las CNNs mediante el uso de técnicas de TDA y formalizar una propuesta en base a las conclusiones obtenidas. Para cumplir con estos objetivos, se han superados los siguientes hitos en la **Parte III**:

- Se analiza cómo transforman diferentes modelos y elementos de las CNNs la homología persistente de los datos en el Capítulo 10.
- Se propone un regularizador topológico con el objetivo de mejorar la tasa de clasificación y la transferibilidad de los modelos en los Capítulos 8 y 10.

## 5. Presupuesto

La primera consideración en el coste de la elaboración del estudio viene dada por la mano de obra empleada. El equipo empleado por el trabajador se trata de un ordenador portátil valorado en 600€ y una vida útil de 6 años. El proyecto se ha realizado durante un periodo de 10 meses y medio, donde en promedio se le ha dedicado 4 horas diarias en los días de semana, lo que se traduce en 20 horas semanales. Por otro lado, según el portal de transparencia empresarial Glassdoor<sup>1</sup>, el salario de un científico de datos promedio en España se comprende en el rango de los 30.000€ a 45.000€. Dado que el perfil del empleado es de junior, supondremos un salario de 30.000€ anuales, lo que implica un salario de 15€ por hora.

El coste derivado del entrenamiento de modelos de aprendizaje profundo suele ser elevado debido al consumo energético de las GPUs y el tiempo de entrenamiento necesario. Además, ha de tenerse en cuenta gastos derivados como mantenimiento y refrigeración. Dado que se han entrenado un total de 88 modelos con tiempos de entrenamiento medios de una hora, más algunas pruebas y experimentos, llegamos a la conclusión que el tiempo total de GPU empleado es de unas 100 horas. En particular, se han empleado dos Quadro RTX 8000, con un consumo de 260W por hora. Sabiendo que el precio de la luz en Granada ronda los 0,1€/kWh en promedio, nos queda el presupuesto total que figura en la Tabla 1.

Categoría	Costo
Mano de obra	13,640€
Amortización del equipo	87,50€
Coste de energía (GPU)	26€
Coste de mantenimiento y refrigeración	20€
<b>Total</b>	<b>13,773,50€</b>

Tabla 1.: Presupuesto detallado del estudio.

## 6. Planificación

La organización de este trabajo supuso un reto desde comienzos del curso 2023-2024. El hecho de tener que compaginar un curso completo con un trabajo de tal calibre requería de una organización minuciosa. Por ello, desde comienzos de septiembre de 2023 se comenzó a investigar y profundizar en los conceptos teóricos requeridos para tener una comprensión profunda de la homología persistente. Esta fase requirió de más tiempo del esperado, tal y como puede observarse en la Figura 1. La principal complicación surgió debido a la aparición de requisitos no contemplados en demostraciones puntuales que, junto a los exámenes, retrasó la finalización del marco teórico matemático.

<sup>1</sup>[https://www.glassdoor.es/Sueldos/data-scientist-sueldo-SRCH\\_KOo,14.htm](https://www.glassdoor.es/Sueldos/data-scientist-sueldo-SRCH_KOo,14.htm)

## 6. Planificación

La etapa final del desarrollo teórico se compaginó con reuniones con los tutores para ir concretando el camino de los experimentos y el planteamiento de hipótesis. Esto llevó a que la implementación del código necesario se realizara en su mayoría durante los meses de abril y mayo, que posteriormente iría siendo modificado en función de las necesidades de la investigación y los resultados obtenidos.

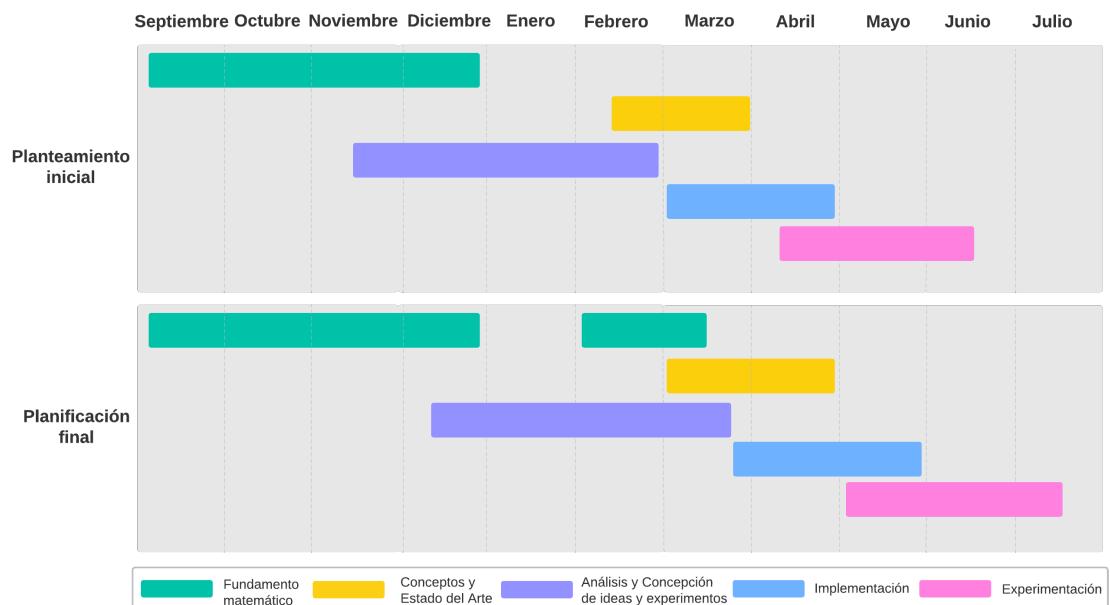


Figura 1.: Planificación temporal propuesta frente a la finalmente realizada.



**Parte I.**

**Fundamento matemático**



# 1. Fundamentos del álgebra homológica

La teoría de homología es una rama de la topología que trata de resolver problemas topológicos en el ámbito del álgebra. Por este motivo es importante conocer muy bien algunas herramientas algebraicas que iremos utilizando con frecuencia. En todo el capítulo usaremos como referencia principal [67].

## 1.1. Módulos

La estructura de módulo surge con la idea de generalizar el concepto de espacio vectorial sobre un cuerpo a un anillo. Nuestro interés en ellos radica en que la teoría de homología se construye sobre módulos y por ello es necesario hacer una introducción al campo. Esta sección recoge algunas definiciones y resultados de interés vistos en la asignatura de Álgebra Moderna y complementada con los contenidos de [31].

**Definición 1.1.** Sea  $R$  un anillo con elemento identidad  $1 \neq 0$ . Un  **$R$ -módulo izquierdo**  $M$  es un grupo abeliano aditivo junto con una función  $p : R \times M \rightarrow M$  con  $(r, m) \mapsto rm$  tal que dados  $r, r' \in R, m, m' \in M$  se tiene

1.  $(r + r')m = rm + r'm,$
2.  $(rr')m = r(r'm),$
3.  $r(m + m') = rm + rm',$
4.  $1m = m.$

De la definición anterior se sigue que  $0m = 0$  y  $(-1)m = -m$ .

De manera análoga, definimos  **$R$ -módulo derecho** donde el anillo actúa por la derecha en vez de por la izquierda de forma que  $p : M \times R \rightarrow M$ . Si  $R$  es un anillo comunitativo, los  $R$ -módulos izquierdos y derechos coinciden y les llamamos simplemente  $R$ -módulos. Como los resultados de  $R$ -módulos izquierdos y derechos son análogos, trabajaremos con los  $R$ -módulos izquierdos y nos referiremos a ellos como  **$R$ -módulos o módulos** a menos que se indique explícitamente lo contrario.

**Ejemplo 1.1.** El interés de los  $R$ -módulos subyace en la cantidad de estructuras conocidas que engloba. Si por ejemplo consideramos el  $K$ -módulo donde  $K$  es un cuerpo, éste adquiere la estructura de **espacio vectorial**. Ahora sea  $M$  un  $\mathbb{Z}$ -módulo. Definimos el producto  $p$  de forma que para  $n \in \mathbb{Z}$  y  $m \in M$  con  $n > 0$ ,  $nm = m + m + \dots + m$  ( $n$  veces),  $0m = 0$  y  $(-n)m = -(nm)$ . Entonces  $M$  ha de tener estructura de **grupo abeliano**. En particular, si  $R$  es un anillo entonces es también un  $R$ -módulo.

**Definición 1.2.** Sea  $M$  un  $R$ -módulo izquierdo y  $N$  un subconjunto de  $M$ . Diremos que  $N$  es un **submódulo** de  $M$ , esto es,  $N \subset M$ , si  $N$  es cerrado respecto a la suma y si  $r \in R, n \in N$  entonces  $rn \in N$ .

De la definición anterior se deduce que  $N$  es un  $R$ -módulo.

## 1. Fundamentos del álgebra homológica

**Definición 1.3.** Sea  $R$  un  $R$ -módulo. Si un submódulo de  $R$  es un subconjunto  $I \subset R$  cerrado respecto a la suma tal que  $\langle I \rangle = \{ri : i \in I\} \subset I$  para todo  $r \in R$ , lo llamaremos **ideal** de  $R$ . En particular, si  $I$  consta de un único elemento  $i \in I$ , diremos que es el **ideal generado por  $i$**  y lo denotaremos por  $\langle i \rangle$ .

**Definición 1.4.** Sea  $M$  un  $R$ -módulo y sea  $m \in M$ . El conjunto  $\langle m \rangle = \{rm : r \in R\}$  es un submódulo de  $M$  que denominaremos **submódulo cíclico generado por  $m$** .

*Observación 1.1.* Nótese que si  $R$  es un  $R$ -módulo, el submódulo cíclico generado por un elemento es el ideal generado por el mismo elemento.

**Definición 1.5.** Sea  $M$  un  $R$ -módulo y sean  $S$  un subconjunto de  $M$ . Sea  $\langle S \rangle$  el submódulo formado por la intersección de todos los submódulos de  $M$  que contienen a  $S$ . Diremos entonces que  $\langle S \rangle$  es el **submódulo generado por  $S$**  y los elementos de  $S$  los llamaremos **generadores** de  $\langle S \rangle$ .

**Definición 1.6.** Sea  $M$  un  $R$ -módulo. Un submódulo  $N$  de  $M$  es **finitamente generado** si existe un subconjunto finito  $S \subset M$  tal que  $N = \langle S \rangle$ .

**Lema 1.1.** *Sea  $M$  un  $R$ -módulo.  $M$  es finitamente generado por  $n$  elementos si, y sólo si, existe un epimorfismo  $\phi : R^n \rightarrow M$ .*

**Definición 1.7.** Sea  $M$  un  $R$ -módulo finitamente generado por  $n$  elementos y sea  $\phi : R^n \rightarrow M$  un epimorfismo. Diremos que  $M$  es **finitamente presentado** si  $\ker \phi$  es finitamente generado.

**Definición 1.8.** Sean  $M, N$   $R$ -módulos. Definimos el **homomorfismo de  $R$ -módulos** de  $M$  a  $N$  como la aplicación  $\alpha : M \rightarrow N$  tal que

1.  $\alpha(m + m') = \alpha(m) + \alpha(m')$ ,
2.  $\alpha(rm) = r\alpha(m)$

para todo  $m, m' \in M, r \in R$ .

Cuando  $\alpha : M \rightarrow N$  sea un homomorfismo de  $R$ -módulos, diremos que  $M$  es el **dominio** y  $N$  el **rango**. La **imagen** de  $\alpha$  es el conjunto  $\text{Im}(\alpha) = \{\alpha(m) : m \in M\}$ . El **núcleo** será el conjunto de elementos que se anulan en su imagen, esto es,  $\ker(\alpha) = \{m \in M : \alpha(m) = 0\}$ . Diremos que  $\alpha$  es un **epimorfismo** cuando  $\alpha$  sea sobreyectiva, un **monomorfismo** cuando  $\alpha$  sea inyectiva y un **isomorfismo** si  $\alpha$  es un epimorfismo y un monomorfismo a la vez. Si existe un isomorfismo entre  $M$  y  $N$  diremos que son **isomorfos** y lo notaremos  $A \cong B$ . Un homomorfismo  $\alpha : M \rightarrow M$  lo llamaremos **endomorfismo**.

Dado que el núcleo y la imagen de un homomorfismo de  $R$ -módulos coincide con el de los grupos abelianos subyacentes, la siguiente caracterización es inmediata de la ya conocida para grupos:

**Proposición 1.1.** *Sea  $\alpha : M \rightarrow N$  un homomorfismo de  $R$ -módulos. Entonces*

1.  $\alpha$  es un monomorfismo si, y sólo si,  $\ker(\alpha) = 0$ .
2.  $\alpha$  es un epimorfismo si, y sólo si,  $\text{Im}(\alpha) = N$ .

Es frecuente escribir el homomorfismo de  $R$ -módulos  $\alpha : M \rightarrow N$  como  $M \xrightarrow{\alpha} N$ . Respecto a la notación de la imagen de un elemento  $m \in M$  por  $\alpha$ , pondremos  $\alpha(m)$  o simplemente

$\alpha m$ . En cuanto a la imagen de  $A$  por  $\alpha$ , lo representaremos de manera análoga por  $\alpha(M)$  o  $\alpha M$ .

Dados dos homomorfismos de  $R$ -módulos  $\alpha_1, \alpha_2 : M \rightarrow N$ , su **suma**  $\alpha_1 + \alpha_2$  la definimos como  $(\alpha_1 + \alpha_2)(m) = \alpha_1(m) + \alpha_2(m)$  para todo  $m \in M$ . Además, dados dos homomorfismos de  $R$ -módulos  $\alpha : M \rightarrow N$ ,  $\beta : N \rightarrow P$ , su **composición**  $\beta \circ \alpha : M \rightarrow P$  es también un homomorfismo de  $R$ -módulos. Nótese que para que la composición sea posible, el rango de  $\alpha$  tiene que ser igual al dominio de  $\beta$ . En ocasiones usaremos la notación por yuxtaposición  $\alpha\beta = \alpha \circ \beta$ . Llamaremos **inversa** (por ambos lados) de  $\alpha : M \rightarrow N$  al homomorfismo  $\alpha^{-1} : N \rightarrow M$  tal que  $\alpha^{-1} \circ \alpha = \text{id}_M$  y  $\alpha \circ \alpha^{-1} = \text{id}_N$ . Una **inversa izquierda** de  $\alpha$  es una función  $\gamma : N \rightarrow M$  tal que  $\gamma \circ \alpha = \text{id}_M$ . De manera análoga, el homomorfismo  $\theta : M \rightarrow N$  es **inversa derecha** de  $\alpha$  si  $\alpha \circ \theta = \text{id}_N$ .

Si  $T \subseteq N$ , el conjunto  $\alpha^{-1}T = \{m \in M : \alpha(m) \in T\}$  es un submódulo de  $M$ , llamado la **imagen inversa** (completa) de  $T$ . En particular,  $\ker \alpha = \alpha^{-1}0$ , donde  $0$  denota el submódulo de  $N$  que consiste solo en el elemento cero.

Sea  $T \subseteq N$  donde  $N$  es un  $R$ -módulo, llamaremos **inclusión** o **inyección canónica** al homomorfismo  $i : T \rightarrow N$  tal que  $i(t) = t$  para todo  $t \in T$ . En particular,  $i$  es un monomorfismo. Las **clases laterales** de  $T$  en  $N$  son los conjuntos  $n + T = \{n + t : t \in T\}$  donde  $n \in N$ . Dos clases laterales  $n_1 + T, n_2 + T$  son iguales si  $n_1 - n_2 \in T$ . Como  $T$  es un submódulo, el grupo abeliano  $N/T$  se convierte en un  $R$ -módulo cuando  $r(n + T) = rn + T$  para todo  $r \in R$ . A este  $R$ -módulo lo llamaremos el **módulo cociente** de  $N$  sobre  $T$ . El homomorfismo  $\pi : N \rightarrow N/T$  tal que  $\pi(n) = n + T$  es un epimorfismo que llamaremos **proyección canónica** de  $N$ .

**Proposición 1.2** (Teorema de factorización). *Sea  $\beta : M \rightarrow M'$  un homomorfismo de módulos con  $T \subset \ker \beta$ . Existe entonces un único homomorfismo de módulos  $\beta' : M/T \rightarrow M'$  con  $\beta'\pi = \beta$ ; es decir, el siguiente diagrama con  $\beta(T) = 0$*

$$\begin{array}{ccc} N & \xrightarrow{\pi} & M/T \\ & \searrow \beta & \downarrow \beta' \\ & & M' \end{array}$$

es comutativo. Al homomorfismo  $\beta'$  lo llamaremos **homomorfismo inducido** por  $\beta$ .

**Teorema 1.1** (Primer teorema de isomorfía). *Sea  $\beta : M \rightarrow M'$  un homomorfismo de  $R$ -módulos. Entonces*

$$\frac{M}{\ker \beta} \cong \text{Im } \beta.$$

**Definición 1.9.** Sea  $\{M_i\}_{i \in I}$  una familia de  $R$ -módulos indexada por  $I$ . Definimos el **producto directo** o **producto directo externo** de  $\{M_i\}_{i \in I}$  como el producto cartesiano

$$\prod_{i \in I} M_i = \{(x_i)_{i \in I} : x_i \in M_i\},$$

donde las operaciones se definen componente a componente:

$$\begin{aligned} (x_i)_{i \in I} + (y_i)_{i \in I} &= (x_i + y_i)_{i \in I}, \\ r(x_i)_{i \in I} &= (rx_i)_{i \in I}, \end{aligned}$$

para todo  $r \in R$ ,  $x_i, y_i \in M_i$ ,  $i \in I$ .

## 1. Fundamentos del álgebra homológica

**Definición 1.10.** Sea  $\{M_i\}_{i \in I}$  una familia de  $R$ -módulos indexada por  $I$ . Definimos la **suma directa** o **suma directa interna** de  $\{M_i\}_{i \in I}$  como el submódulo de  $\prod_{i \in I} M_i$  tal que

$$\bigoplus_{i \in I} M_i = \{(x_i)_{i \in I} : x_i = 0 \text{ p.c.t. } i \in I\}.$$

*Nota.* Recordemos que una condición se cumple «para casi todo» (p.c.t.) elemento de un conjunto si se cumple para todo elemento en él salvo en un subconjunto finito de elementos.

**Definición 1.11.** Sea  $B$  un conjunto y sea  $M$  el  $R$ -módulo tal que  $M = \bigoplus_{b \in B} R_b$  donde  $R_b = R$  para todo  $b \in B$ . Llamaremos a dicho  $R$ -módulo el  **$R$ -módulo libre de base**  $B$  y lo notaremos por  $R\langle B \rangle$ . De esta forma, cada  $x \in R\langle B \rangle$  se representa por  $x = \sum_{b \in B} \lambda_b b$  donde  $\lambda_b \in R$  son coeficientes no nulos en un número finito de posiciones  $b$ .

**Teorema 1.2** (Propiedad universal de los módulos libres). *Sean  $B$  un conjunto,  $M$  un  $R$ -módulo y  $\varphi : B \rightarrow M$  una aplicación entre conjuntos. Entonces existe un único homomorfismo de  $R$ -módulos  $\phi : R\langle B \rangle \rightarrow M$  de forma que  $\phi(b) = \varphi(b)$  para todo  $b \in B$ . Es decir, el diagrama*

$$\begin{array}{ccc} B & \xrightarrow{\varphi} & M \\ \downarrow i & \nearrow \phi & \\ R\langle B \rangle & & \end{array}$$

commuta.

**Definición 1.12.** Sea  $M$  un  $R$ -módulo libre. Si para toda base  $B$  de  $M$ ,  $B$  tiene la misma cardinalidad, entonces decimos que  $M$  tiene **rango**  $\text{rg } M = \#B$ , donde  $\#B$  es la cardinalidad alguna base de  $M$ .

**Definición 1.13.** Sea  $x$  un elemento de un  $R$ -módulo. Decimos que  $x$  es un **elemento de torsión** si existe un  $r \in R \setminus \{0\}$  tal que  $rx = 0$ . Por otro lado,  $x$  es un **elemento sin torsión** si el único elemento  $r \in R$  que satisface  $rx = 0$  es  $r = 0$ . Un  $R$ -módulo se clasifica como **módulo de torsión** si cada uno de sus elementos es un elemento de torsión. Análogamente, un **módulo sin torsión** es aquel cuyos elementos no nulos son elementos sin torsión.

**Definición 1.14.** Sea  $M$  un  $R$ -módulo. Definimos el **anulador de  $M$**  como el submódulo  $\text{Ann}(M) = \{r \in R : rm = 0, \forall m \in M\}$ . De manera análoga, llamaremos **anulador de  $m \in M$**  al submódulo  $\text{Ann}(m) = \{r \in R : rm = 0, \forall m \in M\}$ .

**Definición 1.15.** Definimos el **submódulo de torsión** de un  $R$ -módulo  $M$  como el conjunto  $\text{Tor}(M) = \{x \in M : \text{Ann}(x) \neq \{0\}\}$ . Es decir, el conjunto de todos los elementos de torsión de  $M$ .

**Teorema 1.3** (Descomposición cíclica primaria). *Sea  $R$  un dominio de ideales principales y sea  $M$  un  $R$ -módulo finitamente generado. Entonces  $M$  se descompone como la suma directa*

$$M \cong R^f \oplus \bigoplus_{i=1}^k \frac{R}{\langle r_i \rangle}$$

donde  $R^f$  es un módulo libre de rango  $f$  y  $R/\langle r_1 \rangle, \dots, R/\langle r_k \rangle$  son módulos cíclicos con anuladores  $\langle r_1 \rangle, \dots, \langle r_k \rangle$ . Además,  $f$  y los ideales  $\langle r_1 \rangle, \dots, \langle r_k \rangle$  de  $R$  generados por  $r_1, \dots, r_k \in R$  están determinados de manera única salvo el orden por  $M$ .

**Ejemplo 1.2.** Consideremos un espacio vectorial  $V$ . Entonces podemos considerar, por ejemplo, su base canónica y generar todo  $V$  a partir de ella. En consecuencia,  $V$  es un módulo libre. Además, aplicando el teorema de **Descomposición cíclica primaria**, es claro que la parte libre de la descomposición es isomorfa a  $V$  y por tanto  $V$  carece de parte cíclica.

## 1.2. Sucesiones exactas

**Definición 1.16.** Sea  $\{A_i, \alpha_i\}_{i \in \mathbb{Z}}$  una familia de  $R$ -módulos  $A_i$  y homomorfismos entre ellos tal que  $\alpha_i : A_i \rightarrow A_{i+1}$ . Diremos que la sucesión

$$\dots \xrightarrow{\alpha_{i-2}} A_{i-1} \xrightarrow{\alpha_{i-1}} A_i \xrightarrow{\alpha_i} A_{i+1} \xrightarrow{\alpha_{i+1}} \dots$$

es **exacta** en  $A_i$  cuando  $\text{Im } \alpha_i = \ker \alpha_{i+1}$ . Si la sucesión es exacta para todo  $i \in \mathbb{Z}$ , diremos que la sucesión es **exacta larga** o simplemente **exacta**.

**Definición 1.17.** Sean  $A, B$  y  $C$   $R$ -módulos y  $\sigma : A \rightarrow B$ ,  $\gamma : B \rightarrow C$  homomorfismos entre ellos. Diremos que la **sucesión exacta** es **corta** si

$$(\sigma, \gamma) : 0 \rightarrow A \xrightarrow{\sigma} B \xrightarrow{\gamma} C \rightarrow 0.$$

Es decir, una sucesión exacta de cinco  $R$ -módulos con los dos módulos exteriores siendo cero (y por lo tanto las dos funciones exteriores triviales).

**Proposición 1.3.** Sean  $A, B$  y  $C$   $R$ -módulos y  $\sigma : A \rightarrow B$ ,  $\gamma : B \rightarrow C$  homomorfismos entre ellos. Entonces

1. La sucesión  $0 \rightarrow A \xrightarrow{\sigma} B$  es exacta (en  $A$ ) si, y sólo si,  $\sigma$  es inyectiva.
2. La sucesión  $B \rightarrow C \xrightarrow{\gamma} 0$  es exacta (en  $C$ ) si, y sólo si,  $\gamma$  es sobreyectiva.

**Demostración.** El único homomorfismo que cumple  $0 \rightarrow A$  tiene imagen 0 en  $A$  y por tanto, el núcleo de  $\sigma$  será este si, y sólo si,  $\sigma$  es inyectiva. De manera similar, el único homomorfismo  $C \rightarrow 0$  es el homomorfismo nulo para todo elemento de  $C$ , que es la imagen de  $\gamma$  si, y sólo si,  $\gamma$  es sobreyectiva.  $\square$

**Corolario 1.1.** La sucesión  $0 \rightarrow A \xrightarrow{\sigma} B \xrightarrow{\gamma} C \rightarrow 0$  es exacta si, y sólo si,  $\sigma$  es inyectiva,  $\gamma$  es sobreyectiva y  $\text{Im } \sigma = \ker \gamma$ .

Como acabamos de probar, la exactitud en  $A$  significa que  $\sigma$  es un monomorfismo, en  $B$  significa que  $\sigma A = \ker \gamma$  y en  $C$  que  $\gamma$  es un epimorfismo. Así la sucesión exacta corta puede escribirse como  $A \xrightarrow{\sigma} B \xrightarrow{\gamma} C$ , con exactitud en  $B$ . Ahora  $\sigma$  induce un isomorfismo  $\sigma' : A \rightarrow \sigma A$  y  $\gamma$  un isomorfismo  $\gamma' : B/\sigma A \rightarrow C$ ; juntos estos proveen un isomorfismo de sucesiones exactas cortas, en la forma de un diagrama comutativo

$$\begin{array}{ccccccc} 0 & \longrightarrow & A & \xrightarrow{\sigma} & B & \xrightarrow{\gamma} & C \longrightarrow 0 \\ & & \downarrow \sigma' & & \parallel & & \downarrow (\gamma')^{-1} \\ 0 & \longrightarrow & \sigma A & \xrightarrow{i} & B & \longrightarrow & B/\sigma A \longrightarrow 0. \end{array}$$

## 1. Fundamentos del álgebra homológica

En resumen, una sucesión exacta corta es simplemente otro nombre para un submódulo y su cociente.

**Ejemplo 1.3.** Respecto al [Teorema de factorización](#), la inclusión  $i$  y la proyección  $\pi$  producen una sucesión exacta corta.

$$0 \rightarrow T \xrightarrow{i} M \xrightarrow{\pi} M/T \rightarrow 0.$$

## 1.3. Categorías y funtores

La teoría de categorías fue introducida por primera vez por Samuel Eilenberg y Saunders MacLane en [34]. En particular, las categorías son estructuras algebraicas que capturan la noción de composición. Gracias a ellas podemos analizar y comparar estructuras algebraicas, permitiendo sacar conclusiones comunes y trasladar problemas complejos a otros espacios donde resolverlos es más sencillo. En esta sección haré una breve introducción de las categorías apoyándome en [66].

**Definición 1.18.** Una **categoría**  $\mathcal{C}$  es una tripleta  $(\mathcal{O}, \text{hom}, \circ)$  formada por:

1. Una clase  $\mathcal{O}$ , cuyos elementos denominamos **objetos** de  $\mathcal{C}$  y notamos por  $\text{Obj}(\mathcal{C})$ .
2. Por cada par de objetos  $(A, B)$  de  $\mathcal{C}$ , un conjunto  $\text{hom}(A, B)$  cuyos elementos son llamados **morfismos** de  $A$  a  $B$ . Si  $f \in \text{hom}(A, B)$ , normalmente escribiremos  $f : A \rightarrow B$  o  $A \xrightarrow{f} B$ .
3. Una **ley de composición** que asocia a cada morfismo  $f : A \rightarrow B$  y a cada morfismo  $g : B \rightarrow C$  un morfismo  $g \circ f : A \rightarrow C$  que satisface
  - **Asociatividad.** Si  $f : A \rightarrow B$ ,  $g : B \rightarrow C$  y  $h : C \rightarrow D$  son morfismos de  $\mathcal{C}$ , entonces  $h \circ (g \circ f) = (h \circ g) \circ f$ .
  - **Identidad.** A cada objeto  $B$  le podemos asociar un morfismo identidad  $\text{id}_B : B \rightarrow B$  tal que si  $f : A \rightarrow B$  y  $g : B \rightarrow C$  entonces  $g \circ \text{id}_B = g$  y  $\text{id}_B \circ f = f$ .

Llamaremos a este morfismo la **composición** de  $f$  y  $g$ .

**Ejemplo 1.4.** Como veremos a continuación, la definición anterior nos va a permitir trabajar con un gran número de espacios matemáticos que ya conocemos en el contexto de la teoría de categorías. Algunos de ellos son:

- **La categoría de conjuntos Set**, cuyos objetos son todos los conjuntos y sus morfismos todas las aplicaciones entre conjuntos.
- **La categoría de grupos Grp**, donde los objetos son todos los grupos y los morfismos todos los homomorfismos de grupos.
- **La categoría de espacios topológicos Top**, donde los objetos son todos los espacios topológicos y los morfismos todas las aplicaciones continuas entre espacios topológicos  $f : X \rightarrow Y$ .
- **La categoría de  $R$ -módulos  $R\text{-Mod}$** , donde los objetos son todos los  $R$ -módulos y los morfismos todos los homomorfismos de módulos.

- **La categoría de sucesiones exactas de  $R$ -módulos de longitud  $n$ .** Los objetos son dichas sucesiones  $S : A_1 \rightarrow \cdots \rightarrow A_n$ . Para dos sucesiones  $S$  y  $S'$ , los morfismos son de la forma  $\Gamma : S \rightarrow S'$  tal que  $\Gamma = (\gamma_1, \dots, \gamma_n)$  es una tupla donde los  $\gamma_i : A_i \rightarrow A'_i$  son homomorfismos de  $R$ -módulos tal que

$$\begin{array}{ccccccc} A_1 & \longrightarrow & A_2 & \longrightarrow & \cdots & \longrightarrow & A_{n-1} \longrightarrow A_n \\ \gamma_1 \downarrow & & \gamma_2 \downarrow & & & & \gamma_{n-1} \downarrow \\ A'_1 & \longrightarrow & A'_2 & \longrightarrow & \cdots & \longrightarrow & A'_{n-1} \longrightarrow A'_n \end{array}$$

comuta para todo  $i \in \{1, \dots, n\}$ .

También es posible encontrar colecciones propias de una categoría que siguen manteniendo dicha estructura.

**Definición 1.19.** Una **subcategoría**  $\mathcal{D}$  de una categoría  $\mathcal{C}$  consiste en:

1. Una subclase de objetos de  $\mathcal{C}$ , denotada por  $Obj(\mathcal{D})$ , tal que cada objeto de  $\mathcal{D}$  es también un objeto de  $\mathcal{C}$ .
2. Para cada par de objetos  $A, B \in Obj(\mathcal{D})$ , un subconjunto de morfismos  $hom_{\mathcal{D}}(A, B) \subseteq hom_{\mathcal{C}}(A, B)$  cuyos elementos son también morfismos en  $\mathcal{C}$ .
3. La ley de composición en  $\mathcal{D}$  es inducida por la ley de composición en  $\mathcal{C}$ , y esta composición es cerrada en  $\mathcal{D}$ .

Además, diremos que  $\mathcal{D}$  es una **subcategoría plena** si para cada par de objetos  $A, B \in Obj(\mathcal{D})$ ,  $hom_{\mathcal{D}}(A, B) = hom_{\mathcal{C}}(A, B)$ .

**Definición 1.20.** Sea  $f \in hom(A, B)$  un morfismo en la categoría  $\mathcal{C}$ . Diremos que  $f$  es una **equivalencia** en  $\mathcal{C}$  si existe en  $\mathcal{C}$  otro morfismo  $g \in hom(B, A)$  tal que  $g \circ f = id_A$  y  $f \circ g = id_B$ .

*Observación 1.2.* Nótese que si  $f \in hom(A, B)$  es una equivalencia en  $\mathcal{C}$ ,  $g \in hom(B, A)$  debe ser única. En efecto, si suponemos que existe  $g' \in hom(B, A)$  tal que  $g' \circ f = id_A$ , entonces  $g = g' \circ f \circ g = g' \circ id_B = g'$ .

Dentro de la teoría de categorías, los funtores tienen un papel principal, pues nos van a permitir llevar objetos y morfismos de una categoría a otra preservando identidades y composiciones.

**Definición 1.21.** Sean  $\mathcal{C}, \mathcal{D}$  dos categorías. Un **funtor covariante** de  $\mathcal{C}$  a  $\mathcal{D}$  es una pareja de funciones denotadas por la misma letra  $T$  tal que:

1. Una **función objeto** que asigna a cada objeto  $C \in \mathcal{C}$  un objeto  $T(C) \in \mathcal{D}$ .
2. Una **función de morfismos** que asigna a cada morfismo  $\gamma : C \rightarrow C'$  de  $\mathcal{C}$  un morfismo  $T(\gamma) : T(C) \rightarrow T(C')$  de  $\mathcal{D}$ . Este par de funciones satisfacen las siguientes condiciones:

$$T(1_C) = id_{T(C)}, \quad C \in \mathcal{C},$$

$$T(\beta\gamma) = T(\beta)T(\gamma), \quad \beta\gamma \text{ definido en } \mathcal{C}.$$

## 1. Fundamentos del álgebra homológica

Es decir, un funtor covariante  $T : \mathcal{C} \rightarrow \mathcal{D}$  es una aplicación que preserva el rango, dominio, identidades y composiciones de  $\mathcal{C}$  en  $\mathcal{D}$ .

**Definición 1.22.** Sean  $\mathcal{C}, \mathcal{D}$  dos categorías. Diremos que  $\mathcal{C}$  y  $\mathcal{D}$  son **isomorfas** si existen funtores covariantes  $F : \mathcal{C} \rightarrow \mathcal{D}$  y  $G : \mathcal{D} \rightarrow \mathcal{C}$  tales que  $G \circ F = \text{id}_{\mathcal{C}}$  y  $F \circ G = \text{id}_{\mathcal{D}}$ .

## 1.4. Módulos diferenciales

Comenzaremos definiendo lo que es un módulo de homología y estableceremos la terminología que emplearemos cuando trabajemos con ellos.

**Definición 1.23.** Sea  $C$  un  $R$ -módulo junto a un endomorfismo  $d : C \rightarrow C$  tal que  $d^2 = d \circ d = 0$ . Diremos entonces que  $C$  es un **módulo diferencial** y llamaremos a  $d$  **operador borde** de  $C$ .

Llamaremos a los elementos de  $C$  **cadenas**. El submódulo de **ciclos** será  $Z(C) = \ker d$ , y el submódulo de **bordes**  $B(C) = \text{Im } d$ . Si nos fijamos, el requisito  $d^2 = 0$  es equivalente a exigir que  $\text{Im } d \subset \ker d$ .

**Definición 1.24.** Sea  $C$  un módulo diferencial. Definimos el  **$R$ -módulo de homología** de  $C$  como el módulo cociente  $H(C; R)$  tal que

$$H(C; R) = \frac{Z(C)}{B(C)}$$

En particular, cuando  $C$  sea un  $\mathbb{Z}$ -módulo diferencial, lo llamaremos **grupo diferencial** y notaremos  $H(C; \mathbb{Z})$  simplemente por  $H(C)$ .

Por tanto, el módulo de homología de un módulo diferencial  $C$  está formado por las clases laterales  $[c] = c + B(C)$  donde  $c$  es un ciclo de  $C$ . A los elementos de  $H(C; R)$  los llamaremos **clases de homología**. Dos ciclos  $c$  y  $c'$  diremos que son **homólogos** si ambos pertenecen a la misma clase de homología, esto es,  $c \sim c'$ .

**Definición 1.25.** Sean  $C$  y  $C'$  dos módulos diferenciales y  $d, d'$  sus respectivos operadores borde. Diremos que  $f : C \rightarrow C'$  es un **homomorfismo de módulos diferenciales** si  $f$  es un homomorfismo de módulos y además  $d'f = fd$ .

La anterior definición nos permite preservar la estructura algebraica del módulo diferencial. De esta forma, si tomamos una cadena  $c \in C$ , que sea un ciclo o un borde, y  $f : C \rightarrow C'$  es un homomorfismo de módulos diferenciales,  $f(c) \in C'$  seguirá siendo un ciclo o un borde de manera correspondiente. En efecto, si  $z \in Z(C)$ , entonces

$$d'f(z) = f(dz) = f(0) = 0.$$

Esto es,  $f(z) \in \ker d'$ . Ahora, si  $b \in B(C)$ , entonces existe  $c \in C$  tal que  $dc = b$ . En consecuencia,

$$d'f(c) = f(dc) = f(b),$$

y por tanto,  $f(b) \in \text{im } d'$ .

Los grupos diferenciales definen una categoría  $\mathcal{C}$  donde los objetos son los módulos diferenciales y los morfismos son los homomorfismos de módulos diferenciales. Tomemos como ley de composición interna la composición de dichos homomorfismos. Claramente es

asociativa pues si  $C, C', \bar{C}, \tilde{C} \in Obj(\mathcal{C})$ , y  $f: C \rightarrow C', g: C' \rightarrow \bar{C}, h: \bar{C} \rightarrow \tilde{C}$ , entonces  $h \circ (g \circ f)$  se cumple si, y sólo si,

$$\begin{aligned}\tilde{d}(h \circ (g \circ f)) &= (\tilde{d}h) \circ (g \circ f) = (h\bar{d}) \circ (g \circ f) = h \circ (\bar{d}g) \circ f \\ &= h \circ (gd') \circ f = h \circ g \circ (d'f) = h \circ g \circ (fd) = (h \circ g) \circ fd \\ &= ((h \circ g) \circ f)d\end{aligned}$$

y por tanto  $h \circ (g \circ f) = (h \circ g) \circ f$ . La propiedad de identidad se sigue de existir el homomorfismo identidad de módulos.

**Definición 1.26.** Sean  $C, C'$  módulos diferenciales y  $f: C \rightarrow C'$  un homomorfismo de módulos diferenciales. Definimos la función  $f_* = H(f): H(C; R) \rightarrow H(C'; R)$  tal que

$$f_*([c]) = [f(c)]$$

Diremos que  $H(f)$  es el **homomorfismo inducido** por  $f$ .

**Proposición 1.4.** En estas condiciones,  $H$  es un funtor covariante de la categoría de módulos diferenciales a la categoría de módulos.

*Demostración.* Por la definición dada del módulo de homología, es claro que la función objeto  $H$  asigna a cada grupo diferencial  $C$  un grupo de homología  $H(C; R)$ . En cuanto a la función de morfismos, la identidad de grupos diferenciales se preserva pues  $H(id)([c]) = id_*([c]) = [id(c)] = [c]$  para todo  $c \in C$ . Además, si  $f, g \in \text{hom}(C)$ , entonces

$$\begin{aligned}H(g \circ f)([c]) &= (g \circ f)_*([c]) = [(g \circ f)(c)] = [g(f(c))] \\ &= g_*([f(c)]) = g_*(f_*([c])) = (H(g) \circ H(f))([c])\end{aligned}$$

para todo  $c \in C$ , manteniendo la ley de composición.  $\square$

## 1.5. Complejos de cadenas

**Definición 1.27.** Sea  $R$  un anillo. Un **complejo de cadenas**  $C_\bullet$  de  $R$ -módulos es una familia  $\{C_n, \partial_n\}$  donde  $C_n$  son  $R$ -módulos y  $\partial_n: C_n \rightarrow C_{n-1}$  homomorfismos de  $R$ -módulos tales que  $\partial_n \partial_{n+1} = 0$  para todo  $n \in \mathbb{Z}$ .

*Nota.* Usualmente notaremos directamente  $\partial$  al homomorfismo  $\partial_n$  independientemente del valor de  $n$  siempre y cuando se sobrentienda por el contexto.

*Observación 1.3.* La última condición es equivalente a que  $\text{Im } \partial_{n+1} \subset \ker \partial_n$ .

Un complejo  $C_\bullet$  es por tanto una sucesión doblemente infinita

$$C_\bullet: \dots \rightarrow C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} C_{-1} \rightarrow \dots$$

donde toda composición de homomorfismos de dicha familia es el homomorfismo nulo. La **homología**  $H(C_\bullet)$  es la familia de  $R$ -módulos

$$H_n(C_\bullet) = \frac{\ker \partial_n}{\text{Im } \partial_{n+1}}$$

## 1. Fundamentos del álgebra homológica

donde  $H_n(C_\bullet)$  es el  $n$ -ésimo módulo de homología de  $C_\bullet$ .

Luego  $H_n(C_\bullet) = 0$  implica que la sucesión  $C_\bullet$  es exacta en  $C_n$ . A los elementos de  $C_n$  los llamaremos **n-cadenas** o **cadenas de dimensión n**. Un **n-ciclo** o **ciclo de dimensión n** de  $C_\bullet$  es un elemento del submódulo  $Z_n(C_\bullet) = \ker \partial_n$ . Un **n-borde** o **borde de dimensión n** es un elemento de  $B_n(C_\bullet) = \text{Im } \partial_{n+1}$ . La clase lateral de un ciclo  $c$  la notaremos por  $[c] = c + \partial_{n+a} C_{n+1}$ . Dos  $n$ -ciclos  $c, c' \in C_n$  pertenecientes a la misma clase lateral  $[c]$  decimos que son **homólogos**, es decir,  $c \sim c'$ .

*Nota.* Si la dimensión se sobrentiende en estos casos, no la indicaremos de manera explícita.

**Definición 1.28.** Sea  $\{C_\bullet^i, \partial^i\}_{i \in I}$  una familia de complejos de cadenas. Su **suma directa** la definimos como el complejo de cadenas  $\bigoplus_{i \in I} C_\bullet^i$  cuyos operadores borde vienen dados por  $\bigoplus_{i \in I} \partial_n^i : \bigoplus_{i \in I} C_n^i \rightarrow \bigoplus_{i \in I} C_{n-1}^i$  para todo  $n \in \mathbb{Z}$ .

**Proposición 1.5.** Sea  $\{C_\bullet^i, \partial^i\}_{i \in I}$  una familia de complejos de cadenas. Entonces su homología conmuta con la suma directa, esto es,  $H_n(\bigoplus_{i \in I} C_\bullet^i) \cong \bigoplus_{i \in I} H_n(C_\bullet^i)$  para todo  $n \in \mathbb{Z}$ .

*Demostración.* Para demostrar que la homología conmuta con sumas directas, queremos mostrar que para una colección de complejos de cadenas  $\{C_\bullet^i, \partial^i\}_{i \in I}$ , los homomorfismos

$$\begin{aligned}\phi : H_n \left( \bigoplus_{i \in I} C_\bullet^i \right) &\rightarrow \bigoplus_{i \in I} H_n(C_\bullet^i) : [(c_i)] \mapsto ([c_i]), \\ \psi : \bigoplus_{i \in I} H_n(C_\bullet^i) &\rightarrow H_n \left( \bigoplus_{i \in I} C_\bullet^i \right) : ([c_i]) \mapsto [(c_i)],\end{aligned}$$

están bien definidos y son inversos uno del otro.

En primer lugar, para comprobar que dichas aplicaciones están bien definidas, observemos que  $[(c_i)] = [(c'_i)]$  si, y sólo si,  $[0] = [(c_i - c'_i)]$ . Esto ocurre si, y sólo si, existe un  $b_i \in C_\bullet^i$  tal que  $\partial_i(b_i) = (c_i - c'_i)$ , lo cual es equivalente a  $c_i + \partial_i(b_i) = c'_i$  para cada  $i \in I$ . Por lo tanto,  $[(c_i)] = [(c'_i)]$  si, y sólo si,  $\phi([(c_i)]) = \phi([(c'_i)]) = [(c'_i + \partial_i(b_i))] = [(c'_i)]$ . De manera análoga,  $[(c_i)] = [(c'_i)]$  si, y sólo si,  $\psi([(c_i)]) = \psi([(c'_i)])$ . Esto implica que tanto  $\phi$  como  $\psi$  están bien definidos.

En segundo lugar, es claro que  $\phi$  y  $\psi$  son homomorfismos de  $R$ -módulos. Además,  $\phi$  lleva la clase de equivalencia  $[(c_i)]$  a  $([c_i])$ , mientras que  $\psi$  lleva  $([c_i])$  a  $[(c_i)]$ , lo que demuestra que son inversos el uno del otro.  $\square$

**Definición 1.29.** Sean  $C_\bullet, C'_\bullet$  complejos de cadenas. Una **aplicación de cadenas** o **morfismo de cadenas**  $f : C_\bullet \rightarrow C'_\bullet$  es una familia de homomorfismos de  $R$ -módulos  $f_n : C_n \rightarrow C'_n$  tal que  $\partial'_n f_n = f_{n-1} \partial_n$  para todo  $n \in \mathbb{Z}$ .

$$\begin{array}{ccccccc} \cdots & \longrightarrow & C_{n+1} & \xrightarrow{\partial_{n+1}} & C_n & \xrightarrow{\partial_n} & C_{n-1} \longrightarrow \cdots \\ & & \downarrow f_{n+1} & & \downarrow f_n & & \downarrow f_{n-1} \\ \cdots & \longrightarrow & C'_{n+1} & \xrightarrow{\partial'_{n+1}} & C'_n & \xrightarrow{\partial'_n} & C'_{n-1} \longrightarrow \cdots \end{array}$$

Cuando se sobrentienda del contexto, notaremos simplemente por  $\partial$  a los correspondientes  $\partial_n$  y  $\partial'_n$ .

Los complejos de cadenas, junto con sus morfismos, forman una categoría que denotaremos por  $R\text{-Ch}_\bullet$ . Esto se debe a que existe la identidad y se cumple la propiedad de asociatividad

para la composición de morfismos. En efecto, para tres morfismos  $f : C_\bullet \rightarrow C'_\bullet$ ,  $g : C'_\bullet \rightarrow C''_\bullet$  y  $h : C''_\bullet \rightarrow \bar{C}_\bullet$ , se tiene que

$$h_n \circ (g_n \circ f_n) = (h_n \circ g_n) \circ f_n$$

para cada  $n \in \mathbb{Z}$ , ya que son homomorfismos de módulos diferenciales. En consecuencia,  $h \circ (g \circ f) = (h \circ g) \circ f$ .

Sea  $f : C_\bullet \rightarrow C'_\bullet$  un morfismo de complejos de cadenas. Definimos  $H_n(f) = f_* : H_n(C_\bullet) \rightarrow H_n(C'_\bullet)$  dado que

$$f_*([c]) = f_*(c + \partial C_{n+1}) = fc + \partial C'_{n+1} = [f(c)].$$

Entonces  $f_*$  es un homomorfismo de  $R$ -módulos, como recoge el siguiente resultado.

**Proposición 1.6.** *Cada  $H_n$  es un funtor covariante de la categoría de complejos de cadenas y morfismos de cadenas a la categoría de  $R$ -módulos.*

*Demostración.* Como vimos anteriormente,  $H_n$  asigna a cada complejo de cadenas  $C_\bullet$  un  $R$ -módulo. Para demostrar que  $H_n$  también asigna a cada morfismo de cadenas un homomorfismo de  $R$ -módulos, consideremos  $[c], [c'] \in H_n(C_\bullet)$  y  $r, s \in R$ . Entonces, podemos ver que

$$\begin{aligned} f_*(r[c] + s[c']) &= f_*(r(c + \partial C_{n+1}) + s(c' + \partial C_{n+1})) \\ &= f_*(rc + \partial C_{n+1} + sc' + \partial C_{n+1}) \\ &= f_*(rc + sc' + \partial C_{n+1}) \\ &= f(rc + sc') + \partial C'_{n+1} \\ &= rf(c) + sf(c') + \partial C'_{n+1} \\ &= r(f(c) + \partial C'_{n+1}) + s(f(c') + \partial C'_{n+1}) \\ &= r(f_*(c + \partial C_{n+1})) + s(f_*(c' + \partial C_{n+1})) \\ &= rf_*([c]) + sf_*([c']). \end{aligned}$$

Además, si consideramos  $f = \text{id}$  la identidad, es claro que  $\text{id}_*$  es la identidad de  $R$ -módulos.  $\square$

**Definición 1.30.** Sean  $C_\bullet, C'_\bullet$  complejos de cadenas y  $f, g : C_\bullet \rightarrow C'_\bullet$  dos aplicaciones de cadenas entre ellos. Una **homotopía de cadenas** u **homotopía algebraica**  $s$  es una familia de homomorfismos de módulos  $s_n : C_n \rightarrow C'_{n+1}$  para cada  $n \in \mathbb{Z}$  tal que

$$\partial'_{n+1}s_n + s_{n-1}\partial_n = f_n - g_n.$$

Diremos entonces que  $f$  y  $g$  son **algebraicamente homotópicas** y escribiremos  $f \simeq g$ .

**Teorema 1.4.** *Si  $s$  es una homotopía de cadenas entre  $f, g : C_\bullet \rightarrow C'_\bullet$ , entonces*

$$H_n(f) = H_n(g) : H_n(C_\bullet) \rightarrow H_n(C'_\bullet).$$

*Demostración.* Si  $c$  es un ciclo de  $C_n$ , tenemos que  $\partial_n c = 0$ . Por la Def. 1.30 se cumple que  $f_n c - g_n c = \partial s_n c$ . Como consecuencia  $f_n c$  y  $g_n c$  son homólogos lo que implica que  $[f_n c] = [g_n c]$  en  $H_n(C'_\bullet)$ , como queríamos demostrar.  $\square$

## 1. Fundamentos del álgebra homológica

**Definición 1.31.** Una aplicación de cadenas  $f : C_\bullet \rightarrow C'_\bullet$  es una **equivalencia de cadenas** si existe otra aplicación  $h : C'_\bullet \rightarrow C_\bullet$  y homotopías  $s : h \circ f \rightarrow \text{id}_{C_\bullet}$ ,  $t : f \circ h \rightarrow \text{id}'_{C'_\bullet}$  tales que  $h \circ f \simeq \text{id}_{C_\bullet}$ ,  $f \circ h \simeq \text{id}'_{C'_\bullet}$ .

Como  $H_n(\text{id}_{C_\bullet}) = \text{id}_{H_n(C_\bullet)}$ , del anterior teorema se deduce lo siguiente.

**Corolario 1.2.** Si  $f : C_\bullet \rightarrow C'_\bullet$  es una equivalencia de cadenas, la aplicación inducida  $H_n(f) : H_n(C_\bullet) \rightarrow H_n(C'_\bullet)$  es un isomorfismo para cada  $n \in \mathbb{Z}$ .

**Proposición 1.7.** Sean  $f, g : C_\bullet \rightarrow C'_\bullet$  y  $f', g' : C'_\bullet \rightarrow C''_\bullet$  aplicaciones de cadenas. Sean  $s : f \rightarrow g$ ,  $s' : f' \rightarrow g'$  homotopías de cadenas entre ellas tales que  $f \simeq g$ ,  $f' \simeq g'$ . Entonces la composición

$$f's + s'g : f' \circ f \rightarrow g' \circ g \quad g' \circ g : C_\bullet \rightarrow C''_\bullet$$

es una homotopía de cadenas.

*Demostración.* Por ser  $s, s'$  homotopías de cadenas tenemos que  $\partial s + s\partial = f - g$  y  $\partial s' + s'\partial = f' - g'$ . Aplicando  $f'$  a la izquierda de la primera expresión y  $g$  a la derecha de la segunda nos queda

$$\begin{cases} f'\partial s + f's\partial = f' \circ f - f' \circ g, \\ \partial s'g + s'\partial g = f' \circ g - g' \circ g. \end{cases}$$

Sumando ambas igualdades

$$\begin{aligned} f'\partial s + f's\partial + \partial s'g + s'\partial g &= f' \circ f - f' \circ g + f' \circ g - g' \circ g, \\ f'\partial s + f's\partial + \partial s'g + s'\partial g &= f' \circ f - g' \circ g, \\ \partial f's + f's\partial + \partial s'g + s'\partial g &= f' \circ f - g' \circ g, \end{aligned}$$

donde finalmente queda

$$\partial(f's + s'g) + (f's + s'g)\partial = f' \circ f - g' \circ g.$$

□

## 1.6. Subcomplejos y complejos cociente

**Definición 1.32.** Un **subcomplejo**  $S_\bullet$  de  $C_\bullet$  es una familia de submódulos  $S_n \subset C_n$  tal que para cada  $n \in \mathbb{Z}$ ,  $\partial S_n \subset S_{n-1}$ .

Por tanto,  $S_\bullet$  es un complejo en sí con el operador borde  $\partial$  inducido de  $C_\bullet$  y la inclusión  $i : S_\bullet \rightarrow C_\bullet$  es una aplicación de cadenas.

**Definición 1.33.** Sea  $S_\bullet$  un subcomplejo de  $C_\bullet$ . El **complejo cociente**  $C_\bullet/S_\bullet$  es la familia  $(C_\bullet/S_\bullet)_n = C_n/S_n$  de módulos cocientes con operador borde  $\partial'_n : C_n/S_n \rightarrow C_{n-1}/S_{n-1}$  inducido por  $\partial_{C_\bullet}$ .

**Definición 1.34.** Sean  $f : C_\bullet \rightarrow C'_\bullet$ ,  $g : C'_\bullet \rightarrow C''_\bullet$  aplicaciones de cadenas. La sucesión de complejos

$$C_\bullet \xrightarrow{f} C'_\bullet \xrightarrow{g} C''_\bullet$$

es **exacta** en  $C'_\bullet$  si  $\text{Im}(f) = \ker(g)$ ; es decir, si cada sucesión  $C_n \xrightarrow{f_n} C'_n \xrightarrow{g_n} C''_n$  de módulos es exacta en  $C'_n$ .

**Definición 1.35.** Un complejo  $C_\bullet$  es **positivo** si  $C_n = 0$  para todo  $n < 0$  con  $n \in \mathbb{Z}$ . Su  $n$ -ésimo módulo de homología es entonces positivo ya que  $H_n(C_\bullet) = 0$  para todo  $n < 0$ . De manera análoga, un complejo  $C_\bullet$  es **negativo** si  $C_n = 0$  para todo  $n > 0$  con  $n \in \mathbb{Z}$ .

**Definición 1.36.** Sea  $C_\bullet$  un complejo positivo de  $R$ -módulos. Denominaremos **aumento de  $C_\bullet$**  al homomorfismo sobreyectivo  $\varepsilon : C_0 \rightarrow R$  de forma que  $\varepsilon \circ \partial_1 = 0$ .

**Definición 1.37.** Sea  $C_\bullet$  un complejo de cadenas positivo,  $\varepsilon : C_0 \rightarrow R$  un aumento de  $C_\bullet$  y sea  $n \in \mathbb{Z}$ . Consideremos el complejo positivo  $\tilde{C}_\bullet$  tal que  $\tilde{C}_n = C_n$  para todo  $n \geq 0$ ,  $\tilde{C}_n = 0$  para todo  $n < -1$  y  $\tilde{C}_{-1} = R$ . Consideremos también  $\tilde{\partial}_n = \partial_n$  para todo  $n \geq 1$  y  $\tilde{\partial}_0 = \varepsilon$ . Llamaremos a este complejo **complejo aumentado** de  $C_\bullet$ .

**Definición 1.38.** Sea  $A$  un módulo. Definimos el siguiente complejo positivo donde  $A_0 = A$ ,  $A_n = 0$  para  $n \neq 0$  y  $\partial = 0$ . Un **complejo sobre  $A$**  es un complejo positivo  $C_\bullet$  junto con una aplicación de cadenas  $\varepsilon : C_\bullet \rightarrow A$  donde  $\varepsilon$  es un homomorfismo de módulos  $\varepsilon : C_0 \rightarrow A$  tal que  $\varepsilon \partial = 0 : C_1 \rightarrow A$ .

**Definición 1.39.** Una **homotopía contrátil** para  $\varepsilon : C_\bullet \rightarrow A$  es una aplicación de cadenas  $f : A \rightarrow C_\bullet$  tal que  $\varepsilon f = \text{id}_A$  junto con una homotopía  $s : \text{id}_{C_\bullet} \rightarrow f\varepsilon$  donde  $\text{id}_{C_\bullet} \simeq f\varepsilon$ . En otras palabras, una homotopía contrátil consiste en homomorfismos de módulos  $f : A \rightarrow C_0$  y una homotopía de cadenas formada por los homomorfismos  $s_n : C_n \rightarrow C_{n+1}$  donde  $n \in \mathbb{N} \cup \{0\}$  tal que

$$\varepsilon f = \text{id}_A, \quad \partial_1 s_0 + f\varepsilon = \text{id}_{C_0}, \quad \partial_{n+1} s_n + s_{n-1} \partial_n = \text{id}_{C_n},$$

para todo  $n > 0$ .

Podemos extender el complejo estableciendo  $C_{-1} = A$ ,  $\partial_0 = \varepsilon : C_0 \rightarrow C_{-1}$  y  $s_{-1} = f$ . Por la definición anterior, la homotopía contrátil  $s : \text{id}_{C_\bullet} \rightarrow 0$  es una homotopía de cadenas. Si  $\varepsilon : C_\bullet \rightarrow A$  tiene una homotopía contrátil, sus grupos de homología son isomorfos por  $\varepsilon_* : H_0(C_\bullet) \rightarrow A$  para  $n = 0$  y  $H_n(C_\bullet) = 0$  para  $n > 0$ .

Consideremos un complejo de cadenas  $C_\bullet = \{C_n, d_n\}_{n \in \mathbb{Z}}$ , donde cada  $C_n$  es un  $\mathbb{Z}$ -módulo libre y  $d_n : C_n \rightarrow C_{n-1}$  es el operador diferencial de  $C_\bullet$  que cumple  $d_{n-1} \circ d_n = 0$  para todo  $n$ . Este tipo de complejos aparece frecuentemente en el estudio de espacios topológicos.

Supongamos además que cada  $C_n$  es finitamente generado. Entonces, el  $n$ -ésimo grupo de homología de  $C_\bullet$ , definido como

$$H_n(C_\bullet) = \frac{\ker(d_n)}{\text{Im}(d_{n+1})},$$

es un grupo abeliano finitamente generado. Este resultado se sigue del hecho de que el núcleo y la imagen de los morfismos entre  $\mathbb{Z}$ -módulos libres finitamente generados son también finitamente generados.

El teorema de **Descomposición cíclica primaria** y, en particular, el teorema de estructura para grupos abelianos finitamente generados afirma que cualquier grupo abeliano finitamente generado  $G$  puede expresarse como una suma directa de grupos cílicos de la forma

$$G \cong \mathbb{Z}^\beta \oplus \mathbb{Z}_{m_1} \oplus \dots \oplus \mathbb{Z}_{m_k},$$

donde  $\beta$  es el rango de  $G$  y cada  $\mathbb{Z}_{m_i}$  es un grupo cíclico de orden  $m_i$ , donde  $m_i$  divide a  $m_{i+1}$  para cada  $i \in \{1, \dots, k\}$ . Aplicando este teorema al  $n$ -ésimo módulo de homología  $H_n(C_\bullet)$ ,

## 1. Fundamentos del álgebra homológica

obtenemos que

$$H_n(C_\bullet) \cong \mathbb{Z}^{\beta_n} \oplus \mathbb{Z}_{m_1} \oplus \dots \oplus \mathbb{Z}_{m_k},$$

donde  $\beta_n$  es el rango de  $H_n(C_\bullet)$ , conocido como el **n-ésimo número de Betti** de  $C_\bullet$ , y los  $m_i$  son los **n-ésimos coeficientes de torsión**, donde  $m_i$  divide a  $m_{i+1}$  para cada  $i \in \{1, \dots, k\}$ .

**Definición 1.40.** Sea  $C_\bullet$  un complejo de cadenas y  $n$  un entero no negativo. El **n-ésimo número de Betti**,  $\beta_n(C_\bullet)$ , se define como el rango del  $n$ -ésimo módulo de homología de  $C_\bullet$  sobre el dominio de ideales principales  $R$ ,  $H_n(C_\bullet; R)$ . Esto es,  $\beta_n(C_\bullet) = \text{rg}(H_n(C_\bullet; R))$ . Si no hay lugar a confusión lo notaremos simplemente por  $\beta_n$ .

**Definición 1.41.** Dado un complejo de cadenas  $C_\bullet$  y considerando la descomposición en sumas directas del  $n$ -ésimo módulo de homología  $H_n(C_\bullet; R)$  sobre un dominio de ideales principales  $R$ , definimos los **n-ésimos coeficientes de torsión** como los  $m_i \in R$  donde  $i \in \{1, \dots, k\}$  que aparecen en la descomposición

$$H_n(C_\bullet; R) \cong R^{\beta_n} \oplus \frac{R}{\langle m_1 \rangle} \oplus \dots \oplus \frac{R}{\langle m_k \rangle},$$

donde cada  $m_i$  divide a  $m_{i+1}$ .

Los números de Betti  $\beta_n$  proporcionan una medida de la dimensionalidad de la n-ésima homología, mientras que los coeficientes de torsión  $\{m_i\}$  representan los órdenes de los sumandos cíclicos de torsión en la descomposición de homología y reflejan la estructura torsional de  $H_n(C_\bullet; R)$ .

La homología con coeficientes en  $\mathbb{Z}$  es fundamental en topología algebraica, proporcionando una rica estructura para el estudio de espacios topológicos. Sin embargo, el uso de coeficientes en otros anillos, particularmente en cuerpos, puede simplificar significativamente los cálculos y aún así ofrecer información valiosa para ciertas aplicaciones. La elección de un cuerpo como coeficientes, por ejemplo  $\mathbb{Z}_2$ , reduce la complejidad de los cálculos al evitar problemas relacionados con la torsión, facilitando la manipulación algebraica. Al estudiar la homología con estos coeficientes alternativos, aunque se obtiene una visión menos detallada del espacio, a menudo es suficiente para resolver problemas específicos dentro de un contexto dado, tales como la detección de características topológicas esenciales o la simplificación de la clasificación de espacios topológicos.

## 2. Símplices y complejos simpliciales

Los espacios topológicos pueden llegar a ser complicados de estudiar. Los complejos simpliciales tienen la ventaja de ser estructuras fáciles de estudiar. Por este motivo, los dotaremos de cierta topología que nos permitirá construir homeomorfismos a un gran número de espacios topológicos. En este capítulo nos centraremos en la definición y el estudio de estos objetos en profundidad en la línea de [71] y lo complementaremos con algunas aportaciones de [62].

### 2.1. Símplices

Con la finalidad de generalizar estructuras como el triángulo y el tetraedro, a finales del siglo XIX nace un nuevo concepto: el símplice. Su sencillez y propiedades lo convirtieron en una herramienta muy versátil en el estudio de la topología algebraica, dando lugar a lo que hoy conocemos como homología simplicial. En esta sección definiremos lo que es un símplice y algunos conceptos asociados a él que nos serán de gran utilidad en el estudio de dicho campo. Comenzamos recordando algunos conceptos de la geometría afín.

Como tan sólo será necesario trabajar en el espacio afín usual  $N$ -dimensional, lo notaremos simplemente por  $\mathbb{R}^N$ .

**Definición 2.1.** Sea  $\{a_0, \dots, a_p\}$  un conjunto de puntos en  $\mathbb{R}^N$ . Diremos que dicho conjunto es **afínmente independiente** si para cualesquiera  $t_i \in \mathbb{R}$ , las ecuaciones

$$\sum_{i=0}^p t_i = 0 \quad \text{y} \quad \sum_{i=0}^p t_i a_i = 0$$

implican que  $t_0 = t_1 = \dots = t_p$ .

**Definición 2.2.** Sea  $\{a_0, \dots, a_p\}$  un conjunto de puntos afínmente independiente. Definimos el **plano afín**  $P$  generado por  $\{a_0, \dots, a_p\}$  como el conjunto de puntos  $x \in \mathbb{R}^N$  tales que

$$x = \sum_{i=0}^p t_i a_i = a_0 + \sum_{i=1}^p t_i (a_i - a_0)$$

para algunos  $t_1, \dots, t_p \in \mathbb{R}$ . Diremos entonces que  $P$  es el plano que pasa por  $a_0$  paralelo a los vectores  $a_i - a_0$ ,  $i \in \{1, \dots, p\}$ .

Nótese que la transformación afín  $T$  de  $\mathbb{R}^N$  tal que  $T(x) = x - a_0$  es una traslación que lleva el plano  $P$  al subespacio vectorial de  $\mathbb{R}^N$  con base  $a_1 - a_0, a_2 - a_0, \dots, a_p - a_0$ . Si componemos dicha transformación con una aplicación lineal que lleve cada vector  $a_1 - a_0, a_2 - a_0, \dots, a_p - a_0$  a los primeros  $N$  vectores de la base usual  $\{e_1, \dots, e_N\}$ , obtenemos una transformación afín  $S : P \rightarrow \mathbb{R}^N \times \{0\}$  tal que  $S(a_i) = e_i$  para cada  $i \in \{1, \dots, p\}$ .

## 2. Símplices y complejos simpliciales

**Definición 2.3.** Sea  $\{a_0, \dots, a_p\}$  un conjunto de puntos afínmente independiente en  $\mathbb{R}^N$ . Definimos el **p-símplex o símplex**  $\sigma = [a_0, \dots, a_p]$  generado por  $a_0, \dots, a_p$  como el conjunto de todos los  $x \in \mathbb{R}^N$  tales que

$$x = \sum_{i=0}^p t_i a_i \quad \text{y} \quad \sum_{i=0}^p t_i = 1$$

con  $t_i \geq 0, i \in \{0, 1, \dots, p\}$ . Diremos que  $t_i$  es la *i-ésima coordenada baricéntrica* de  $x$  respecto a  $a_0, a_1, \dots, a_p$ .



Figura 2.1.: Símplices de dimensión 0, 1, 2 y 3.

**Proposición 2.1.** Sea  $\sigma$  un  $k$ -símplex definido como en 2.3. Entonces, para cualquier  $p \in \sigma$ , las coordenadas baricéntricas  $t_0, \dots, t_k$  de  $p$  están determinadas de manera única.

*Demostración.* Por definición, cualquier punto arbitrario  $p \in \sigma$  puede escribirse como una combinación convexa de los puntos  $a_i$ . Esto garantiza la existencia de una solución (no negativa) al sistema lineal

$$At = \begin{pmatrix} a_{01} & \cdots & a_{k1} \\ \vdots & \ddots & \vdots \\ a_{0N} & \cdots & a_{kN} \\ 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} t_0 \\ \vdots \\ t_k \end{pmatrix} = \begin{pmatrix} p_1 \\ \vdots \\ p_N \\ 1 \end{pmatrix} = p^*,$$

donde  $A$  es la matriz que contiene a los  $a_i$  como columnas, extendidos con un 1 en la última fila para incorporar la condición de que la suma de  $t_i$  sea igual a 1, asegurando que estamos considerando combinaciones convexas.

Para demostrar la unicidad, supongamos la existencia de otro vector  $t'$  tal que  $At' = p^*$ . Esto lleva a  $A(t - t') = 0$ . Supongamos que  $A(t - t') = Av = 0$ , donde  $v = t - t'$ . Esto implica que para  $v_i = t_i - t'_i$  para todo  $i \in \{0, \dots, k\}$

$$\sum_{i=0}^k v_i \cdot \begin{pmatrix} a_{0i} \\ \vdots \\ a_{ki} \\ 1 \end{pmatrix} = 0,$$

lo que lleva a que  $v_0 = v_1 = \dots = v_k = 0$ , debido a la independencia lineal de las columnas de  $A$ . En consecuencia,  $t = t'$ , demostrando así que las coordenadas baricéntricas son únicas para cualquier punto  $p$  en  $\sigma$ .  $\square$

Los puntos  $a_0, \dots, a_p$  que generan  $\sigma$  los llamaremos **vértices** de  $\sigma$  y al número  $p$  lo llamaremos la **dimensión** de  $\sigma$ , que notaremos por  $\dim \sigma$ .

**Definición 2.4.** Sea  $\sigma = [a_0, \dots, a_p]$  un simplice. Una **cara de dimensión  $p$**  de  $\sigma$  será cualquier simplice generado por un subconjunto no vacío de  $\{a_0, \dots, a_p\}$ .

En particular, la cara de  $\sigma$  generada por  $a_0, \dots, a_{i-1}, a_{i+1}, \dots, a_p$  la llamamos la **cara opuesta** de  $a_i$ ,  $i \in \{0, \dots, p\}$ . Las caras de  $\sigma$  diferentes de  $\sigma$  diremos que son **caras propias** de  $\sigma$  y la unión de todas ellas la llamaremos el **borde** de  $\sigma$  y lo notaremos  $Bd \sigma$ . Finalmente, definimos el **interior** de  $\sigma$ ,  $Int \sigma$ , como el conjunto de puntos de  $\sigma$  que no pertenecen a su borde.

En ocasiones, para dos simplices  $\sigma$  y  $\tau$ , escribiremos  $\tau \preceq \sigma$  si  $\tau$  es cara de  $\sigma$ . En caso de ser cara propia, lo notaremos por  $\tau \prec \sigma$ .

**Proposición 2.2.** Si  $\sigma$  es un simplice, entonces es unión disjunta del interior de todas sus caras.

*Demostración.* Sea  $x$  un elemento del simplice  $\sigma = [a_0, \dots, a_p]$  y sean  $t_0, \dots, t_p$  sus coordenadas baricéntricas. Consideremos ahora  $\sigma_k$  el simplice resultante de eliminar los vértices cuya coordenada tenía valor nulo. Esto es, tomamos el simplice  $\sigma_k = [a_{i_1}, \dots, a_{i_k}]$  donde  $t_{i_s} > 0$  para todo  $s \in \{1, \dots, k\}$ . Por la construcción de  $\sigma_k$ , tenemos que  $x$  pertenece a su interior.

Ahora sabemos que todo punto de un simplice pertenece al interior de una cara. Finalmente, la unicidad de las coordenadas baricéntricas nos garantiza que la unión del interior de dos caras es disjunta.  $\square$

## 2.2. Complejos simpliciales

La importancia de los complejos simpliciales reside en su capacidad para descomponer espacios topológicos en componentes manejables, permitiendo un análisis detallado de su estructura. Al considerar la forma en que estos simplices se conectan y orientan entre sí, los complejos simpliciales facilitarán la definición de cadenas y ciclos simpliciales que serán indispensables en el estudio de la homología simplicial.

**Definición 2.5.** Un **complejo simplicial geométrico** (finito), o simplemente **complejo simplicial** (finito),  $K$  en  $\mathbb{R}^N$ , es una colección (finita) de simplices en  $\mathbb{R}^N$  tal que:

1. Toda cara de un simplice de  $K$  está en  $K$ .
2. La intersección de cualesquiera dos simplices de  $K$  es o el vacío o una cara de ambos simplices.

*Nota.* Si bien los complejos simpliciales se pueden formular sin la restricción de finitud, nosotros trabajaremos principalmente en el caso finito por simplicidad en algunos resultados.

En ciertas ocasiones puede ser interesante saber si dada una colección cualquiera de simplices, esta es un complejo simplicial o no. Para ello, el siguiente lema nos puede ser de utilidad.

**Lema 2.1.** Una colección  $K$  de simplices es un complejo simplicial si, y sólo si, se cumplen las siguientes condiciones:

1. Toda cara de un simplice de  $K$  está en  $K$ .
2. La intersección dos a dos del interior de los simplices de  $K$  es vacía.

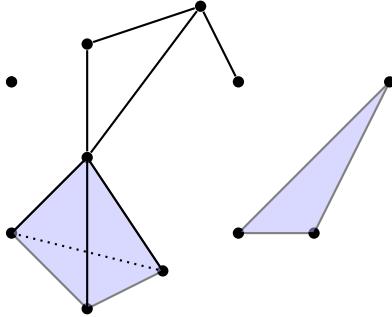


Figura 2.2.: Visualización de un complejo simplicial.

*Demostración.* Primero, asumamos que  $K$  es un complejo simplicial. Dados dos símplices  $\sigma, \tau \in K$  veamos que si el interior de ambos tiene un punto  $x$  en común, entonces  $\sigma = \tau$ . Sea  $s = \sigma \cap \tau$  y considero  $x \in s$ . Si  $s$  fuera una cara propia de  $\sigma$ , entonces  $x$  pertenecería a la frontera de  $\sigma$ , lo cual no se cumple ya que  $x$  pertenece al interior de  $\sigma$ . Por tanto  $s = \sigma$ . De manera análoga,  $s = \tau$ , luego  $\sigma = \tau$ .

Asumamos ahora que se cumplen (1) y (2). Queremos ver que si el conjunto  $\sigma \cap \tau \neq \emptyset$ , dicha intersección es la cara  $\sigma'$  de  $\sigma$  generada por los vértices  $b_0, \dots, b_m$  de  $\sigma$  que están en  $\tau$ . Primero,  $\sigma' \subset \sigma \cap \tau$  por ser  $\sigma \cap \tau$  convexa y contener a  $b_0, \dots, b_m$ . Para la otra inclusión supongamos que  $x \in \sigma \cap \tau$ . Esto implica que  $x \in \text{Int } s \cap \text{Int } t$  para alguna cara  $s$  de  $\sigma$  y alguna cara  $t$  de  $\tau$ . Se sigue de (2) que  $s = t$  por lo que los vértices de  $s$  están en  $\tau$  y por definición, son elementos del conjunto  $\{b_0, \dots, b_m\}$ . Concluimos entonces que  $s$  es una cara de  $\sigma'$ , lo que implica que  $x \in \sigma'$ , como queríamos ver.  $\square$

**Definición 2.6.** Si  $L$  es una subcolección del complejo simplicial  $K$  que contiene todas las caras de sus elementos, entonces  $L$  es un complejo simplicial que llamaremos **subcomplejo** de  $K$ .

**Definición 2.7.** Sea  $K$  un complejo simplicial. Diremos **p-esqueleto** de  $K$  al subcomplejo formado por todas las caras de  $K$  cuya dimensión sea menor o igual que  $p$ . Lo denotaremos por  $K^{(p)}$ . En particular,  $K^{(0)}$  lo llamaremos el **conjunto de vértices** de  $K$ .

**Definición 2.8.** Sea  $K$  un complejo simplicial de  $\mathbb{R}^N$  y sea  $|K|$  el subconjunto de  $\mathbb{R}^N$  tal que  $|K|$  es la unión de todos los símplices de  $K$ . Definimos el **politopo** o **espacio subyacente** de  $K$  como el espacio topológico  $(|K|, \mathcal{T})$  donde los abiertos de  $\mathcal{T}$  son aquellos  $O \subseteq |K|$  tal que  $O \cap \sigma$  es abierto en  $\sigma$  con la topología inducida de  $\mathbb{R}^N$  para todo  $\sigma \in K$ .

Veamos que en efecto  $(|K|, \mathcal{T})$  es un espacio topológico.  $\emptyset, |K| \in \mathcal{T}$  ya que son abiertos trivialmente en  $\sigma$ , pues  $\emptyset \cap \sigma = \emptyset$  y  $|K| \cap \sigma = \sigma$  para todo  $\sigma \in K$ . Si  $O_1, O_2 \in \mathcal{T}$ , entonces  $O_1 \cap \sigma, O_2 \cap \sigma$  son abiertos en  $\sigma$  luego  $(O_1 \cap O_2) \cap \sigma = (O_1 \cap \sigma) \cap (O_2 \cap \sigma)$  es abierto en  $\sigma$  para todo  $\sigma \in K$ . Por tanto  $O_1 \cap O_2 \in \mathcal{T}$ . Finalmente, consideremos una familia  $\{O_i\}_{i \in I} \subset \mathcal{T}$  donde  $I$  es un conjunto de índices. Para cada  $\sigma \in K$ ,  $(\cup_{i \in I} O_i) \cap \sigma = \cup_{i \in I} (O_i \cap \sigma)$  que efectivamente es una unión arbitraria de abiertos de  $\sigma$ . En consecuencia,  $\cup_{i \in I} O_i \in \mathcal{T}$ .

En general, la topología de  $|K|$  es más fina que la inducida de la topología usual de  $\mathbb{R}^N$ . Si  $A$  es cerrado en  $|K|$  con la topología inducida de la usual,  $A = B \cap |K|$  para algún cerrado  $B$  de  $\mathbb{R}^N$  y por tanto  $B \cap \sigma$  sería cerrado en  $\sigma$  para cada símplice  $\sigma$  de  $K$ . Como consecuencia,  $B \cap |K| = A$  es cerrado en  $|K|$  con la topología  $\mathcal{T}$  definida anteriormente. No obstante, la otra inclusión no tiene por qué cumplirse:

**Ejemplo 2.1.** Consideremos el complejo no finito  $K$  en  $\mathbb{R}$  cuyos simplices son todos los intervalos  $[m, m+1]$  con  $m \in \mathbb{Z} \setminus \{0\}$ , todos los intervalos de la forma  $[1/(n+1), 1/n]$  donde  $n \in \mathbb{N}$  y todas sus respectivas caras. Como resultado tenemos que  $|K| = \mathbb{R}$ , donde  $F = \{1/n : n \in \mathbb{N}\}$  es cerrado en nuestra topología  $\mathcal{T}$  pero no en la inducida por la usual. Dicho de otra forma,  $\mathbb{R} \setminus F$  es abierto en  $\mathcal{T}$  pero no en la usual.

Si no hay lugar a confusión, simplemente notaremos al politopo de  $K$  por  $|K|$  y lo llamaremos el **poliedro**  $|K|$ .

A continuación, mencionemos algunas propiedades relevantes de este espacio topológico. Para ello fijaremos un complejo simplicial finito  $K$  en  $\mathbb{R}^N$ .

**Proposición 2.3.** *Sea  $K$  un complejo simplicial finito. Entonces el poliedro  $|K|$  es compacto.*

*Demostración.* Si  $K$  es un complejo simplicial, sus simplices son conjuntos cerrados y acotados. En consecuencia,  $|K|$  es unión finita de conjuntos cerrados y acotados, luego es cerrado y acotado en  $\mathbb{R}^N$ . Por lo tanto, es compacto.  $\square$

**Proposición 2.4.** *Sea  $K$  un complejo simplicial. Si  $x \in |K|$ , entonces existe un único simplex en  $K$  tal que  $x$  pertenece a su interior.*

*Demostración.* Si  $x \in |K|$ , entonces existe algún simplex  $\sigma$  de  $K$  tal que  $x \in \sigma$ . Por la [Proposición 2.2](#),  $x$  pertenece al interior de alguna cara  $\tau$  de  $\sigma$ . Supongamos ahora que existe otro simplex  $\rho$  de  $K$  tal que  $x \in \text{Int } \rho$ . Por consiguiente, si  $x \in \text{Int } \rho \cap \text{Int } \tau$ , entonces  $x$  pertenecería a una cara común  $\mu$  de  $\rho$  y  $\tau$ . Esto es,  $\mu = \rho \cap \tau$ . Ahora si  $\rho \neq \mu$ , el elemento  $x$  debería tener alguna coordenada baricéntrica nula respecto a los vértices de  $\rho$ , en contradicción con que  $x$  pertenece al interior de  $\rho$ . En consecuencia,  $\rho = \mu$ . De manera análoga obtenemos  $\tau = \mu$  y por tanto,  $\rho = \tau$ .  $\square$

**Definición 2.9.** Sea  $K$  un complejo simplicial y sea  $x \in |K|$ . Llamaremos **símplice soporte de  $x$**  al único simplex que contiene a  $x$  en su interior y lo notaremos por  $\text{sop}(x)$ .

**Corolario 2.1.** *Sean  $\sigma, \tau$  simplices de  $K$  tal que  $\text{Int } \sigma \cap \tau$  es no vacía. Entonces  $\sigma$  es una cara de  $\tau$ .*

*Demostración.* Consideremos  $x \in \text{Int } \sigma \cap \tau$ . Por la [Proposición 2.2](#) sabemos que  $\tau$  es la unión de todas sus caras lo que implica que existe una cara  $\mu$  de  $\tau$  cuyo interior contiene a  $x$ . Por lo tanto,  $x \in \text{Int } \mu \cap \text{Int } \sigma$  y como consecuencia de la [Proposición 2.4](#),  $\mu = \sigma$ .  $\square$

**Lema 2.2.** *Sea  $K$  un complejo simplicial y  $X$  un espacio topológico. Una aplicación  $f : |K| \rightarrow X$  es continua si, y sólo si,  $f|_{\sigma}$  es continua para cada  $\sigma \in K$ .*

*Demostración.* Si  $f$  es continua, también lo es  $f|_{\sigma}$  por ser  $\sigma$  un subespacio de  $K$ . Supongamos ahora que  $f|_{\sigma}$  es continua para cada  $\sigma \in K$ . Si  $C$  es un cerrado de  $X$ ,  $f^{-1}(C) \cap \sigma = f|_{\sigma}^{-1}(C)$  es un cerrado en  $\sigma$  por la continuidad de  $f|_{\sigma}$ . Concluimos que  $f^{-1}(C)$  es cerrado en  $|K|$  por definición.  $\square$

Para finalizar esta sección, comentaremos la principal utilidad del estudio de complejos simpliciales introduciendo el concepto de triangulación.

**Definición 2.10.** Un espacio topológico  $X$  es **triangulable** si existe un complejo simplicial  $K$  cuyo espacio subyacente es homeomorfo a  $X$ . Diremos entonces que el homeomorfismo  $h : |K| \rightarrow X$  es una **triangulación**.

## 2. Símplices y complejos simpliciales

La triangulación nos permite representar un espacio topológico usando un complejo simplicial, lo que es útil porque muchos espacios topológicos importantes pueden ser triangulados. En el [Capítulo 3](#), veremos que estudiar ciertas propiedades topológicas en estos complejos es más sencillo y manejable.

El siguiente ejemplo muestra la triangulación de una esfera en  $\mathbb{R}^3$  por el método de «proyección radial». Para ello, recordemos el siguiente resultado:

**Proposición 2.5.** *Sean  $X, Y$  espacios topológicos tales que  $X$  es compacto e  $Y$  es Hausdorff. Si además  $f : X \rightarrow Y$  es una aplicación continua, entonces  $f$  es cerrada. En particular, si  $f$  es biyectiva, entonces es un homeomorfismo.*

**Ejemplo 2.2.** Consideremos el complejo simplicial  $K$ , que es un tetraedro en  $\mathbb{R}^3$  formado por símplices de dimensión 2. Denotamos su poliedro asociado por  $|K|$ . Supongamos que el origen de  $\mathbb{R}^3$  es el centro de  $|K|$  y que  $S^2$  es la esfera unidad con radio 1 con la topología inducida de la usual. Definimos la función  $f : |K| \rightarrow S^2$  por

$$f(t) = \frac{t}{\|t\|}$$

que es continua en todo  $|K|$ , pues  $0 \notin |K|$ . Para cada  $s \in S^2$ , definimos  $g : S^2 \rightarrow |K|$  tal que

$$g(s) = t = s \cdot \sup\{\lambda : \lambda s \in \text{envolvente convexa de } |K|\},$$

donde el supremo asegura que  $t$  está en la frontera de  $|K|$ . De este modo, se verifica que  $g(f(t)) = t$  para todo  $t \in |K|$ . Por lo tanto,  $f$  es una biyección continua desde un conjunto compacto hacia un espacio Hausdorff, luego es un homeomorfismo y en consecuencia, una triangulación de la esfera  $S^2$ .

En particular, las esferas pertenecen a una familia de espacios topológicos conocidos como **variedades topológicas**. Estos espacios son el objeto central de estudio en la topología, siendo el objetivo su clasificación bajo homeomorfismos.

**Definición 2.11.** Sea  $X$  un espacio topológico Hausdorff no vacío. Diremos que  $X$  es una  **$m$ -variedad** o **variedad de dimensión  $n$**  si cada punto de  $X$  tiene un entorno homeomorfo a un subconjunto abierto de  $\mathbb{R}^m$  con la topología usual.

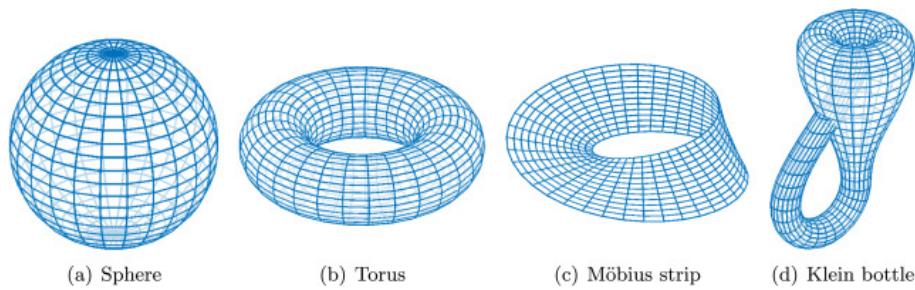


Figura 2.3.: Ejemplos de variedades de dimensión 2: (a) Esfera, (b) Toro, (c) Banda de Möbius y (d) Botella de Klein. Fuente [26].

Aunque existen un gran número de resultados fundamentales relacionados con el estudio de las variedades topológicas, este análisis escapa de los contenidos de este trabajo. Sin

embargo, veremos que estos espacios serán necesarios en la aplicación experimental realizada para comprender cómo las redes neuronales transforman los datos desde el punto de vista de la topología.

## 2.3. Celdas y CW-complejos

A continuación presentamos una generalización del concepto de complejo simplicial, propuesta por J.H.C. Whitehead en [120]. Los CW-complejos reemplazan la estructura simplicial tradicional por estructuras homeomorfas a bolas abiertas, facilitando el estudio de una gama más amplia de espacios topológicos.

Iniciaremos esta sección estableciendo la notación que utilizaremos. Denotaremos la **bola abierta** centrada en  $x_0$  y de radio  $r$  en el espacio  $\mathbb{R}^N$  con la topología usual por el conjunto  $B_r(x_0) = \{x \in \mathbb{R}^N : \|x - x_0\| < r\}$ . Además, para cualquier subconjunto  $U$  de  $\mathbb{R}^N$ , denotaremos su **clausura** como  $\overline{U}$  y su **frontera** como  $\text{Bd } U$ . Por último, la **esfera unidad** de dimensión  $n$  será representada simplemente como  $S^{n-1}$ .

**Definición 2.12.** Sea  $X$  un espacio topológico. Diremos que  $X$  es una **celda** abierta (cerrada) de dimensión  $p$  o  $p$ -celda si  $X$  es homeomorfo a la bola unidad abierta (cerrada) de dimensión  $p$ .

Sería interesante disponer de resultados que nos digan cuándo un subconjunto dado puede ser una celda. La siguiente proposición será de gran utilidad para ver que los complejos simpliciales no son más que un caso particular de los CW-complejos.

**Proposición 2.6.** Si  $D \subseteq \mathbb{R}^n$  es un subconjunto convexo compacto con interior no vacío, entonces  $D$  es una  $n$ -celda cerrada y su interior es una  $n$ -celda abierta. De hecho, dado cualquier punto  $p \in \text{Int } D$ , existe un homeomorfismo  $F : \overline{B}_1(0) \rightarrow D$  que envía  $0$  a  $p$ ,  $B_1(0)$  a  $\text{Int } D$ , y  $S^{n-1}$  a  $\text{Bd } D$ .

*Demostración.* Sea  $p$  un punto interior de  $D$ . Reemplazando  $D$  por su imagen bajo la traslación  $x \mapsto x - p$ , podemos suponer que  $p = 0 \in \text{Int } D$ . Entonces existe algún  $\varepsilon > 0$  tal que la bola  $B_\varepsilon(0)$  está contenida en  $D$ . Utilizando la dilatación  $x \mapsto x/\varepsilon$ , podemos asumir  $B_1(0) \subseteq D$ . A continuación demostraremos que cada semirecta cerrada que comienza en el origen interseca  $\text{Bd } D$  en exactamente un punto. Sea  $R$  tal semirecta cerrada. Dado que  $D$  es compacto, su intersección con  $R$  es compacta; por lo tanto, hay un punto  $x_0$  en esta intersección donde la distancia al origen alcanza su máximo. Este punto se identifica fácilmente como parte del borde de  $D$ . Para demostrar que solo puede haber un punto así, mostramos que el segmento de línea desde  $0$  hasta  $x_0$  consta enteramente de puntos interiores de  $D$ , excepto  $x_0$  mismo. Cualquier punto en este segmento que no sea  $x_0$  se puede escribir en la forma  $\lambda x_0$  para  $0 \leq \lambda < 1$ . Supongamos que  $z \in B_{1-\lambda}(\lambda x_0)$ , y sea  $y = (z - \lambda x_0)/(1 - \lambda)$ . En consecuencia  $|y| < 1$ , por lo que  $y \in B_1(0) \subseteq D$ . Como  $y$  y  $x_0$  están ambos en  $D$  y  $z = \lambda x_0 + (1 - \lambda)y$ , se sigue de la convexidad que  $z \in D$ . Así, la bola abierta  $B_{1-\lambda}(x_0)$  está contenida en  $D$ , lo que implica que  $\lambda x_0$  es un punto interior.

Ahora definamos una aplicación  $f : \text{Bd } D \rightarrow S^{n-1}$  por

$$f(x) = \frac{x}{\|x\|}.$$

Básicamente,  $f(x)$  es el punto donde el segmento de línea que parte del origen hasta  $x$  interseca la esfera unidad. Puesto que  $f$  es la restricción de una aplicación continua, es

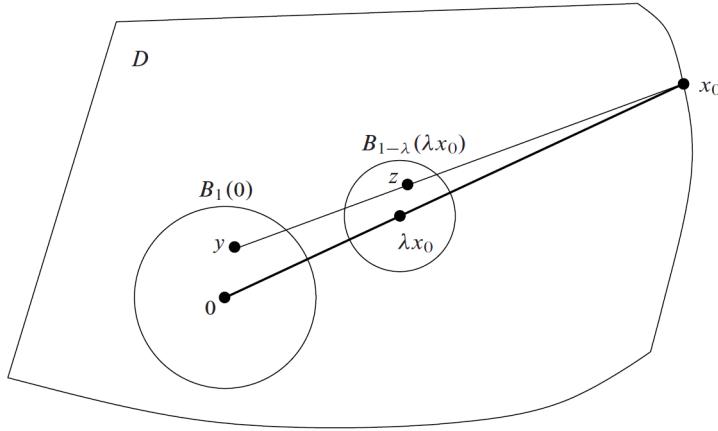


Figura 2.4.: Esquema que muestra la idea de que cada rayo tan sólo tiene un punto en la frontera en la demostración de la [Proposición 2.6](#). Fuente [62].

continua, por lo que la discusión del párrafo anterior muestra que es biyectiva. Dado que  $\text{Bd } D$  es compacto, la [Proposición 2.5](#) nos garantiza que  $f$  es cerrada. Por ser  $f$  biyectiva, continua y cerrada, entonces  $f$  es un homeomorfismo.

Finalmente, definamos  $F : \bar{B}_1(0) \rightarrow D$  de forma que

$$F(x) = \begin{cases} \|x\| f^{-1}\left(\frac{x}{\|x\|}\right), & x \neq 0; \\ 0, & x = 0. \end{cases}$$

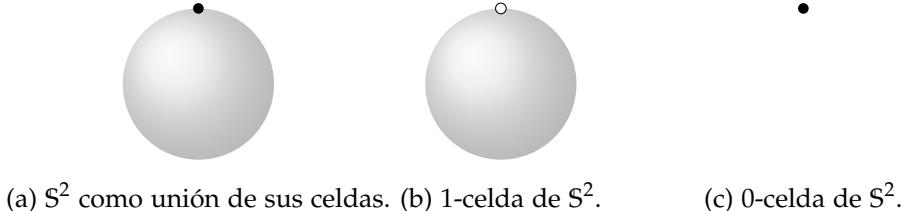
Entonces  $F$  es continua lejos del origen porque  $f^{-1}$  lo es, y también lo es en el origen pues la acotación de  $f^{-1}$  implica que  $F(x) \rightarrow 0$  conforme  $x \rightarrow 0$ . Geométricamente,  $F$  lleva cada segmento de línea radial que conecta 0 con un punto  $\omega \in S^{n-1}$  linealmente sobre el segmento radial de 0 al punto  $f^{-1}(\omega) \in \text{Bd } D$ . Por convexidad,  $F$  toma sus valores en  $D$ . La aplicación  $F$  es inyectiva, ya que puntos en rayos distintos van a rayos distintos, y cada segmento radial se lleva linealmente a su imagen. Es sobreyectiva porque cada punto  $y \in D$  está en algún rayo desde 0. Por la [Proposición 2.5](#),  $F$  es un homeomorfismo.  $\square$

**Definición 2.13.** Sea  $(X, \mathcal{E})$ , donde  $X$  es un espacio topológico Hausdorff y  $\mathcal{E}$  una colección de celdas abiertas. Diremos entonces que  $(X, \mathcal{E})$  es un **CW-complejo** si se cumple que:

- (C) Consideremos la bola unidad  $B_1(0)$  de dimensión  $p$ . Para cada  $p$ -celda  $e \in \mathcal{E}$ , existe una aplicación continua  $f_e : \bar{B}_1(0) \rightarrow X$  de forma que la restricción a  $B_1(0)$  es homeomorfa a la celda  $e$  y la restricción a la frontera  $\text{Bd } \bar{B}_1(0)$  está contenida en una unión finita de celdas de dimensión menor a  $p$ . A dicha función la llamaremos **función característica**.
- (W) Un subconjunto  $F$  de  $X$  es cerrado si, y sólo si,  $F \cap \bar{e}$  es cerrado en  $\bar{e}$  con la topología inducida de  $X$  para todo  $e \in \mathcal{E}$ .

Normalmente denotaremos al CW-complejo  $(X, \mathcal{E})$  simplemente por  $X$ .

Una propiedad importante de los CW-complejos es que mantienen su estructura en subconjuntos bajo ciertas condiciones razonables.

Figura 2.5.: Visualización de un CW-complejo para  $S^2$ .

**Definición 2.14.** Sea  $X$  un CW-complejo. Diremos que  $Y \subseteq X$  es un **subcomplejo** de  $X$  si es unión de celdas de  $X$  de forma que si  $Y$  contiene una celda, entonces también contiene su clausura.

**Teorema 2.1.** *Sea  $X$  un CW-complejo y sea  $Y$  un subcomplejo de  $X$ . Entonces  $Y$  es cerrado en  $X$  y, además, es un CW-complejo con la topología y la colección de celdas inducidas.*

*Demostración.* Es claro que  $Y$  es Hausdorff. Además, por definición tenemos que  $Y$  es la unión disjunta de sus celdas. Sea  $e \subseteq Y$  una celda abierta de  $Y$ . Como su clausura también está contenida en  $Y$ , entonces existe un número finito de celdas de  $X$  con intersección no vacía con  $\bar{e}$  que, a su vez, son celdas de  $Y$ . En consecuencia, la condición (C) se cumple. Es más, cualquier aplicación característica  $f_e : \rightarrow X$  de  $X$  lo es también de  $Y$  para cualquier celda  $e \subseteq Y$ .

En cuanto a la condición (W), supongamos que  $S$  es un subconjunto de  $Y$  tal que  $S \cap \bar{e}$  es cerrado en  $\bar{e}$  con la topología inducida de  $Y$  para toda celda en  $Y$ . Sea ahora  $e$  una celda de  $X$  que no esté contenida en  $Y$ . Sabemos que  $\bar{e} \setminus e$  está contenido en la unión de un número finito de celdas de  $X$ , de las cuales un subconjunto de ellas están contenidas en  $Y$ . Llamemos a dichas celdas  $e_1, \dots, e_n$ . Por consiguiente,  $\bar{e}_1 \cup \dots \cup \bar{e}_n \subseteq Y$  y además,

$$S \cap \bar{e} = S \cap (\bar{e}_1 \cup \dots \cup \bar{e}_n) \cap \bar{e} = ((S \cap \bar{e}_1) \cup \dots \cup (S \cap \bar{e}_n)) \cap \bar{e},$$

luego  $S \cap \bar{e}$  es cerrado en  $\bar{e}$  con la topología inducida de  $Y$ . Es decir,  $S$  es cerrado en  $X$  y por tanto en  $Y$ . Finalmente, concluimos que  $Y$  es cerrado en  $X$  tomando  $S = Y$ .  $\square$

**Definición 2.15.** Sea  $X$  un CW-complejo. Diremos que el subespacio  $X^{(p)}$  de  $X$  es el **p-esqueleto** de  $X$  si es igual a la unión de todas las celdas de dimensión menor o igual que  $p$ . En particular, es un subcomplejo de dimensión  $p$  de  $X$ .

**Teorema 2.2.** *Sea  $X$  un CW-complejo. Entonces las siguientes propiedades son equivalentes:*

1.  $X$  es conexo por arcos.
2.  $X$  es conexo.
3. El 1-esqueleto de  $X$  es conexo.
4. Algun  $n$ -esqueleto de  $X$  es conexo para algún  $n$ .

*Demostración.* Obviamente, (1)  $\Rightarrow$  (2) y (3)  $\Rightarrow$  (4), por lo que basta con demostrar que (2)  $\Rightarrow$  (3) y (4)  $\Rightarrow$  (1).

Para probar (2)  $\Rightarrow$  (3) razonaremos por contrarrecíproco. Supongamos que  $X^{(1)} = X'^{(1)} \cup X''^{(1)}$  es una unión no conexa del 1-esqueleto de  $X$ . Veamos por inducción en  $n$  que para cada

## 2. Símplices y complejos simpliciales

$n > 1$ , el  $n$ -esqueleto  $X^{(n)}$  puede expresarse como unión no conexa  $X^{(n)} = X'^{(n)} \cup X''^{(n)}$  tal que  $X'^{(n)} \subseteq X'^{(n-1)}$  y  $X''^{(n)} \subseteq X''^{(n-1)}$  para cada  $n$ . Supongamos  $X^{(n-1)} = X'^{(n-1)} \cup X''^{(n-1)}$  es una unión no conexa de  $X^{(n-1)}$  para algún  $n > 1$ . Para cada celda  $n$ -dimensional  $e$ , la restricción de su aplicación función característica  $f_e: D^n \rightarrow X^{(n)}$  a  $\partial D^n$  es continua en  $X^{(n-1)}$ . Dado que  $\partial D^n \cong S^{n-1}$  es conexo, su imagen debe estar contenida en uno de los conjuntos  $X'^{(n)}$  o  $X''^{(n)}$ . Por lo tanto,  $\overline{f_e(D)}$  tiene una intersección no trivial con  $X'^{(n)}$  o  $X''^{(n)}$ , pero no con ambos. Dividimos las  $n$ -celdas en dos colecciones disjuntas  $\mathcal{E}'$  y  $\mathcal{E}''$ , según si sus clausuras intersecan  $X'^{(n-1)}$  o  $X''^{(n-1)}$ , respectivamente, y definimos

$$X'^{(n)} = X'^{(n-1)} \cup \left( \bigcup_{e \in \mathcal{E}'} \overline{f_e(e)} \right), \quad X''^{(n)} = X''^{(n-1)} \cup \left( \bigcup_{e \in \mathcal{E}''} \overline{f_e(e)} \right).$$

Claramente,  $X^{(n)}$  es la unión disjunta de  $X'^{(n)}$  y  $X''^{(n)}$ , y ambos conjuntos son no vacíos debido a la hipótesis de inducción.

Ahora, definamos  $X' = \bigcup_n X'^{(n)}$  y  $X'' = \bigcup_n X''^{(n)}$ . Como antes,  $X = X' \cup X''$ , y ambos conjuntos son no vacíos. Por el mismo argumento que arriba, si  $e$  es cualquier celda de  $X$  de cualquier dimensión, su clausura debe estar contenida en uno de estos conjuntos. Así,  $X'$  y  $X''$  son ambos abiertos y cerrados en  $X$ , lo que implica que  $X$  no es conexo.

Para demostrar (4)  $\Rightarrow$  (1), supongamos que  $X$  es un CW-complejo cuyo  $n$ -esqueleto es conexo para algún  $n \geq 0$ . Mostremos por inducción en  $k$  que  $X^{(k)}$  es conexo por arcos para cada  $k \geq n$ . Primero, necesitamos mostrar que  $X^{(n)}$  en sí mismo es conexo por arcos. Si  $n = 0$ , entonces  $X^{(n)}$  es discreto y conexo, así que es un conjunto unitario y por lo tanto conexo por arcos. En caso contrario, elijamos cualquier punto  $x_0 \in X^{(n)}$  y consideremos  $S_n$  la componente arcoconexa de  $X^{(n)}$  que contiene a  $x_0$ . Para cada celda  $e$  de  $X^{(n)}$ , notemos que  $\overline{f_e(e)}$  es la imagen continua de un espacio conexo por arcos, así que es conexo por arcos. Por lo tanto, si  $\overline{f_e(e)}$  tiene una intersección no trivial con la componente arcoconexa  $S_n$ , debe estar contenida en  $S_n$ . En consecuencia,  $S_n$  es cerrado y abierto en  $X^{(n)}$ . Como estamos asumiendo que  $X^{(n)}$  es conexo, entonces  $S_n = X^{(n)}$ .

Ahora, supongamos que hemos demostrado que  $X^{(k-1)}$  es conexo por arcos para algún  $k > n$  y sea  $S_k$  la componente arcoconexa de  $X^{(k)}$  que contiene a  $X^{(k-1)}$ . Para cada  $k$ -celda  $e$ , su clausura  $\overline{f_e(e)}$  es un subconjunto de  $X^{(k)}$  conexo por arcos que tiene intersección no trivial con  $X^{(k-1)}$  y, por lo tanto, está contenido en  $S_k$ . Se sigue que  $X^{(k)} = S_k$ , completando la inducción.  $\square$

**Lema 2.3.** *Sea  $X$  un CW-complejo. Entonces la clausura de cada celda está contenida en un subcomplejo finito.*

*Demostración.* Consideremos cualquier  $n$ -celda  $e \in X$  y probemos el lema por inducción. Para el caso  $n = 0$ ,  $\bar{e} = e$  es trivialmente un subcomplejo finito. Supongamos ahora el lema cierto para las celdas de dimensión menor o igual que  $n$  y veámoslo para  $n + 1$ . Por la condición (C),  $\bar{e} \setminus e$  está contenido en la unión de un número finito de celdas de dimensión menor que  $n + 1$ . Dichas celdas están contenidas en subcomplejos finitos por hipótesis de inducción. Sin embargo, la unión de dichos subcomplejos finitos con  $e$  es de hecho un subcomplejo finito que contiene a  $\bar{e}$ .  $\square$

**Lema 2.4.** *Sea  $X$  un CW-complejo. Un subconjunto de  $X$  es discreto si, y sólo si, su intersección con cada celda es finita.*

*Demostración.* Sea  $S$  un subconjunto discreto de  $X$ . Entonces, la intersección de la clausura de cada celda  $e$  de  $X$  con  $S$  es un subconjunto discreto de un conjunto compacto, luego es finito. En consecuencia,  $S \cap e$  también lo es.

Para la otra implicación supongamos que  $S$  es un subconjunto cuya intersección con cualquier celda es finita. Como la clausura de cada celda está contenida en un subcomplejo finito, entonces por hipótesis tenemos que  $S \cap \bar{e}$  es finito para cada celda  $e$  de  $X$ . Esto significa que  $S \cap \bar{e}$  es cerrado en  $\bar{e}$  y por la condición  $(W)$ ,  $S$  es cerrado en  $X$ . Sin embargo, este argumento podemos aplicarlo a cualquier subconjunto de  $S$ , luego todo subconjunto de  $S$  es cerrado en  $X$ . Por lo tanto, la topología inducida en  $S$  es discreta.  $\square$

**Teorema 2.3.** *Sea  $X$  un CW-complejo. Un subconjunto de  $X$  es compacto si, y sólo si, es cerrado en  $X$  y está contenido en un subcomplejo finito.*

*Demostración.* Todo subcomplejo finito de  $X$  es compacto pues es unión finita de clausuras de celdas, las cuales son compactas. En consecuencia, si  $K$  es un subconjunto cerrado de  $X$  contenido en un subcomplejo finito, entonces es compacto.

Supongamos ahora que  $K \subseteq X$  es compacto. Si  $K$  intersecara una cantidad infinita de celdas, podríamos tomar un punto de cada intersección de forma que tuviéramos un subconjunto infinito discreto de  $K$ , lo cual es imposible. Es decir,  $K$  está contenido en la unión de un número finito de celdas y por el [Lema 2.3](#), está contenido en un subcomplejo finito.  $\square$

**Corolario 2.2.** *Un CW-complejo es compacto si, y sólo si, es un complejo finito.*

**Proposición 2.7.** *Todo  $p$ -simplice es una celda cerrada de dimensión  $p$ .*

*Demostración.* Inmediato por la [Proposición 2.6](#).  $\square$

Una vez discutidas algunas propiedades básicas de los CW-complejos, ya estamos en condiciones de verificar que efectivamente los complejos simpliciales son CW-complejos.

**Proposición 2.8.** *Si  $K$  es un complejo simplicial finito, entonces el poliedro  $|K|$  junto con la colección  $\mathcal{E}$  de interiores de los simplices de  $K$  forman un CW-complejo.*

*Demostración.* Supongamos que  $K$  es un complejo simplicial finito en  $\mathbb{R}^N$ . La condición  $(C)$  se obtiene de manera directa a partir de la [Proposición 2.6](#).

En cuanto a la propiedad  $(W)$ , consideremos  $F$  como un subconjunto de  $|K|$ . Sea  $\{x_n\}_{n \in \mathbb{N}}$  una sucesión que converge a  $x$  en  $|K|$  y sea  $U$  un entorno de  $x$ . Por la compacidad de  $|K|$  y el hecho de que  $\mathcal{E}$  es un recubrimiento por abiertos de  $|K|$ , podemos escoger un subrecubrimiento finito  $e_1, \dots, e_k$  tal que  $x \in U \subseteq \bar{e}_1 \cup \dots \cup \bar{e}_k$ .

Fijemos  $n_0 \in \mathbb{N}$  tal que  $x_{n_i} \in U$  para todo  $n_i \geq n_0$ . Como hay un número finito de  $e_j$  e infinitos  $x_{n_i}$ , existe una parcial convergente  $\{x_{n_i}\}_{i \in \mathbb{N}}$  que converge a  $x$  contenido en algún  $\bar{e}_j$  para cierto  $j \in \{1, \dots, k\}$ . Esto muestra que  $x \in \bar{e}_j$  y, puesto que  $x_{n_i} \in F \cap \bar{e}_j$  para todo  $n_i \geq n_0$ , y  $F \cap \bar{e}_j$  es cerrado en  $\bar{e}_j$ , concluimos que  $x \in F \cap \bar{e}_j$ .  $\square$

## 2.4. Aplicaciones simpliciales

Cuando trabajemos con complejos simpliciales, será interesante tener en cuenta cuándo las transformaciones entre ellos pueden ser continuas o incluso homeomorfismos.

## 2. Símplices y complejos simpliciales

**Lema 2.5.** Sean  $K$  y  $L$  dos complejos simpliciales y sea  $f : K^{(0)} \rightarrow L^{(0)}$  una aplicación entre los conjuntos de vértices de  $K$  y  $L$ . Supongamos que siempre que los vértices  $v_0, \dots, v_n$  de  $K$  generen un simplex en  $K$ , los puntos  $f(v_0), \dots, f(v_n)$  son vértices de un simplex de  $L$ . Entonces podemos extender  $f$  a una aplicación continua  $|f| : |K| \rightarrow |L|$  tal que

$$x = \sum_{i=0}^n t_i v_i \implies |f|(x) = \sum_{i=0}^n t_i f(v_i)$$

Llamaremos a  $|f|$  la **aplicación simplicial (lineal) inducida por  $f$** .

*Demostración.* Por hipótesis, los vértices  $f(v_0), \dots, f(v_n)$  generan un simplex  $\tau$  en  $L$ . Por ser  $K$  un complejo simplicial, la suma de sus coeficientes  $t_i$ , con  $i \in \{0, \dots, n\}$ , es igual a uno, luego  $|f|(x) = \sum_{i=0}^n t_i f(v_i)$  es un punto de  $\tau$ . Es decir,  $|f|$  es una aplicación lineal del simplex  $\sigma$  generado por  $v_0, \dots, v_n$  al simplex  $\tau$  generado por  $f(v_0), \dots, f(v_n)$ . Por ser  $|f| : \sigma \rightarrow \tau$  lineal en un espacio de dimensión finita, entonces es continua.

Ahora tan solo nos queda ver que  $|f| : |K| \rightarrow |L|$  es continua. Bien, pues por ser  $|f| : \sigma \rightarrow \tau$  continua, también lo es  $|f| : \sigma \rightarrow |L|$ . Finalmente por el [Lema 2.2](#),  $|f| : |K| \rightarrow |L|$  es continua.  $\square$

Consideremos las funciones de la forma de  $f$  descrita en [2.5](#). Para cualquier complejo  $K$ , existe una aplicación identidad  $\text{id}_K : K \rightarrow K$  que corresponde a la aplicación identidad en los vértices. Dadas tres aplicaciones  $f : K \rightarrow L$ ,  $g : L \rightarrow M$  y  $h : M \rightarrow N$ , la aplicación compuesta  $h \circ (g \circ f) = (h \circ g) \circ f$ , pues es una composición de aplicaciones de conjuntos que preserva simplices. Por lo tanto, existe una categoría de complejos simpliciales y estas funciones que denotaremos por **Csim**.

Por otro lado, veamos que el [Lema 2.5](#) nos garantiza la existencia de un funtor covariante entre esta categoría y los espacios topológicos.

**Proposición 2.9.** Existe un funtor covariante  $|\cdot| : \mathbf{Csim} \rightarrow \mathbf{Top}$  de la categoría de aplicaciones simpliciales a la categoría de espacios topológicos.

*Demostración.* Para cada complejo simplicial  $K$ , la identidad en **Csim** es la función identidad  $\text{id}_K : K \rightarrow K$ . La aplicación simplicial inducida  $|\text{id}_K| : |K| \rightarrow |K|$  es tal que

$$|\text{id}_K| \left( \sum_{i=0}^n t_i v_i \right) = \sum_{i=0}^n t_i i_K(v_i) = \sum_{i=0}^n t_i v_i,$$

lo cual es precisamente la identidad en el espacio topológico  $|K|$ . Esto muestra que  $|\cdot|$  preserva las identidades.

Sean ahora  $f : K \rightarrow L$  y  $g : L \rightarrow M$  dos morfismos en **Csim**. La composición en **Csim** es  $g \circ f : K \rightarrow M$ , y necesitamos demostrar que  $|(g \circ f)| = |g| \circ |f|$ . Para cualquier punto  $x = \sum_{i=0}^n t_i v_i$  en  $|K|$ ,

$$|(g \circ f)|(x) = \sum_{i=0}^n t_i (g \circ f)(v_i) = \sum_{i=0}^n t_i g(f(v_i)).$$

Por otro lado,

$$(|g| \circ |f|)(x) = |g| \left( |f| \left( \sum_{i=0}^n t_i v_i \right) \right) = |g| \left( \sum_{i=0}^n t_i f(v_i) \right) = \sum_{i=0}^n t_i g(f(v_i)).$$

Ambas expresiones son iguales y por tanto,  $|\cdot|$  preserva la composición de morfismos.  $\square$

Normalmente abusaremos de la notación de forma que escribiremos la aplicación simplicial inducida  $|f| : |K| \rightarrow |L|$  simplemente por  $f : |K| \rightarrow |L|$ .

**Lema 2.6.** *Supongamos que  $f : K^{(0)} \rightarrow L^{(0)}$  es una aplicación biyectiva tal que los vértices  $v_0, \dots, v_n$  de  $K$  generan un simplice de  $K$  si, y sólo si,  $f(v_0), \dots, f(v_n)$  generan un simplice de  $L$ . Entonces la aplicación simplicial inducida  $g : |K| \rightarrow |L|$  es un homeomorfismo. Diremos entonces que  $g$  es un **homeomorfismo simplicial** de  $K$  con  $L$ .*

*Demostración.* Por hipótesis, cada simplice  $\sigma \in K$  se identifica con otro simplice  $\tau \in L$ . Por tanto, debemos comprobar que la aplicación lineal  $h : \tau \rightarrow \sigma$  inducida por la correspondencia de vértices  $f^{-1}$  es la inversa de  $g : \sigma \rightarrow \tau$ . Si consideramos  $x = \sum_{i=0}^n t_i v_i$ , entonces por definición  $g(x) = \sum_{i=0}^n t_i f(v_i)$ . Luego

$$h(g(x)) = h\left(\sum_{i=0}^n t_i f(v_i)\right) = \sum_{i=0}^n t_i f^{-1}(v_i) = \sum_{i=0}^n t_i v_i = x$$

$\square$

## 2.5. Complejos simpliciales abstractos

Si bien la definición actual de los complejos simpliciales puede llegar a ser de gran utilidad, en la práctica muchas veces no es necesario usar las herramientas que nos proporciona la geometría afín. Es por ello que vamos a introducir una descripción puramente combinatoria de los complejos simpliciales que, aun siendo más simple, nos serán de gran utilidad a la hora de trabajar con espacios topológicos.

**Definición 2.16.** Un **complejo simplicial abstracto** (o simplemente **complejo abstracto**) es una colección  $\mathcal{S}$  de conjuntos finitos no vacíos tal que si  $A \in \mathcal{S}$ , entonces para todo  $B \subset A$  con  $B$  no vacío,  $B \in \mathcal{S}$ . Además, diremos que el complejo abstracto es **finito** si dicha colección es finita.

Al elemento  $A$  de  $\mathcal{S}$  lo llamaremos **simplice** de  $A \in \mathcal{S}$ . La **dimensión** de  $A$  es una menos que el número de elementos que le pertenecen. Todo subconjunto de  $A$  lo llamaremos **cara** de  $A$ . En cuanto a la **dimensión** de  $\mathcal{S}$ , diremos que es igual al máximo de las dimensiones de sus elementos o en caso de no haberlo, diremos que la dimensión de  $\mathcal{S}$  es infinita. El **conjunto de vértices**  $V$  de  $\mathcal{S}$  diremos que es la unión de elementos de  $\mathcal{S}$  que contienen un único punto. Llamaremos **subcomplejo** de  $\mathcal{S}$  a cualquier subcolección de  $\mathcal{S}$  que sea un complejo simplicial abstracto en sí.

Sean  $V_S, V_T$  los conjuntos de vértices de los complejos abstractos  $\mathcal{S}, \mathcal{T}$  respectivamente. Dos complejos abstractos  $\mathcal{S}$  y  $\mathcal{T}$  diremos que son **isomorfos** si existe una aplicación biyectiva  $f : V_S \rightarrow V_T$  tal que  $\{a_0, \dots, a_p\} \in \mathcal{S}$  si, y sólo si,  $\{f(a_0), \dots, f(a_p)\} \in \mathcal{T}$ .

**Definición 2.17.** Sean  $K$  un complejo simplicial y  $V$  su conjunto de vértices. Sea  $\mathcal{K}$  la colección de todos los subconjuntos  $\{a_0, \dots, a_p\} \subset V$  tales que los vértices  $a_0, \dots, a_p$  generan un simplice de  $K$ . Entonces llamaremos a la colección  $\mathcal{K}$  el **esquema de vértices** de  $K$ .

**Definición 2.18.** Si el complejo simplicial abstracto  $\mathcal{S}$  es isomorfo al esquema de vértices del complejo simplicial  $K$ , diremos que  $K$  es una **realización geométrica** de  $\mathcal{S}$ .

## 2. Símplices y complejos simpliciales

**Proposición 2.10.** *Sea  $\mathcal{S}$  un complejo simplicial abstracto finito de dimensión  $N$ . Entonces existe una realización geométrica de  $\mathcal{S}$  en  $\mathbb{R}^{2N+1}$ .*

*Demostración.* Consideremos un conjunto de puntos  $p_i \in \mathbb{R}^{2N+1}$  de forma que sus componentes son potencias de su índice  $i$ . Veamos que cualquier conjunto de  $2N + 2$  de estos puntos es afínmente independiente. Es decir, que los vectores formados por las diferencias entre estos puntos son linealmente independientes.

Para demostrarlo, consideremos un subconjunto de puntos  $\{p_{j_k} : 1 \leq k \leq 2N + 2\}$  de esta forma y analicemos el determinante de la matriz formada por los vectores correspondientes,

$$\begin{vmatrix} j_2 - j_1 & j_3 - j_1 & \cdots & j_{2N+2} - j_1 \\ j_2^2 - j_1^2 & j_3^2 - j_1^2 & \cdots & j_{2N+2}^2 - j_1^2 \\ \vdots & \vdots & \ddots & \vdots \\ j_2^{2n+1} - j_1^{2n+1} & j_3^{2n+1} - j_1^{2n+1} & \cdots & j_{2N+2}^{2n+1} - j_1^{2n+1} \end{vmatrix}.$$

Simplificando mediante operaciones elementales de fila, este determinante se transforma en el determinante de Vandermonde, cuyo valor es conocido y se calcula como el producto de las diferencias entre los términos seleccionados,

$$\prod_{1 \leq k < l \leq 2N+2} (j_k - j_l).$$

Este resultado no es cero siempre que todos los  $j_k$  sean distintos, asegurando así la independencia lineal.

Respecto a la construcción del complejo simplicial, tomemos un símplice abstracto  $A$  en  $\mathcal{S}$  con vértices  $\{v_{i_0}, v_{i_1}, \dots, v_{i_m}\}$  y consideremos el símplice geométrico  $\sigma_A = [p_{i_0}, p_{i_1}, \dots, p_{i_m}]$  en  $\mathbb{R}^{2N+1}$ . Dado que  $m + 1 \leq 2N + 2$ , el símplice  $\sigma_A$  tiene dimensión  $m$ . Definimos  $K$  como el conjunto que contiene todos los símplices  $\sigma_A$  para cada  $A \in \mathcal{S}$ . Veamos que la intersección de dos símplices  $\sigma_A$  y  $\sigma_B$  en  $K$  es igual a  $\sigma_{A \cap B}$  con  $A, B \in \mathcal{S}$ . Consideremos  $\tau$  como el símplice en  $\mathbb{R}^{2N+1}$  cuyos vértices son la unión de los vértices pertenecientes a  $\sigma_A$  y a  $\sigma_B$ , lo cual es posible ya que la suma de sus dimensiones no supera  $2N$ . De esta manera, la intersección  $\sigma_A \cap \sigma_B$  resulta ser la cara de  $\tau$  determinada por los vértices que  $\sigma_A$  y  $\sigma_B$  comparten, es decir, aquellos asociados a  $A \cap B$ . Concluimos entonces que  $\sigma_A \cap \sigma_B = \sigma_{A \cap B}$ .  $\square$

Como consecuencia inmediata de la proposición anterior y del [Lema 2.6](#), tenemos el siguiente corolario.

**Corolario 2.3.** *Las siguientes afirmaciones son ciertas:*

- (a) *Todo complejo abstracto finito  $\mathcal{S}$  es isomorfo al esquema de vértices de algún complejo simplicial  $K$ .*
- (b) *Dos complejos simpliciales son afínmente isomorfos si, y sólo si, sus esquemas de vértices son isomorfos como complejos simpliciales abstractos.*

## 3. Homología simplicial

Este capítulo se centra en la homología simplicial, una rama de estudio crucial de la topología algebraica que utiliza complejos simpliciales para analizar y comprender la estructura de espacios topológicos triangulables. Tras explorar los fundamentos del álgebra homológica y la teoría de complejos simpliciales, ahora profundizamos en las propiedades teóricas y aplicaciones prácticas de la homología simplicial siguiendo los contenidos de [82].

### 3.1. Homología simplicial orientada

Dado un simplice  $\sigma$ , podemos definir un orden sobre sus vértices. Dos órdenes de  $\sigma$  los consideraremos equivalentes si podemos pasar de uno a otro con un número par de permutaciones. Además, en el caso donde  $\sigma$  sea un 0-simplice, claramente existe una única orientación. Así, los ordenamientos posibles para los vértices de  $\sigma$  se pueden agrupar en dos clases de equivalencia distintas, que definimos como las **orientaciones del simplice  $\sigma$** .

**Definición 3.1.** Decimos que un simplice  $\sigma = [a_0, a_1, \dots, a_p]$  está **orientado** si se le ha asignado una de estas orientaciones. Utilizaremos  $[a_0 a_1 \dots a_p]$  para denotar la clase de equivalencia dada por la orientación  $a_0 < a_1 < \dots < a_p$  del simplice generado por los vértices  $a_0, a_1, \dots, a_p$ .

Consideremos  $\Sigma_p$  el conjunto de todos los simplices de dimensión  $p$  de un complejo simplicial geométrico  $K$  y el conjunto de sus clases de equivalencia por la relación de orientación. Para cada  $\sigma \in \Sigma_p$ , definimos  $\Sigma_p^+$  y  $\Sigma_p^-$  como los conjuntos que contienen, respectivamente, un simplice orientado  $\sigma^+$  y el simplice con orientación opuesta  $\sigma^-$ . En lo que sigue,  $R$  siempre será un **anillo unitario conmutativo**, a menos que se indique de manera explícita lo contrario.

**Definición 3.2.** Sea  $K$  un complejo simplicial y sea  $R$  un anillo. Consideremos los conjuntos definidos anteriormente. Definimos el  **$R$ -módulo de las  $p$ -cadenas simpliciales orientadas** de  $K$ ,  $C_p(K; R)$ , como el cociente del  $R$ -módulo libre generado por  $\Sigma_p^+ \cup \Sigma_p^-$  sobre el submódulo generado por el conjunto  $\{\sigma^+ + \sigma^- : \sigma \in \Sigma_p\}$ . Esto es,

$$C_p(K; R) = \frac{R\langle \Sigma_p^+ \cup \Sigma_p^- \rangle}{\langle \sigma^+ + \sigma^- : \sigma \in \Sigma_p \rangle}.$$

Para  $p < 0$  o  $p > \dim(K)$ , definimos  $C_p(K; R)$  como el  $R$ -módulo trivial.

El interés de definir el  $R$ -módulo de  $p$ -cadenas simpliciales orientadas radica tanto en la identificación de los elementos que contiene como en las operaciones algebraicas aplicables sobre ellos. Esta construcción nos permite manejar un simplice orientado y su opuesto como opuestos algebraicos en un marco formal. Veámoslo.

Nuestro objetivo es demostrar que efectivamente

$$\frac{R\langle \Sigma_p^+ \cup \Sigma_p^- \rangle}{\langle \sigma^+ + \sigma^- : \sigma \in \Sigma_p \rangle} \cong R\langle \tilde{\Sigma}_p \rangle,$$

### 3. Homología simplicial

donde  $\tilde{\Sigma}_p$  representa el conjunto de  $p$ -símplices en  $\Sigma_p$  con una orientación arbitrariamente fija para cada uno.

Para ello, definamos la aplicación  $f : \Sigma_p^+ \cup \Sigma_p^- \rightarrow R\langle \tilde{\Sigma}_p \rangle$ . Esta aplicación asigna a cada símplice orientado  $\sigma^+$  en  $\Sigma_p^+$ , un representante  $\sigma$  en  $R\langle \tilde{\Sigma}_p \rangle$  con una orientación fija elegida arbitrariamente, y a cada  $\sigma^-$  en  $\Sigma_p^-$ , le asigna  $-\sigma$  en  $R\langle \tilde{\Sigma}_p \rangle$ , donde  $-\sigma$  refleja el elemento opuesto de  $\sigma$ .

La aplicación  $f$  respeta las relaciones de orientación al asignar a símplices con orientaciones opuestas a elementos que son opuestos algebraicos en  $R\langle \tilde{\Sigma}_p \rangle$ . Por la **Propiedad universal de los módulos libres**, esta aplicación induce un homomorfismo  $\tilde{f} : R\langle \Sigma_p^+ \cup \Sigma_p^- \rangle \rightarrow R\langle \tilde{\Sigma}_p \rangle$  que resulta ser sobreyectivo, ya que cada elemento en  $R\langle \tilde{\Sigma}_p \rangle$  tiene al menos una preimagen en  $R\langle \Sigma_p^+ \cup \Sigma_p^- \rangle$ .

Por definición de  $f$ , para cada elemento de la forma  $\sigma^+ + \sigma^-$  en  $\langle \sigma^+ + \sigma^- : \sigma \in \Sigma_p \rangle$ , tenemos que  $\tilde{f}(\sigma^+ + \sigma^-) = f(\sigma^+) + f(\sigma^-) = \sigma - \sigma = 0$ , demostrando que todo el submódulo  $\langle \sigma^+ + \sigma^- : \sigma \in \Sigma_p \rangle$  tiene imagen cero por  $\tilde{f}$  y, por ende, está contenido en el núcleo de  $\tilde{f}$ .

Además, si consideramos un elemento  $x$  en  $R\langle \Sigma_p^+ \cup \Sigma_p^- \rangle$  tal que  $\tilde{f}(x) = 0$ , este elemento puede expresarse como una combinación lineal de elementos en  $\Sigma_p^+$  y  $\Sigma_p^-$ . La condición  $\tilde{f}(x) = 0$  implica que la suma de las imágenes bajo  $f$  de los términos en esta combinación lineal debe ser cero en  $R\langle \tilde{\Sigma}_p \rangle$ . Esto solo ocurre si para cada  $\sigma$ , la suma total de los coeficientes correspondientes a  $\sigma^+$  y  $\sigma^-$  es cero, lo que significa que cada término en  $x$  que contribuye a esta suma cero debe ser de la forma  $\sigma^+ + \sigma^-$  o un múltiplo de este, luego  $\tilde{f}(x) = 0$  implica que  $x \in \langle \sigma^+ + \sigma^- : \sigma \in \Sigma_p \rangle$ .

Por tanto, el núcleo de  $\tilde{f}$  coincide precisamente con  $\langle \sigma^+ + \sigma^- : \sigma \in \Sigma_p \rangle$ , y aplicando el **Primer teorema de isomorfía**, concluimos que

$$\frac{R\langle \Sigma_p^+ \cup \Sigma_p^- \rangle}{\langle \sigma^+ + \sigma^- : \sigma \in \Sigma_p \rangle} \cong R\langle \tilde{\Sigma}_p \rangle,$$

estableciendo la estructura algebraica deseada y completando la prueba.

*Observación 3.1.* En particular, la anterior construcción asigna a cada símplice orientado una cadena cuyo coeficiente del anillo es 1, 0 o  $-1$ . A estas cadenas las llamaremos  **$p$ -cadenas elementales**. En ocasiones abusaremos de la notación para designar por  $\sigma$  a la cadena elemental respectiva del símplice orientado  $\sigma$ .

**Definición 3.3.** Sea  $K$  un complejo simplicial y sean  $C_p(K; R), C_{p-1}(K; R)$   $R$ -módulos de  $p$ -cadenas. Definimos el **operador borde de  $p$ -cadenas** como el homomorfismo  $\partial_p : C_p(K; R) \rightarrow C_{p-1}(K; R)$  tal que

$$\partial_p(\sigma) = \partial_p([v_0, v_1, \dots, v_p]) = \sum_{i=0}^p (-1)^i [v_0, \dots, \hat{v}_i, \dots, v_p].$$

donde  $\hat{v}_i$  denota el vértice a eliminar.

**Lema 3.1.** El operador borde  $\partial_p : C_p(K; R) \rightarrow C_{p-1}(K; R)$  está bien definido. En particular, si  $\sigma^+$  y  $\sigma^-$  son las dos orientaciones del  $p$ -símplice  $\sigma$ , tenemos que

$$\partial_p(\sigma^+ + \sigma^-) = 0$$

*Demostración.* Probaremos que la suma de la imagen por el operador borde de  $\sigma^+ = [v_0 v_1 \dots v_p]$

y  $\sigma^- = [v_1 v_0 \dots v_p]$  es igual a 0. Para ello, observamos que

$$\begin{aligned}\partial_p \sigma^+ &= [v_1 v_2 \dots] - [v_0 v_2 \dots] + \sum_{i \neq 0,1} (-1)^i [v_0 v_1 \dots \hat{v}_i \dots v_p], \\ \partial_p \sigma^- &= [v_0 v_2 \dots] - [v_1 v_2 \dots] + \sum_{i \neq 0,1} (-1)^i [v_1 v_0 \dots \hat{v}_i \dots v_p].\end{aligned}$$

Al sumar ambas expresiones, los dos primeros términos de  $\partial_p \sigma^+$  y  $\partial_p \sigma^-$  se cancelan entre sí. Como consecuencia de la definición de  $C_{p-1}(K; R)$ , los términos restantes definen orientaciones opuestas del mismo simplice por lo que se cancelan y  $\partial_p(\sigma^+ + \sigma^-) = 0$ .  $\square$

**Lema 3.2.** Sean  $\partial_p : C_{p+1}(K; R) \rightarrow C_p(K; R)$ ,  $\partial_p : C_p(K; R) \rightarrow C_{p-1}(K; R)$  operadores borde. Entonces  $\partial_p \circ \partial_{p+1} = 0$ .

*Demostración.*

$$\begin{aligned}\partial_p \partial_{p+1}[v_0, \dots, v_{p+1}] &= \partial_p \left( \sum_{i=0}^{p+1} (-1)^i [v_0 \dots \hat{v}_i \dots v_{p+1}] \right) \\ &= \sum_{i=0}^{p+1} (-1)^i \left[ \sum_{j>i}^{p+1} (-1)^j [v_0 \dots, \hat{v}_i \dots \hat{v}_j \dots v_{p+1}] + \sum_{j=0}^{i-1} (-1)^j [v_0 \dots \hat{v}_j \dots \hat{v}_i \dots v_{p+1}] \right].\end{aligned}$$

Es decir, el simplice  $[v_0 \dots, \hat{v}_k \dots, \hat{v}_t \dots, v_{p+1}]$  aparece dos veces en la anterior expresión con signos opuestos, donde  $k, t \in \{0, \dots, p+1\}$ . Esto nos lleva a discutir los siguientes casos. Supongamos sin pérdida de generalidad que  $k < t$ . En el primer caso,  $i = k < j = t$  donde el coeficiente es  $(-1)^k(-1)^{t-1}$ . En el segundo caso,  $i = t > j = k$  con coeficiente  $(-1)^t(-1)^k$ . Concluimos por tanto que todo simplice de la expresión se anula y al anularse sobre los generadores,  $\partial_{p-1} \partial_p$  es el homomorfismo nulo.  $\square$

**Definición 3.4.** El complejo de cadenas positivo  $C_\bullet(K; R) = \{C_p(K; R), \partial_p\}$  lo llamaremos **complejo de cadenas simpliciales** de  $K$ . La homología de dicho complejo la notaremos por  $H_p(K; R)$  y lo llamaremos  **$p$ -ésimo  $R$ -módulo de homología** de  $K$ .

Si  $R = \mathbb{Z}$ , el módulo  $H_p(K; \mathbb{Z})$  lo notaremos simplemente por  $H_p(K)$  y diremos que es el  **$p$ -ésimo grupo de homología** de  $K$ .

**Proposición 3.1.** Sea  $K$  un complejo simplicial no vacío. Entonces el complejo de cadenas positivo  $\{C_p(K; R), \partial_p\}$  admite un aumento.

*Demostración.* Sea  $\varepsilon : C_0(K; R) \rightarrow R$  el homomorfismo que extiende linealmente  $\varepsilon(v) = 1$  para todo vértice  $v \in K$ . Veamos que  $\varepsilon \circ \partial_1 : C_1(K; R) \rightarrow R$  es nulo. Tomando  $[v_0, v_1] \in C_1(K; R)$  obtenemos que  $\varepsilon(\partial_1[v_0, v_1]) = \varepsilon(v_1 - v_0) = 1 - 1 = 0$ , como queríamos ver.  $\square$

**Definición 3.5.** Sea  $\tilde{C}_\bullet(K; R)$  el complejo aumentado del complejo de cadenas simpliciales  $C_\bullet(K; R)$ . Denominaremos  **$p$ -ésimo módulo de homología reducida** de  $K$  al módulo de homología  $H_p(\tilde{C}_\bullet; R)$  y lo denotaremos por  $\tilde{H}(K; R)$ .

**Proposición 3.2.** Sean  $K$  y  $L$  dos complejos simpliciales junto con una aplicación simplicial  $f : |K| \rightarrow |L|$ . Esta aplicación induce un homomorfismo entre los complejos de cadenas,  $C(f)$ , el cual se define extendiendo linealmente la función

$$C(f)([v_0 \dots v_p]) = \begin{cases} [f(v_0) \dots f(v_p)] & \text{si los vértices son distintos entre sí,} \\ 0 & \text{en caso contrario.} \end{cases}$$

### 3. Homología simplicial

En particular, si  $f$  es la identidad, entonces  $C(f)$  es simplemente la identidad también. Además, si  $g : |L| \rightarrow |M|$  es otra aplicación simplicial, se cumple que  $C(g \circ f) = C(g) \circ C(f)$ .

*Demostración.* Para demostrar esto, primero observamos que la definición de  $C(f)$  es independiente de la orientación de los simplices. Luego, verificamos la igualdad  $\partial_p \circ C(f) = C(f) \circ \partial_p$ . Si no hay vértices repetidos, se tiene que:

$$\begin{aligned} C(f)\partial_p([v_0 \dots v_p]) &= C(f) \left( \sum_{i=0}^p (-1)^i [v_0 \dots \hat{v}_i \dots v_p] \right) = \\ &\sum_{i=0}^p (-1)^i [f(v_0) \dots \widehat{f(v_i)} \dots f(v_p)] = \partial_p C(f)([v_0 \dots v_p]). \end{aligned}$$

Si hay vértices repetidos, digamos  $f(v_i) = f(v_j)$ , entonces  $\partial_p C(f)([v_0 \dots v_p]) = 0$ . Por otro lado,

$$\sum_{i=0}^p (-1)^i C(f)([v_0 \dots \hat{v}_i \dots v_p]) = 0$$

debido a que  $C(f)([v_0 \dots \hat{v}_k \dots v_p]) = 0$  para  $k \neq i, j$  y cuando  $i < j$ ,

$$(-1)^i [f(v_0) \dots \widehat{f(v_i)} \dots f(v_j) \dots f(v_p)] + (-1)^j [f(v_0) \dots f(v_i) \dots \widehat{f(v_j)} \dots f(v_p)] = 0$$

también se anula. Esto se debe a que si no hay más vértices repetidos, como  $f(v_i) = f(v_j)$ , el número de trasposiciones necesarias para cambiar de un simplex orientado al otro es  $j - i - 1$ , dado que  $f(v_j)$  ocupa el lugar  $j - 1$  en el primer simplex. La fórmula  $C(g \circ f) = C(g)C(f)$  se sigue directamente de la definición de  $C(f)$ .  $\square$

*Observación 3.2.* El resultado anterior nos garantiza que  $C : \mathbf{Csim} \rightarrow R\text{-Ch}_\bullet$  es un functor covariante entre la categoría de complejos simpliciales y la categoría de complejos de cadenas.

**Definición 3.6.** Sea  $f : |K| \rightarrow |L|$  una aplicación simplicial y sea  $C(f) : C_\bullet(K; R) \rightarrow C_\bullet(L; R)$  una aplicación de cadenas definida como en la [Proposición 3.2](#). Llamaremos a  $C(f)$  la **aplicación de cadenas inducida por  $f$**  y la notaremos por  $f_*$ .

**Corolario 3.1.** *Toda aplicación simplicial inducida  $f : |K| \rightarrow |L|$  induce un homomorfismo de  $R$ -módulos*

$$H_p(f) : H_p(K; R) \rightarrow H_p(L; R)$$

que notaremos por  $f_*$  y que cumple que si  $g : |L| \rightarrow |M|$  es otra aplicación simplicial, entonces  $(g \circ f)_* = g_* \circ f_*$  e  $\text{id}_* = \text{id}$ .

*Observación 3.3.* La última implicación del corolario se traduce en que tenemos un functor covariante que va de la categoría de complejos simpliciales con los homeomorfismos simpliciales a la categoría de  $R$ -módulos con sus homomorfismos.

**Lema 3.3.** *La aplicación de cadenas  $f_\# : C_\bullet(K; R) \rightarrow C_\bullet(L; R)$  preserva el homomorfismo de aumento y como resultado, induce un homomorfismo  $f_*$  de módulos de homología reducida.*

*Demostración.* Sea  $f : |K| \rightarrow |L|$  una aplicación simplicial,  $f_\#$  su aplicación de cadenas inducida y sean  $\varepsilon : C_0(K; R) \rightarrow R$ ,  $\varepsilon : C_0(L; R) \rightarrow R$  aumentos de  $C_\bullet(K; R), C_\bullet(L; R)$  respectivamente. Llamemos indistintamente  $\varepsilon$  a ambos aumentos en función del dominio en el que nos encontramos. Ahora definamos  $\varepsilon(f_\#(v)) = 1$  y  $\varepsilon(v) = 1$  para todo vértice de  $K$  y

extendamos por linealidad. Por consiguiente  $\varepsilon \circ f_\# = \varepsilon$ . Esta ecuación implica que  $f_\#$  lleva el núcleo de  $\varepsilon_K : C_0(K; R) \rightarrow R$  al núcleo de  $\varepsilon_L : C_0(L; R) \rightarrow R$ , lo que induce un homomorfismo  $f_* : \tilde{H}_0(K; R) \rightarrow \tilde{H}_0(L; R)$ .  $\square$

**Teorema 3.1.** Sean  $f, g$  aplicaciones simpliciales de  $K$  a  $L$ ;  $f_\#, g_\#$  sus aplicaciones de cadenas inducidas y sea  $s : f_\# \rightarrow g_\#$  una homotopía de cadenas entre ellas. Entonces los homomorfismos inducidos  $f_*, g_*$  para sus módulos de homología son iguales.

*Demostración.* Sea  $z$  un  $p$ -ciclo de  $K$ . Entonces

$$g_*(z) - f_*(z) = \partial sz + s\partial z = \partial sz + 0$$

por lo que  $f(z)$  y  $g(z)$  tienen la misma clase de homología. Por tanto,  $f_*([z]) = g_*([z])$  como se quería.  $\square$

## 3.2. Homología del complejo cono

A continuación, exploraremos un nuevo complejo simplicial que construiremos a partir de otro dado. El complejo cono nos facilitará la obtención de algunos resultados relevantes en homología.

**Definición 3.7.** Sea  $K$  un complejo simplicial de  $\mathbb{R}^N$  y sea  $w \in \mathbb{R}^N$  tal que cada semirrecta con origen  $w$  corta a  $|K|$  a lo sumo en un punto. Definimos el **cono sobre  $K$  con vértice  $w$**  como el conjunto cuyos elementos son los simplices de  $K$  o simplices de la forma  $[w, v_0, \dots, v_p]$ , donde  $[v_0, \dots, v_p] \in K$ . Lo denotaremos por  $w * K$ .

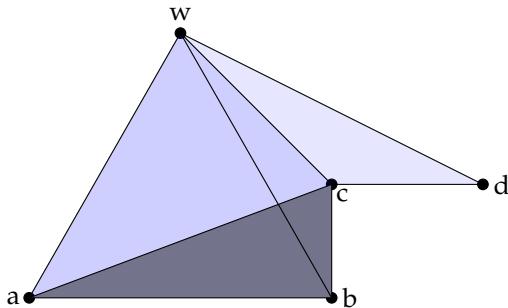


Figura 3.1.: Cono sobre el complejo formado por el 2-símplex  $[a, b, c]$ , el 1-símplex  $[c, d]$  y todas sus caras con vértice  $w$ .

**Lema 3.4.** El cono  $w * K$  es un complejo simplicial.

*Demostración.* Sea  $\sigma = [v_0, \dots, v_p]$  un símplex de  $K$ . Primero veamos que el conjunto  $\{w, v_0, \dots, v_p\}$  es afínmente independiente. Si  $w$  perteneciera al plano  $P$  generado por los puntos  $v_0, \dots, v_p$ , podríamos considerar el segmento que une  $w$  con un punto de  $x \in \text{Int } \sigma$ . Dicho conjunto, por ser abierto en  $P$ , contendría un intervalo de puntos en el segmento, contradiciendo la hipótesis de que las semirrectas que parten de  $w$  cortan a lo sumo en un punto a  $|K|$ .

Veamos ahora que  $w * K$  es un complejo simplicial. Los simplices de  $w * K$  pueden ser de tres tipos:

### 3. Homología simplicial

1. Símplices  $[v_0, \dots, v_p]$  pertenecientes a  $K$ .
2. Símplices de la forma  $[w, v_0, \dots, v_p]$ .
3. El 0-símplex  $[w]$ .

Si  $\sigma, \tau$  son símplices del primer tipo, entonces  $\text{Int } \sigma \cap \text{Int } \tau = \emptyset$  puesto que  $K$  es un complejo simplicial. El símplex  $\text{Int}[w, v_0, \dots, v_p]$  es la unión de todos los segmentos abiertos que unen  $w$  con  $v_0, \dots, v_p$ , luego dos símplices de esta forma tienen intersección vacía pues las semirectas que parten de  $w$  cortan a  $K$  a lo sumo en un punto. Finalmente, si  $\sigma$  es del primer tipo y  $\tau$  del segundo,  $\text{Int } \sigma \cap \text{Int } \tau = \emptyset$  por el mismo argumento recién dado.  $\square$

**Proposición 3.3.** *Sea  $K$  un complejo simplicial y sea  $w * K$  el cono sobre  $K$  de vértice  $w$ . Entonces la homología orientada de  $w * K$  es  $H_p(w * K; R) = 0$  para todo  $p \neq 0$  y  $H_0(w * K; R) \cong R$ . En el caso de la homología reducida,  $\tilde{H}_0(w * K; R) = 0$  para todo  $p \in \mathbb{Z}$ .*

*Demostración.* Sea  $D_\bullet = \{D_p, \partial_p\}$  un complejo de cadenas tal que  $D_p = 0$  para todo  $p \neq 0$  y  $D_0 = R$ . Definimos la aplicación de cadenas  $f : D_\bullet \rightarrow C_\bullet(w * K; R)$  de forma que  $f_p = 0$  para todo  $p \neq 0$  y  $f_0(r) = rw$ . Por otro lado, por la [Proposición 3.1](#) podemos definir el aumento  $\varepsilon : C_\bullet(w * K; R) \rightarrow D_\bullet$  dado por  $\varepsilon_p = 0$  para todo  $p \neq 0$  y  $\varepsilon_0(v) = 1$  para todo vértice  $v$  del cono. Nuestro objetivo es ver que efectivamente  $f$  es una equivalencia de cadenas junto a  $\varepsilon$ . De manera directa tenemos que  $\varepsilon \circ f = \text{id}_D$ , luego  $\varepsilon \circ f \simeq \text{id}_D$ . Veamos ahora que  $f \circ \varepsilon$  es homotópica a la identidad. Para ello vamos a definir  $s$  como la familia  $\{s_p\}$  de homomorfismos  $s_p : C_p(w * K; R) \rightarrow C_{p+1}(w * K; R)$  tal que

$$s_p([v_0 \dots v_p]) = \begin{cases} [wv_0 \dots v_p] & \text{si } v_i \neq w \quad 0 \leq i \leq p, \quad p \geq 0, \\ 0 & \text{en caso contrario,} \end{cases}$$

induce una extensión lineal. Dicha familia está bien definida para  $C_p(w * K; R)$ . Veamos que  $\partial_{p+1}s_p + s_{p-1}\partial_p = \text{id}_{C_p(w * K; R)} - f_p\varepsilon_p$  se cumple, por lo que  $s$  es una homotopía de cadenas. Para el caso en que  $p \in \mathbb{Z}$  es menor que 0, se cumple de manera trivial. Si  $p = 0$ , distinguimos dos casos. Cuando  $v \neq w$  tenemos que  $(\partial_1 s_0 + s_{-1}\partial_0)(v) = \partial_1[w, v] = v - w = (\text{id}_0 - f_0\varepsilon_0)(v)$ . Por el contrario si  $v = w$ ,  $(\partial_1 s_0 + s_{-1}\partial_0)(v) = 0$  y también  $(\text{id}_0 - f_0\varepsilon_0)(v) = \text{id}_0(w) - (f_0\varepsilon_0)(w) = w - w = 0$ . Por último, veamos que sucede cuando  $p > 0$ . Supongamos primero que  $w \neq v_i$ . Entonces

$$\begin{aligned} (\partial_{p+1}s_p + s_{p-1}\partial_p)[v_0 \dots v_p] &= \partial_{p+1}[wv_0 \dots v_p] + s_{p-1} \left( \sum_{i=0}^p (-1)^i [v_0 \dots \hat{v}_i \dots v_p] \right) \\ &= [v_0 \dots v_p] + \sum_{i=0}^p (-1)^{i+1} [wv_0 \dots \hat{v}_i \dots v_p] + \sum_{i=0}^p (-1)^i [wv_0 \dots \hat{v}_i \dots v_p] \\ &= [v_0 \dots v_p] = (\text{id}_{C_p} - f_p\varepsilon_p)[v_0 \dots v_p]. \end{aligned}$$

Finalmente si  $w = v_{i_0}$  para algún  $i_0$  entonces

$$\begin{aligned} (\partial_{p+1}s_p + s_{p-1}\partial_p)[v_0 \dots v_p] &= s_{p-1}\partial_p[v_0 \dots v_p] = s_{p-1} \left( \sum_{i=0}^{p-1} (-1)^i [v_0 \dots \hat{v}_i \dots v_p] \right) \\ &= (-1)^{i_0} s_{p-1}[v_0 \dots \hat{v}_{i_0} \dots v_p] = (-1)^{i_0} [wv_0 \dots \hat{v}_{i_0} \dots v_p] \\ &= (-1)^{i_0} [v_{i_0} v_0 \dots \hat{v}_{i_0} \dots v_p] = [v_0 \dots v_p]. \end{aligned}$$

Es decir,  $f \circ \varepsilon \simeq \text{id}_{C(w*K; R)}$  y por el **Corolario 1.2** induce un isomorfismo  $\varepsilon_* : H_p(w*K; R) \rightarrow H_p(D; R)$ .

Para el caso reducido consideremos el complejo aumentado  $D_\bullet$  dado por el aumento  $\text{id}_R : D_0 \rightarrow R$ . Como consecuencia, la homología de  $\tilde{D}$  es trivial. Además, podemos extender los homomorfismos  $\varepsilon$  y  $f$  a homomorfismos  $\tilde{\varepsilon}$  y  $\tilde{f}$  para los complejos aumentados de forma que  $\tilde{\varepsilon}_{-1} = \tilde{f}_{-1} = \text{id}_R$ . Por la misma homotopía  $s$  obtenemos que  $\tilde{\varepsilon}$  y  $\tilde{f}$  son equivalencias homotópicas entre los complejos aumentados y por tanto,  $\tilde{H}_p(w*K; R) = 0$  para todo  $p \in \mathbb{Z}$ .  $\square$

**Corolario 3.2.** *La homología simplicial reducida de cualquier simplice es nula.*

**Corolario 3.3.** *Sea  $\sigma$  un  $n$ -simplice y sea  $\text{Bd } \sigma$  su borde. Entonces  $\tilde{H}_p(\text{Bd } \sigma; R) = 0$  es trivial si  $p \neq n-1$  y  $\tilde{H}_{n-1}(\text{Bd } \sigma; R) \cong R$ . Además, para el caso no trivial, un generador es la clase de la cadena  $\partial(\sigma)$ .*

*Demostración.* Dado el simplice anterior, los complejos de cadenas aumentados de  $\sigma$  y su borde coinciden hasta dimensión  $p \leq n-1$ . Por el **Corolario 3.2** deducimos que  $\tilde{H}_p(\text{Bd } \sigma; R) = 0$  para  $p \leq n-2$ . Además,  $C_p(\text{Bd } \sigma; R) = 0$  para  $p \geq n$ . Por lo tanto,  $\tilde{H}_{n-1}(\text{Bd } \sigma; R) = \ker \partial_{n-1}$ . Aquí,  $\partial_{n-1}$  representa el operador borde en ambos complejos aumentados (es decir,  $\partial_0 = \varepsilon$  indica el aumento). Dado que el complejo aumentado de  $\sigma$  tiene homología trivial, entonces  $\ker \partial_{n-1} = \text{Im } \partial_n$ , y además  $\partial_n$  es inyectivo donde el operador borde  $\partial_n : C_n(\sigma; R) \rightarrow C_{n-1}(\sigma; R) = C_{n-1}(\text{Bd } \sigma; R)$  aparece en el complejo de  $\sigma$ . Puesto que  $C_n(\sigma; R)$  es isomorfo a  $R$  generado por  $\sigma$ , se sigue que  $\text{Im } \partial_n = R$ , y por tanto  $\tilde{H}_{n-1}(\text{Bd } \sigma; R)$ , es isomorfo a  $R$  generado por  $\partial(\sigma)$ .  $\square$

### 3.3. Sucesión de Mayer-Vietoris

Nombrada en honor a los matemáticos austriacos Walther Mayer y Leopold Vietoris, la sucesión de Mayer-Vietoris es una herramienta esencial en la topología algebraica y la teoría de homología. Esta sucesión permite analizar la homología de un complejo simplicial a partir de la homología de sus subcomplejos, de manera análoga a como el teorema de Seifert-van Kampen describe el grupo fundamental de un espacio topológico a partir de subespacios abiertos y conexos por arcos.

**Lema 3.5** (Lema de la serpiente). *Sean  $A_\bullet = \{A_n, \partial_A\}$ ,  $B_\bullet = \{B_n, \partial_B\}$  y  $C_\bullet = \{C_n, \partial_C\}$  complejos de cadenas y sean  $f, g$  aplicaciones de cadenas tales que la sucesión*

$$0 \rightarrow A_\bullet \xrightarrow{f} B_\bullet \xrightarrow{g} C_\bullet \rightarrow 0$$

*es exacta. Existe entonces una sucesión exacta de homología*

$$\begin{array}{ccccccc} \dots & \longrightarrow & H_p(A_\bullet; R) & \xrightarrow{f_*} & H_p(B_\bullet; R) & \xrightarrow{g_*} & H_p(C_\bullet; R) \\ & & & & & & \boxed{\quad} \\ & & \xrightarrow{f_*} & H_{p-1}(A_\bullet; R) & \xrightarrow{\partial_*} & H_{p-1}(B_\bullet; R) & \xrightarrow{g_*} H_{p-1}(C_\bullet; R) \longrightarrow \dots \end{array}$$

*donde  $\partial_*$  es el operador borde inducido en  $B_\bullet$  que llamaremos **operador conector**.*

### 3. Homología simplicial

*Demostración.* Para realizar esta prueba usaremos una persecución de diagramas. Usaremos el siguiente diagrama como guía:

$$\begin{array}{ccccccc}
 0 & \longrightarrow & A_{p+1} & \xrightarrow{f} & B_{p+1} & \xrightarrow{g} & C_{p+1} \longrightarrow 0 \\
 & & \downarrow \partial_A & & \downarrow \partial_B & & \downarrow \partial_C \\
 0 & \longrightarrow & A_p & \xrightarrow{f} & B_p & \xrightarrow{g} & C_p \longrightarrow 0 \\
 & & \downarrow \partial_A & & \downarrow \partial_B & & \downarrow \partial_C \\
 0 & \longrightarrow & A_{p-1} & \xrightarrow{f} & B_{p-1} & \xrightarrow{g} & C_{p-1} \longrightarrow 0
 \end{array}$$

*Paso 1.* Para definir el operador conector  $\partial_*$ , primero tenemos que comprobar que si tenemos un ciclo de  $C_p$ , entonces podemos asignarle un único ciclo en  $A_{p-1}$ . Por tanto, sea  $c_p$  un ciclo de  $C_p$  (esto es,  $c_p \in \ker \partial_C$ ) y escojamos  $b_p \in B_p$  tal que  $g(b_p) = c_p$  (recordemos que  $g$  es sobreyectiva por ser la sucesión exacta corta). El elemento  $\partial_B b_p$  de  $B_{p-1}$  pertenece al núcleo de  $g$  pues  $g(\partial_B b_p) = \partial_C g(b_p) = \partial_C c_p = 0$ . Por tanto, existe un elemento  $a_{p-1} \in A_{p-1}$  tal que  $f(a_{p-1}) = \partial_B b_p$ , pues  $\ker g = \text{Im } f$ . Tenemos que dicho elemento es único por ser  $f$  inyectiva. Además,  $a_{p-1}$  es un ciclo. Como  $f(\partial_A a_{p-1}) = \partial_B f(a_{p-1}) = \partial_B \partial_B b_p = 0$ , entonces  $\partial_A a_{p-1} = 0$  por ser  $f$  inyectiva. Definimos  $\partial_*[c_p] = [a_{p-1}]$  donde los corchetes denotan la clase de homología.

*Paso 2.* Queremos probar ahora que  $\partial_*$  es un homomorfismo de módulos bien definido. Sean  $c_p, c'_p$  dos elementos del núcleo de  $\partial_C : C_p \rightarrow C_{p-1}$ . Sean  $b_p, b'_p$  elementos de  $B_p$  tal que  $g(b_p) = c_p$  y  $g(b'_p) = c'_p$ . Escojamos ahora  $a_{p-1}$  y  $a'_{p-1}$  tal que  $f(a_{p-1}) = \partial_B b_p$  y  $f(a'_{p-1}) = \partial_B b'_p$ .

Para probar que  $\partial_*$  está bien definido, veamos que no depende del  $b_p$  y  $c_p$  escogido. Supongamos que  $c_p \sim c'_p$  y veamos entonces que  $a_{p-1}$  y  $a'_{p-1}$  también lo son. Por tanto, supongamos que  $c_p - c'_p = \partial_C c_{p+1}$ . Escogemos  $b_{p+1}$  tal que  $g(b_{p+1}) = c_{p+1}$ . Esto implica que

$$f(b_p - b'_p - \partial_B b_{p+1}) = c_p - c'_p - \partial_C g(b_{p+1}) = c_p - c'_p - \partial_C c_{p+1} = 0.$$

En consecuencia, podemos tomar  $a_p$  tal que  $f(a_p) = b_p - b'_p - \partial_B b_{p+1}$  luego

$$f(\partial_A a_p) = \partial_B f(a_p) = \partial_B(b_p - b'_p) - 0 = f(a_{p-1} - a'_{p-1}).$$

Por ser  $f$  inyectiva,  $\partial_A a_p = a_{p-1} - a'_{p-1}$ , como buscábamos.

Ya sabemos que  $\partial_*$  está bien definido, veamos que es un homomorfismo de módulos. Para ello basta fijarnos en que  $g(b_p + b'_p) = c_p + c'_p$  y que  $f(a_{p-1} + a'_{p-1}) = \partial_B(b_p + b'_p)$ . Por tanto  $\partial_*[c_p + c'_p] = [a_{p-1} + a'_{p-1}]$  por definición y en consecuencia,  $\partial_*[c_p + c'_p] = \partial_*[c_p] + \partial_*[c'_p]$ . Ahora si  $\lambda \in R$ , de manera análoga obtenemos que  $\lambda \partial_*[b_p] = \lambda [c_p] = [\lambda c_p] = \partial_*[\lambda b_p]$ .

*Paso 3.* Probaremos la exactitud de  $H_p(B_\bullet; R)$  por doble inclusión. Como  $g \circ f = 0$  tenemos que  $g_* \circ f_* = 0$ . Esto implica que si  $\gamma \in \text{Im } f_*$ , entonces  $g_*(\gamma) = 0$ .

Para probar la otra inclusión, consideremos  $\gamma = [b_p]$  y supongamos que  $g_*(\gamma) = 0$ . Entonces  $g(b_p) = \partial_C c_{p+1}$  para algún  $c_{p+1} \in C_p$ . Escojamos  $b_{p+1}$  de manera que  $g(b_{p+1}) = c_{p+1}$ . Entonces

$$g(b_p - \partial_B b_{p+1}) = g(b_p) - \partial_C g(b_{p+1}) = g(b_p) - \partial_C c_{p+1} = 0,$$

luego  $b_p - \partial_B b_{p+1} = f(a_p)$  para algún  $a_p$ . Ahora,  $a_p$  es un ciclo pues

$$f(\partial_A a_p) = \partial_B f(a_p) = \partial_B b_p - 0 = 0$$

y  $f$  es inyectiva. Es más,  $f_*[a_p] = [f(a_p)] = [b_p - \partial_B b_{p+1}] = [b_p]$  y por tanto  $[b_p] \in \text{Im } f_*$  como queríamos.

*Paso 4.* Probemos la exactitud en  $H_p(C_\bullet; R)$ . Sea  $\alpha = [c_p]$  un elemento de  $H_p(C_\bullet; R)$ . Escogeremos  $b_p$  tal que  $g(b_p) = c_p$  y ahora tomemos  $a_{p-1}$  tal que  $f(a_{p-1}) = \partial_B b_p$ . En consecuencia,  $\partial_* \alpha = [a_{p-1}]$  por definición.

Procederemos de nuevo por doble inclusión. Consideraremos primero que  $\alpha \in \text{Im } g_*$ . Entonces  $\alpha = [g(b_p)]$  donde  $b_p$  es un ciclo en  $B$ . Esto implica que  $f(a_{p-1}) = 0$  de donde  $a_{p-1} = 0$  y por tanto  $\partial_* \alpha = 0$ .

Supongamos ahora que  $\partial_* \alpha = 0$ . Entonces  $a_{p-1} = \partial_A a_p$  para algún  $a_p$ . Deducimos entonces que  $b_p - f(a_p)$  es un ciclo y que  $\alpha = g_*[b_p - f(a_p)]$  luego  $\alpha \in \text{Im } g_*$ . Realizando los cálculos obtenemos que

$$\partial_B(b_p - f(a_p)) = \partial_B(b_p) - \partial_B(f(a_p)) = \partial_B(b_p) - f(a_{p-1}) = 0,$$

$$g_*[b_p - f(a_p)] = [g(b_p) - 0] = [c_p] = \alpha.$$

*Paso 5.* Finalmente obtengamos la exactitud para  $H_{p-1}(A_\bullet; R)$ . Si  $\beta \in \text{Im } \partial_*$ , entonces  $\beta = [a_{p-1}]$  donde  $f(a_{p-1}) = \partial_B b_p$  para algún  $b_p$  por definición. En consecuencia,

$$f_*(\beta) = [f(a_{p-1})] = [\partial_B b_p] = 0.$$

Consideraremos ahora el caso donde  $f_*(\beta) = 0$ . Sea  $\beta = [a_{p-1}]$ . Entonces  $[f(a_{p-1})] = 0$  por lo que  $f(a_{p-1}) = \partial_B b_p$  para algún  $b_p$ . Definimos  $c_p = g(b_p)$ . En consecuencia,  $c_p$  es un ciclo ya que  $\partial_c c_p = g(\partial_B b_p) = g(f(a_{p-1})) = 0$  y  $\beta = \partial_* [c_p]$  por definición. Esto es,  $\beta \in \text{Im } \partial_*$ .  $\square$

**Definición 3.8.** En las condiciones del anterior lema, llamaremos a la sucesión obtenida **sucesión exacta larga de homología**.

Una consecuencia importante del resultado anterior es su naturalidad, un concepto de gran interés en teoría de categorías.

**Teorema 3.2.** Sean  $\alpha : A_\bullet \rightarrow A'_\bullet$ ,  $\beta : B_\bullet \rightarrow B'_\bullet$  y  $\gamma : C_\bullet \rightarrow C'_\bullet$  aplicaciones de cadenas. Consideremos el siguiente diagrama comutativo

$$\begin{array}{ccccccc} 0 & \longrightarrow & A_\bullet & \xrightarrow{f} & B_\bullet & \xrightarrow{g} & C_\bullet \longrightarrow 0 \\ & & \downarrow \alpha & & \downarrow \beta & & \downarrow \gamma \\ 0 & \longrightarrow & A'_\bullet & \xrightarrow{f'} & B'_\bullet & \xrightarrow{g'} & C'_\bullet \longrightarrow 0 \end{array}$$

donde las sucesiones horizontales son sucesiones exactas de complejos de cadenas. Entonces el diagrama

$$\begin{array}{ccccccc} \longrightarrow H_p(A_\bullet; R) & \xrightarrow{f_*} & H_p(B_\bullet; R) & \xrightarrow{g_*} & H_p(C_\bullet; R) & \xrightarrow{\partial_*} & H_{p-1}(A_\bullet; R) \longrightarrow \\ & \downarrow \alpha_* & & \downarrow \beta_* & & \downarrow \gamma_* & \downarrow \alpha_* \\ \longrightarrow H_p(A'_\bullet; R) & \xrightarrow{f'_*} & H_p(B'_\bullet; R) & \xrightarrow{g'_*} & H_p(C'_\bullet; R) & \xrightarrow{\partial'_*} & H_{p-1}(A'_\bullet; R) \longrightarrow \end{array}$$

### 3. Homología simplicial

es comutativo.

*Demostración.* Es claro que el diagrama

$$\begin{array}{ccccc} H_p(A_\bullet; R) & \xrightarrow{f_*} & H_p(B_\bullet; R) & \xrightarrow{g_*} & H_p(C_\bullet; R) \\ \downarrow \alpha_* & & \downarrow \beta_* & & \downarrow \gamma_* \\ H_p(A'_\bullet; R) & \xrightarrow{f'_*} & H_p(B'_\bullet; R) & \xrightarrow{g'_*} & H_p(C'_\bullet; R) \end{array}$$

es comutativo, pues los homomorfismos inducidos de las aplicaciones de cadenas conservan la comutatividad. Por tanto, basta estudiar la comutatividad en

$$\begin{array}{ccc} H_p(C_\bullet; R) & \xrightarrow{\partial_*} & H_{p-1}(A_\bullet; R) \\ \downarrow \gamma_* & & \downarrow \alpha_* \\ H_p(C'_\bullet; R) & \xrightarrow{\partial'_*} & H_{p-1}(A'_\bullet; R) . \end{array}$$

Sea  $[a] \in H_p(A_\bullet; R)$  y tomemos  $b_p$  de manera que  $g(b_p) = c_p$ . Además tomemos  $a_{p-1} \in A_{p-1}$  de forma que  $f(a_{p-1}) = \partial_B b_p$ . En consecuencia,  $\partial'_*[c_p] = [a_{p-1}]$  por definición. Consideremos ahora  $c'_p = \gamma(c_p)$ . Nuestro objetivo es ver que  $\partial'_*[c'_p] = \alpha_*[a_{p-1}]$ . Está claro que  $\beta(b_p)$  es preimagen de  $c_p$  por  $g'$ , pues  $g'\beta(b_p) = \gamma g(b_p) = \gamma(c_p) = e'_p$ . Así mismo,  $\alpha(c_{p-1})$  lo es de  $\partial'_D \beta(b_p)$ , pues  $f' \alpha(a_{p-1}) = \beta f(a_{p-1}) = \beta(\partial_B b_p) = \partial'_D \beta(b_p)$ . Esto es,  $\partial'_*[c_p] = [\alpha(a_{p-1})]$  por definición.  $\square$

**Proposición 3.4** (Sucesión de Mayer-Vietoris). *Sea  $K$  un complejo simplicial y sean  $K_1, K_2$  subcomplejos de  $K$  tales que  $K = K_1 \cup K_2$ . Entonces existe una sucesión exacta*

$$\cdots \rightarrow H_p(K_1 \cap K_2; R) \xrightarrow{f} H_p(K_1; R) \oplus H_p(K_2; R) \xrightarrow{g} H_p(K; R) \rightarrow H_{p-1}(K_1 \cap K_2; R) \rightarrow \cdots$$

tal que  $f(c) = (i_{1\#}(c), -i_{2\#}(c))$ ,  $g(d, e) = j_{1\#}(d) + j_{2\#}(e)$  donde  $i_t : K_1 \cap K_2 \rightarrow K_t$  y  $j_t : K_t \rightarrow K_1 \cup K_2$  para  $t \in \{1, 2\}$  son las respectivas inclusiones.

*Demostración.* La demostración consiste en construir la sucesión exacta corta de complejos de cadena

$$0 \rightarrow C_\bullet(K_1 \cap K_2; R) \xrightarrow{f} C_\bullet(K_1; R) \oplus C_\bullet(K_2; R) \xrightarrow{g} C_\bullet(K; R) \rightarrow 0$$

y aplicar el **Lema de la serpiente**.

Para ello comenzemos describiendo el complejo de cadenas  $C_\bullet(K_1; R) \oplus C_\bullet(K_2; R)$ . Recordemos que la suma directa de un complejo de cadenas se definía como la suma directa de los  $R$ -módulos de dimensión  $p$   $C_p(K_1; R) \oplus C_p(K_2; R)$ , cuyo operador borde  $\partial'(d, e) = (\partial_1 d, \partial_2 e)$  donde  $\partial_1, \partial_2$  corresponden a los operadores borde de  $K_1$  y  $K_2$  respectivamente.

Para comprobar la exactitud de la sucesión, comenzemos estudiando la exactitud en los extremos de ésta. Es claro que  $f$  es inyectiva por ser una inclusión. En cuanto a la sobreyectividad de  $g$ , tomemos  $d \in C_p(K; R)$  donde  $d$  sea la suma de simplices orientados. Notemos por  $d_1$  a los elementos de dicha suma provenientes de  $K_1$ . Entonces  $d - d_1 \in K_2$  y  $g(d_1, d - d_1) = d$ .

Para estudiar la exactitud en  $C_\bullet(K_1; R) \oplus C_\bullet(K_2; R)$ , consideremos la inclusión  $k : K_1 \cap K_2 \rightarrow K$  y la respectiva inclusión de cadenas inducida  $k_\# : C_\bullet(K_1 \cap K_2; R) \rightarrow C_\bullet(K; R)$ . Nótese

### 3.4. Conexión y el módulo de homología $H_0(K; R)$

que  $g(f(c)) = k_{\#}(c) - k_{\#}(c) = 0$ . Sea ahora  $g(d, e) = 0$ , entonces  $d = -e$  si las consideramos como cadenas de  $K$ . Como  $d$  proviene de  $K_1$  y  $e$  de  $K_2$ , ambas deben de provenir de  $K_1 \cap K_2$  y en consecuencia,  $(d, e) = (d, -d) = f(d)$ , como queríamos.

La homología de  $C_{\bullet}(K_1; R) \oplus C_{\bullet}(K_2; R)$  de dimensión  $p$ , que notaremos por  $H_p(K_1 \oplus K_2; R)$ , es entonces

$$H_p(K_1 \oplus K_2; R) \cong H_p(K_1; R) \oplus H_p(K_2; R)$$

por la [Proposición 1.5](#). Finalmente aplicamos el [Lema de la serpiente](#) y en consecuencia tenemos la sucesión deseada.

Para obtener la sucesión de Mayer-Vietoris de homología reducida, reemplazaremos los complejos de cadenas anteriores por sus correspondientes complejos de cadenas aumentados. Consideremos para ello el siguiente diagrama

$$\begin{array}{ccccccc} 0 & \longrightarrow & C_0(K_1 \cap K_2; R) & \longrightarrow & C_0(K_1; R) \oplus C_0(K_2; R) & \longrightarrow & C_0(K; R) \longrightarrow 0 \\ & & \downarrow \varepsilon_{K_1 \cap K_2} & & \downarrow \varepsilon_1 \oplus \varepsilon_2 & & \downarrow \varepsilon \\ 0 & \longrightarrow & R & \xrightarrow{\tilde{f}} & R \oplus R & \xrightarrow{\tilde{g}} & R \longrightarrow 0 \end{array}$$

La comutatividad y la exactitud se mantienen en la parte inferior del diagrama si definimos  $\tilde{f}(r) = (r, r)$  y  $\tilde{g}(r', r) = r' + r$ . Las aplicaciones  $\varepsilon_{K_1 \cap K_2}, \varepsilon_1 \oplus \varepsilon_2$  y  $\varepsilon$  son sobreyectivas pues la intersección de  $K_1$  y  $K_2$  es no vacía. De este modo, la homología de sus respectivos complejos de cadenas es nula en dimensión  $-1$  y en dimensión  $0$  es igual a la de sus respectivos módulos de homología reducida  $\tilde{H}_0(K_1 \cap K_2; R)$ ,  $\tilde{H}_0(K_1; R) \oplus \tilde{H}_0(K_2; R)$  y  $\tilde{H}_0(K; R)$ . Para finalizar, aplicamos de nuevo el [Lema de la serpiente](#).  $\square$

## 3.4. Conexión y el módulo de homología $H_0(K; R)$

Uno de los resultados más destacados en la teoría de homología simplicial es su capacidad para identificar y clasificar las componentes conexas de un complejo simplicial. Utilizando el módulo de homología de dimensión cero,  $H_0$ , veremos que es posible determinar directamente el número de componentes conexas en el complejo.

**Proposición 3.5.** *Sea  $K$  un complejo simplicial. Entonces  $K$  se puede partir en subcomplejos disjuntos  $K_1, K_2, \dots, K_s$  cuyos poliedros son las componentes conexas del poliedro  $|K|$ .*

**Demostración.** Consideremos las componentes conexas  $X_1, X_2, \dots, X_s$  del politopo de  $K$ . Para cada  $j$ , consideremos  $K_j$  como la colección de todos los simplices  $\sigma$  de  $K$  tales que  $\sigma \subset X_j$ . Si un simplex pertenece a  $K_j$  para algún  $j$ , entonces todas sus caras también pertenecen a  $K_j$ . Por lo tanto,  $K_1, K_2, \dots, K_s$  son subcomplejos de  $K$ . Estos subcomplejos son disjuntos entre sí, debido a que las componentes conexas  $X_1, X_2, \dots, X_s$  del poliedro  $|K|$  son disjuntas. Además, si  $\sigma \in K$  entonces  $\sigma \subset X_j$  para algún  $j$ , ya que  $\sigma$  es un subconjunto conexo del espacio topológico  $|K|$ , y todo subconjunto conexo de un espacio topológico se encuentra contenido en alguna componente conexa. Por consiguiente,  $\sigma$  pertenece a  $K_j$ . En consecuencia,  $K = K_1 \cup K_2 \cup \dots \cup K_s$  y  $|K| = |K_1| \cup |K_2| \cup \dots \cup |K_s|$ .  $\square$

**Definición 3.9.** Sea  $K$  un complejo simplicial y sean  $v, w$  dos vértices de  $K$ . Diremos que  $v, w$  pueden unirse por un **caminio de aristas** si existen vértices  $v_0, \dots, v_k$  en  $K$  de forma que  $v_0 = v$ ,  $v_k = w$  y  $[v_i, v_{i+1}]$  es un 1-simplice para todo  $i \in \{0, \dots, k-1\}$ .

### 3. Homología simplicial

**Lema 3.6.** *El políedro  $|K|$  de un complejo simplicial  $K$  es un espacio topológico conexo si, y sólo si, cualesquiera dos vértices de  $K$  pueden ser unidos por un camino de aristas.*

*Demostración.* Consideremos un par de vértices cualesquiera del camino de aristas  $v_{i_0}, v_{j_0}$  de  $K$ . Claramente si  $i_0 = j_0$  entonces es trivialmente conexo. Supongamos entonces sin pérdida de generalidad  $i_0 < j_0$ . Entonces podemos definir una aplicación lineal y continua  $\alpha_i : \left[ \frac{i-i_0}{j_0-i_0}, \frac{i+1-i_0}{j_0-i_0} \right] \rightarrow [v_i, v_{i+1}]$  tal que

$$\alpha_i(\lambda) = (1-\lambda)v_{i_0} + \lambda v_{j_0+1}$$

para todo  $i \in \{i_0, \dots, j_0 - 1\}$ . Por tanto, la función  $\alpha : [0, 1] \rightarrow [v_{i_0}, v_{j_0}]$  tal que  $\alpha(\lambda) = \alpha_i(\lambda)$  si  $\lambda \in [i, i+1]$  es un arco que conecta ambos vértices. En consecuencia,  $[v, w]$  es arco conexo. Por ser cada simplice convexo, y por tanto arco conexo,  $|K|$  es arco conexo. Concluimos aplicando el [Teorema 2.2](#).  $\square$

**Teorema 3.3.** *Sea  $K$  un complejo simplicial y  $R$  un anillo unitario conmutativo. Supongamos que el políedro  $|K|$  de  $K$  es conexo. Entonces  $H_0(K; R) \cong R$ .*

*Demostración.* Consideremos el complejo de cadenas aumentado  $\tilde{C}(K; R)$  y su respectiva homología reducida  $\tilde{H}(K; R)$ . Es claro que el submódulo de bordes del complejo aumentado  $B_0(K; R)$  está contenido en  $\ker \tilde{\partial}_0$ , dado que  $\tilde{\partial}_0 \circ \tilde{\partial}_1 = 0$ .

Para la otra inclusión, consideremos  $w_0, w_1, \dots, w_m$  vértices de  $K$  que determinan un camino de aristas. Cada  $w_j - w_{j-1}$  es una arista de  $K$  para  $j = 1, 2, \dots, m$ , y se sigue que:

$$[w_m] - [w_0] = \sum_{j=1}^m ([w_j] - [w_{j-1}]) = \tilde{\partial}_1 \left( \sum_{j=1}^m [w_j, w_{j-1}] \right) \in B_0(K; R).$$

Dado que  $|K|$  es conexo, por el [Lema 3.6](#) sabemos que cualquier par de vértices de  $K$  puede ser unido por un camino de aristas. Por lo tanto,  $v - u \in B_0(K; R)$  para cualquier par de vértices  $u$  y  $v$  de  $K$ .

Escojamos un vértice  $u \in K$ . Entonces, para cualquier conjunto de coeficientes  $r_1, r_2, \dots, r_s \in R$  y vértices  $v_1, v_2, \dots, v_s$  de  $K$ , tenemos que

$$\sum_{j=1}^s r_j [v_j] = \sum_{j=1}^s r_j ([v_j] - [u]) + \left( \sum_{j=1}^s r_j \right) [u],$$

y, por lo tanto,

$$\sum_{j=1}^s r_j ([v_j] - [u]) \in B_0(K; R).$$

En consecuencia,

$$z - \tilde{\partial}_0([u]) \in B_0(K; R)$$

para todo  $z \in \tilde{C}_0(K; R)$ . Esto muestra que  $\ker \tilde{\partial}_0 \subseteq B_0(K; R)$ . Finalmente, el homomorfismo  $\tilde{\partial}_0 : \tilde{C}_0(K; R) \rightarrow R$  es sobreyectivo y su núcleo es  $B_0(K; R)$ . Además, sabemos que  $Z_0(K; R) = C_0(K; R)$ , pues  $\tilde{\partial}_0$  es el homomorfismo nulo. Entonces

$$H_0(K; R) = \frac{Z_0(K; R)}{B_0(K; R)} = \frac{C_0(K; R)}{B_0(K; R)}.$$

### 3.4. Conexión y el módulo de homología $H_0(K; R)$

Por el **Primer teorema de isomorfía**, el homomorfismo  $\tilde{\delta}_0$  induce un isomorfismo de  $H_0(K; R)$  a  $R$ , y por lo tanto  $H_0(K; R) \cong R$ , como se requería.  $\square$

**Corolario 3.4.** *Sea  $K$  un complejo simplicial y sea  $R$  un anillo unitario conmutativo. Entonces  $H_0(K; R) \cong R^s$ , donde  $s$  es el número de componentes conexas del poliedro  $|K|$ .*

*Demostración.* Procederemos por inducción sobre el número de componentes conexas de  $|K|$ . Si  $|K|$  es conexo, entonces el resultado se sigue del **Teorema 3.3**. Supongamos ahora que podemos descomponer  $K$  en subcomplejos  $K_1, \dots, K_s$  disjuntos dos a dos. Por la **Sucesión de Mayer-Vietoris**, tenemos que la sucesión

$$\begin{aligned} \cdots &\rightarrow H_0(K_1 \cap K \setminus K_1; R) \rightarrow H_0(K_1; R) \oplus H_0(K \setminus K_1; R) \rightarrow \\ &\rightarrow H_0(K; R) \rightarrow H_{-1}(K_1 \cap K \setminus K_1; R) \rightarrow \cdots \end{aligned}$$

es exacta, donde  $K \setminus K_1 = \bigcup_{i=2}^{n+1} K_i$ . Sin embargo, la intersección  $K_1 \cap K \setminus K_1$  es vacía luego su módulo de homología de dimensión 0 y  $-1$  es el trivial. Por hipótesis de inducción,  $H_0(K_1) \oplus H_0(K \setminus K_1; R) \cong R \oplus R^{s-1} = R^s$ . Finalmente, por ser la secuencia exacta en  $H_0(K; R)$  y ser  $H_{-1}(K_1 \cap K \setminus K_1; R)$  el módulo trivial, el núcleo del operador conector es todo  $H_0(K; R)$  y por tanto,  $H_0(K; R) \cong H_0(K_1; R) \oplus H_0(K \setminus K_1; R)$ .  $\square$



## 4. Homología persistente

Este capítulo se dedica a la homología persistente, un concepto de gran relevancia en la topología computacional que proporciona herramientas poderosas para analizar y entender la estructura subyacente de los datos a través de múltiples escalas. Originada en los trabajos iniciales de matemáticos como Edelsbrunner, Letscher y Zomorodian [33], la homología persistente permite la identificación y el análisis de características topológicas que persisten a lo largo de variaciones en la escala en la que se observa.

Este capítulo se centrará en detallar los principios teóricos detrás de la homología persistente. En particular, estudiaremos en profundidad el Teorema de correspondencia, un resultado central en la teoría que muestra cómo transformar el nacimiento y muerte de las clases de homología en objetos matemáticos más sencillos de analizar y manipular computacionalmente. Seguiremos como esquema principal los resultados de [122] y [27].

### 4.1. Complejos de Čech y Vietoris-Rips

La homología persistente es comúnmente empleada para analizar conjuntos de datos representados como nubes de puntos. Aunque la homología en sí de estos conjuntos puede no ser de gran interés debido a su simplicidad o falta de estructura topológica relevante, la homología persistente permite revelar información de interés mediante la construcción de estructuras topológicas construidas a partir de los datos.

En este contexto, los complejos de Čech y Vietoris-Rips se emplean frecuentemente para capturar la estructura topológica subyacente de las nubes de puntos. Estos complejos dotan de estructura de complejo simplicial a los datos, facilitando su representación, el estudio de su forma y sus características a múltiples escalas.

**Definición 4.1.** Sea  $X$  un espacio topológico y sea  $\mathcal{U} = \{U_v\}_{v \in V}$  un recubrimiento de  $X$  por puntos  $v \in V$ . Llamaremos **nervio** de  $\mathcal{U}$  al complejo simplicial abstracto con conjunto de vértices  $V$  tal que la familia  $v_0, \dots, v_p \in V$  genera un  $p$ -símplice si, y sólo si,  $U_{v_0} \cap \dots \cap U_{v_p} \neq \emptyset$ . Lo notaremos por  $N(\mathcal{U})$ .

**Definición 4.2.** Sea  $(X, d)$  un espacio métrico y sea  $V$  un subconjunto de puntos finito de  $X$ . Dado  $\varepsilon > 0$ , definimos el **complejo de Čech**  $C(V, \varepsilon)$  como el nervio  $N(\mathcal{B}_\varepsilon)$ , donde

$$\mathcal{B}_\varepsilon = \{B_\varepsilon(v) : v \in V\},$$

siendo  $B_\varepsilon(v)$  la bola abierta de centro  $v$  y radio  $\varepsilon > 0$ .

**Proposición 4.1.** El complejo de Čech  $C(V, \varepsilon)$  es un complejo simplicial abstracto.

*Demostración.* Sea  $V = \{v_i\}_{i=1}^M$  un subconjunto de puntos en el espacio métrico  $X$ . Fijado cierto  $\varepsilon > 0$ , definimos  $\mathcal{B}_\varepsilon = \{B_\varepsilon(v) : v \in V\}$  como un recubrimiento por abiertos por puntos de  $V$  del subespacio topológico  $Y = \bigcup_{v \in V} B_\varepsilon(v)$ . Veamos que el nervio  $N(\mathcal{B}_\varepsilon)$  de  $\mathcal{B}_\varepsilon$  es el complejo abstracto cuyos vértices son los conjuntos  $B_\varepsilon(v)$  y los símplices se forman por colecciones de estos conjuntos que tienen intersecciones no vacías.

#### 4. Homología persistente

Supongamos que tenemos un simplice  $\sigma = \{v_{i_1}, \dots, v_{i_k}\}$  en  $N(\mathcal{B}_\varepsilon)$ . Esto implica que

$$\bigcap_{j=1}^k B_\varepsilon(v_{i_j}) \neq \emptyset.$$

Consideremos ahora cualquier subconjunto de bolas de la forma  $\{B_\varepsilon(v_{i_j})\}_{j \in J}$  donde  $J \subseteq \{1, \dots, k\}$ . Es claro que por ser  $\sigma$  un simplice,

$$\bigcap_{j \in \{1, \dots, k\} \setminus J} B_\varepsilon(v_{i_j}) \neq \emptyset,$$

por lo que el conjunto de vértices restantes tambien forma un simplice en  $N(\mathcal{B}_\varepsilon)$ . Por lo tanto, todas las caras de cualquier simplice  $\sigma$  son tambien simplices en  $N(\mathcal{B}_\varepsilon)$ . Es decir, el nervio  $N(\mathcal{B}_\varepsilon)$  es cerrado bajo inclusiones.

Dado que  $V$  es finito y cada simplice se define como un subconjunto de  $V$ , entonces cada simplice sólo puede tener un número finito de subconjuntos. En consecuencia, el número de caras de cada simplice tambien es finito.  $\square$

El siguiente teorema sera de utilidad para comprender y estudiar el espacio topológico que definen los complejos de Čech. Por motivos de espacio, no trataremos la prueba y simplemente enunciaremos el resultado. Una demostración del teorema enunciado de forma más general puede encontrarse en [45].

**Teorema 4.1** (Teorema del nervio). *Sea  $X$  un espacio métrico y sea  $\mathcal{U} = \{U_v\}_{v \in V}$  un recubrimiento por abiertos finito de  $X$ . Supongamos además que para todo subconjunto no vacío de vértices  $S \subseteq V$  tenemos que  $\bigcap_{s \in S} U_s$  es contráctil o vacío. Entonces, el politopo de la realización geométrica del nervio de  $\mathcal{U}$ ,  $|N(\mathcal{U})|$ , es homotópicamente equivalente a  $X$ .*

Claramente, la intersección de bolas abiertas es vacía o contráctil (pues es convexa). Por el **Teorema del nervio**, tenemos que el poliedro de la realización geométrica del complejo de Čech es homotópicamente equivalente al subespacio métrico formado por la unión de dichas bolas. En consecuencia, el estudio de la topología del complejo de Čech se resume al estudio de las bolas que recubren sus vértices. Sin embargo, el complejo de Čech es costoso de obtener mediante métodos computacionales. Por ello, se propone el complejo de Vietoris-Rips para resolver este problema.

**Definición 4.3.** Sea  $(X, d)$  un espacio métrico y sea  $V$  un subconjunto de puntos de  $X$ . Dado  $\varepsilon > 0$ , definimos el **complejo de Vietoris-Rips**  $VR(V, \varepsilon)$  como el complejo abstracto cuyo conjunto de vértices es  $V$ , de forma que  $\{v_0, v_1, \dots, v_p\} \subseteq V$  genera un  $p$ -simplice si, y solo si,  $d(v_i, v_j) \leq \varepsilon$  para todo  $0 \leq i, j \leq p$ .

**Proposición 4.2.** *El complejo de Vietoris-Rips  $VR(V, \varepsilon)$  es un complejo simplicial abstracto.*

*Demostración.* Primero veamos que el conjunto es cerrado bajo inclusiones. Supongamos que  $\sigma = \{v_0, v_1, \dots, v_p\}$  es un  $p$ -simplice en  $VR(V, \varepsilon)$ . Por definición, esto significa que para todo  $i, j$  tal que  $0 \leq i, j \leq p$ , se cumple que  $d(v_i, v_j) \leq \varepsilon$ . Consideremos ahora un subconjunto no vacío  $\tau = \{v_{i_1}, v_{i_2}, \dots, v_{i_k}\} \subseteq \sigma$ . Para cualquier par de índices  $a, b$  con  $1 \leq a, b \leq k$ , los vértices  $v_{i_a}$  y  $v_{i_b}$  tambien cumplen que  $d(v_{i_a}, v_{i_b}) \leq \varepsilon$ , pues  $\tau \subseteq \sigma$ . Por lo tanto,  $\tau$  es un  $(k-1)$ -simplice en  $VR(V, \varepsilon)$ .

Por otro lado, cada  $p$ -simplice  $\sigma$  en  $VR(V, \varepsilon)$  es un subconjunto finito de  $V$ . El número de subconjuntos de cualquier conjunto finito es finito, y en particular, el número de subconjuntos

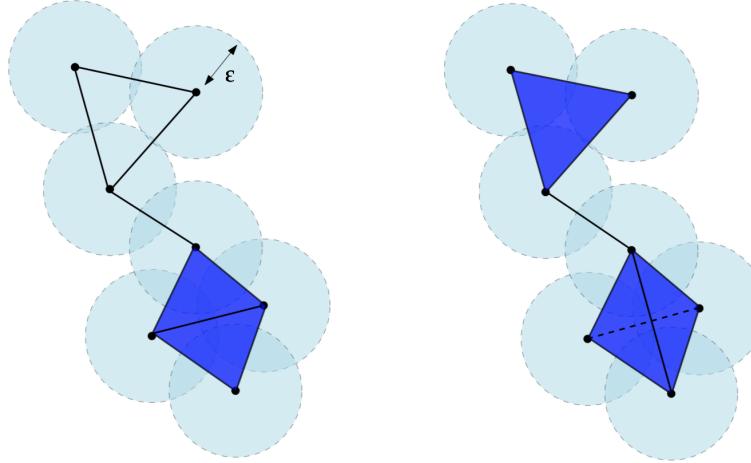


Figura 4.1.: Representación de complejos simpliciales Čech (izquierda) y Vietoris-Rips (derecha) para un conjunto de puntos en  $\mathbb{R}^2$ . En el complejo de Čech, los simplices se forman por la intersección no vacía de círculos de radio  $\varepsilon$  centrados en los puntos. El complejo de Vietoris-Rips conecta puntos que distan hasta  $2\varepsilon$ , independientemente de las intersecciones de los círculos. Los simplices de mayor dimensión están coloreados en azul, resaltando las diferencias en la estructura simplicial generada por cada método. Fuente [14].

de  $\sigma$  es  $2^{|\sigma|}$ , donde  $|\sigma|$  es el número de vértices en  $\sigma$ . En consecuencia, cada simplex en  $VR(V, \varepsilon)$  tiene un número finito de caras.  $\square$

El complejo de Vietoris-Rips es interesante en el estudio de complejos de Čech. Este hecho se debe a que estos últimos pueden ser aproximados en cierto sentido por complejos de Vietoris-Rips:

**Proposición 4.3.** *Sea  $(X, d)$  un espacio métrico y sea  $V$  un subconjunto de puntos de  $X$ . Entonces*

$$C(V, \varepsilon) \subseteq VR(V, 2\varepsilon) \subseteq C(V, 2\varepsilon).$$

*Demostración.* La primera inclusión es inmediata pues si un punto  $x$  pertenece a la intersección  $\bigcap_{v \in V} B(v, \varepsilon)$ , entonces la distancia para cada par de puntos de  $V$  es, a lo sumo,  $2\varepsilon$ . En consecuencia, cualquier simplex de  $C(V, \varepsilon)$  se encuentra en  $VR(V, 2\varepsilon)$ .

Para la segunda inclusión, consideremos ahora un simplex  $\sigma = \{v_0, \dots, v_p\}$  de  $VR(V, 2\varepsilon)$ . Por la definición de complejo de Vietoris-Rips, tenemos que  $d(v_i, v_j) \leq 2\varepsilon$  para todo  $i, j \in \{0, \dots, p\}$ . Considerando las bolas abiertas de radio  $2\varepsilon$  centradas en  $v_i$  y en  $v_j$ , tenemos que su intersección es no vacía, pues  $v_i \in \overline{B}_{2\varepsilon}(v_j)$  y  $v_j \in \overline{B}_{2\varepsilon}(v_i)$ . En el supuesto de que los puntos pertenecieran a la frontera de las bolas, la intersección de las bolas abiertas también sería no vacía pues  $\varepsilon > 0$ . En consecuencia, tenemos que  $\sigma \in C(V, 2\varepsilon)$ .  $\square$

## 4.2. Módulos de homología persistente

El módulo de homología persistente es el objeto central de estudio en este capítulo. A partir de filtraciones de complejos simpliciales, esta estructura nos va a permitir realizar un estudio

#### 4. Homología persistente

de la homología simplicial de dicha filtración.

**Definición 4.4.** Sea  $K$  un complejo simplicial. Una **filtración**  $\mathcal{F}$  de  $K$  es una familia totalmente ordenada de subcomplejos  $\{K^n\}_{n \in \mathbb{N}}$  tal que  $\emptyset, K \in \mathcal{F}$  y si  $i \leq j$ , entonces  $K^i \subseteq K^j$ . En particular, llamaremos a dicho orden **filtro**.

A partir de la definición anterior, podemos construir los complejos de cadenas asociados  $C(K^i; R)$  para todo  $i \in \mathbb{N}$ . Así mismo, podemos obtener sus respectivos submódulos de ciclos  $Z_p(K^i)$  y bordes  $B_p(K^i)$  para cada  $R$ -módulo de cadenas orientadas  $C_p(K^i; R)$ .

**Definición 4.5.** Sea  $\mathcal{F}$  una filtración, sea  $p$  un número natural y sean  $i, j \in \{0, \dots, n\}$  tales que  $i \leq j$ . Definimos el  $(i, j)$ -ésimo  $R$ -módulo de homología persistente de dimensión  $p$  asociado a  $\mathcal{F}$  como

$$H_p^{i \rightarrow j}(\mathcal{F}; R) := \text{Im } f_p^{i \rightarrow j},$$

donde  $f_p^{i \rightarrow j}$  es el homomorfismo inducido entre las clases de homología de la inclusión que va de  $K^i$  a  $K^j$ . El rango de  $H_p^{i \rightarrow j}(\mathcal{F}; R)$  diremos que es el  $(i, j)$ -ésimo número de Betti de persistencia de dimensión  $p$  y lo notaremos por  $\beta_p^{i \rightarrow j}$ .

**Proposición 4.4.** Sea  $\mathcal{F}$  una filtración del complejo simplicial  $K$ . Entonces

$$H_p^{i \rightarrow j}(\mathcal{F}; R) \cong \frac{Z_p(K^i)}{B_p(K^j) \cap Z_p(K^i)}$$

es un isomorfismo de  $R$ -módulos.

*Demostración.* Sabemos que el cociente anterior está bien definido, pues  $Z_p(K^i) \cap B_p(K^j)$  es un submódulo de  $Z_p(K^i)$ . Para ver que en efecto existe un isomorfismo, consideraremos la proyección  $\pi_i : Z_p(K^i) \rightarrow H_p(K^i; R)$ . Aplicando el **Primer teorema de isomorfía**, tenemos que

$$\frac{Z_p(K^i)}{\ker \pi_i} \cong \text{Im } \pi_i$$

es un isomorfismo. Sin embargo, nótese que

$$\begin{aligned} \ker \pi_i &= \{z \in Z_p(K^i) : \pi_i(z) = [0]\} = \{z \in Z_p(K^i) : [z] = [0]\} \\ &= \{z \in Z_p(K^i) : z \in B_p(K^j)\} = B_p(K^j) \cap Z_p(K^i). \end{aligned}$$

Además,

$$\begin{aligned} H_p^{i \rightarrow j}(\mathcal{F}; R) &= \text{Im } f_p^{i \rightarrow j} = \{f_p^{i \rightarrow j}([z]) : [z] \in H_p(K^i; R)\} \\ &= \{[(i_{i,j})_p(z)] : z \in Z_p(K^i)\} = \{\pi_i(z) : z \in Z_p(K^i)\} = \text{Im } \pi_i. \end{aligned}$$

□

La homología persistente facilita la interpretación de la homología en los distintos niveles de la filtración, permitiendo un análisis cuantitativo de su evolución. Al observar el nacimiento y desaparición de las clases de homología mediante los módulos de homología persistente, se obtiene información detallada de la estructura topológica general de la filtración.

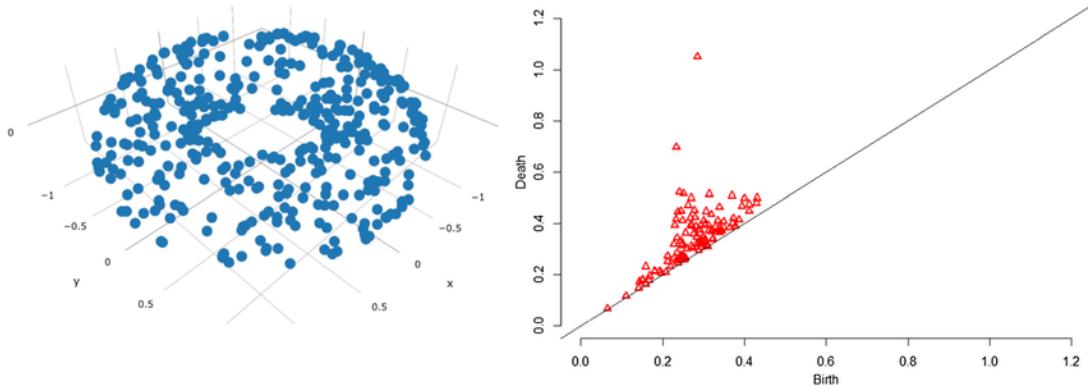


Figura 4.2.: Diagrama de persistencia para la homología de dimensión 1 de la filtración de Vietoris-Rips de 2000 puntos i.i.d. en un toro. El gráfico destaca dos puntos significativos que representan las clases de equivalencia de los ciclos unidimensionales del toro, destacando su persistencia topológica. Fuente [28].

**Definición 4.6.** Dada una filtración  $\mathcal{F}$ , decimos que un elemento  $\alpha \neq 0$  en  $H_p(K^i; R)$  **nace** en  $K^i$  si  $\alpha \notin H_p^{i-1 \rightarrow i}(\mathcal{F}; R)$ . Además, decimos que  $\alpha$  **muerde** en  $K^j$  si se fusiona con una clase proveniente de una dimensión anterior cuando se desplaza de  $K^{j-1}$  a  $K^j$ . Esto es, si  $f_p^{i \rightarrow j-1}(\alpha) \notin H_p^{i-1 \rightarrow j-1}(\mathcal{F}; R)$  pero  $f_p^{i \rightarrow j}(\alpha) \in H_p^{i-1 \rightarrow j}(\mathcal{F}; R)$ .

**Ejemplo 4.1.** Consideremos un conjunto de vértices  $\{0, e_1, e_2, e_3\} = V$  formado por los vectores de la base usual y el vector nulo en  $\mathbb{R}^3$  con la topología usual. Definamos el complejo simplicial  $K$  a partir del 2-símplice formado por dichos vectores y todas sus caras. A continuación, vamos a estudiar la filtración inducida por inclusiones  $\mathcal{K}$ :

$$\emptyset = K^0 \rightarrow K^1 \rightarrow K^2 \rightarrow K^3 \rightarrow K^4 = K,$$

donde

$$\begin{aligned} K^1 &= \{[0], [e_1], [e_2], [e_3]\}, \\ K^2 &= K^1 \cup \{[0, e_1], [0, e_2], [0, e_3]\}, \\ K^3 &= K^2 \cup \{[0, e_1, e_2], [0, e_1, e_3], [0, e_2, e_3]\}. \end{aligned}$$

Consideremos además las cadenas de  $\mathbb{R}$ -módulos asociadas, donde cada  $C_p(K^j; \mathbb{R})$  es un espacio vectorial con cuerpo  $\mathbb{R}$  para cada dimensión  $p \in \mathbb{Z}$ .

En  $K^1$ , el módulo de cadenas  $C_0(K^1; \mathbb{R})$  es generado por los 0-símplices  $\{[0], [e_1], [e_2], [e_3]\}$ , lo que resulta en un espacio vectorial  $\mathbb{R}^4$ , mientras que los módulos  $C_1(K^1; \mathbb{R})$  y  $C_2(K^1; \mathbb{R})$  son ambos cero, debido a la ausencia de 1-símplices y 2-símplices.

En cuanto a  $K^2$ , el módulo  $C_0(K^2; \mathbb{R})$  conserva la misma estructura que  $C_0(K^1; \mathbb{R})$ , pues no se añaden nuevos 0-símplices. Sin embargo,  $C_1(K^2; \mathbb{R})$  se genera ahora a partir de las aristas  $\{[0, e_1], [0, e_2], [0, e_3]\}$ , formando un espacio vectorial isomorfo a  $\mathbb{R}^3$ . El módulo  $C_2(K^2; \mathbb{R})$  sigue siendo nulo debido a la falta de 2-símplices.

Llegando a  $K^3$ , observamos que  $C_2(K^3; \mathbb{R})$  ahora se genera por  $\{[0, e_1, e_2], [0, e_1, e_3], [0, e_2, e_3]\}$ , generando otro espacio isomorfo a  $\mathbb{R}^3$ . Los módulos  $C_0(K^3; \mathbb{R})$  y  $C_1(K^3; \mathbb{R})$  son idénticos a sus pares en  $K^2$ , pues tienen los mismos símplices de dimensión 1 y 2.

#### 4. Homología persistente

Finalmente, en  $K^4 = K$ , todos los módulos  $C_p(K^4; \mathbb{R})$  para  $p \in \{0, 1, 2\}$  son idénticos a los de  $K^3$ . Sin embargo,  $K^4$  tiene un 3-símplice formado por todos sus vértices, de forma que  $C_3(K^4; \mathbb{R}) = \mathbb{R}$ .

En  $K^2$ , todos los 1-símplices son ciclos porque no existen 2-símplices que puedan generar bordes, de modo que  $Z_1(K^2) = C_1(K^2)$  y  $B_1(K^2) = 0$ . En  $K^3$ , todos los 2-símplices son ciclos, ya que no hay 3-símplices en esta etapa; así,  $Z_2(K^3) = C_2(K^3)$  y los bordes de  $C_1(K^3)$  son generados por las aristas de los 2-símplices, indicando cierta interacción entre ellos.

En  $K^4$ , el único 3-símplice forma un ciclo por sí mismo, puesto que  $C_4(K^4) = 0$  y, por lo tanto,  $Z_3(K^4) = C_3(K^4)$ . Los bordes de los 2-símplices en  $C_2(K^4)$  resultan de aplicar  $\partial_3$  al 3-símplice, produciendo una combinación de todos los 2-símplices, por lo tanto,  $B_2(K^4)$  se compone de estos bordes.

Para  $K^1$ , tenemos  $H_0(K^1; \mathbb{R}) = \mathbb{R}^4$ , mostrando cuatro componentes conexas. En  $K^2$ , la homología se simplifica a  $H_0(K^2; \mathbb{R}) = \mathbb{R}$ , indicando que todos los vértices forman una única componente conexa, mientras que  $H_1(K^2; \mathbb{R}) = 0$  y  $H_2(K^2; \mathbb{R}) = 0$  dado que no hay 2-símplices. En  $K^3$ , la homología permanece sin cambios para  $H_0$  y  $H_2$ , pero todos los posibles 1-ciclos se convierten en bordes de los 2-símplices, lo que resulta en  $H_1(K^3; \mathbb{R}) = 0$ .

Finalmente, en  $K^4$ , observamos que  $H_0(K^4; \mathbb{R}) = \mathbb{R}$ ,  $H_1(K^4; \mathbb{R}) = 0$ ,  $H_2(K^4; \mathbb{R}) = 0$  y  $H_3(K^4; \mathbb{R}) = \mathbb{R}$ , indicando la presencia de una cavidad tridimensional que no es el borde de ningún símplice de mayor dimensión.

Tras este exhaustivo estudio de la homología de la filtración, por fin estamos en condiciones de estudiar el nacimiento o muerte de las clases de homología. Por ejemplo, podemos observar que en el instante 1 de la filtración nacen cuatro clases de homología de dimensión 0, pues  $H_0(K^1; \mathbb{R}) = \mathbb{R}^4$ . Sin embargo, en el segundo instante de la filtración,  $H_0(K^2; \mathbb{R}) = \mathbb{R}$ , lo que implica la muerte de tres clases de homología de dimensión 0. De esta forma, tendríamos que el módulo de homología persistente  $H_0^{1 \rightarrow 2}(K; \mathbb{R}) = \text{Im } i_*^{1 \rightarrow 2} = \mathbb{R}$ , siendo  $i_*^{1 \rightarrow 2} : H_0(K^1; \mathbb{R}) \rightarrow H_0(K^2; \mathbb{R})$  el homomorfismo de inclusión inducido. En consecuencia, es claro que (1, 2)-ésimo número de Betti de persistencia de dimensión 0 es 1.

Como hemos visto en el [E.j. 4.1](#), el estudio de la homología persistente requiere de un amplio estudio de la homología de la filtración a todos sus niveles. Esta información a menudo es numerosa y difícil de interpretar, por lo que en la siguiente sección mostraremos una representación de la homología persistente que facilite su estudio e interpretación.

### 4.3. Representación de la homología persistente

Nuestro objetivo ahora es obtener una representación de la homología persistente que nos permita interpretarla. Originalmente, esta representación se ha realizado mediante los **diagramas de persistencia**. En estos diagramas, cada clase de homología que nace en un complejo simplicial  $K^i$  y muere en  $K^j$  se representa por un punto  $(i, j)$  en el plano cartesiano, para cada dimensión  $p$ .

Aunque intuitivos, los diagramas de persistencia presentan ciertas limitaciones. El principal problema radica en que no logran capturar de manera efectiva la multiplicidad de la persistencia; es decir, múltiples clases que nacen y mueren simultáneamente se representan mediante un mismo punto, lo que puede conducir a ambigüedades en la interpretación de los datos.

Para abordar estos problemas, Zomorodian y Carlsson introdujeron un enfoque innovador mediante el uso del **código de barras** [122]. Este método no solo ofrece una visualización clara

y detallada de la vida de cada clase de homología, sino que también facilita la interpretación de su multiplicidad y relevancia en el contexto de los datos analizados. Esta interpretación fue posible ya que demostraron que todo módulo de persistencia puede identificarse de manera biyectiva con una estructura algebraica más general, conocida como **módulos graduados**, bajo ciertas hipótesis.

A continuación profundizaremos en la aplicación y las implicaciones de estos métodos, explorando en detalle el Teorema de correspondencia como se discute en [22].

**Definición 4.7.** Sea  $\mathcal{M} = \{\{M_i\}_{i \in \mathbb{N}}, \{f_{i,j}\}_{i \leq j \in \mathbb{N}}\}$  una familia de  $R$ -módulos. Diremos que dicha familia es un **módulo de persistencia discreto** sobre  $R$  si para cada  $i \leq j$  existe un homomorfismo de  $R$ -módulos  $f_{i,j} : M_i \rightarrow M_j$  tal que:

1. El homomorfismo  $f_i = f_{i,i} = \text{id}_{M_i}$  para todo  $i \in \mathbb{N}$ .
2. La composición  $f_{i,k} \circ f_{i,j} = f_{i,k}$  para todo  $i \leq j \leq k$ .

*Observación 4.1.* Nótese que con esta definición, todo módulo de homología persistente es de hecho un módulo de persistencia discreto. Dada una dimensión  $p \in \mathbb{Z}$  fija, los  $R$ -módulos los componen los módulos de homología y sus homomorfismos inducidos por la inclusión cumplen dichas propiedades.

**Definición 4.8.** Sean  $\mathcal{M} = \{\{M_i\}_{i \in \mathbb{N}}, \{f_{i,j}\}_{i \leq j \in \mathbb{N}}\}$ ,  $\mathcal{N} = \{\{N_i\}_{i \in \mathbb{N}}, \{g_{i,j}\}_{i \leq j \in \mathbb{N}}\}$  dos módulos de persistencia discretos. Diremos que la familia de homomorfismos  $\varphi_\bullet = \{\varphi_i\}_{i \in \mathbb{N}}$  tales que  $\varphi_i : M_i \rightarrow N_i$  es un **homomorfismo de módulos de persistencia discreto** si  $g_{i,j} \circ \varphi_i = \varphi_j \circ f_{i,j}$ .

La anterior definición es equivalente a decir que el diagrama

$$\begin{array}{ccccccccc} M_0 & \xrightarrow{f_0} & M_1 & \xrightarrow{f_1} & \cdots & \xrightarrow{f_{i-1}} & M_i & \xrightarrow{f_i} & M_{i+1} \xrightarrow{f_{i+1}} \cdots \\ \downarrow \varphi_0 & & \downarrow \varphi_1 & & & & \downarrow \varphi_i & & \downarrow \varphi_{i+1} \\ N_0 & \xrightarrow{g_0} & N_1 & \xrightarrow{g_1} & \cdots & \xrightarrow{g_{i-1}} & N_i & \xrightarrow{g_i} & N_{i+1} \xrightarrow{g_{i+1}} \cdots \end{array}$$

comuta. En las condiciones anteriores, los módulos de persistencia discretos junto a sus homomorfismos forman una categoría que notaremos por  $R\text{-PersMod}$ . Claramente se tiene que existe el homomorfismo identidad  $\text{id}_\mathcal{M} : \mathcal{M} \rightarrow \mathcal{M}$ , donde a cada módulo y homomorfismo de módulos se le asocia él mismo. Además, la identidad comuta con la familia de homomorfismos de módulos, pues es la composición con la identidad de homomorfismos de módulos componente a componente. Veamos ahora la asociatividad en la composición de los homomorfismos de módulos de persistencia discretos. Consideremos cuatro módulos de persistencia discretos  $\mathcal{M} = \{\{M_i\}_{i \in \mathbb{N}}, \{f_{i,j}\}_{i \leq j \in \mathbb{N}}\}$ ,  $\mathcal{N} = \{\{N_i\}_{i \in \mathbb{N}}, \{g_{i,j}\}_{i \leq j \in \mathbb{N}}\}$ ,  $\mathcal{P} = \{\{P_i\}_{i \in \mathbb{N}}, \{h_{i,j}\}_{i \leq j \in \mathbb{N}}\}$  y  $\mathcal{Q} = \{\{Q_i\}_{i \in \mathbb{N}}, \{p_{i,j}\}_{i \leq j \in \mathbb{N}}\}$ ; y tres homomorfismos  $\varphi_\bullet : \mathcal{M} \rightarrow \mathcal{N}$ ,  $\psi_\bullet : \mathcal{N} \rightarrow \mathcal{P}$ , y  $\theta_\bullet : \mathcal{P} \rightarrow \mathcal{Q}$ . Para demostrar la asociatividad, necesitamos verificar que

$$(\theta \circ (\psi \circ \varphi))_i = ((\theta \circ \psi) \circ \varphi)_i \quad \forall i \in \mathbb{N}.$$

Sin embargo, dicha composición es la asociatividad de la composición de homomorfismos de módulos componente a componente, por lo que tenemos que:

$$\theta_i \circ (\psi_i \circ \varphi_i) = (\theta_i \circ \psi_i) \circ \varphi_i.$$

#### 4. Homología persistente

Además, para cualquier  $i \leq j$  en  $\mathbb{N}$ , necesitamos mostrar que:

$$h_{i,j} \circ (\theta \circ (\psi \circ \varphi))_i = (\theta \circ (\psi \circ \varphi))_j \circ f_{i,j}.$$

Sustituyendo la definición de composición, esto se reduce a verificar que

$$h_{i,j} \circ (\theta_i \circ (\psi_i \circ \varphi_i)) = (\theta_j \circ (\psi_j \circ \varphi_j)) \circ f_{i,j} \quad i \leq j \text{ en } \mathbb{N}.$$

O equivalentemente, que  $((h_{i,j} \circ \theta_i) \circ \psi_i) \circ \varphi_i = \theta_j \circ (\psi_j \circ (\varphi_j \circ f_{i,j}))$ . Debido a que cada homomorfismo respeta las estructuras de los módulos de persistencia, tenemos:

$$h_{i,j} \circ \theta_i = \theta_j \circ p_{i,j},$$

$$p_{i,j} \circ \psi_i = \psi_j \circ g_{i,j},$$

$$g_{i,j} \circ \varphi_i = \varphi_j \circ f_{i,j}.$$

Aplicando estas igualdades y la asociatividad de homomorfismos de módulos, concluimos que:

$$\begin{aligned} h_{i,j} \circ (\theta_i \circ (\psi_i \circ \varphi_i)) &= ((h_{i,j} \circ \theta_i) \circ \psi_i) \circ \varphi_i = ((\theta_j \circ p_{i,j}) \circ \psi_i) \circ \varphi_i \\ &= (\theta_j \circ (p_{i,j} \circ \psi_i)) \circ \varphi_i = (\theta_j \circ (\psi_j \circ g_{i,j})) \circ \varphi_i \\ &= \theta_j \circ ((\psi_j \circ g_{i,j}) \circ \varphi_i) = \theta_j \circ (\psi_j \circ (g_{i,j} \circ \varphi_i)) = \theta_j \circ (\psi_j \circ (\varphi_j \circ f_{i,j})). \end{aligned}$$

**Definición 4.9.** Sea  $R$  un anillo. Diremos que  $R$  es un **anillo graduado** si puede descomponerse como una suma directa

$$R = \bigoplus_{n=0}^{\infty} R_n,$$

donde  $R_m R_n \subseteq R_{m+n}$  para todo  $m, n \in \mathbb{Z}$ , siendo el anillo  $R_m R_n = \{r_m r_n : r_m \in R_m, r_n \in R_n\}$ . Los elementos de  $R_n$  distintos de cero se denominan **homogéneos de grado  $n$** .

**Definición 4.10.** Sea  $R$  un anillo graduado y sea  $M$  un  $R$ -módulo. Diremos que  $M$  es un **módulo graduado** si puede escribirse como

$$M = \bigoplus_{n=0}^{\infty} M_n,$$

donde  $M_n$  son grupos abelianos y  $R_m M_n \subseteq M_{m+n}$  para todo  $m, n \in \mathbb{Z}$ , siendo  $R_m M_n$  el  $R_m$ -módulo con grupo abeliano  $M_n$ . Un elemento de  $M_n$  distinto de cero se llama **homogéneo de grado  $n$** .

Los módulos graduados no son más que un tipo particular de módulos. Para cada módulo graduado  $M$ , el morfismo identidad  $\text{id}_M : M \rightarrow M$  es simplemente el morfismo identidad de  $R$ -módulos. Este morfismo es claramente un homomorfismo de módulos graduados que preserva la graduación. Consideremos ahora dos módulos graduados  $M, N$  y un homomorfismo de módulos graduados  $f : M \rightarrow N$ . Si tenemos otro módulo graduado  $P$  y un homomorfismo  $g : N \rightarrow P$ , su composición  $g \circ f$  es la composición de homomorfismos de  $R$ -módulos. Por lo tanto, la asociatividad de la composición se sigue directamente de la asociatividad de la composición de homomorfismos de  $R$ -módulos. En consecuencia, los módulos graduados sobre  $R$  junto con los homomorfismos de módulos graduados forman una categoría, que

denotaremos por  $R\text{-GrMod}$ .

Los módulos de persistencia discretos sobre un anillo  $R$  y los  $R[t]$ -módulos graduados son conceptos íntimamente relacionados. Si  $\mathcal{M}$  es un módulo de persistencia discreto, podemos definir un  $R[t]$ -módulo graduado  $\alpha(\mathcal{M})$  como

$$\alpha(\mathcal{M}) = \bigoplus_{i \in \mathbb{N}} M_i,$$

donde el producto por  $t$  lo definimos como  $t \cdot m_i = f_{i,i+1}(m_i)$  para todo  $m_i \in M_i$ . Análogamente, podemos definir un módulo de persistencia discreto a partir de un  $R[t]$ -módulo  $\bigoplus_{i \in \mathbb{N}} M_i$ , de forma que

$$\beta \left( \bigoplus_{i \in \mathbb{N}} M_i \right) = \mathcal{M}.$$

Aquí, los morfismos los obtenemos a partir del producto por  $t$ , esto es,  $f_{i,i+1}(m_i) = t \cdot m_i$  para todo  $m_i \in M_i$ . El siguiente resultado nos proporciona formalmente cómo de íntima es esta relación.

**Lema 4.1.** *Las aplicaciones  $\alpha$  y  $\beta$  definidas anteriormente forman una pareja isomorfa de funtores covariantes entre  $R\text{-PersMod}$  y  $R[t]\text{-GrMod}$ . En particular, ambas categorías son isomórficas.*

*Demostración.* Sea  $\varphi_\bullet : \mathcal{M} \rightarrow \mathcal{N}$  un morfismo de módulos de persistencia discretos. Definamos

$$\alpha(\varphi_\bullet) : \bigoplus_{i \in \mathbb{N}} M_i \rightarrow \bigoplus_{i \in \mathbb{N}} N_i$$

donde a cada  $m_i \in M_i$  le asignamos  $\varphi_i(m_i)$  para cada  $i \in \mathbb{N}$ . Veamos que  $\alpha : R\text{-PersMod} \rightarrow R[t]\text{-GrMod}$  es un functor covariante. Primero veamos que  $\alpha(\varphi_\bullet)$  es un morfismo de módulos graduados. Tenemos que  $\alpha(\varphi_\bullet)$  es un homomorfismo de grupos pues cada  $\varphi_i$  lo es, cumple que  $\alpha(\varphi_i)(M_i) \subseteq N_i$  y además, si  $m = (m_0, m_1, \dots)$  es un elemento de  $\mathcal{M}$ , entonces

$$\begin{aligned} \alpha(\varphi_\bullet)(tm) &= \alpha(\varphi_\bullet)(0, tm_0, tm_1, \dots) = (0, \varphi_0(tm_0), \varphi_1(tm_1), \dots) \\ &= (0, t\varphi_0(m_0), t\varphi_1(m_1), \dots) = t\alpha(\varphi_\bullet)(m), \end{aligned}$$

donde la última igualdad es consecuencia de la segunda propiedad de los morfismos de módulos de persistencia discretos. En cuanto a las propiedades funtoriales, es evidente que  $\alpha$  lleva identidades en identidades. Además, si  $\psi_\bullet$  es otro morfismo de módulos de persistencia discretos, tenemos que

$$(\alpha(\psi_\bullet \circ \varphi_\bullet))(m) = (\psi_i(\varphi_i(m_i)))_{i \in \mathbb{N}} = \alpha(\psi_\bullet)(\varphi_i(m_i))_{i \in \mathbb{N}} = (\alpha(\psi_\bullet) \circ \alpha(\varphi_\bullet))(m).$$

Consideremos ahora el homomorfismo de  $R[t]$ -módulos graduados

$$\eta : \bigoplus_{i \in \mathbb{N}} M_i \rightarrow \bigoplus_{i \in \mathbb{N}} N_i,$$

que para cada  $i \in \mathbb{N}$  induce un homomorfismo  $\eta_i : M_i \rightarrow N_i$  compatible con el producto por

#### 4. Homología persistente

t. En consecuencia, el diagrama

$$\begin{array}{ccccccc} M_0 & \xrightarrow{t} & M_1 & \xrightarrow{t} & \cdots & \xrightarrow{t} & M_i & \xrightarrow{t} & M_{i+1} & \xrightarrow{t} & \cdots \\ \downarrow \eta_0 & & \downarrow \eta_1 & & & & \downarrow \eta_i & & \downarrow \eta_{i+1} & & & \\ N_0 & \xrightarrow{t} & N_1 & \xrightarrow{t} & \cdots & \xrightarrow{t} & N_i & \xrightarrow{t} & N_{i+1} & \xrightarrow{t} & \cdots \end{array}$$

es comutativo. Definamos ahora  $\beta(\eta) = (\eta_0, \eta_1, \dots)$  y veamos que es un homomorfismo de módulos de persistencia discretos entre  $\mathcal{M}$  y  $\mathcal{N}$ . En consecuencia,  $\beta$  nos da homomorfismos de grupos  $\eta_i : M_i \rightarrow N_i$  que, a su vez, son homomorfismos de  $R$ -módulos. Para comprobarlo, basta tomar cualquier  $r \in R$  y  $m_i \in M_i$  y vemos que  $\eta_i(rm_i) = \eta(rm_i) = r\eta(m_i) = r\eta_i(m_i)$ . Como los homomorfismos de  $R$ -módulos de  $\mathcal{M}$  y  $\mathcal{N}$  se obtienen mediante la multiplicación por  $t$ , entonces para todo  $m_i \in M_i$  tenemos que

$$\eta_{i+1}(tm_i) = \eta(tm_i) = t\eta(m_i) = t\eta_i(m_i),$$

por lo que  $\beta(\eta)$  es un homomorfismo de módulos de persistencia discretos. Claramente  $\beta$  conserva la identidad. Luego para otro  $\theta : \mathcal{M} \rightarrow \mathcal{N}$  y cualquier  $m = (m_i)_{i \in \mathbb{N}} \in \mathcal{M}$ ,

$$(\beta(\theta \circ \eta))(m) = (\theta(\eta(m_i)))_{i \in \mathbb{N}} = \beta(\theta)(\eta(m_i))_{i \in \mathbb{N}} = (\beta(\theta) \circ \beta(\eta))(m).$$

Esto es,  $\beta$  es un funtor covariante. Finalmente, por la construcción de  $\alpha$  y  $\beta$  tenemos que  $\beta \circ \alpha$  es el funtor identidad en  $R[t]\text{-GrMod}$  y que  $\alpha \circ \beta$  es el funtor identidad en  $R\text{-PersMod}$ .  $\square$

En la práctica generalmente trabajaremos con módulos de persistencia que cumplen ciertas condiciones de finitud. Por ello, resulta de gran interés conocer si la correspondencia recién realizada se sigue cumpliendo bajo estos casos.

**Definición 4.11.** Diremos que un módulo de persistencia discreto  $\mathcal{M}$  es de **tipo finito** si existe  $N \in \mathbb{N}$  de forma que para todo  $i, j \in \mathbb{N}$  tal que  $N \leq i \leq j$  el homomorfismo  $f_{i,j}$  es un isomorfismo.

**Definición 4.12.** Diremos que un módulo de persistencia discreto  $\mathcal{M}$  es de **tipo finitamente presentado (generado)** si es de tipo finito y además,  $M_i$  es finitamente presentado (generado) para todo  $i \in \mathbb{N}$ .

Nuestro objetivo será probar que el [Lema 4.1](#) sigue siendo cierto para los módulos de persistencia discretos de tipo finitamente presentados.

**Lema 4.2.** *Sea  $\mathcal{M}$  un módulo de persistencia discreto. Si  $\mathcal{M}$  es de tipo finitamente presentado, entonces  $\alpha(\mathcal{M})$  es finitamente presentado.*

*Demostración.* Consideremos  $N \in \mathbb{N}$  de forma que  $f_{i,j} : M_i \rightarrow M_j$  es un isomorfismo para todo  $N \leq i \leq j$ . Sea  $G$  un conjunto de generadores de  $M_i$ . Queremos ver que  $G = \bigcup_{i=1}^N G_i$  es un sistema de generadores también para  $\alpha(\mathcal{M})$ . Para ello, veamos que todo elemento homogéneo de  $\alpha(\mathcal{M})$  está generado por la unión de los  $G_i$ . Fijemos  $k \in \mathbb{N}$  y sea  $m_k \in \alpha(\mathcal{M})$  un elemento homogéneo de grado  $k$ . Si  $k \leq N$ , entonces  $m_k$  está generado por los elementos de  $G_k$  por construcción. Si  $k > N$ , veamos que  $m_k$  está generado por  $G_N$ . Por ser  $f_{N,k}$  un isomorfismo, podemos tomar  $m_N = f_{N,k}^{-1}(m_k)$ . Pero como  $m_D$  está generado por  $G_N$ , entonces  $m_k$  está generado por  $f_{N,k}(G_N)$ . Por como hemos construido  $\alpha$ ,  $f_{N,k}(G_N) = t^{k-N}G_N$  y como

$t^{k-N} \in R[t]$ , entonces  $m_k$  está generado por  $G_N$ . En consecuencia,  $\alpha(\mathcal{M})$  es finitamente generado.

Para ver que  $\alpha(\mathcal{M})$  es finitamente presentado, consideremos el epimorfismo  $\mu_i : R^{n_i} \rightarrow M_i$  que genera  $M_i$  por extensión lineal sobre  $G_i$ . Considerando  $n = \sum_{i=1}^N n_i$ , existe una aplicación  $\mu : R[t]^N \rightarrow \alpha(\mathcal{M})$  que corresponde al sistema de generadores  $G$ . Para cada  $g_i \in G$ , denotemos por  $e_i$  a su correspondiente elemento en el sistema de generadores de  $R[t]^N$ .

A continuación definamos un conjunto finito de elementos del núcleo de  $\mu$ . sea  $H_i$  el sistema de generadores de  $\ker \mu_i$  para cada  $0 \leq i \leq N$ . Es claro que  $H_i \subseteq \ker \mu_i$ . Es más, para cualquier  $0 \leq i < j \leq N$  y cualquier  $g_i \in G_i$  tal que  $f_{i,j}(g_i) \neq 0$ , tenemos que

$$f_{i,j}(g_i) = \sum_{k=0}^{n_j} \lambda_k g_{j,k}$$

donde  $\lambda_k \in R$  y  $G_i = \{g_{j_0}, g_{j_1}, \dots, g_{j_k}\}$ . Por tanto, el correspondiente elemento

$$t^{j-i} e_i - \sum_{k=0}^{n_j} \lambda_k e_{j,k}$$

pertenece al  $\ker \mu$ . Denotemos ahora por  $H_{i,j}$  al conjunto finito obtenido tomando los elementos de la forma de la expresión anterior para cada  $g_i \in G_i$  tal que  $f_{i,j}(g_i) \neq 0$ . Sea  $H = \bigcup_{i=0}^N H_i \cup \bigcup_{0 < i \leq j \leq N} H_{i,j}$ .

A continuación, fijemos un elemento  $x$  del núcleo de  $\mu$  de la forma

$$x = \sum_l \lambda_l e_l$$

de forma que  $\lambda_l \in R[t]$  y  $e_l$  es un generador de  $R[t]^n$ . Podemos suponer sin pérdida de generalidad que  $x$  es homogéneo de algún grado  $k$ . Veamos por casos que  $x$  es finitamente generado por los elementos de  $H_k$ .

Supongamos que  $k \leq N$  y que todos los escalares  $\lambda_l$  son de grado 0. Entonces, todos los  $e_l$  que aparecen en  $x$  son del mismo grado y por tanto, sus imágenes por  $\mu$  son generadores de  $M_k$ . Es decir,  $x$  está generado por  $H_k$ .

Supongamos ahora que  $k \leq N$  y que algún  $\lambda_l$  es de grado positivo. Por ser  $x$  homogéneo, entonces  $\lambda_l$  es de la forma  $r_l t^{d_l}$ , donde  $r_l \in R$  y  $d_l > 0$ . Como el grado de  $e_l$  es  $k - d_l$ , entonces existe un elemento  $h_l \in H_{k-d_l, k}$  de la forma

$$h_l = t^{d_l} e_l - \sum_{m=0}^{n_l} \tilde{\lambda}_m e_{l,m},$$

donde todos los  $e_{l,m}$  son de grado  $k$  y  $\tilde{\lambda}_m \in R$ . Por consiguiente, en  $x - r_l h_l$  el coeficiente de  $e_l$  en  $x$  es 0 en  $t$  y por tanto, sólo estamos introduciendo sumandos de grado 0 en  $t$ .

Iterando esta construcción para cada sumando con coeficiente de grado positivo, obtenemos un elemento  $x' = x - \sum_w r_w h_w$ , donde  $r_w \in R$ ,  $h_w \in H$  y  $x'$  tiene solamente coeficientes de grado 0 en  $t$ . Esto es,  $x = x' + \sum_w r_w h_w$ . Finalmente, aplicando la primera parte de la demostración tenemos que  $x$  es generado por  $H$ .

Para concluir, consideremos  $k > N$ . En dicho caso, cada  $\lambda_l$  es de grado al menos  $k - N$ , pues el grado maximal de  $e_l$  es  $N$ . Luego  $x = t^{k-N} x'$ , donde  $x'$  es homogéneo de grado  $N$ .

#### 4. Homología persistente

Como  $0 = \mu(x) = t^{k-N}\mu(x')$ , entonces  $x' \in \ker \mu$ . Por la segunda parte de la demostración, concluimos que  $x'$  es generado por  $H$  y por tanto,  $x$  también.  $\square$

Para los siguientes dos lemas, fijaremos el  $R[t]$ -módulo graduado finitamente presentado  $\mathbf{M} = \bigoplus_{i \in \mathbb{N}} M_i$  junto con la aplicación  $\mu : R[t]^n \rightarrow \mathbf{M}$  cuyo núcleo es finitamente generado. Consideraremos además el sistema de generadores  $G = \{g_1, \dots, g_n\}$  de  $\mathbf{M}$  y sea  $H = \{h_1, \dots, h_m\}$  un sistema de generadores de  $\ker \mu$ . Además, consideraremos que tanto los elementos de  $G$  como de  $H$  son homogéneos del grado del respectivo módulo. Finalmente, vamos a asumir que dichos elementos están ordenados por grado en orden no decreciente.

**Lema 4.3.** *Cada  $M_i$  de  $\mathbf{M}$  es finitamente presentado como un  $R$ -módulo.*

*Demostración.* Veamos primero que  $M_i$  es finitamente generado. Sea  $d_j$  el grado de  $g_j$  para  $1 \leq j \leq n$ . Sea  $n_i$  el número de elementos de  $G$  con grado menor o igual que  $i$ . Definamos  $\mu_i : R^{n_i} \rightarrow M_i$  de forma que  $\mu_i$  asigne al  $j$ -ésimo generador  $e_{ij} \in R^{n_i}$  el elemento  $t^{i-d_j}g_j$ . Cada elemento  $x \in M_i$  es un componente homogéneo de grado  $i$  de una combinación lineal de los generadores de  $\mathbf{M}$ . Dado que solo los generadores  $g_j$  cuyo grado  $d_j$  es menor o igual que  $i$  pueden contribuir a esta combinación para formar un elemento de grado  $i$ ,  $x$  puede expresarse como:

$$x = \sum_{d_j=1}^i r_j t^{i-d_j} g_j,$$

donde  $r_j \in R$ . Considerando el generador en  $R^{n_i}$  cuyas entradas son los coeficientes  $r_j$ , entonces

$$\mu_i((r_1, \dots, r_{n_i})) = \sum_{d_j=1}^i r_j t^{i-d_j} g_j = x$$

y en consecuencia,  $\mu_i$  es sobreyectivo. Esto es,  $M_i$  es finitamente generado.

A continuación veamos que  $\ker \mu_i$  también es finitamente generado. Sean  $e_1, \dots, e_n$  los generadores de  $R[t]^n$  con imagen  $g_1, \dots, g_n$  por  $\mu$  respectivamente. Sea  $m_i$  el número de elementos  $h_j$  de  $H$  cuyo grado  $d'_j$  es menor o igual que  $i$ . Para cada  $h_j$  tal que  $1 \leq j \leq m_i$ , consideremos  $t^{i-d'_j}h_j$  que podemos reescribir como

$$t^{i-d'_j}h_j = \sum_{k=1}^{m_i} r_k t^{i-d_k} e_k$$

para ciertos  $r_k \in R$ . Definamos ahora

$$h_{j_i} = \sum_{k=1}^{n_i} r_k e_{ki}$$

y definamos  $H_i = \{h_{j_i} : 1 \leq i \leq m_i\}$ . Veamos que  $H_i$  genera el núcleo de  $\mu$ . Es claro que  $\mu_i(h_{j_i}) = \mu(h_j) = 0$ . Fijemos ahora un elemento arbitrario  $x$  de  $\ker \mu_i$ . Tenemos entonces que  $x$  es combinación lineal de  $\{e_{1i}, \dots, e_{ni}\}$  con coeficientes en  $R$ . Reemplazando  $e_{j_i}$  por  $t^{i-d_i}e_j$ , obtenemos un elemento homogéneo  $x' \in R[t]^n$  de grado  $i$ . Por hipótesis, podemos escribir  $x'$  como combinación de elementos de  $H$  de forma que

$$x' = \sum_{k=1}^{m_i} r'_k t^{i-d'_k} h_{ki}$$

donde  $r'_k \in R$ . En consecuencia, veamos que

$$x = \sum_{k=1}^{m_i} r'_k h_{ki}.$$

Para ello, procederemos comparando coeficientes. Consideremos  $j \in \{1, \dots, n_i\}$  y sea  $c_j \in R$  el coeficiente de  $e_{j_i}$  en  $x$ . Sea  $c'_j$  el coeficiente de  $e_{j_i}$  en la suma de la expresión anterior, escribiendo cada  $h_{ki}$  como combinación lineal de los  $e_{j_i}$ . Por la construcción realizada,  $c_j$  es el coeficiente de  $t^{i-d_j} e_j$  en  $x'$  y  $c'_j$  es el coeficiente de  $t^{i-d_j} e_j$  en la suma  $\sum_{k=1}^{m_i} r'_k t^{i-d'_k} h_{ki}$ . Esto es,  $c_j = c'_j$ . Como  $x$  se escogió de manera arbitraria de  $\ker \mu_i$ , entonces  $H_i$  lo genera.  $\square$

**Lema 4.4.**  $\beta(\mathbf{M})$  es de tipo finito. En particular, es de tipo finitamente presentado.

*Demostración.* Sea  $N$  el grado máximo de los  $g_j \in G_j$ ,  $h_k \in H_k$  de forma que  $1 \leq j \leq n$ ,  $1 \leq k \leq m$ . Veamos que la multiplicación por  $t$  induce un isomorfismo entre  $M_i$  y  $M_{i+1}$  para todo  $i \geq N$ .

Si  $y \in M_{i+1}$ , entonces existen  $\lambda_j \in R[t]$  de grado al menos 1 de forma que  $y = \sum_{j=1}^n \lambda_j g_j$ . Por tanto,  $y = ty'$  donde  $y' \in M_i$  mostrando que la multiplicación por  $t$  es sobreyectiva.

Para ver que es inyectiva, consideremos  $y \in M_i$  de forma que  $ty = 0$ . Sea  $x \in R[t]^n$  tal que  $\mu(x) = y$ . Entonces  $\mu(tx) = ty = 0$  y por tanto, veamos  $tx$  se puede escribir como

$$tx = \sum_{j=0}^m \tilde{\lambda}_j h_j,$$

donde cada  $\lambda_j$  no trivial es un polinomio de grado al menos 1. Es inmediato, pues cada  $h_j$  es de grado menor o igual que  $N$  y  $tx$  es de grado mayor o igual que  $N + 1$ . En consecuencia, también podemos descomponer  $tx$  como

$$tx = \sum_{j=0}^m t\lambda_j h_j = t \sum_{j=0}^m \lambda_j h_j.$$

Por ser  $R[t]^n$  un módulo libre, tenemos que  $x = \sum_{j=0}^m \lambda_j h_j$  y por tanto,  $x \in \ker \mu$  lo que implica que  $y = 0$ .  $\square$

**Teorema 4.2** (Teorema de correspondencia). *Sea  $R$  un anillo unitario. Entonces, existe un isomorfismo entre la categoría de  $R[t]$ -módulos graduados finitamente presentados y la categoría de módulos de persistencia discretos de tipo finitamente presentados.*

*Demostración.* Estas categorías son subcategorías de  $R[t]\text{-GrMod}$  y  $R\text{-PersMod}$  respectivamente. Restringiendo  $\alpha$  y  $\beta$  a dichas subcategorías, por el Lema 4.2 tenemos que  $\alpha$  es un funtor covariante de los módulos de persistencia discretos de tipo finitamente presentados a los  $R[t]$ -módulos graduados finitamente presentados. Así mismo, por los lemas 4.3 y 4.4,  $\beta$  es un funtor covariante de los  $R[t]$ -módulos graduados finitamente presentados a los módulos de persistencia discretos de tipo finitamente presentados. En consecuencia, estas subcategorías son isomorfas.  $\square$

El Teorema de descomposición de módulos graduados nos dará la clave para representar los módulos de persistencia. La idea general de la prueba consiste en aplicar el Descomposición cíclica primaria y ajustar la graduación mediante las potencias de  $t$  para garantizar

#### 4. Homología persistente

el isomorfismo. Dicha demostración hace uso de algunos resultados que, por motivo de extensión, no trataremos en el presente trabajo. Para el lector interesado en los detalles de la prueba véase [118].

**Teorema 4.3** (Teorema de descomposición de módulos graduados). *Sea  $F$  un cuerpo y sea  $M$  un  $F[t]$ -módulo graduado finitamente generado. Entonces  $M$  se descompone de manera única, salvo isomorfismos, como*

$$M \cong \left( \bigoplus_{i=1}^{n-m} t^{a_i} \cdot R[t] \right) \oplus \left( \bigoplus_{j=1}^m t^{b_j} \cdot \frac{R[t]}{\langle t^{c_j} \rangle} \right),$$

donde  $a_i, b_j, c_j \in \mathbb{N}$ , y para cada  $j$ ,  $t^{c_j}$  es un elemento homogéneo tal que divide a  $t^{c_{j+1}}$ .

De manera intuitiva, primero hemos construido una única estructura algebraica que contiene todos los complejos de la filtración. Después, hemos seguido calculando una suma directa a partir de los complejos, llegando así a un espacio mucho más grande que está graduado según el orden inducido de la filtración. A continuación, recordamos el momento en que cada simplice entra utilizando un coeficiente polinomial, codificando así el orden de filtración en el anillo de coeficientes polinomiales. A partir de aquí, el **Teorema de descomposición de módulos graduados** nos proporciona una factorización simple cuando el anillo base es un cuerpo  $F$ . Aquí, el anillo graduado  $F[t]$  es un DIP y sus únicos ideales graduados son homogéneos de la forma  $\langle t^n \rangle = t^n \cdot R[t]$ , donde  $n \geq 0$ . Ahora que hemos transformado los módulos de persistencia discretos en objetos más manejables, queremos parametrizar las clases de isomorfismo de los  $F[t]$ -módulos por objetos más sencillos de interpretar.

**Definición 4.13.** Definimos un  **$\mathcal{P}$ -intervalo** como un par ordenado  $(i, j)$  tal que  $0 \leq i \leq j$  donde  $i, j \in \mathbb{Z}^\infty = \mathbb{Z} \cup \{+\infty\}$ .

Nuestro objetivo ahora es asociar un  $F[t]$ -módulo graduado a un conjunto  $S$  de  $\mathcal{P}$ -intervalos mediante una biyección  $Q$ . Para ello, definimos  $Q(i, j) = t^i F[t]/\langle t^{j-i} \rangle$  para el  $\mathcal{P}$ -intervalo  $(i, j)$ . Es claro que  $Q(i, +\infty) = t^i F[t]$ . Además, para un conjunto de  $\mathcal{P}$ -intervalos  $S = \{(i_1, j_1), (i_2, j_2), \dots, (i_n, j_n)\}$ , definimos

$$Q(S) = \bigoplus_{l=1}^n Q(i_l, j_l).$$

Nuestra correspondencia puede ahora redefinirse como sigue.

**Corolario 4.1.** La correspondencia  $S \mapsto Q(S)$  define una biyección entre los conjuntos finitos de  $\mathcal{P}$ -intervalos y los módulos graduados finitamente generados sobre el anillo graduado  $F[t]$ . Consecuentemente, las clases de isomorfismo de los módulos de persistencia de tipo finito sobre  $F$  están en correspondencia biyectiva con los conjuntos finitos de  $\mathcal{P}$ -intervalos.

En el contexto de filtraciones de complejos simpliciales, este resultado da lugar a la descripción de la homología persistente conocida como **código de barras**: cada  $\mathcal{P}$ -intervalo  $(i, j)$  describe un ciclo que nace en la filtración  $i$  y especifica su clase de homología hasta que se convierte en borde (es decir, hasta que muere) en el instante  $j$ .

Los diagramas de código de barras derivados de la homología persistente han demostrado ser herramientas muy útiles para la extracción de información relevante en campos como el análisis de datos. Este hecho es debido a ciertas propiedades de la homología persistente que la hacen relevante para un uso práctico. Además de las ya comentadas, también destaca su

**estabilidad**, lo que significa que pequeñas perturbaciones en los datos no producen grandes cambios en los diagramas de código de barras. Esto es fundamental porque asegura que las características topológicas que se detectan son inherentes a la estructura de los datos y no consecuencia del ruido o de variaciones menores [21, 13]. Dicha propiedad contrasta con la homología simplicial, pues dos complejos de Čech o Vietoris-Rips con distinto radio pueden tener invariantes topológicos completamente diferentes. Estos métodos no solo ofrecen una representación visual intuitiva de las características de los datos, sino que también proporcionan un marco robusto para la cuantificación y comparación de estructuras topológicas.

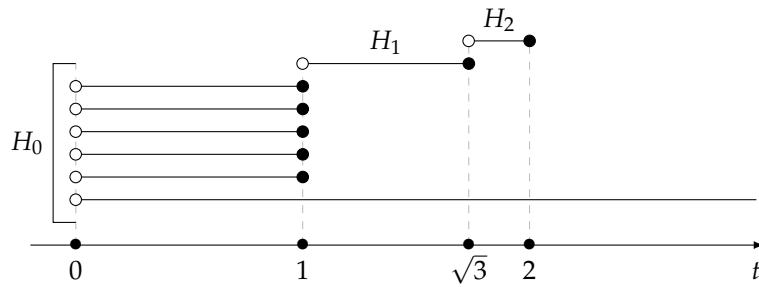


Figura 4.3.: Ejemplo de código de barras mostrando la persistencia de clases de homología a través del tiempo. Cada línea horizontal representa una clase de homología que persiste a través de un intervalo específico, indicando la escala a la que dichas características topológicas son detectables.



## **Parte II.**

### **Aprendizaje profundo**



## 5. Aprendizaje automático y visión artificial

La **inteligencia artificial** (IA) se define en [89] como el campo de estudio de la informática que se centra en la creación de agentes inteligentes, es decir, sistemas que perciben su entorno y toman acciones que maximizan sus posibilidades de éxito en algún objetivo o tarea.

La IA abarca una amplia variedad de problemas y aplicaciones, entre los que destacan:

- **Resolución de Problemas y Búsqueda:** La IA desarrolla algoritmos para encontrar soluciones óptimas o satisfactorias en espacios de búsqueda complejos. Esto incluye problemas clásicos como el ajedrez [95], el rompecabezas del cubo de Rubik [4] o la planificación de rutas [7].
- **Representación del Conocimiento y Razonamiento:** Este área se centra en cómo representar información sobre el mundo de manera que un ordenador pueda utilizarla para resolver problemas complejos y tomar decisiones. Ejemplos incluyen sistemas expertos [42] y razonamiento basado en casos [2].
- **Aprendizaje Automático:** Como veremos posteriormente, el aprendizaje automático se enfoca en el desarrollo de algoritmos que permitan a los ordenadores aprender a partir de los datos y mejorar con la experiencia.
- **Procesamiento del Lenguaje Natural (NLP):** El NLP busca permitir que los ordenadores comprendan, interpreten y respondan al lenguaje humano de manera útil. Algunas aplicaciones de este campo incluyen la traducción automática [6], el análisis de sentimientos [24] y la implementación de agentes conversacionales [10].
- **Visión Artificial:** Este campo se enfoca en que los ordenadores sean capaces de interpretar y comprender el contenido de imágenes y vídeos. Aplicaciones incluyen el reconocimiento facial, la conducción autónoma y la detección de objetos.
- **Robótica:** La IA aplicada a la robótica busca diseñar agentes que interactúen con el mundo físico. Esto incluye tareas como la navegación [16], la manipulación de objetos [38] y la interacción humano-robot [91].

Es importante destacar que los problemas que aborda la IA pueden involucrar múltiples áreas. Por ejemplo, un sistema de conducción autónoma requiere la integración de visión artificial para interpretar el entorno, aprendizaje automático para mejorar su desempeño a partir de la experiencia y toma de decisiones, y algoritmos de planificación para determinar la mejor ruta a seguir. Esta interrelación de diferentes áreas y técnicas permite a los sistemas de IA abordar problemas complejos y dinámicos de manera más eficaz.

### 5.1. Aprendizaje automático

El **aprendizaje automático** o *machine learning* es la subdisciplina de la IA que se centra en el desarrollo de algoritmos y técnicas que permiten a los ordenadores aprender y hacer

## 5. Aprendizaje automático y visión artificial

predicciones o tomar decisiones basadas en datos [44]. Esta capacidad de aprendizaje se logra a través de la construcción de modelos que pueden mejorar su rendimiento en tareas específicas mediante la experiencia y los datos disponibles. Este enfoque supuso un cambio de paradigma respecto al diseño de algoritmos tradicional, donde su desempeño dependía explícitamente del conocimiento sobre el problema de los ingenieros que lo implementaban.

Existen varios paradigmas del aprendizaje automático en función del nivel de supervisión humana necesaria durante el proceso de aprendizaje. Los tres tipos principales son:

- **Aprendizaje supervisado:** En este tipo, el algoritmo aprende a partir de un conjunto de datos etiquetados [3]. Cada ejemplo en el conjunto de datos incluye una entrada y una salida esperada. El objetivo es que el modelo aprenda a identificar las salidas correctas a partir de las entradas para poder hacer predicciones sobre datos nuevos. Aplicaciones comunes incluyen la clasificación y la regresión.
- **Aprendizaje no supervisado:** A diferencia del aprendizaje supervisado, en el aprendizaje no supervisado los datos no están etiquetados [41]. El objetivo es descubrir estructuras o patrones ocultos en los datos. Las técnicas de agrupamiento o *clustering* y la reducción de dimensionalidad son ejemplos comunes de aprendizaje no supervisado.
- **Aprendizaje por refuerzo:** En este enfoque, un agente aprende a tomar decisiones mediante la interacción con un entorno dinámico [101]. El agente recibe recompensas o penalizaciones en función de las acciones que realiza, y su objetivo es maximizar la recompensa acumulada a lo largo del tiempo. Este tipo de aprendizaje es especialmente útil en problemas de toma de decisiones secuenciales, como en juegos y robótica.

El aprendizaje automático se utiliza para abordar una amplia variedad de problemas en diferentes dominios. Algunos de los problemas más destacados incluyen:

- **Clasificación:** El objetivo de la clasificación es asignar una etiqueta a una entrada entre un conjunto de etiquetas posibles. Ejemplos de problemas de clasificación incluyen el reconocimiento de imágenes [90], el filtrado de *spam* [23] y el diagnóstico médico [97].
- **Regresión:** La regresión se centra en predecir un valor continuo a partir de una entrada. Un ejemplo clásico es la predicción de precios de viviendas basándose en características como el tamaño y la ubicación [115].
- **Agrupamiento (*Clustering*):** Este problema implica agrupar un conjunto de datos en subconjuntos, o clusters, de tal manera que los datos en el mismo *cluster* sean más similares entre sí que con los datos de otros *clusters*. Aplicaciones incluyen segmentación de mercado [57] y análisis de imágenes [81].
- **Reducción de Dimensionalidad:** La reducción de dimensionalidad se utiliza para simplificar modelos complejos reduciendo el número de variables en el análisis de datos. Técnicas como el Análisis de Componentes Principales (PCA) [121] o *Locally Linear Embedding* (LLE) [86] son comunes en este campo.
- **Detección de Anomalías:** También conocido como detección de *outliers*, este problema implica identificar datos que no se ajustan al comportamiento normal esperado. Es crucial en áreas como la detección de fraudes [80] y la monitorización de sistemas [73].

El avance en el aprendizaje automático ha llevado a desarrollos significativos en múltiples campos, transformando la manera en que se analizan los datos y se toman decisiones en el mundo real.

## 5.2. Visión artificial

Como hemos comentado anteriormente de manera breve, la **visión artificial** es la rama de la IA cuyo objetivo es comprender el mundo que percibimos a partir de una o más imágenes y reconstruir propiedades suyas como su forma, iluminación o distribución de colores [103]. Su objetivo final es desarrollar sistemas que puedan interpretar y comprender el entorno visual de manera efectiva.

La visión artificial es un campo muy diverso que integra métodos de varias áreas del conocimiento para lograr su objetivo principal. Como resultado, abarca una amplia gama de aplicaciones y tareas. Algunas de las más destacadas incluyen:

- **Clasificación de objetos:** identificar y categorizar objetos en una imagen, lo que es fundamental en aplicaciones como la detección de alimentos en mal estado [49] o la clasificación de productos en una tienda *online* [55].
- **Generación de imágenes:** crear imágenes a partir de modelos o datos, lo que tiene aplicaciones en la creación de gráficos 3D [65], la síntesis de imágenes para la publicidad [76] o la creación de contenido en redes sociales [19].
- **Reconstrucción de entornos 3D:** reconstruir entornos tridimensionales a partir de imágenes 2D, lo que es fundamental en aplicaciones como la creación de modelos 3D de edificios [63] o el mapeo en un entorno desconocido [99].
- **Detección de objetos:** detectar la presencia de objetos en una imagen, lo que es fundamental en aplicaciones como la seguridad en espacios públicos [78] o la detección de anomalías en la producción industrial [5].
- **Segmentación de imágenes:** dividir una imagen en regiones significativas, lo que es fundamental en aplicaciones como la medicina [46], la agricultura [17] o la minería [108].

La visión artificial es un campo en constante evolución que se ha desarrollado significativamente en las últimas décadas, al punto de haber superado a los seres humanos en ciertas aplicaciones específicas, como el reconocimiento de imágenes [25] y la detección de objetos [114]. Sin embargo, aún quedan numerosos desafíos por abordar, especialmente en contextos más complejos y variados, como la interpretación de escenas en tiempo real y la comprensión de imágenes en condiciones adversas.



## 6. Redes neuronales artificiales

Dentro de los distintos modelos de aprendizaje automático, nosotros trabajaremos con redes neuronales artificiales. Para entender mejor su funcionamiento, primero hablaremos de las neuronas biológicas y su inspiración en los modelos artificiales.

### 6.1. Neuronas biológicas

Las neuronas biológicas son células especializadas del sistema nervioso que desempeñan roles clave en la recepción, procesamiento y transmisión de señales [85]. Estas células son fundamentales para la comunicación neuronal, permitiendo el flujo de información a través de complejas redes en el cerebro y otros componentes del sistema nervioso.

Las neuronas biológicas están compuestas por tres partes principales: el **soma**, que es el cuerpo celular conteniendo el núcleo y orgánulos celulares; las **dendritas**, extensiones que reciben señales de otras neuronas; y el **axón**, una prolongación especializada que conduce los potenciales de acción desde el soma hasta las terminaciones sinápticas. Estas terminaciones sinápticas se encuentran en las sinapsis, donde se produce la transmisión de señales a otras neuronas o tejidos ([Figura 6.1](#)).

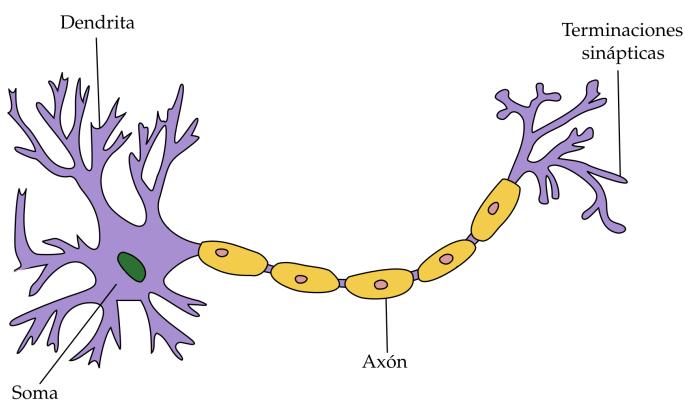


Figura 6.1.: Representación esquemática de una neurona biológica, mostrando el soma, las dendritas y el axón. Fuente [User:Dhp1080, CC BY-SA 3.0](#), via Wikimedia Commons.

Esta compleja red de conexiones y la regulación electroquímica hacen que las neuronas biológicas sean fundamentales para coordinar y ejecutar numerosas funciones, desde simples reflejos hasta procesos cognitivos avanzados.

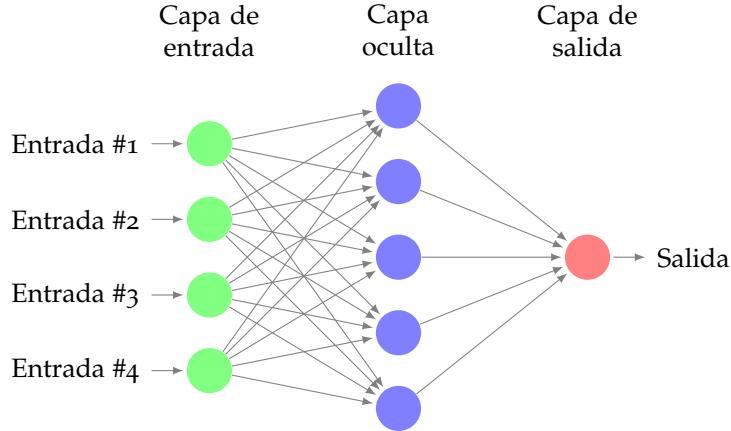


Figura 6.2.: Estructura de una FNN. Muestra una red neuronal hacia adelante con tres tipos de capas: entrada (verde), oculta (azul), y salida (rojo). Cada neurona de la capa de entrada está conectada a todas las neuronas en la capa oculta, demostrando un ejemplo de capas totalmente conectadas.

## 6.2. Redes neuronales artificiales

Las **neuronas artificiales** se inspiran en las neuronas biológicas en el sentido de que ambas tienen la capacidad de recibir, procesar y transmitir señales. En el caso de las neuronas artificiales, estas reciben señales de otras neuronas dentro de una red y las procesan mediante una combinación lineal de una matriz de **pesos** ajustables  $\mathbf{W}$  y un vector de **sesgo** o **bias**  $\mathbf{b}$ . Durante el entrenamiento de la red, estos parámetros se refinan para optimizar el rendimiento de la red en tareas específicas como clasificación o regresión. Cada neurona artificial aplica una función escalar denominada **función de activación**  $\theta : \mathbb{R}^m \rightarrow \mathbb{R}$ , que transforma la combinación lineal de sus entradas:

$$f(\mathbf{x}, \mathbf{W}, \mathbf{b}) = \theta(\mathbf{W}\mathbf{x} + \mathbf{b}), \quad \forall \mathbf{x} \in \mathbb{R}^d, \mathbf{W} \in \mathbb{R}^{m \times d}, \mathbf{b} \in \mathbb{R}^m.$$

Aquí,  $d$  es la dimensión de entrada y  $m$  es el número de neuronas en la capa.

Estas neuronas se agrupan en **capas** dentro de una **red neuronal artificial** (ANN). La estructura de la red se define por su **profundidad**, que corresponde al número de capas ocultas, y su **anchura**, que se refiere al número de neuronas en cada capa.

En el caso de las **redes neuronales hacia adelante** o *feedforward networks* (FNNs), la red está compuesta por varias capas cuyas neuronas solamente están conectadas a neuronas en capas posteriores (Figura 6.2.). En el caso donde todas las salidas de una capa están conectadas a todas las neuronas de la siguiente capa se les conoce como **capas totalmente conectadas** o *fully connected layers* (FC). Las funciones de cada capa en una FNN pueden describirse como:

$$f_j(\mathbf{x}) = \theta(\mathbf{W}_j\mathbf{x} + \mathbf{b}_j),$$

donde  $\mathbf{W}_j$  es la matriz de pesos y  $\mathbf{b}_j$  es el vector de sesgo para la capa  $j$ . Estos elementos son fundamentales para determinar cómo se transforman las entradas en salidas que serán utilizadas por la siguiente capa. La estructura funcional de las FNNs se describe entonces

mediante la composición de estas funciones organizadas en capas:

$$v = s \circ f_l \circ f_{l-1} \circ \dots \circ f_2 \circ f_1(\mathbf{x}),$$

donde  $l$  denota el número total de capas. La última capa de la red utiliza una función de activación  $s$ , que convierte la salida de la capa final en uno o varios valores en función de la tarea designada a la red. Esta función de salida es crítica para aplicaciones como clasificación, donde el resultado necesita expresarse como una probabilidad, o para regresión, donde se ajusta a un rango específico.

### 6.3. Funciones de activación

Las funciones de activación son de gran importancia en la arquitectura de las ANNs, ya que introducen no linealidades en el modelo, permitiendo a la red aprender y modelar relaciones complejas en los datos. Sin estas funciones, la red no podría resolver tareas más complejas que una simple regresión lineal.

Anteriormente, hemos definido las funciones de activación en un contexto vectorial, aplicando una única transformación a todas las entradas procedentes de la capa anterior. Sin embargo, en la práctica es habitual emplear una función de activación escalar  $\theta : \mathbb{R} \rightarrow I$ , donde  $I$  es un intervalo específico, de forma que se aplica componente a componente a cada combinación lineal de entrada.

Una de las funciones de activación más antiguas es la **sigmoide**, que se define como:

$$\theta(z) = \frac{1}{1 + e^{-z}},$$

donde  $\theta : \mathbb{R} \rightarrow [0, 1]$  y  $z$  es la combinación lineal de entradas, pesos y sesgo de la neurona. La función sigmoide transforma los valores de entrada a un rango entre 0 y 1, modelando de esta forma probabilidades. Aunque su uso ha disminuido en redes profundas debido al problema del **desvanecimiento del gradiente**, problema que consiste en que los gradientes van disminuyendo progresivamente conforme profundizan en las capas de la red, impidiendo así su actualización. A pesar de ello, sigue siendo relevante en la capa de salida de ANNs para clasificación binaria.

La función **Softmax** es una generalización de la sigmoide para múltiples clases y se define para un vector  $z \in \mathbb{R}^K$  como:

$$\theta(z)_i = \frac{e^{z_i}}{\sum_{k=1}^K e^{z_k}} \quad \text{para } i = 1, \dots, K,$$

donde  $\theta : \mathbb{R}^K \rightarrow [0, 1]^K$ . Cada componente de la salida,  $\theta(z)_i$ , representa la probabilidad de que la entrada pertenezca a la clase  $i$ , y se utiliza principalmente en la capa de salida de las redes neuronales para tareas de clasificación multiclas.

La **tangente hiperbólica** ( $\tanh$ ), que se define como:

$$\theta(z) = \tanh(z) = \frac{2}{1 + e^{-2z}} - 1,$$

donde  $\theta : \mathbb{R} \rightarrow [-1, 1]$ , es preferida sobre la sigmoide en algunos escenarios por su salida centrada en cero, que facilita la optimización durante el entrenamiento. Aunque también

## 6. Redes neuronales artificiales

sufre del problema de desvanecimiento de gradiente, su uso en capas ocultas es bastante frecuente.

La *Rectified Linear Unit* (ReLU), definida como:

$$\theta(z) = \max(0, z),$$

donde  $\theta : \mathbb{R} \rightarrow [0, \infty)$ , es la más utilizada en aprendizaje profundo por su simplicidad y eficiencia computacional. ReLU se utiliza generalmente en capas ocultas, pero no es adecuada para la capa de salida en tareas de clasificación debido a su rango no acotado. Además, ReLU sufre de un problema de «**muerte de neuronas**» o «*dying ReLU*», que sucede cuando los valores de entrada son menores o iguales a cero. Debido a que la salida de ReLU es cero para estos valores, los gradientes también son cero durante la retropropagación. Como resultado, estas neuronas dejan de aprender y contribuyen poco o nada al modelo, afectando negativamente el rendimiento de la red.

La *Leaky ReLU* es una variante de la ReLU definida como:

$$\theta(z) = \max(\alpha z, z),$$

donde  $\alpha$  es un coeficiente real positivo y  $\theta : \mathbb{R} \rightarrow \mathbb{R}$ . Esta función se utiliza también en capas ocultas para evitar la muerte de neuronas que puede ocurrir con ReLU.

Finalmente tenemos la *Sigmoid Linear Unit* (SiLU), también conocida como **Swish**, es una de las funciones de activación más recientes que combina elementos de la función sigmoide y linealidad. Se define como:

$$\theta(z) = \frac{z}{1 + e^{-z}},$$

donde  $\theta : \mathbb{R} \rightarrow \mathbb{R}$ . A diferencia de la ReLU y sus variantes, la SiLU permite que valores negativos tengan una contribución, aunque moderada, a la activación, lo que resulta en una capacidad mejorada para ajustar gradientes durante el entrenamiento. Esta característica hace que la SiLU sea particularmente útil en capas profundas de redes neuronales. Esta función ha mostrado ventajas en términos de rendimiento de convergencia y eficacia en comparación con funciones más tradicionales [83].

Un aspecto muy importante de estas funciones es que **tienen que ser derivables** y, en caso de no serlo, adaptarlas para que sea derivable en todo punto donde se evalúe (Figura 6.3). El principal motivo es el método de aprendizaje empleado que, como veremos a continuación, ajusta el valor de los pesos en función de la derivada de las funciones que la componen.

Cada una de estas funciones de activación juega un papel importante en la arquitectura de una red neuronal. La elección de una sobre otra puede depender del problema específico que se esté abordando, del comportamiento de la red durante el entrenamiento y de la naturaleza de los datos.

## 6.4. Optimización en redes neuronales

El entrenamiento de estas redes se realiza mediante el uso de métodos de optimización que ajustan iterativamente los parámetros de la red para minimizar una función denominada **función de pérdida** o **coste**, denotada por  $\mathcal{L}$  [116]. Esta función evalúa la diferencia entre las salidas predichas por la red  $\hat{y}$  y los valores reales o esperados  $y$ .

Una elección típica de función de pérdida en tareas de clasificación multiclase es la **entropía cruzada** o *cross entropy*, que se utiliza en combinación con la función de activación **Softmax**.

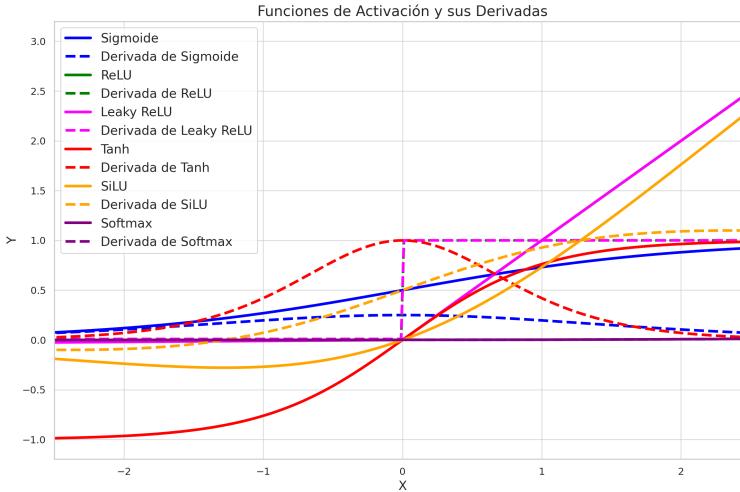


Figura 6.3.: Funciones de activación comunes en el campo del aprendizaje profundo: Sigmoid, ReLU, Leaky ReLU, Tanh, SiLU y Softmax, cada una representada por una línea continua. Las derivadas de cada función están indicadas con líneas discontinuas del mismo color.

Softmax se aplica en la última capa de la red para convertir las salidas lineales en un vector de probabilidades que suman uno, siendo cada componente  $i$  la evaluación de Softmax en la respectiva clase. De esta forma,

$$\hat{\mathbf{y}} = \text{Softmax}(\mathbf{z}) = \left( \frac{e^{z_1}}{\sum_{j=1}^N e^{z_j}}, \frac{e^{z_2}}{\sum_{j=1}^N e^{z_j}}, \dots, \frac{e^{z_N}}{\sum_{j=1}^N e^{z_j}} \right),$$

donde  $\mathbf{z}$  representa las salidas de la última capa de la red antes de la aplicación de Softmax. Con el vector de probabilidades establecido por Softmax, la entropía cruzada se calcula entonces como:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = - \sum_{i=1}^N y_i \log(\hat{y}_i).$$

Aquí,  $\mathbf{y} = (y_1, y_2, \dots, y_N)$  es el vector de etiquetas reales en formato *one-hot*, donde  $y_i = 1$  si la clase  $i$  es la correcta y  $y_i = 0$  en caso contrario. Cada elemento  $\hat{y}_i$  en  $\hat{\mathbf{y}}$  indica la probabilidad predicha de que la entrada pertenezca a la clase  $i$ , y la suma de todas las probabilidades es igual a uno. La función de entropía cruzada mide entonces la diferencia entre las distribuciones de las etiquetas reales y las probabilidades predichas, y su minimización lleva al modelo a mejorar la precisión de sus predicciones.

En el ámbito del aprendizaje profundo también es común notar la función de pérdida como  $\mathcal{L}(\mathcal{W}; \mathbf{x}, \mathbf{y})$ , siendo  $\mathcal{W}$  el conjunto de pesos de la red y  $\mathbf{x}$  el vector de entrada, de forma que  $\mathcal{L}(\mathcal{W}; \mathbf{x}, \mathbf{y}) = \mathcal{L}(\nu(\mathbf{x}), \mathbf{y})$ .

El principal método de entrenamiento de redes neuronales consta de dos pasos: el **paso hacia adelante** o *forward pass*, que calcula la salida de la red a partir de la entrada; y la **propagación hacia atrás**, también conocida como **retropropagación** o *backpropagation*. Este algoritmo es un método para calcular el gradiente de la función de pérdida con respecto a

## 6. Redes neuronales artificiales

cada parámetro de la red aplicando la regla de la cadena. Este proceso se inicia en la capa de salida y se propaga hacia atrás a través de la red, de la siguiente manera:

$$\frac{\partial \mathcal{L}}{\partial w_{ij}^{(l)}} = \frac{\partial \mathcal{L}}{\partial x_i^{(l+1)}} \cdot \frac{\partial x_i^{(l+1)}}{\partial z_i^{(l+1)}} \cdot \frac{\partial z_i^{(l+1)}}{\partial w_{ij}^{(l)}},$$

donde  $\mathcal{L}$  es la función de pérdida,  $w_{ij}^{(l)}$  son los pesos de la capa  $l$ ,  $x_i^{(l+1)}$  es la salida de la neurona  $i$  en la capa  $l + 1$ , y  $z_i^{(l+1)}$  es la entrada en forma de combinación lineal a la neurona  $i$  en la capa  $l + 1$ , que se calcula como  $z_i^{(l+1)} = \sum_j w_{ij}^{(l)} x_j^{(l)} + b_i^{(l)}$ . La derivada  $\frac{\partial x_i^{(l+1)}}{\partial z_i^{(l+1)}}$  es la derivada de la función de activación  $\theta$  aplicada a  $z_i^{(l+1)}$ .

Como los métodos de optimización que emplearemos están basados en el cálculo del gradiente, restringiremos la optimización a los vectores de pesos  $\mathbf{w}$  en las distintas capas. En consecuencia, definiremos la función a optimizar como **función objetivo**  $J$  tal que  $J(\mathcal{W}) = \mathcal{L}(\mathcal{W}; \mathbf{x}, \mathbf{y})$ . Sin embargo, por comodidad abusaremos de la notación y escribiremos el gradiente de  $J$  respecto al vector  $\mathbf{w}$ ,  $\nabla_{\mathbf{w}} J(\mathcal{W})$ , simplemente por  $\nabla J(\mathbf{w})$ .

La optimización efectiva de estos pesos es crucial para el rendimiento de la red, y se lleva a cabo a través de varias iteraciones de entrenamiento, ajustando progresivamente los parámetros para reducir el error y mejorar la precisión de la clasificación o la predicción de la red.

### 6.4.1. Descenso del gradiente

El algoritmo de **descenso del gradiente** [12] es el enfoque más básico de optimización, donde los pesos se actualizan en la dirección opuesta al gradiente de la función objetivo:

$$\mathbf{w}_{\text{nuevo}} = \mathbf{w}_{\text{viejo}} - \alpha \nabla J(\mathbf{w}_{\text{viejo}}),$$

donde  $\alpha$  representa la **tasa de aprendizaje** y  $\nabla J(\mathbf{w}_{\text{viejo}})$  es el gradiente de la función de pérdida con respecto a los pesos anteriores. El descenso del gradiente puede implementarse de varias maneras dependiendo de cómo se seleccionan y utilizan los datos para calcular el gradiente de la función de pérdida. Una forma es el **descenso del gradiente estocástico** (SGD) [56], que actualiza el conjunto de pesos después de cada ejemplo de entrenamiento, siendo muy utilizado en grandes conjuntos de datos. Otra variante es el **descenso del gradiente por lotes**, que calcula el gradiente utilizando todo el conjunto de datos antes de realizar una actualización, asegurando una actualización consistente de los pesos. Finalmente, el **descenso del gradiente por mini-lotes** divide el conjunto de datos en pequeños lotes y realiza actualizaciones de los pesos después de procesar cada mini-lote, combinando aspectos de las dos técnicas anteriores.

### 6.4.2. Momentum

Una técnica que mejora la eficacia del descenso del gradiente es el **momentum** [88], que ayuda a acelerar el algoritmo en la dirección correcta mientras suaviza las actualizaciones de pesos. En lugar de actualizar los pesos basándose únicamente en el gradiente actual, el **momentum** también considera los gradientes anteriores para obtener una dirección más estable y consistente. Esto se logra mediante la introducción de una variable  $\mathbf{v}_t$ , conocida como el

**término de momentum**, que acumula los gradientes pasados con un **factor de descuento**  $\gamma$ . La fórmula de actualización con *momentum* es entonces:

$$\mathbf{v}_t = \gamma \mathbf{v}_{t-1} + \alpha \nabla J(\mathbf{w}_{\text{viejo}}),$$

$$\mathbf{w}_{\text{nuevo}} = \mathbf{w}_{\text{viejo}} - \mathbf{v}_t,$$

donde  $\alpha$  es la tasa de aprendizaje. Este enfoque no solo acelera la convergencia, sino que también puede ayudar a evitar los mínimos locales subóptimos, haciendo que el algoritmo sea más eficaz y robusto en práctica.

### 6.4.3. Adagrad

*Adaptive Gradient Algorithm* (Adagrad) [30] es un algoritmo de optimización que adapta individualmente la tasa de aprendizaje de cada parámetro basándose en la magnitud acumulada de sus gradientes. La actualización de los pesos  $\mathbf{w}$  se realiza mediante la siguiente fórmula:

$$\mathbf{w}_{\text{nuevo}} = \mathbf{w}_{\text{viejo}} - \frac{\alpha}{\sqrt{\mathbf{G}^{(t)} + \epsilon}} \odot \nabla J(\mathbf{w}_{\text{viejo}}),$$

donde  $\mathbf{G}^{(t)}$  es una matriz diagonal en la que cada elemento  $G_{ii}^{(t)}$  representa la suma acumulada de los cuadrados de las derivadas parciales de los gradientes con respecto a cada componente  $w_i$  de  $\mathbf{w}$  hasta el instante de tiempo  $t$ . El **producto de Hadamard**  $\odot : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}^N$  se define como el producto escalar componente a componente. El término  $\epsilon$  es un pequeño valor constante añadido a cada elemento de  $\mathbf{G}^{(t)}$  para garantizar estabilidad numérica y evitar la división por cero. Cada elemento  $G_{ii}^{(t)}$  de la matriz se actualiza como sigue:

$$G_{ii}^{(\text{nuevo})} = G_{ii}^{(\text{viejo})} + \left( \frac{\partial J}{\partial w_i}(\mathbf{w}_{\text{viejo}}) \right)^2,$$

permitiendo que las tasas de aprendizaje sean más bajas para parámetros con gradientes altos y mayores para aquellos con gradientes menores, lo cual facilita una convergencia más rápida y estable del algoritmo.

### 6.4.4. RMSProp

*Root Mean Square Propagation* (RMSProp) [48] modifica Adagrad para mejorar su rendimiento manteniendo un promedio móvil del cuadrado de los gradientes. Esto ajusta la tasa de aprendizaje de manera más adecuada para problemas a largo plazo:

$$\mathbf{v}_t = \beta \mathbf{v}_{t-1} + (1 - \beta)(\nabla J(\mathbf{w}_{\text{viejo}}))^2,$$

$$\mathbf{w}_{\text{nuevo}} = \mathbf{w}_{\text{viejo}} - \frac{\alpha}{\sqrt{\mathbf{v}_t + \epsilon}} \odot \nabla J(\mathbf{w}_{\text{viejo}}),$$

donde  $\mathbf{v}_t$  es la media móvil del cuadrado de los gradientes y  $\beta$  es un factor de descuento que determina la ponderación de dicha media.

## 6. Redes neuronales artificiales

### 6.4.5. Adam

*Adaptive Moment Estimation* (Adam) [58] combina las ideas detrás de momentum y RMSprop. Mantiene estimaciones de los primeros y segundos momentos de los gradientes para ajustar la tasa de aprendizaje de cada parámetro de manera individual:

$$\begin{aligned}\mathbf{m}_t &= \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \nabla J(\mathbf{w}_{\text{viejo}}), \\ \mathbf{v}_t &= \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) (\nabla J(\mathbf{w}_{\text{viejo}}))^2, \\ \hat{\mathbf{m}}_t &= \frac{\mathbf{m}_t}{1 - \beta_1^t}, \\ \hat{\mathbf{v}}_t &= \frac{\mathbf{v}_t}{1 - \beta_2^t}, \\ \mathbf{w}_{\text{nuevo}} &= \mathbf{w}_{\text{viejo}} - \frac{\alpha}{\sqrt{\hat{\mathbf{v}}_t + \epsilon}} \odot \hat{\mathbf{m}}_t,\end{aligned}$$

donde  $\mathbf{m}_t$  y  $\mathbf{v}_t$  son estimaciones del primer y segundo momento respectivamente,  $\hat{\mathbf{m}}_t$  y  $\hat{\mathbf{v}}_t$  son sus correcciones sesgadas, y  $\beta_1, \beta_2$  son tasas de decaimiento exponencial para los momentos estimados.

La selección del algoritmo de optimización es una decisión clave que puede variar dependiendo de la naturaleza del problema y el tipo de red neuronal que se está entrenando. SGD y Adam son especialmente populares debido a su robustez y buen desempeño en una amplia variedad de problemas. Sin embargo, la eficiencia de estos algoritmos no solo se debe a su diseño, sino también a la selección adecuada de hiperparámetros, como la tasa de aprendizaje y los coeficientes de momentum. Una tasa de aprendizaje mal configurada puede llevar al algoritmo a converger demasiado lentamente o a no converger en absoluto [9]. Por lo tanto, el ajuste fino de estos hiperparámetros, que a menudo requiere experimentación y experiencia, es importante para maximizar el rendimiento del modelo.

## 6.5. Regularización en redes neuronales

Un problema muy conocido en el ámbito del aprendizaje profundo es el **sobreajuste** u *overfitting*, el cual ocurre cuando una red neuronal aprende patrones específicos del conjunto de entrenamiento en lugar de las verdaderas relaciones subyacentes entre los datos. Este fenómeno resulta en un rendimiento deficiente cuando la red se expone a datos nuevos y no vistos durante el entrenamiento. Para combatir el sobreajuste y mejorar la capacidad de generalización de las redes neuronales, se utilizan técnicas de **regularización**.

### 6.5.1. Regularización L1 y L2

Entre las técnicas más populares de regularización se encuentran la **regularización L1** y **L2** [74], también conocidas como *Lasso* y *Ridge* respectivamente. Estas técnicas modifican la función objetivo añadiendo términos que penalizan los pesos grandes de la red.

La regularización *L1*, o *Lasso*, añade a la función objetivo original  $J(\mathbf{w})$  un término proporcional a la suma de los valores absolutos de los pesos:

$$J_{L1}(\mathbf{w}) = J(\mathbf{w}) + \lambda \sum_i |w_i|,$$

donde  $\lambda$  es el parámetro de regularización. Este método es particularmente útil para generar modelos más interpretables al promover la **dispersión** o *sparsity* de los pesos, lo que resulta en que algunos de ellos sean exactamente cero, reduciendo así la complejidad del modelo.

Por otro lado, la regularización *L2*, o *Ridge*, añade un término proporcional a la suma de los cuadrados de los pesos:

$$J_{L2}(\mathbf{w}) = J(\mathbf{w}) + \lambda \sum_i w_i^2.$$

A diferencia de la regularización *L1*, la regularización *L2* penaliza más agresivamente los valores grandes de los pesos, favoreciendo soluciones con pesos más uniformemente distribuidos y pequeños. Esto contribuye a una mejor generalización del modelo al desincentivar los pesos grandes, lo que lleva a soluciones más homogéneas y en consecuencia, a modelos que pueden generalizar mejor ante nuevos datos.

### 6.5.2. Dropout

**Dropout** [100] es otra técnica ampliamente utilizada que implica desactivar aleatoriamente una proporción de neuronas durante cada iteración del entrenamiento:

$$x' = x \odot \mathcal{B}(p),$$

donde  $x$  es la salida de la función de activación de una capa y  $\mathcal{B}(p)$  es un vector binario aleatorio donde cada elemento tiene una probabilidad  $p \in [0, 1]$  de ser cero. Esta técnica reduce el sobreajuste al forzar a la red a aprender representaciones redundantes.

### 6.5.3. Early stopping

El **early stopping** [70] es una técnica de regularización diseñada para prevenir el sobreajuste al detener el entrenamiento de un modelo de aprendizaje automático antes de que se manifieste el sobreajuste. Este método consiste en monitorear el rendimiento del modelo en un conjunto de validación separado durante el entrenamiento. Si el error de validación no solo comienza a incrementar, sino que lo hace por un margen mayor a un **umbral** definido  $\delta$ , entonces indica que el modelo está empezando a aprender el ruido y las particularidades del conjunto de entrenamiento en lugar de las relaciones generales, por lo que el entrenamiento se detiene. La implementación de *early stopping* requiere definir otro hiperparámetro comúnmente denominado **paciencia**, que establece el número de épocas que se permite que el error de validación continúe aumentando antes de cesar el entrenamiento. Esta técnica no solo ayuda a mejorar la generalización del modelo sino que también puede reducir el tiempo de entrenamiento al evitar iteraciones innecesarias.

### 6.5.4. Regularización de datos

La **regularización de datos** abarca técnicas que modifican los datos de entrada para hacer el modelo menos sensible a pequeñas variaciones en los datos de entrenamiento. Una de las formas más comunes es el **aumento de datos** o *data augmentation*, que consiste en aplicar transformaciones a los datos de entrenamiento para generar nuevas muestras. Este método de regularización es muy empleado sobre conjuntos de imágenes dado sus buenos resultados [59]. Ejemplos de este tipo incluyen rotación, escalado, traslación y cambios en la intensidad del color de imágenes. Esta práctica enriquece el conjunto de entrenamiento y ayuda a que

## 6. Redes neuronales artificiales

el modelo sea más robusto frente a variaciones en la entrada, lo que es especialmente útil en tareas de visión artificial.

Por último, la **normalización por lotes** o *batch normalization* [53] es una técnica que normaliza las salidas de una capa a una media y desviación típica calculadas sobre el conjunto de datos de un mini-lote. Se define por:

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}},$$

donde  $x_i$  son las salidas de la capa anterior antes de la normalización,  $\mu_{\mathcal{B}}$  y  $\sigma_{\mathcal{B}}^2$  son la media y varianza calculadas para el mini-lote  $\mathcal{B}$  y  $\epsilon$  es un pequeño valor para asegurar la estabilidad numérica. La normalización por lotes no solo reduce el problema del desplazamiento de covarianza durante el entrenamiento, sino que también permite el uso de tasas de aprendizaje más altas y reduce la sensibilidad a la inicialización de los pesos.

Estas técnicas, ya sea de forma independiente o combinada, ayudan a asegurar que las redes neuronales no solo minimicen el error en el conjunto de entrenamiento, sino que también mantengan una buena capacidad de generalización a nuevos datos.

## 7. Redes neuronales convolucionales

Las **redes neuronales convolucionales** (CNNs) [59] son un tipo de modelo de ANN fundamental en el campo de la visión artificial y el procesamiento de imágenes. Fue introducido por Yann LeCun et al. en 1998 [61] y desde entonces ha sido ampliamente utilizado en una gran variedad de aplicaciones, desde problemas como la detección de objetos hasta la segmentación de imágenes.

### 7.1. La corteza visual y el neocognitrón

La comprensión del funcionamiento de la corteza visual en los seres humanos y otros animales ha sido una fuente de inspiración significativa para el desarrollo de algoritmos en el campo del aprendizaje profundo, especialmente en el diseño de CNNs. La corteza visual, ubicada en el lóbulo occipital del cerebro, es fundamental para el procesamiento de información visual. Estudios realizados por Hubel y Wiesel en la década de 1960 demostraron que ciertas neuronas en la corteza visual responden preferentemente a bordes específicos y orientaciones espaciales dentro de una región visual limitada [51]. Estas neuronas, conocidas como **células de orientación selectiva**, muestran una organización jerárquica que permite la percepción compleja a partir de la combinación de respuestas simples.

Inspirado en estas observaciones, Kunihiko Fukushima desarrolló el **neocognitrón** en 1980, una red neuronal que es considerada uno de los precursores de las modernas CNNs [39]. El neocognitrón fue diseñado para reconocer patrones visuales complejos de manera robusta frente a traslaciones y otras pequeñas distorsiones de la imagen. Esta red consta de múltiples capas que alternan entre capas convolucionales, que detectan características locales y capas que agregan las respuestas de los detectores de características sobre áreas locales (Figura 7.1). La estructura de esta red capta de manera efectiva la forma en que la corteza visual procesa la información visual, implementando una forma primitiva de invarianza a la traslación y la capacidad de extraer características jerárquicas.

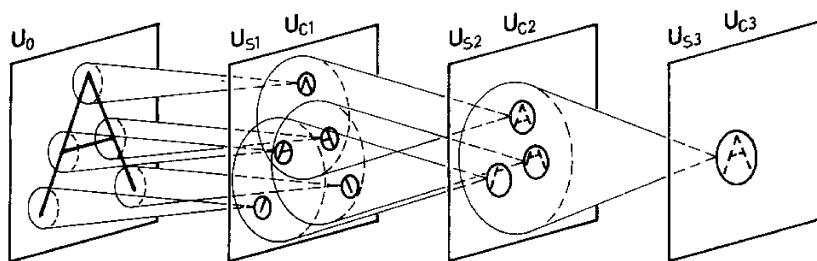


Figura 7.1.: Diagrama del neocognitrón mostrando el flujo de procesamiento de una imagen a través de varias capas. Las capas  $U_S$  (simple) y  $U_C$  (compleja) procesan la información de forma jerárquica, comenzando con la imagen original  $U_0$  y extrayendo características cada vez más complejas en  $U_{S1}, U_{C1}, U_{S2}, U_{C2}, U_{S3}$  y  $U_{C3}$ . Fuente [39].

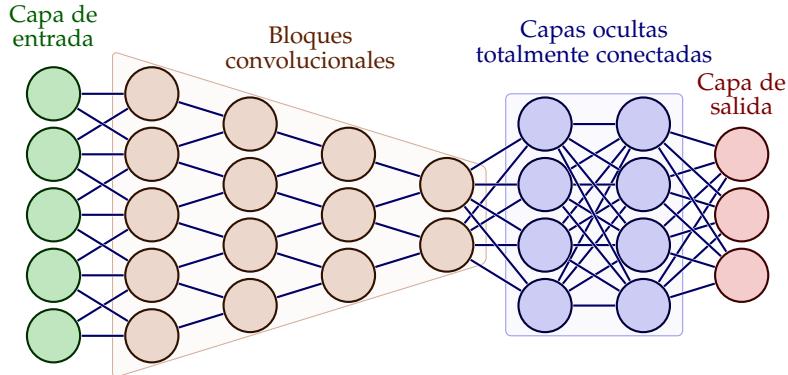


Figura 7.2.: Arquitectura de una CNN. Muestra la secuencia de capas en la red, iniciando con la capa de entrada, seguida por múltiples bloques convolucionales (resaltados en naranja), continuando con capas ocultas totalmente conectadas (resaltadas en azul) y finalmente la capa de salida.

La influencia de la corteza visual y el neocognitrón en el diseño de las CNNs es una muestra de la utilidad de estudios interdisciplinarios entre neurociencia y aprendizaje automático. Estos estudios no solo han facilitado avances tecnológicos en visión artificial sino que también han ofrecido nuevas perspectivas sobre cómo los seres humanos procesamos la información visual, proponiendo un puente entre la inteligencia artificial y la biológica [94].

## 7.2. Arquitectura de una CNN

Las CNNs, introducidas por Yann LeCun, supusieron un avance significativo respecto al neocognitrón. Mientras que el neocognitrón sentó las bases al proponer una arquitectura inspirada en el procesamiento visual del cerebro, LeCun implementó un método de entrenamiento supervisado utilizando retropropagación, lo que permitió mejorar notablemente el rendimiento y la capacidad de generalización de las redes neuronales [61]. Además, LeCun sentó las bases de la arquitectura de CNNs, optimizando la eficiencia y escalabilidad de los modelos, incorporando capas como las capas convolucionales y de muestreo [60].

Una CNN típicamente consiste en una secuencia de capas que transforman la entrada de imagen bruta en representaciones cada vez más abstractas y útiles para la tarea en cuestión. Cada tipo de capa dentro de una CNN tiene un propósito específico y contribuye de manera distinta al proceso de aprendizaje. Generalmente, las distintas capas de una CNN suelen agruparse para cumplir distintas funciones. Las principales agrupaciones que componen una CNN son:

- **Capa de entrada:** Es la primera capa de la red, donde se introduce la imagen original o preprocesada. Su función principal es preparar y escalar la imagen para las operaciones de las capas siguientes.
- **Bloques convolucionales:** Estos bloques contienen una o más capas convolucionales seguidas frecuentemente por capas de normalización y funciones de activación. Cada capa convolucional aplica diferentes filtros a la entrada para crear mapas de características que resalten aspectos específicos de la imagen. Estos bloques suelen ir consecutivos

intercalando capas de muestreo.

- **Capas totalmente conectadas:** Después de varias capas convolucionales y de muestreo, la información en forma de matrices y tensores se aplana en vectores y se pasa a través de capas FC. Estas capas integran la información aprendida por las capas anteriores para realizar la clasificación final.
- **Capa de salida:** La última capa de una CNN, donde se obtiene el resultado final. En tareas de clasificación, esta capa suele usar una función de activación como la Softmax para asignar probabilidades a las distintas clases posibles.

A continuación, vamos a explorar las diferentes capas que componen estos bloques y cómo trabajan juntas para detectar y aprender patrones importantes en los datos.

### 7.2.1. Capa convolucional

La **capa convolucional** es el bloque de construcción fundamental de una CNN. Utiliza un conjunto de filtros que se aplican a la entrada mediante el operador de **convolución discreta**. Cada filtro detecta características específicas en una región local de la entrada. El operador de convolución discreta se puede expresar como:

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n),$$

donde  $m, n$  son las coordenadas en la imagen o matriz de entrada  $I$  de la región a convolucionar,  $i, j$  son las coordenadas del centro del **kernel** o **filtro**  $K$  y  $S$  es la matriz de salida o **mapa de características**. La convolución discreta es un **operador lineal**, lo que implica que satisface propiedades como la **comutatividad**, **asociatividad** y **distributividad**.

Estos filtros se desplazan sobre toda la superficie de la entrada, generando un mapa de características que resume la presencia de las particularidades de dicha entrada.

0	1	1	1	0	0	0	0
0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0
0	0	0	1	1	0	0	0
0	0	1	1	0	0	0	0
0	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0

**I**

1	0	1
0	1	0
1	0	1

**K**

1	4	3	4	1
1	2	4	3	3
1	2	3	4	1
1	3	3	1	1
3	3	1	1	0

**I \* K**

Figura 7.3.: Ejemplo de convolución discreta. La matriz de entrada  $I$  se muestra con un segmento resaltado en naranja, mostrando la región afectada por el filtro  $K$ , en color verde. La matriz resultante  $I * K$  destaca los valores resultantes de la convolución, con el resultado de la convolución en dicha región en azul.

Además de las propiedades del operador de convolución, las capas convolucionales muestran varias propiedades más que las hacen especialmente adecuadas para tareas de procesamiento de imágenes y visión artificial. Entre estas propiedades destacan:

## 7. Redes neuronales convolucionales

- **Conectividad local:** Cada neurona en una capa convolucional está conectada solo a un pequeño número de neuronas cercanas en la capa anterior. Esta estructura imita la manera en que los campos receptivos en el sistema visual humano se organizan, concentrándose en pequeñas regiones del espacio visual. La conectividad local permite a la red detectar características locales de la entrada sin la influencia de la estructura global, reduciendo la complejidad y el número total de parámetros necesarios.
- **Compartición de parámetros:** En una CNN, el mismo filtro se utiliza para cada posición de la entrada, a diferencia de una FNN donde cada peso es único para cada conexión. Esta compartición de parámetros permite que la red sea más eficiente en términos de memoria y computación. Además, también implica que las características aprendidas por un filtro son útiles en toda la imagen, lo que mejora la eficiencia del aprendizaje y ayuda a la red a generalizar mejor.
- **Equivarianza frente a traslaciones:** Debido al uso de la misma función de convolución a lo largo de toda la entrada, las CNNs son naturalmente equivariantes a las traslaciones. Esto significa que si la entrada se traslada, las características detectadas por la red también se trasladarán de manera correspondiente. Esta propiedad es particularmente interesante en tareas de visión artificial, donde la relevancia de una característica no suele depender de su posición específica en el espacio de entrada.

Los hiperparámetros de **paso** o *stride*, y **relleno** o *padding* son especialmente relevantes en la manipulación dimensional durante la convolución en CNNs. El *stride* define el paso con el que el filtro se desplaza sobre la imagen o mapa de características, afectando la reducción dimensional del mapa resultante y permitiendo la captura de características a diversas escalas. Por otro lado, el *padding* consiste en añadir píxeles artificiales alrededor de la imagen de entrada, lo que permite que el filtro acceda completamente a los bordes y mantiene el tamaño del volumen de salida, preservando la información en los bordes cruciales para la interpretación completa de la imagen.

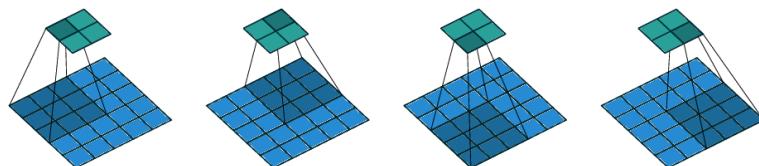


Figura 7.4.: Convolución de un filtro de  $3 \times 3$  sobre una entrada de  $5 \times 5$ . La operación se realiza utilizando *strides* de  $2 \times 2$  y sin agregar relleno (*padding = 0*), lo que resulta en una matriz de salida más pequeña y eficientemente espaciada. Fuente [32].

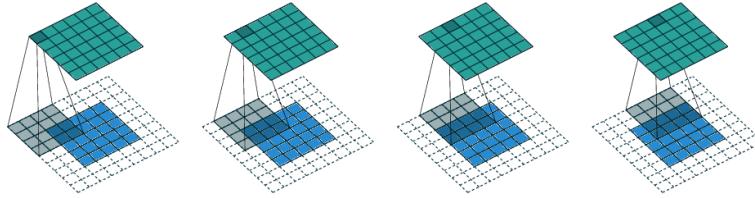


Figura 7.5.: Convolución de un filtro de  $3 \times 3$  sobre una entrada de  $5 \times 5$ . Se emplea un relleno de 2 unidades y pasos unitarios ( $stride = 1$ ), asegurando que la matriz de salida aumente las dimensiones de la matriz de entrada, al expandir el borde de la imagen original. Fuente [32].

La elección de dichos hiperparámetros influye directamente en las dimensiones del mapa de características de salida, cuya altura  $H_{\text{out}}$  y anchura  $W_{\text{out}}$  vienen dadas de la siguiente manera:

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2P - H_f}{S} + 1 \right\rfloor, \quad W_{\text{out}} = \left\lfloor \frac{W_{\text{in}} + 2P - W_f}{S} + 1 \right\rfloor,$$

donde  $H_{\text{in}}$  y  $W_{\text{in}}$  son las dimensiones de entrada,  $H_f$  y  $W_f$  son las del filtro,  $P$  es el *padding*,  $S$  es el *stride* y  $\lfloor \cdot \rfloor$  denota la función suelo. La profundidad del mapa de salida,  $D_{\text{out}}$ , está determinada por el número de filtros aplicados, donde  $D_{\text{out}} = \text{número de filtros}$ .

Estas convoluciones, aunque predominantemente asociadas con dimensiones bidimensionales, pueden extenderse a contextos tridimensionales (3D) donde se aplican filtros 3D sobre varios canales al mismo tiempo. Por ejemplo, un filtro de tamaño  $1 \times 1 \times 1$  puede transformar linealmente los mapas de características en cada ubicación del volumen de entrada. Las dimensiones de salida se mantienen como:

$$H_{\text{out}} = H_{\text{in}}, \quad W_{\text{out}} = W_{\text{in}}, \quad D_{\text{out}} = \text{número de filtros}.$$

Nótese que estas convoluciones son importantes para ajustar la dimensionalidad de los canales dentro de redes profundas, pues con una sola **convolución**  $1 \times 1$  podemos colapsar todos los canales de entrada en uno solo. Esto supone una reducción efectiva de parámetros y de la complejidad computacional en datos tridimensionales.

Otro tipo de convoluciones que surge en este contexto son las **convoluciones en profundidad** o *depthwise convolutions*. En ellas, cada filtro se aplica de manera independiente a cada canal del volumen de entrada, permitiendo un procesamiento separado de las dimensiones espaciales y de profundidad. La fórmula para las dimensiones de salida se mantiene, pero esta vez manteniendo el número de canales:

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2P - H_f}{S} + 1 \right\rfloor, \quad W_{\text{out}} = \left\lfloor \frac{W_{\text{in}} + 2P - W_f}{S} + 1 \right\rfloor, \quad D_{\text{out}} = D_{\text{in}}.$$

Estas convoluciones son eficaces para optimizar el rendimiento computacional en aplicaciones donde el manejo eficiente de los recursos es esencial.

## 7.2.2. Capa de muestreo

Las capas de **muestreo**, conocidas generalmente como capas de *pooling*, buscan reducir la dimensionalidad espacial de los mapas de características, lo que permite disminuir la cantidad de parámetros y de cómputo en la red al mismo tiempo que se mantiene la información

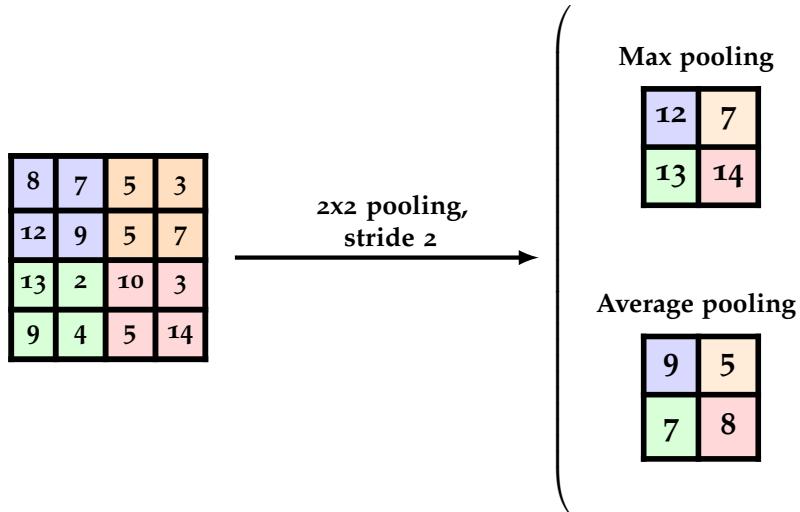


Figura 7.6.: Comparación de técnicas de *pooling*. Arriba, *max pooling* y abajo, *average pooling*, ambas con un filtro de 2x2 y *stride* de 2, mostrando la transformación de la matriz original.

más relevante. Las capas de *pooling* más comunes son las de **máximo** o *max pooling*, y las de **promedio** o *average pooling*. El *max pooling* toma la mayor activación en la ventana del filtro como:

$$P(i, j) = \max_{m, n \in W} I(i + m, j + n),$$

donde  $W$  es la ventana del filtro de *pooling*. Esta operación ayuda a hacer la representación obtenida invariante a pequeñas traslaciones y distorsiones. Por otro lado, el *average pooling* calcula el promedio de las activaciones dentro de la ventana del filtro, proporcionando una representación que suaviza las características de entrada:

$$P(i, j) = \frac{1}{|W|} \sum_{a, b \in W} I(i + a, j + b),$$

donde  $|W|$  es el número de elementos en la ventana del filtro. Ambas técnicas de *pooling* tienen sus aplicaciones específicas dependiendo de la naturaleza del problema y de la arquitectura de la red. Mientras que el *max pooling* es generalmente preferido para tareas relacionadas con la detección de características debido a su capacidad para preservar las activaciones más fuertes, el *average pooling* puede ser más adecuado para tareas donde la uniformidad de las características es más importante [59].

### 7.3. Modelos y estado del arte en CNNs

Desde su origen, las arquitecturas de CNNs han experimentado un desarrollo significativo, marcado por una serie de innovaciones que han mejorado su rendimiento y eficiencia. Como vimos anteriormente, la historia de las CNNs comenzó con el neocognitrón de Fukushima en los años 80 [39], un modelo novedoso que introdujo el concepto de capas convolucionales

y de *pooling*. Posteriormente, la introducción de LeNet-5 por LeCun et al. en los años 90 [61] demostró la eficacia de las CNNs en tareas de reconocimiento de dígitos y documentos, mostrando que este tipo de arquitecturas podían aprender a resolver problemas de aprendizaje supervisado. Posteriormente AlexNet, desarrollada por Krizhevsky et al. en 2012 [59], revolucionó el campo del aprendizaje profundo, utilizando técnicas como paralelización del entrenamiento, profundización de la red, ReLU y *dropout* para ganar el desafío de ImageNet con una reducción drástica en la tasa de error. A partir de ahí, surgieron modelos más sofisticados como VGG [96], GoogLeNet con su módulo Inception [102], o ResNet, que introdujo las conexiones residuales permitiendo entrenar redes mucho más profundas [47]. Cada una de estas arquitecturas ha contribuido a comprender mejor cómo diseñar redes eficientes para procesar y aprender de imágenes a gran escala. En la actualidad, modelos más recientes como DenseNet, que conecta cada capa directamente con todas las anteriores [50], o EfficientNet, que escala de manera uniforme todas las dimensiones de la red [104], han logrado mejoras notables tanto en la eficiencia como en la capacidad de clasificación de las CNNs. Desde entonces, diferentes modelos de redes convolucionales como EfficientNetV2 [105], mejoras en el proceso de entrenamiento [79] o el uso de novedosas arquitecturas como *Transformer* [111, 29] continúan empujando los límites de lo que las CNNs pueden lograr, mejorando su capacidad de clasificación y optimizando su rendimiento y eficiencia para emplearlos en aplicaciones en tiempo real y en dispositivos con recursos limitados.

### 7.3.1. ResNet

Las **Redes Neuronales Residuales** (ResNet) introducidas por He et al. en 2015 [47], supusieron un avance significativo en la arquitectura de redes profundas para el reconocimiento de imágenes. A diferencia de sus predecesores como AlexNet y VGG, ResNet aborda el problema del desvanecimiento del gradiente que suele presentarse en redes muy profundas mediante la introducción de una conexión de identidad que salta una o más capas.

La clave de la arquitectura de ResNet es el **bloque residual**, que incorpora una **conexión residual** directamente conectando la entrada del bloque a su salida, lo que permite que la señal se propague directamente a través de la red. El bloque residual se puede expresar como:

$$H(\mathbf{x}_l) = \mathcal{F}(\mathbf{x}_l, \{W_i\}) + \mathbf{x}_l$$

donde  $\mathbf{x}_l$  y  $H(\mathbf{x}_l)$  son la entrada y la salida del  $l$ -ésimo bloque residual respectivamente,  $\mathcal{F}$  representa las capas intermedias de la red y  $\{W_i\}$  denota el conjunto de pesos de estas capas.

A diferencia de AlexNet, que tiene 8 capas, y VGG, que tiene 16 o 19 capas, ResNet se diseñó con capacidades mucho más profundas, con versiones que van desde 18 hasta 152 capas. Lo revolucionario de ResNet no es simplemente añadir más capas, sino su habilidad para entrenar redes muy profundas sin sufrir desvanecimiento del gradiente gracias a sus conexiones residuales. Estas conexiones ayudan a preservar el gradiente a lo largo del proceso de aprendizaje, lo que permite entrenar redes más profundas que las posibles anteriormente.

El diseño del bloque residual permite que ResNet no solo evite el problema del desvanecimiento del gradiente sino que también mejore la eficiencia del entrenamiento. Los experimentos demuestran que las redes con bloques residuales superan a las arquitecturas convencionales en grandes conjuntos de datos de imágenes como ImageNet.

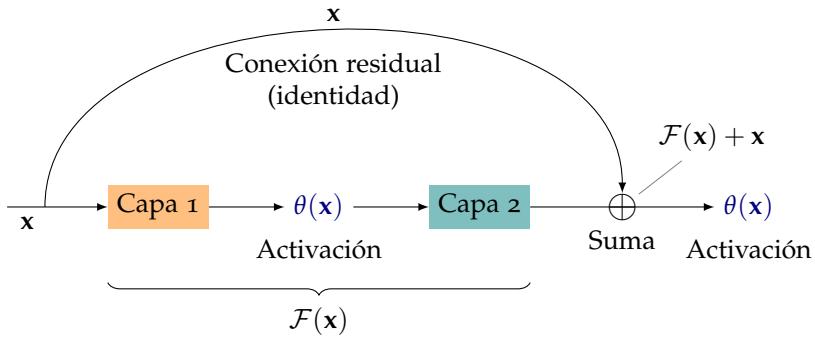


Figura 7.7.: Diagrama de una conexión residual. La entrada  $x$  pasa a través de la Capa 1 y una función de activación, fluyendo luego hacia la Capa 2. Paralelamente,  $x$  se suma directamente al final de la Capa 2 mediante una conexión residual.

### 7.3.2. DenseNet

Las **Redes Convolucionales Densamente Conectadas** (DenseNet), propuestas por Huang et al. en 2017 [50], representan otra evolución significativa en el diseño de redes neuronales profundas. DenseNet mejora la idea de conexiones de salto de ResNet mediante la integración de cada capa directamente con todas las capas posteriores de una manera densamente conectada.

La principal innovación de DenseNet es su estructura de **conexiones densas**, donde cada capa recibe como entrada todas las salidas de las capas anteriores, concatenando estas salidas. Esto se formula como sigue:

$$\mathbf{x}_l = H_l([\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_{l-1}])$$

donde  $\mathbf{x}_l$  es la salida de la capa  $l$ ,  $[\cdot]$  denota la operación de concatenación y  $H_l$  es una función que representa las operaciones dentro de la capa  $l$ , normalmente compuesta por operaciones de *Batch Normalization*, activación ReLU, y convolución.

Aunque tanto ResNet como DenseNet utilizan conexiones que saltan capas, DenseNet ofrece una mejora en la eficiencia y la efectividad del entrenamiento al promover la reutilización de características de mayor manera. Mientras que las conexiones de salto en ResNet suman entradas, en DenseNet la información fluye a través de la red mediante la concatenación de características, lo que resulta en una mejora del flujo de información y gradientes a través de la red, reduciendo así el problema del desvanecimiento del gradiente de manera más efectiva.

DenseNet ha demostrado ser especialmente eficaz en conjuntos de datos como ImageNet y en aplicaciones donde la conservación de la información a lo largo de la red es crítica. Además, DenseNet tiende a ser más eficiente en términos de parámetros que ResNet debido a su capacidad para reutilizar características, lo que permite la construcción de redes profundas que son tanto compactas como potentes.

### 7.3.3. EfficientNet

**EfficientNet**, introducido por Tan y Le en 2019 [104], es un ejemplo de cómo se puede mejorar la eficiencia y la efectividad de las redes neuronales mediante una cuidadosa optimización

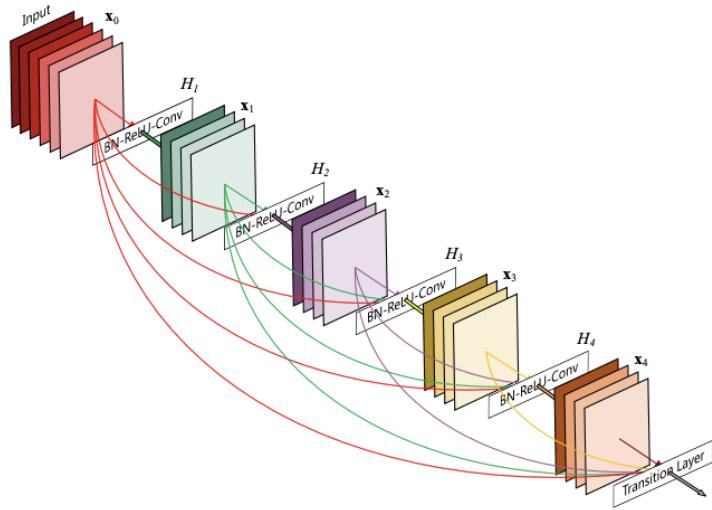


Figura 7.8.: Arquitectura de DenseNet. La figura ilustra la configuración de la red DenseNet, mostrando cómo las capas están conectadas entre sí, donde cada capa recibe como entrada todas las salidas de las capas anteriores. Fuente [50].

de sus dimensiones. Lo innovador de EfficientNet es su metodología de escalado compuesto, que escala uniformemente todas las dimensiones de la red (profundidad, ancho y resolución de la imagen) con un conjunto fijo de coeficientes de escalado. La fórmula de escalado se representa como:

$$\text{profundidad : } d = \alpha^\phi, \quad \text{anchura : } w = \beta^\phi, \quad \text{resolución : } r = \gamma^\phi,$$

tal que

$$\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2, \quad \alpha \geq 1, \quad \beta \geq 1, \quad \gamma \geq 1,$$

donde  $\phi$  es un coeficiente que determina la cantidad de recursos disponibles para el escalado, y  $\alpha, \beta, \gamma$  son constantes que definen cómo deben escalar la profundidad, el ancho y la resolución, respectivamente, para lograr un equilibrio óptimo entre precisión y eficiencia.

**EfficientNet-Bo** es el modelo base en la familia de modelos EfficientNet, que se caracteriza por aplicar un escalado uniforme a una arquitectura optimizada mediante **búsqueda de arquitectura neuronal** (NAS). La arquitectura inicia con una capa convolucional que maneja la imagen de entrada, aplicando la función de activación Swish y normalización por lotes para preparar las características para las etapas siguientes. A continuación, EfficientNet-Bo implementa una serie de **bloques MBConv**. Cada bloque MBConv sigue un patrón específico:

1. **Expansión:** Primero, una convolución  $1 \times 1$  incrementa el número de canales. Esta técnica de expansión prepara los canales para una manipulación más intensiva, facilitando la extracción de características.
2. **Convolución en profundidad:** A continuación, se aplica una convolución en profundidad con un filtro  $3 \times 3$  o  $5 \times 5$ . Esta técnica permite la extracción de características locales de manera eficiente, sin un gran incremento en el número de parámetros como

## 7. Redes neuronales convolucionales

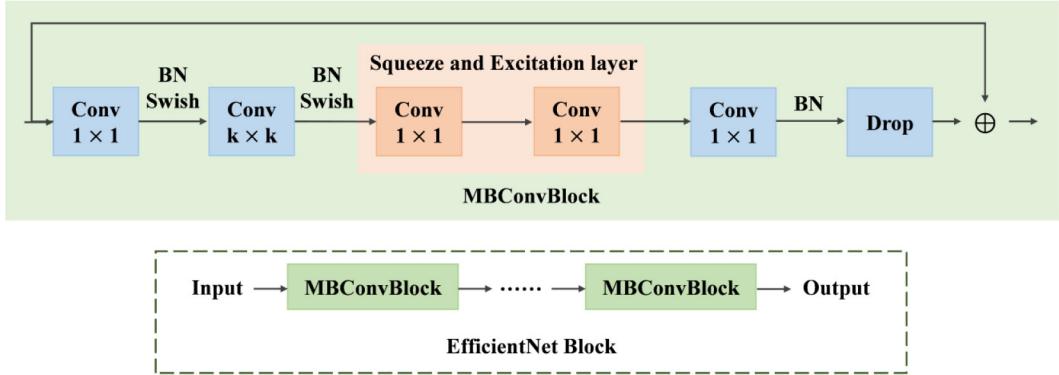


Figura 7.9.: Estructura de un bloque MBConv, la unidad básica en la arquitectura de EfficientNet. Incluye capas convolucionales con activaciones Swish y normalización por lotes, intercaladas con capas SE para un refinamiento de las características. La secuencia de procesamiento termina en una capa de *dropout* antes de la salida final. Fuente [106].

se observaría con convoluciones tradicionales.

3. **Capa Squeeze and Excitation (SE):** Esta capa sigue a la convolución en profundidad. Funciona primero reduciendo espacialmente cada canal a un valor escalar (*squeeze*), que luego se utiliza para recalibrar los canales (*excitation*) mediante operaciones de reescalado. Este proceso permite al modelo aprender a enfocar dinámicamente su atención en características informativas y suprimir las menos útiles.
4. **Compresión:** Después de la capa SE, una convolución  $1 \times 1$  reduce el número de canales, consolidando las características importantes, lo cual asegura la eficiencia de la red y prepara la salida para la siguiente etapa del procesamiento.

Las capas de reducción, situadas entre grupos de bloques MBConv, emplean convoluciones con un paso de 2 para reducir las dimensiones espaciales, aumentando el nivel de abstracción y reduciendo la carga computacional en las capas más profundas.

Finalmente, conexiones residuales se incorporan en cada bloque MBConv para facilitar el entrenamiento de la red al prevenir el desvanecimiento del gradiente. La red termina con una capa de muestreo promedio global que transforma la salida de los bloques MBConv en un vector único por imagen, el cual es procesado por una capa FC para realizar la clasificación final.

En la actualidad, modelos como ResNet, DenseNet y EfficientNet han establecido nuevos estándares de precisión en *benchmarks* como **ImageNet**, donde las tasas de error se han reducido de manera significativa en comparación con los métodos tradicionales basados en características manuales. Estas arquitecturas avanzadas han demostrado no solo una gran capacidad de generalización sobre grandes conjuntos de datos, sino también robustez frente a variaciones y perturbaciones en las imágenes. Además, la integración de técnicas como el aumento de datos y la **transferencia de aprendizaje** o *transfer learning*, han permitido aplicar modelos entrenados en un dominio específico a nuevos conjuntos de datos con poco o ningún ajuste adicional. Este progreso ha transformado la clasificación de imágenes de ser un desafío técnico a una herramienta útil, con aplicaciones en múltiples industrias, desde

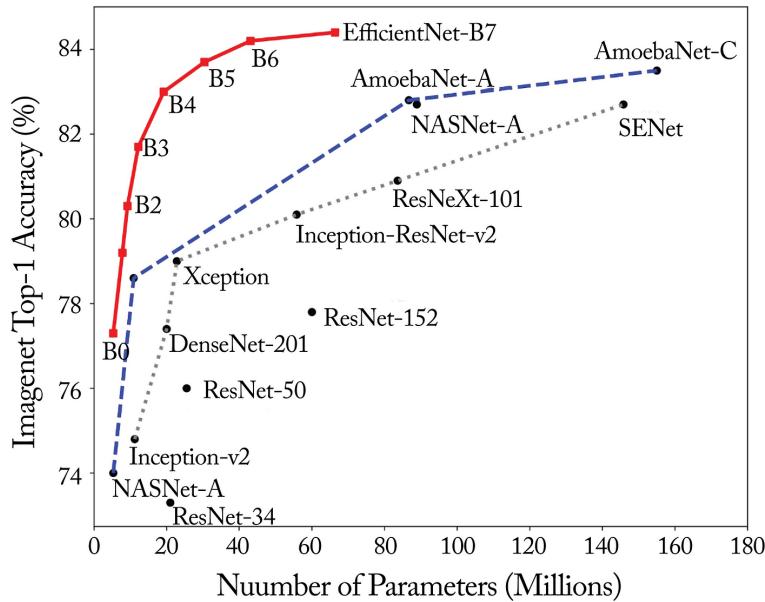


Figura 7.10.: Comparación de tamaño de modelo y precisión en ImageNet. EfficientNet supera notablemente al resto de modelos de CNNs. En general, EfficientNet muestra una mejora destacable en eficiencia y rendimiento comparado con modelos previos como ResNet-152. Fuente [104].

el reconocimiento automático de contenido en redes sociales hasta sistemas avanzados de asistencia al conductor en vehículos autónomos.



## **Parte III.**

### **Análisis de datos topológico para el estudio de CNNs**



## 8. Análisis de Datos Topológico

En este capítulo, nos centraremos en conocer el análisis de datos topológico y en cómo se aplica para el estudio de las CNNs. Con el fin de tratar de dar explicación a cómo transforman los datos las CNNs, proponemos un **término de regularización topológico**, de forma que trataremos de mejorar las capacidades del modelo penalizando el entrenamiento en base a su complejidad topológica. Más concretamente, estudiaremos cómo afecta este término tanto en una transferencia de conocimiento de la red al completo como en un proceso de *fine-tuning* donde tan sólo entrenaremos la cabeza clasificatoria.

### 8.1. Análisis de datos topológico

El **análisis de datos** es una disciplina en la ciencia de datos que emplea métodos estadísticos, aprendizaje automático y sistemas de procesamiento para transformar grandes cantidades de datos en información útil para tomar decisiones en diversos campos [54]. Con el desarrollo de esta área, se han integrado técnicas avanzadas como la **topología algebraica**, que explora propiedades que se mantienen constantes a pesar de las transformaciones físicas de los objetos, enfocándose en aspectos como la continuidad y la conectividad más allá de la forma exacta.

Recordemos que dentro de este contexto, la **homología** ([Capítulo 3](#)) destaca como una herramienta de la topología algebraica diseñada para detectar y analizar características como componentes conexas, agujeros y vacíos en múltiples dimensiones de un espacio. Por otro lado tenemos la **hipótesis de la variedad** [35], la cual postula que los datos de alta dimensionalidad suelen residir en variedades de baja dimensión embebidos en ese espacio de mayor dimensión. Este concepto es importante en el contexto de las CNNs, pues defiende que estas arquitecturas transforman las variedades subyacentes de los datos a clasificar hasta hacerlos linealmente separables [20]. Los conjuntos de datos, que a menudo se presentan como nubes de puntos, representan muestras discretas que podrían sugerir estructuras subyacentes similares a **variedades topológicas** [68]. Para explorar estas estructuras sin introducir sesgos en cómo se conectan estos puntos, se considera una familia de complejos simpliciales ([Capítulo 2](#)) en función de la distancia entre puntos denominados **filtraciones**, dando lugar al cálculo de su **homología persistente** ([Capítulo 4](#)). Esta herramienta no solo identifica características topológicas sino también evalúa su persistencia a lo largo de diversas escalas, lo que ayuda a distinguir entre el ruido y las estructuras relevantes.

El **análisis de datos topológico** (TDA) aprovecha estas técnicas en una metodología que se centra en desentrañar la estructura subyacente de los datos, basándose en su «forma» [11]. Utilizando la homología persistente como su herramienta principal, el TDA ofrece una forma robusta y detallada de identificar características topológicas de los datos, permitiendo revelar patrones y relaciones no evidentes mediante métodos tradicionales. Esta metodología se ha aplicado con éxito en varios campos, como la neurociencia para analizar la conectividad cerebral, en genómica para explorar la interacción entre genes y en ciencia de materiales para estudiar la estructura microscópica de los materiales [15].

## 8. Análisis de Datos Topológico

El proceso de TDA se organiza en tres fases que facilitan el estudio de la homología persistente de los datos ([Figura 8.1](#)):

1. **Preparación de datos:** Inicialmente, se asume que los datos están dispuestos en un espacio métrico, típicamente el espacio euclídeo  $\mathbb{R}^N$ , utilizando normalmente la distancia euclídea. La elección de la distancia es un paso importante, pues determinará la manera en la que se relacionan los puntos dentro del conjunto de datos.
2. **Construcción de filtraciones:** Posteriormente, se construye una filtración de complejos simpliciales de Vietoris-Rips a partir de un intervalo de radios dado. Generalmente, se trabaja con una muestra finita de este intervalo.
3. **Extracción de características topológicas:** Finalmente, se emplea la homología persistente para analizar las filtraciones construidas, obteniendo descriptores topológicos que revelan la presencia de características como agujeros y conexiones en diversas dimensiones. Estos descriptores proporcionan una visión detallada de las propiedades topológicas y geométricas de los datos, ayudando a distinguir entre rasgos estructurales significativos y el ruido.

## 8.2. TDA en el estudio de CNNs

Entre las muchas aplicaciones del TDA, nosotros nos centraremos en su uso para el estudio de las CNNs. Actualmente, existen 3 aproximaciones principales para analizar estas redes neuronales desde el punto de vista de la topología en función del objeto de estudio:

- **TDA para el preprocesamiento de imágenes:** Se obtienen diversos descriptores topológicos a partir de la imagen a analizar y se estudia cómo van cambiando a lo largo de la red. En consecuencia, se proponen transformaciones topológicas a las imágenes con el fin de facilitar el aprendizaje o la clasificación [[40](#)].
- **TDA para el análisis de la arquitectura de CNNs:** Se obtiene el grafo computacional que define las operaciones de la red y se analiza su homología persistente en función de los pesos [[18](#)].
- **TDA para el estudio de CNNs en función del conjunto de datos:** En esta aproximación, se estudia cómo evoluciona la homología persistente de un conjunto de datos a través de las distintas capas de la red. Se aplica para comprender cómo las CNNs y sus distintas componentes transforman los datos [[72](#)].

En este trabajo, nos centramos en analizar cómo las CNNs modifican los conjuntos de datos y cómo la «forma» de los datos influye en su capacidad de transferencia y clasificación. Naitzat et al. en su artículo *Topology of Deep Neural Networks* [[72](#)] exploraron cómo distintas implementaciones básicas de CNNs, con una tasa de error de generalización inferior al 0.01 %, alteran la topología de los datos. En su estudio, realizado sobre conjuntos de datos sintéticos y reales, concluyeron que estas redes simplifican progresivamente la complejidad topológica de los datos y que esta simplificación varía según la arquitectura de la red. Extendiendo este trabajo, German Magai en [[68](#)] comparó la homología persistente de varios modelos de aprendizaje profundo en tareas de clasificación de imágenes utilizando diferentes conjuntos de datos. Magai confirmó las conclusiones de Naitzat y además señaló que la homología

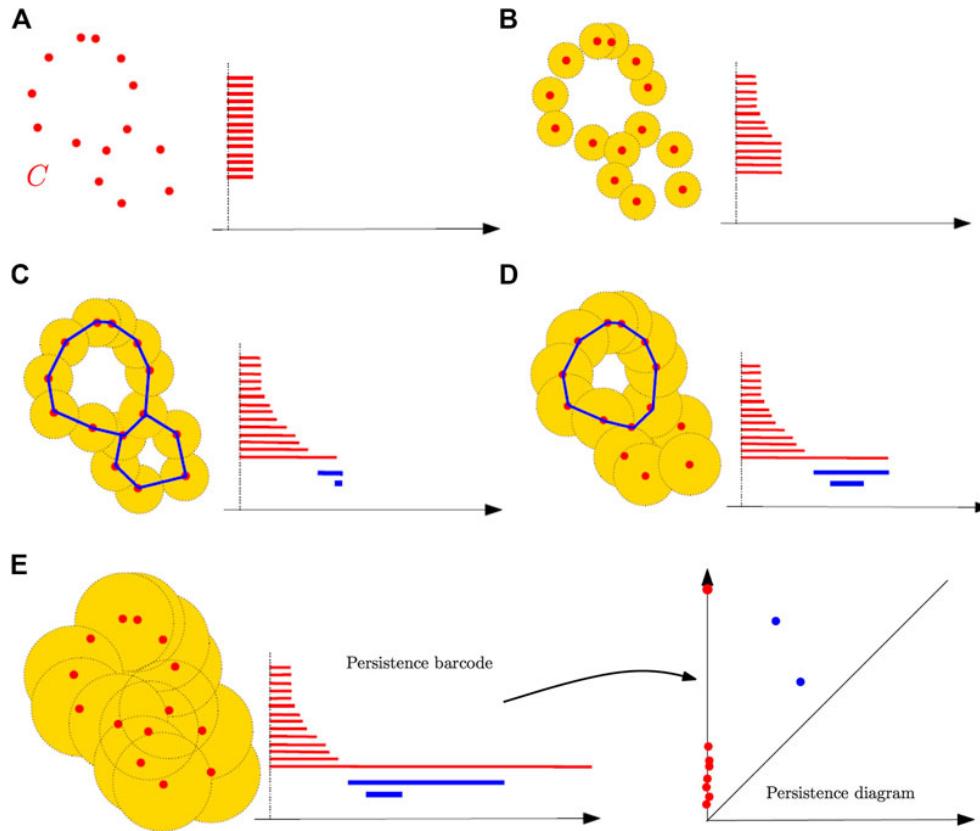


Figura 8.1.: Ejemplo de flujo de trabajo en TDA utilizando complejos de Vietoris-Rips. (A) Conjunto inicial de datos. (B,C,D,E) Desarrollo del complejo de Vietoris-Rips a distintos niveles de la filtración y el respectivo código de barras hasta dicho instante. Finalmente se muestra el diagrama de persistencia resultante, mostrando la homología de dimensión 0 (rojo) y de dimensión 1 (azul). Fuente [14].

persistente podría servir como un indicador del error de generalización sin necesidad de conjuntos de prueba. Adicionalmente, investigaciones como la de Waibel et al. en [113] sugieren que la implementación de **funciones de pérdida topológicas** puede mejorar la capacidad de generalización de los modelos de aprendizaje profundo dadas sus propiedades diferenciables [64].

### 8.3. Descriptores topológicos en TDA

Existen diversas métodos para analizar cómo de compleja es la topología de un conjunto de datos. En el marco del TDA, el descriptor topológico más sencillo que se puede obtener es la **persistencia total**. Dado un código de barras  $\mathcal{B}$ , se conoce como persistencia total a la suma de la longitud de los intervalos que forman el código de barras. Esto es,

$$TP = \sum_{(b_i, d_i) \in \mathcal{B}} d_i - b_i,$$

## 8. Análisis de Datos Topológico

donde  $d_i \in \mathbb{R}$  denota el instante de muerte de la clase de homología y  $b_i \in \mathbb{R}$  el nacimiento de ésta. Pese a su sencillez, la persistencia total permite estudiar de forma absoluta la homología persistente de un conjunto de datos en su totalidad.

Otro descriptor comúnmente empleado en el contexto del TDA es la **entropía persistente** [87]. Como su nombre indica, este descriptor calcula la entropía del código de barras respecto a la persistencia total:

$$PE = - \sum_{(b_i, d_i) \in \mathcal{B}} \frac{d_i - b_i}{TP} \log \left( \frac{d_i - b_i}{TP} \right).$$

Más recientemente están empezando a emplearse métodos más sofisticados para obtener descriptores más informativos. Algunos de ellos buscan lidiar con las clases de homología con una muerte rápida, pues normalmente son consideradas como «ruido». Por ello, Benjamin Schweinhart [92] presentó la **dimensión fractal homológica persistente**, que se define como sigue. Sea  $X$  una nube de puntos perteneciente a un espacio métrico y consideremos  $\mathcal{B}_n$  como sus códigos de barras de dimensión  $n$ . Además, consideremos la suma de potencias de periodos de vida

$$E_\alpha^n(X) = \sum_{(d_i, b_i) \in \mathcal{B}_n} (d_i - b_i)^\alpha,$$

donde  $\alpha \geq 0$  es un valor real y  $n \in \mathbb{N}$ . Dada una medida  $\mu$  definida en  $X$ , para cada dimensión  $n$  definimos la dimensión fractal homológica persistente  $PH\dim_\alpha^n$  de la medida  $\mu$  como:

$$PH\dim_\alpha^n(\mu) = \frac{\alpha}{1 - \beta},$$

donde

$$\beta = \lim_{m \rightarrow \infty} \sup \frac{\log (\mathbb{E}[E_\alpha^n(x_1, \dots, x_m)])}{\log(m)},$$

siendo  $x_1, \dots, x_m$  muestras i.i.d. de  $X$  a partir de la medida  $\mu$  y  $\mathbb{E}$  denota la esperanza de dicha variable aleatoria. Básicamente, este descriptor da más peso a los periodo de vida largos en vez de a los pequeños, permitiendo que nos centremos en las características topológicas más relevantes de nuestro conjunto de datos.

### 8.4. Propuesta: regularización con persistencia total normalizada

Sin duda, la «forma» y estructuras que generan los datos en su paso por la red son características que pueden llegar a ser de gran interés para mejorar diversos aspectos de un modelo. Con dicha idea en mente, se propone un término de regularización topológico basado en la persistencia total con el fin de mejorar la capacidad de clasificación o generalización de la red en función de nuestro interés.

La maximización o minimización de la persistencia total nos permite aumentar o disminuir la complejidad topológica de nuestros datos respectivamente. Un aumento en la complejidad topológica implica unos datos más dispersos con estructuras más complejas, mientras que una minimización de dicha complejidad tiende a agrupar los datos en estructuras más simples. En el ámbito de las CNNs, el estudio de la persistencia total a distintos niveles de la

#### 8.4. Propuesta: regularización con persistencia total normalizada

red es útil para ver en términos absolutos cómo afectan los cambios en la red a la homología persistente de los datos **entre las distintas capas**.

En base a lo recién comentado, proponemos el uso de un nuevo descriptor que denominaremos **persistencia total normalizada**. A diferencia de la persistencia total, la persistencia total normalizada calcula el cociente de cada uno de los intervalos del código de barras respecto a aquel con un mayor ciclo de vida. En caso de que un ciclo de vida fuese infinito, se truncaría en el último instante de muerte de una clase de homología persistente previo a él. Este descriptor es una métrica relativa, de utilidad para estudiar la tendencia **intracapa** de la homología persistente que sufren los datos a lo largo de la red.

En consecuencia, para un código de barras  $\mathcal{B}$  podemos definir un término de regularización  $\Omega(\mathcal{B})$  que minimice la persistencia de forma que

$$\Omega(\mathcal{B}) = \frac{TP}{L} = \frac{1}{L} \cdot \sum_{(b_i, d_i) \in \mathcal{B}} d_i - b_i,$$

donde  $L = \max_{(b_i, d_i) \in \mathcal{B}} \{d_i - b_i\}$  es el intervalo de mayor longitud. Nótese que dicho intervalo siempre va a representar el ciclo de vida de una clase de homología de dimensión 0, pues nacen en el primer instante y solamente mueren si la cadena que representa se ha fusionado con otra componente conexa. Luego siempre existe una componente conexa que vive durante toda la filtración. Además, al aplicar el cociente sobre la longitud del mayor intervalo, logramos relativizar la complejidad respecto al tamaño de la filtración, lo que nos proporciona un mayor control.

Por ejemplo, en el caso  $\Omega(\mathcal{B}) = 1$  esto implica que tan solo existe una componente conexa. El término aplicaría de forma que

$$J_\Omega(\mathbf{w}) = J(\mathbf{w}) + \alpha \Omega(\mathcal{B})$$

siendo  $J$  la función objetivo,  $\mathbf{w}$  los pesos de la red y  $\alpha \in [-1, 1]$ . En el caso donde  $\alpha > 0$ , estaríamos disminuyendo la complejidad topológica mientras que un valor  $\alpha < 0$  implicaría un aumento en la complejidad de la topología.

Nuestro objetivo en esta última fase será aplicar el término de regularización con dos fines distintos. Primero, trataremos de mejorar la capacidad de clasificación del modelo disminuyendo la complejidad topológica de los datos en la cabeza clasificatoria. Por último, trataremos de utilizar dicho término para mejorar la capacidad de transferencia de los modelos. Exploraremos dos vías: buscaremos mejorar la capacidad de transferencia de un modelo más sencillo (con menos clases) a uno más general (con más clases) aumentando su persistencia total normalizada y viceversa disminuyéndola.



## 9. Marco experimental y metodología

Como se ha podido observar en el [Capítulo 8](#), el TDA proporciona una serie de mecanismos que nos permiten estudiar con bastante fidelidad la topología de los datos en CNNs. Además, hemos visto que dichas redes tienden a simplificar la forma de los datos durante su paso a través de las distintas capas y lo hacen de manera diferente en función de la arquitectura propuesta. En este capítulo se describe la metodología así como las herramientas empleadas para analizar distintas CNNs en profundidad.

### 9.1. Experimentos

Los experimentos se han dividido en tres etapas con el fin de comprender cómo las CNNs transforman la topología de los datos. Para ello, se han entrenado una serie de modelos pertenecientes a tres familias de arquitecturas diferentes de CNNs: ResNet-18, DenseNet-121 y EfficientNet-Bo. El principal motivo para seleccionar estos representantes y no alternativas más complejas de la misma familia se debe a que han mostrado una buena capacidad de generalización y eficiencia.

La **primera etapa** ha consistido en la comparación de la complejidad topológica de los datos a lo largo de la red en función de la arquitectura, el tamaño de lote y el optimizador escogido.

En la **segunda etapa**, se ha escogido el mejor modelo para cada una de las dos granularidades del conjunto de datos tratado y se ha comparado la topología de los datos con los mismos modelos entrenados con aumento de datos, la granularidad del conjunto de datos y la partición de datos.

En la **última etapa**, se han realizado dos experimentos en base a las observaciones realizadas en las anteriores etapas. Como veremos en la [Sección 10.2](#), nuestra hipótesis en base a los estudios realizados apuntan a que una complejidad topológica relativa menor en las etapas finales del modelos podrían mejorar su rendimiento en clasificación, mientras que una complejidad mayor podría facilitar la transferencia de conocimiento a conjuntos de datos más complejos. Por ello, se propone tratar de mejorar la clasificación del modelo mediante el uso del regularizador topológico descrito en la [Sección 8.4](#).

### 9.2. Entorno de experimentación

Para la realización de este proyecto se seleccionó Python como lenguaje de programación principal, dada su amplia disponibilidad de herramientas de análisis de datos [\[110\]](#). En particular, se utilizó la versión 3.10.14 por su compatibilidad con una variedad de bibliotecas importantes para la investigación realizada. Para la gestión del entorno de desarrollo se empleó Anaconda [\[1\]](#), facilitando así la configuración reproducible de los entornos de trabajo. El código implementado puede consultarse en el repositorio de GitHub [tda-nn-analysis<sup>1</sup>](#).

<sup>1</sup><https://github.com/pab1s/tda-nn-analysis>

## 9. Marco experimental y metodología

Los modelos de aprendizaje profundo se implementaron utilizando PyTorch 1.13.1 [77], una biblioteca de cálculo tensorial que soporta la aceleración por GPU. PyTorch es conocido por su extensiva colección de herramientas que simplifican el desarrollo de modelos de aprendizaje profundo, siendo ampliamente reconocido y utilizado en la investigación en inteligencia artificial.

El análisis de homología persistente se llevó a cabo mediante la biblioteca giotto-tda 0.6.0 [107], que se apoya en ripser [109] por su eficiencia en la ejecución y el uso de memoria en comparación con otras opciones disponibles [75]. Además, esta herramienta permite el procesamiento multihilo, lo que es un beneficio adicional importante. Para completar los experimentos, se empleó el paquete torch\_topological<sup>2</sup> 0.1.7, que implementa el cálculo de la homología persistente de la filtración de Vietoris-Rips de forma que sea compatible con la retropropagación en PyTorch.

Para la visualización de datos, se optó principalmente por matplotlib 3.8.4 [52] y seaborn 0.13.2 [117], herramientas que ofrecen funcionalidades avanzadas para la creación de gráficos y visualizaciones complejas. También se usó torch-cam 0.4.0 [36] para la visualización de las activaciones en la imagen. Adicionalmente, se integraron bibliotecas auxiliares como pandas 2.2.2 [69] y numpy 1.26.4 [43] para cálculos numéricos adicionales<sup>3</sup>.

Los experimentos se realizaron en el *switch* Dionisio de la partición Dios del clúster de servidores GPU ubicado en el edificio CPD Santa Lucía de la Universidad de Granada, perteneciente al Instituto Andaluz Interuniversitario en *Data Science and Computational Intelligence*<sup>4</sup> (DASCI). Está equipado con dos CPUs Intel Xeon Silver 4216 @ 2.10GHz, 512 Gb de RAM y dos GPUs Quadro RTX 8000 con 48Gb de VRAM.

### 9.3. Conjunto de datos

Durante el desarrollo de los experimentos se ha empleado una versión modificada del conjunto de datos *Vehicle Identification*<sup>5</sup> publicado por el DASCI. Este conjunto de datos está compuesto de imágenes frontales de coches en primer plano proveniente de diversas fuentes. El interés de este conjunto de datos en particular radica en la su disposición de **distintas granularidades**. Es decir, disponemos de distintas clasificaciones en función de lo «fina» que sea nuestra predicción sobre los detalles de los datos. Esta versión se divide en dos conjuntos en función de su especificidad:

- **Especificidad Marca:** consiste en 3232 imágenes etiquetadas en 34 clases que indican la marca del fabricante del vehículo.
- **Especificidad Marca-Modelo:** consiste en 2701 imágenes etiquetadas en 152 clases identificando la marca y el modelo concreto del vehículo.

El principal motivo por el que se ha seleccionado este conjunto de datos es por su disposición en las dos especificidades propuestas. Dado que las instancias son comunes en la mayoría de los casos, resultará de gran interés saber cómo clasificar conjuntos de datos prácticamente idénticos con distintas etiquetas puede afectar al proceso de generalización y a la topología de los datos.

<sup>2</sup><https://pytorch-topological.readthedocs.io/en/latest/>

<sup>3</sup>Una lista completa de paquetes y requisitos se encuentra disponible en el archivo `environment.yaml` en el repositorio del proyecto.

<sup>4</sup><https://dasci.es/>

<sup>5</sup><https://dasci.es/es/transfereencia/open-data/vehicleid-es/>

### 9.3. Conjunto de datos



Figura 9.1.: Ejemplos de instancias por clase en la especificidad de Marca.



Figura 9.2.: Ejemplos de instancias por clase en la especificidad de Marca-Modelo.

La [Figura 9.3](#) muestra un claro desbalance entre las distintas clases ambos conjuntos de datos. De hecho podemos observar que en ciertas ocasiones el número de muestras por clase es tan solo de una instancia, mientras que otras clases están claramente sobrerepresentadas.

Este problema es aún peor cuando nos fijamos en la especificidad Marca-Modelo, tal y como muestra la [Figura 9.4](#). Por este motivo, será interesante emplear métricas cuya sensibilidad respecto a clases desbalanceadas sea mayor.

Ambos *datasets* se dividirán en conjuntos de entrenamiento, validación y test con una distribución 80-10-10 respectivamente durante el proceso de experimentación. El conjunto de entrenamiento se empleará para ajustar los pesos de los modelos directamente de las instancias que lo componen. El conjunto de validación se empleará durante el proceso de entrenamiento para obtener una evaluación no sesgada de los modelos con el objetivo de evitar el sobreajuste y mejorar su capacidad de generalización. Finalmente, emplearemos el conjunto de test para obtener una evaluación final de los modelos entrenados. Es importante destacar la importancia y diferencia de estos dos últimos conjuntos de datos. Si bien ambos se emplean para evaluar los modelos, prescindir del conjunto de validación para emplear también el de test supondría un caso de *data snooping* [119]. Esto significa que nuestros resultados finales se verían sesgados de manera indirecta por las decisiones tomadas sobre las evaluaciones del conjunto de test, de forma que nuestros resultados también estarían sesgados y nuestras conclusiones sobre la generalización de los modelos serían incorrectas.

## 9. Marco experimental y metodología

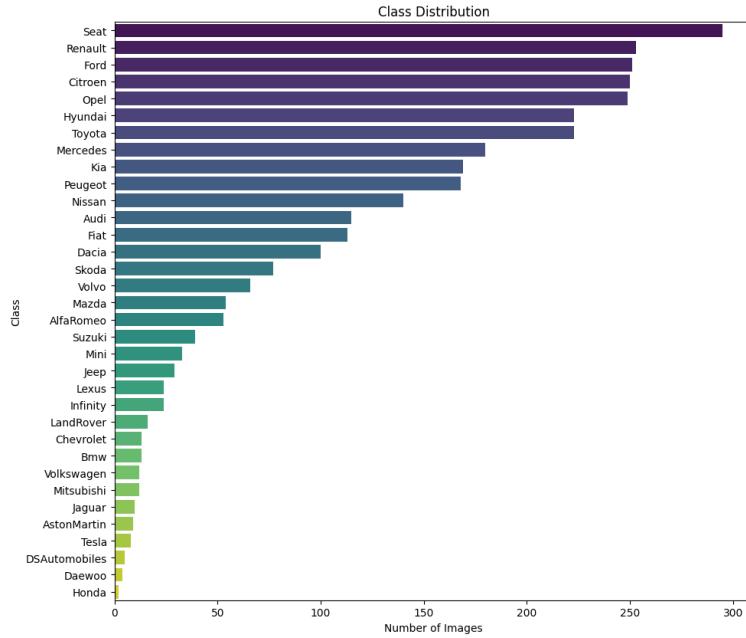


Figura 9.3.: Número de instancias por clase en la especificidad de Marca.

En un inicio, se valoró la posibilidad de realizar una validación cruzada para comparar los resultados de las redes entrenadas. No obstante, dado que nuestro objetivo es estudiar la topología de los datos en estas CNNs respecto a los conjuntos de datos que disponemos y no necesariamente obtener el mejor modelo posible, se descartó dicha opción. Además, el gran coste computacional del entrenamiento de estos modelos, junto a los numerosos entrenamientos realizados, reforzaron esta postura. Este enfoque permite centrar los recursos computacionales en el análisis de la topología de los datos y la realización de los experimentos.

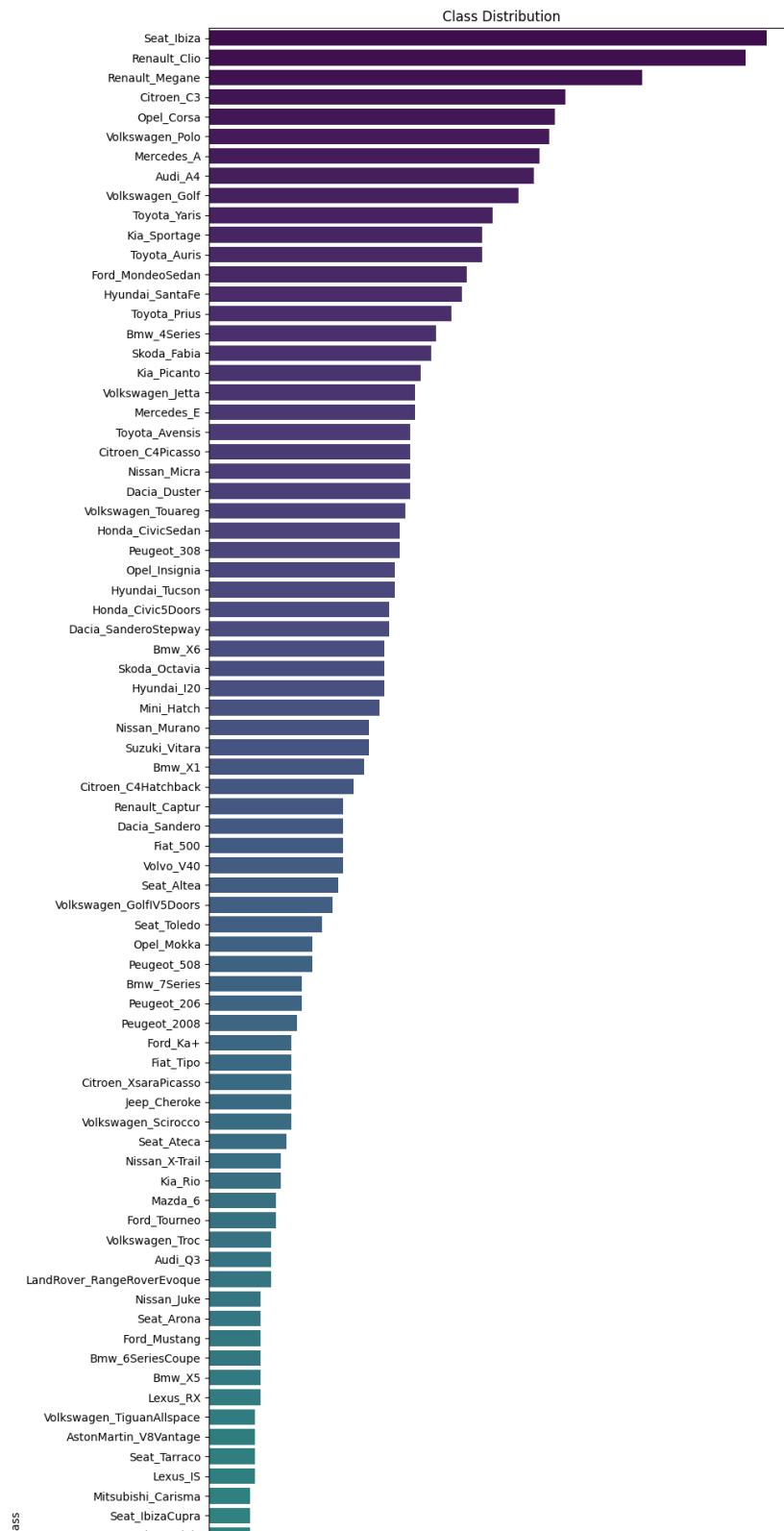
## 9.4. Preprocesamiento de datos

Dada la naturaleza heterogénea de las imágenes empleadas, todas las imágenes se han redimensionado a  $224 \times 224$  con un total de 3 canales normalizado en un rango entre  $[0, 1]$  para los colores RGB. Al estar los modelos preentrenados con ImageNet, los colores de las imágenes se han tipificado respecto a la media y desviación típica de dicho conjunto. Esto es, una media  $\mu = (0.485, 0.456, 0.406)$  y una desviación típica  $\sigma = (0.229, 0.224, 0.225)$ .

Para los modelos entrenados empleando aumento de datos, se han estudiado las siguientes transformaciones:

- **Fluctuación del color:** aplica alteraciones en el brillo, contraste y saturación base de las imágenes. Dada una imagen, se modifica cada uno de estos valores  $\alpha$  en un factor obtenido de una distribución uniforme  $\mathcal{U}([u, v])$ , siendo  $u = \max\{0, 1 - \alpha\}$  y  $v = 1 + \alpha$ .
- **Desenfoque Gaussiano:** se desenfoca la imagen base mediante el uso de una convolución con un *kernel* Gaussiano  $5 \times 5$  y una desviación típica escogida con una probabilidad obtenida de una distribución uniforme discreta para los valores 0.1 y 2.

#### 9.4. Preprocesamiento de datos



## 9. Marco experimental y metodología

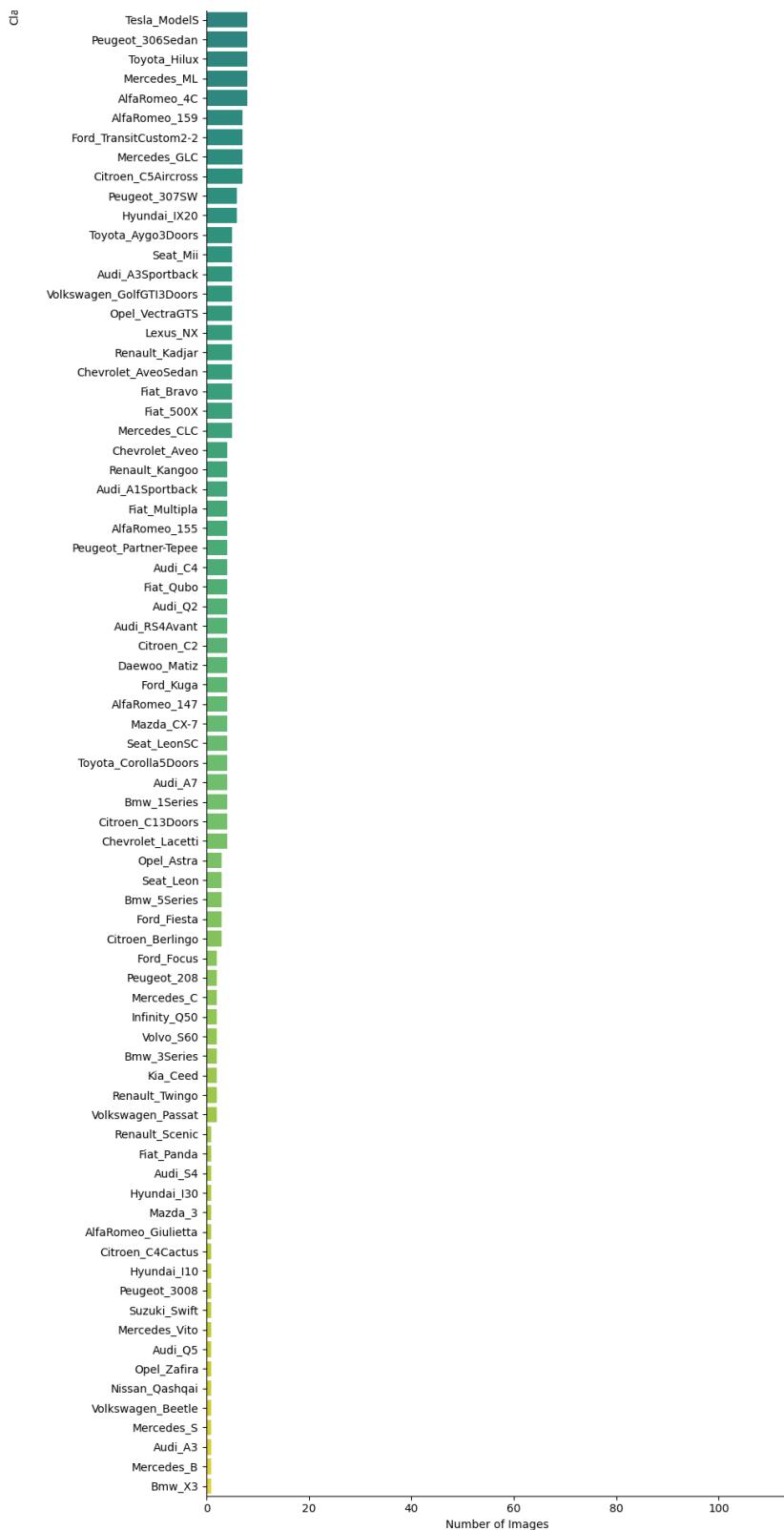


Figura 9.4.: Número de instancias por clase en la especificidad de Marca-Modelo.

- **Transformación especular:** se obtiene la imagen especular con una probabilidad de 0.5.
- **Rotación:** se rotan las imágenes originales un número de grados en cualquiera de los sentidos obtenidos de  $\mathcal{U}([-10, 10])$ .
- **Recorte:** se recortan porciones de  $224 \times 224$  píxeles de la imagen original con un acercamiento de un factor hasta de 1 y un alejamiento de hasta 0.08.

## 9.5. Métricas de evaluación

En el ámbito de los problemas de clasificación binaria, existen numerosas métricas que pueden emplearse para estudiar cómo de bueno es un modelo y hacer un seguimiento de su entrenamiento. Las más populares son aquellas obtenidas a partir de la **matriz de confusión**, la cual muestra el rendimiento del modelo comparando las predicciones realizadas con las etiquetas originales. Para la elaboración de las métricas, los valores de la matriz de confusión se segmentan en cuatro clases:

- **Verdaderos positivos (TP):** son los casos en los que el modelo predice correctamente la clase positiva. Es decir, cuando el modelo predice que un ejemplo es positivo y realmente lo es.
- **Verdaderos negativos (TN):** son los casos en los que el modelo predice correctamente la clase negativa. Es decir, cuando el modelo predice que un ejemplo es negativo y realmente lo es.
- **Falsos positivos (FP):** son los casos en los que el modelo predice incorrectamente la clase positiva. Es decir, cuando el modelo predice que un ejemplo es positivo, pero en realidad es negativo. Esto también se conoce como error de tipo I.
- **Falsos negativos (FN):** son los casos en los que el modelo predice incorrectamente la clase negativa. Es decir, cuando el modelo predice que un ejemplo es negativo, pero en realidad es positivo. Esto también se conoce como error de tipo II.

En los problemas de clasificación multiclas, esta metodología es adaptada para que tenga sentido en el caso donde el número de clases es mayor que 2. Para ello, se utiliza un enfoque conocido como «uno contra todos» (*one-vs-all*), donde cada clase se compara contra todas las demás clases. En este contexto, se considera positivo cuando la etiqueta predicha se corresponde con su valor original y negativa si ha predicho cualquiera de las clases restantes.

Estas clases son posteriormente empleadas para calcular diferentes métricas en función de los requisitos que debe cumplir el modelo. Entre ellas, las más empleadas son las siguientes:

- **Exactitud o accuracy:** Es el cociente entre el número de predicciones correctas y el total de predicciones realizadas. Se calcula como:

$$\text{Exactitud} = \frac{TP + TN}{TP + TN + FP + FN}.$$

- **Precisión o precision:** Es el cociente entre el número de verdaderos positivos y el total de predicciones positivas. Indica la exactitud de las predicciones positivas del modelo. Se calcula como:

$$\text{Precisión} = \frac{TP}{TP + FP}.$$

## 9. Marco experimental y metodología

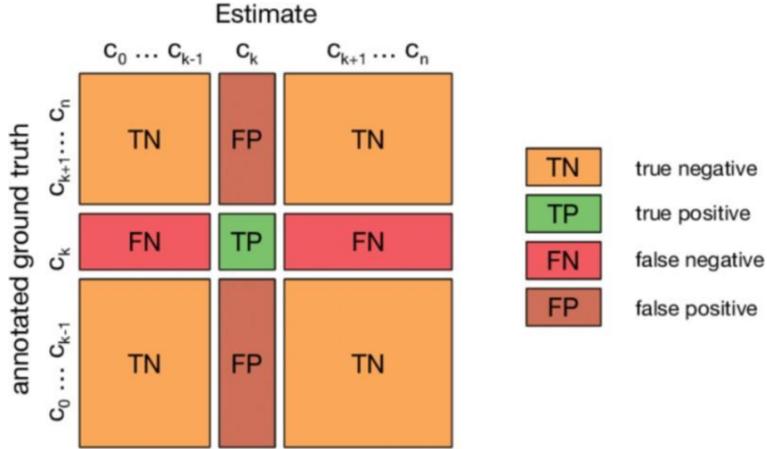


Figura 9.5.: Martiz de confusión multiclasificadora. Dada una predicción de una instancia de la clase  $C_k$ , la tabla muestra cuando se considera TP (verde), TN (naranja), FP (ocre) y FN (rojo). Fuente [112].

- **Sensibilidad o recall:** Es el cociente entre el número de verdaderos positivos y el total de casos reales positivos. Indica la capacidad del modelo para identificar todos los casos positivos. Se calcula como:

$$\text{Sensibilidad} = \frac{TP}{TP + FN}$$

- **F1-Score:** Es la media armónica entre la precisión y la sensibilidad. Proporciona una única métrica que equilibra las dos anteriores y es especialmente útil cuando hay un desequilibrio entre las clases. Se calcula como:

$$F1 = 2 \cdot \frac{\text{Precisión} \cdot \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}}$$

Por otra parte, se ha empleado la persistencia total como métrica para medir la persistencia homológica de los datos en los distintos niveles de la red, además de nuestra propuesta, la persistencia total normalizada.

## 9.6. Proceso de entrenamiento

En esta etapa se ha optado por entrenar los modelos ResNet-18, DenseNet-121 y EfficientNet-Bo bajo las siguientes condiciones para hacer un exhaustivo análisis viendo cómo afectan a la topología de los datos y su rendimiento.

1. **Optimización de Hiperparámetros:** Se realizó una búsqueda en rejilla (*grid search*) para determinar los tamaños de lote óptimos (8, 16, 32 y 64). Además, se emplearon los optimizadores Adam y SGD. Para Adam, se emplearon los hiperparámetros  $\beta_1 = 0.9$ ,  $\beta_2 = 0.999$  y  $\epsilon = 10^{-8}$ .
2. **Aumento de Datos:** Posteriormente, se aplicaron técnicas de aumento de datos al mejor

modelo de cada familia identificado anteriormente. Las transformaciones incluyeron las descritas en la [Sección 9.4](#) para determinar las más beneficiosas en el contexto del entrenamiento final de los modelos, con el fin de mejorar su generalización.

3. **Incorporación de Regularización Topológica:** Finalmente, se ha aplicado una búsqueda en rejilla al término  $\alpha$  en el regularizador de dos formas. Se ha empleado 0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5 y 1 con los extractores de características de EfficientNet-Bo y DenseNet-121 congelados y para transferir DenseNet-121 de la especificidad Marca-Modelo a Marca. Estos mismos valores negados se han empleado para transferir EfficientNet-Bo de Marca a Marca-Modelo.

Todas las tasas de aprendizaje fueron obtenidas a partir de una implementación realizada del método propuesto por Leslie N. Smith [98]. Dicha técnica comienza aplicando una tasa de aprendizaje muy baja que va aumentando exponencialmente en cada lote de entrenamiento hasta un máximo establecido y registrando el error de entrenamiento asociado a cada tasa. Finalmente, se determina el valor óptimo de la tasa escogiéndolo en algún punto previo al mínimo donde la pendiente en la gráfica de error frente a tasa de aprendizaje es más negativa. En particular, para la búsqueda se han empleado un mínimo de  $10^{-7}$  y un máximo de 1. Las Tablas [9.1](#) y [9.2](#) muestran las tasas de aprendizaje empleadas para cada alternativa.

Modelo	SGD				Adam			
	Lote 8	Lote 16	Lote 32	Lote 64	Lote 8	Lote 16	Lote 32	Lote 64
<b>ResNet</b>	0.005	0.001	0.01	0.01	0.0005	0.001	0.001	0.001
<b>DenseNet</b>	0.005	0.01	0.01	0.01	0.0005	0.001	0.001	0.001
<b>EfficientNet</b>	0.001	0.05	0.05	0.05	0.001	0.001	0.001	0.001

Tabla 9.1.: Tasas de aprendizaje escogidas para los diferentes modelos, optimizadores y tamaños de lote en el conjunto de datos Marca.

Modelo	SGD				Adam			
	Lote 8	Lote 16	Lote 32	Lote 64	Lote 8	Lote 16	Lote 32	Lote 64
<b>ResNet</b>	0.005	0.01	0.01	0.05	0.0005	0.001	0.001	0.001
<b>DenseNet</b>	0.005	0.01	0.01	0.01	0.0005	0.001	0.001	0.005
<b>EfficientNet</b>	0.05	0.05	0.05	0.05	0.001	0.005	0.005	0.005

Tabla 9.2.: Tasas de aprendizaje escogidas para los diferentes modelos, optimizadores y tamaños de lote en el conjunto de datos Marca-Modelo.

Se ha empleado *early stopping* como criterio de parada con una paciencia de 5 épocas respecto a los valores de pérdida del conjunto de validación. Todos los modelos han sido entrenados sobre los conjuntos de datos de Marca y Marca-Modelo.

## 9.7. Visualización de la homología persistente

Para cada una de las redes entrenadas y seleccionadas en la [Sección 9.6](#), hemos medido la persistencia de las clases de homología de dimensión 0 y 1 respecto a los conjuntos de datos de entrenamiento, validación y test. En un inicio, se valoró estudiar la topología respecto

## *9. Marco experimental y metodología*

al conjunto de entrenamiento al completo. Debido al gran coste en memoria que suponía, finalmente se optó por evaluarla para un subconjunto escogido de manera aleatoria de los datos de entrenamiento con el mismo número de instancias que los conjuntos de validación y test.

Para la visualización de la homología persistente se han empleado gráficas que muestran la persistencia total y la persistencia total normalizada en función del instante de avance de los datos en la red. Además, se ha utilizado el código de barras descrito en la [Sección 4.3](#) para ofrecer un mayor detalle en ciertos momentos del estudio.

Por último, se ha empleado el método Grad-CAM [\[93\]](#) para visualizar la intensidad de las activaciones mediante un mapa de calor sobre las imágenes originales.

# 10. Resultados experimentales

## 10.1. Rendimiento y selección de modelos

A continuación se presentan los resultados de las métricas respecto a los conjuntos de validación obtenidas tras el entrenamiento de las redes. El objetivo de esta sección será seleccionar los modelos que emplearemos para comparar la evolución de la homología persistente entre ellos.

### 10.1.1. Selección de hiperparámetros

La [Figura 10.1](#) muestra la evolución de la función de pérdida promedio respecto al optimizador a lo largo del entrenamiento. En ella, podemos observar que los modelos entrenados con SGD para los distintos tamaños de lote y arquitecturas han mostrado unas tasas de pérdida más bajas y mayor regularidad en el conjunto de validación. En general, la gráfica muestra un desarrollo correcto del proceso de entrenamiento.

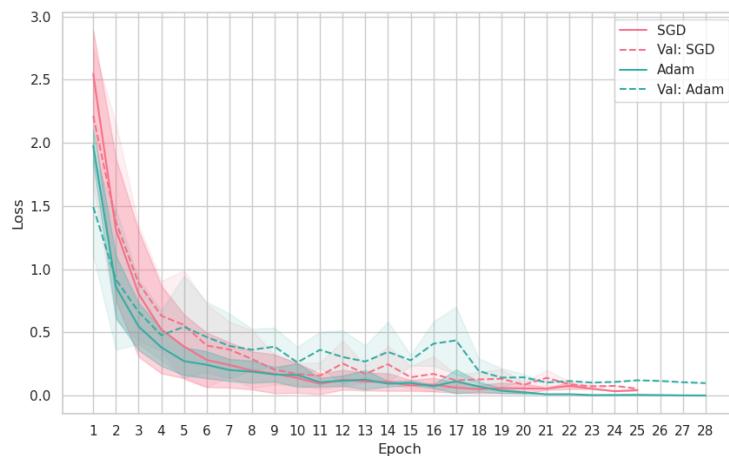


Figura 10.1.: Función de pérdida promedio en entrenamiento y validación de los distintos modelos entrenados en función del optimizador empleado: SGD y Adam. Se muestra de forma sombreada la desviación típica de la función de pérdida.

Por otro lado, la [Figura 10.2](#) muestra como existe una mayor variabilidad en el proceso de entrenamiento entre los modelos con un mismo tamaño de lote. Entre ellos, el que parece presentar una mayor estabilidad para las distintas arquitecturas es con un tamaño de lote 64, pues muestra una menor desviación típica y valores bajos en la función de pérdida.

## 10. Resultados experimentales

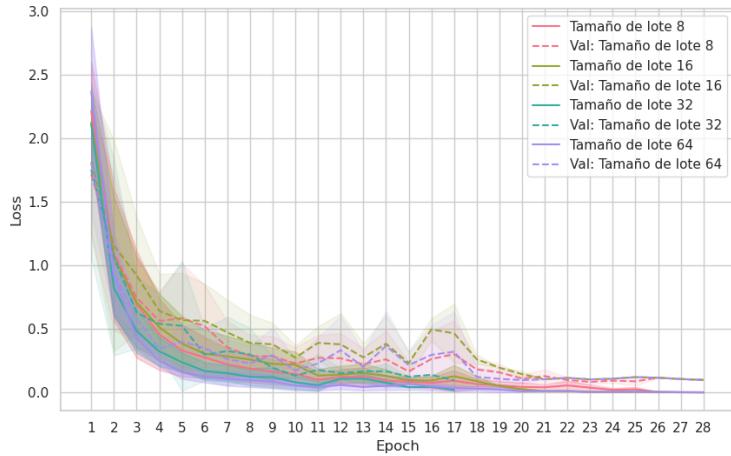


Figura 10.2.: Función de pérdida promedio en entrenamiento y validación de los distintos modelos entrenados en función delos tamaños de lote empleados: 8, 16 ,32 y 64. Se muestra de forma sombreada la desviación típica de la función de pérdida.

Finalmente, las Figuras 10.3 y 10.4, muestran un comportamiento similar a las recién vistas, indicando un correcto desarrollo de la fase de entrenamiento. A diferencia de las anteriores, si bien SGD ha vuelto a mostrar una mejor evolución de la curva de pérdida, es esta vez el tamaño de lote 32 el que ha mostrado unas tasas de pérdida promedio más baja. Otra observación interesante es la disminución en la pendiente de la curva, reflejando una mayor dificultad en el aprendizaje en la especificidad Marca-Modelo.

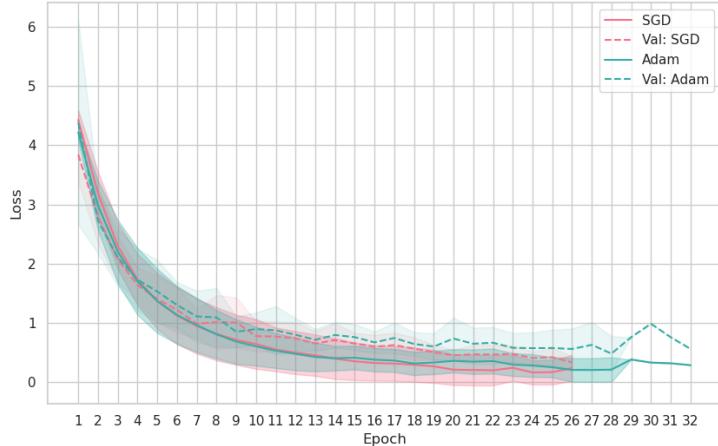


Figura 10.3.: Función de pérdida promedio en entrenamiento y validación de los distintos modelos entrenados en función del optimizador empleado: SGD y Adam. Se muestra de forma sombreada la desviación típica de la función de pérdida.

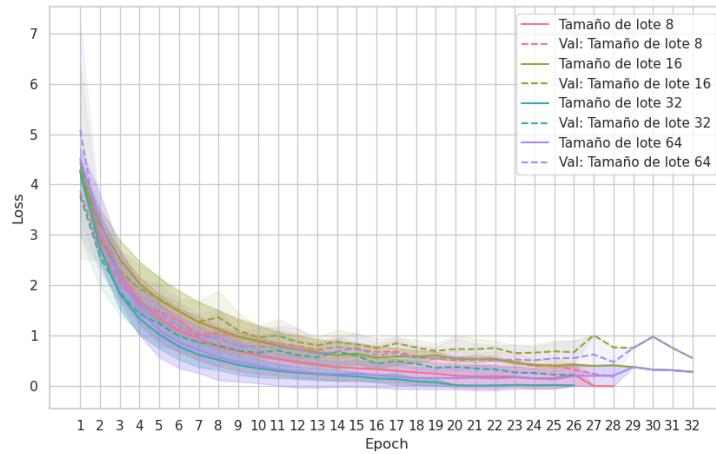


Figura 10.4.: Función de pérdida promedio en entrenamiento y validación de los distintos modelos entrenados en función de los tamaños de lote empleados: 8, 16 ,32 y 64. Se muestra de forma sombreada la desviación típica de la función de pérdida.

Las Tablas 10.1 y 10.2 muestran las métricas obtenidas durante el proceso de entrenamiento tanto para SGD como Adam respectivamente para la especificidad de Marca. Todas ellas han alcanzado el criterio de parada en menos de 2 horas y de 50 épocas. En general, podemos observar que ambos optimizadores logran una puntuación superior al 95 % en la gran mayoría de las métricas. Es interesante observar cómo pese al desbalance de los datos, el F1-Score sigue mostrando resultados excelentes. Entre todos los modelos vemos que los mejores resultados los ha obtenido EfficientNet-Bo con el optimizador SGD y un tamaño de lote 8 y 64. Sin embargo, finalmente se ha optado por escoger el mismo modelo con un tamaño de lote 64, pues su resultados son muy similares y su F1-Score es ligeramente mayor.

Modelo	Tamaño de Lote	Exactitud	Precisión	Sensibilidad	F1-Score
ResNet-18	8	0.9536	0.9416	0.9474	0.9443
	16	0.9102	0.8834	0.8953	0.8891
	32	0.9721	0.9785	0.9753	0.9769
	64	0.9814	0.9758	0.9775	0.9767
DenseNet-121	8	0.9814	0.9850	0.9828	0.9838
	16	0.9752	0.9755	0.9783	0.9768
	32	0.9845	0.9931	0.9863	0.9897
	64	0.9814	0.9738	0.9753	0.9746
EfficientNet-Bo	8	<b>0.9907</b>	0.9907	<b>0.9917</b>	0.9911
	16	0.9474	0.9512	0.9515	0.9512
	32	0.9690	0.9798	0.9689	0.9743
	64	0.9859	<b>0.9925</b>	0.9907	<b>0.9916</b>

Tabla 10.1.: Métricas de validación para los modelos optimizados con SGD en la especificidad Marca.

## 10. Resultados experimentales

Modelo	Tamaño de Lote	Exactitud	Precisión	Sensibilidad	F1-Score
ResNet-18	8	0.9536	0.9496	0.9508	0.9501
	16	0.9536	0.9486	0.9467	0.9475
	32	0.9381	0.9319	0.9235	0.9275
	64	0.9721	0.9713	0.9633	0.9672
DenseNet-121	8	0.9350	0.9428	0.9363	0.9393
	16	0.9536	0.9465	0.9463	0.9462
	32	0.9659	0.9750	0.9694	0.9721
	64	<b>0.9876</b>	<b>0.9916</b>	<b>0.9869</b>	<b>0.9892</b>
EfficientNet-Bo	8	0.9598	0.9588	0.9576	0.9580
	16	0.9752	0.9768	0.9757	0.9762
	32	0.9783	0.9720	0.9698	0.9709
	64	0.9567	0.9330	0.9332	0.9331

Tabla 10.2.: Métricas de validación para los modelos optimizados con Adam en la especificidad Marca.

Por otro lado, los modelos entrenados sobre el conjunto de datos Marca-Modelo han mostrado mayores dificultades durante el proceso de aprendizaje, tal y como muestran los resultados de las Tablas 10.3 y 10.4. Pese a ello, la configuración que ha mostrado mejores resultados en las métricas evaluadas ha sido DenseNet-121 con SGD y un tamaño de lote 32. Esta configuración ha mostrado ser la mejor tanto en exactitud como en F1-Score, además de tener resultados competentes en el resto de métricas. Por ello, emplearemos dicho modelo en las siguientes etapas.

Modelo	Tamaño de Lote	Exactitud	Precisión	Sensibilidad	F1-Score
ResNet-18	8	0.8000	0.7934	0.7968	0.7949
	16	0.8037	0.7866	0.7969	0.7916
	32	0.9000	0.8743	0.8912	0.8826
	64	0.8741	0.8510	0.8540	0.8524
DenseNet-121	8	0.8667	0.8569	0.8624	0.8595
	16	0.8741	0.8739	0.8732	0.8734
	32	<b>0.9407</b>	<b>0.9176</b>	<b>0.9314</b>	<b>0.9243</b>
	64	0.9111	0.8985	0.9009	0.8997
EfficientNet-Bo	8	0.9111	0.9122	0.9116	0.9118
	16	0.8000	0.8020	0.8066	0.8040
	32	0.9148	0.9213	0.9181	0.9197
	64	0.9000	0.8941	0.8920	0.8929

Tabla 10.3.: Métricas de validación para los modelos optimizados con SGD en la especificidad Marca-Modelo.

Modelo	Tamaño de Lote	Exactitud	Precisión	Sensibilidad	F1-Score
ResNet-18	8	0.8556	0.8526	0.8536	0.8529
	16	0.8556	0.8471	0.8543	0.8506
	32	0.7741	0.7367	0.7561	0.7461
	64	0.8556	0.8493	0.8438	0.8464
DenseNet-121	8	0.8741	0.8672	0.8725	0.8696
	16	0.8667	0.8685	0.8687	0.8685
	32	0.9037	0.8943	0.9018	0.8979
	64	<b>0.9201</b>	<b>0.9263</b>	0.9103	<b>0.9182</b>
EfficientNet-Bo	8	0.8481	0.8384	0.8460	0.8420
	16	0.8926	0.8946	0.8924	0.8935
	32	0.9148	0.9044	<b>0.9143</b>	0.9093
	64	0.8111	0.7869	0.7906	0.7887

Tabla 10.4.: Métricas de validación para los modelos optimizados con Adam en la especificidad Marca-Modelo.

### 10.1.2. Selección de transformaciones de datos

Tras la anterior etapa se ha procedido a realizar un aumento de datos sobre los dos conjuntos seleccionados: EfficientNet-Bo con SGD y un tamaño de lote 64 para la especificidad Marca, y DenseNet-121 con SGD y tamaño de lote 32 para la especificidad Marca-Modelo. Las Tablas 10.5 y 10.6 muestran los resultados obtenidos para cada una de las transformaciones descritas en la Sección 9.4.

Si bien el modelo entrenado para el conjunto de datos Marca con las transformaciones no ha mejorado significativamente, vemos que las transformaciones sobre el modelo entrenado para Marca-Modelo han logrado unas mejoras notablemente. El objetivo del aumento de datos en las CNNs es mejorar la capacidad de generalización de los modelos. Sin embargo, el primer modelo ya gozaba de muy buenas métricas por lo que un empeoramiento general al realizar aumento de datos podría significar un sobreajuste del modelo. Por tanto, entremos el modelo de nuevo con las transformaciones que mejor hayan funcionado para estudiar cómo afectan a la topología.

Transformación	Exactitud	Precisión	Sensibilidad	F1-Score
Fluctuación	<b>0.9814</b>	<b>0.9869</b>	<b>0.9844</b>	<b>0.9856</b>
Desenfoque	0.9567	0.9360	0.9388	0.9373
Simetría	0.9721	0.9634	0.9585	0.9609
Recorte	0.9783	0.9765	0.9709	0.9737
Rotación	0.9783	0.9706	0.9657	0.9682

Tabla 10.5.: Métricas de validación para EfficientNet-Bo con transformaciones de datos en la especificidad Marca.

## 10. Resultados experimentales

Transformación	Exactitud	Precisión	Sensibilidad	F1-Score
Fluctuación	0.9783	0.9728	0.9745	0.9736
Desenfoque	0.9783	0.9790	0.9750	0.9769
Simetría	0.9814	0.9804	0.9775	0.9789
Recorte	0.9845	0.9863	0.9817	0.9840
Rotación	<b>0.9876</b>	<b>0.9914</b>	<b>0.9874</b>	<b>0.9894</b>

Tabla 10.6.: Métricas de validación para DenseNet-121 con transformaciones de datos en la especificidad Marca-Modelo.

Finalmente, la [Tabla 10.7](#) muestra los resultados de los modelos reentrenados con las transformaciones escogidas: EfficientNet-Bo con fluctuaciones en el color, recortes aleatorios y rotaciones aleatorias en Marca; y DenseNet-121 con todas las transformaciones propuestas en Marca-Modelo.

Modelo	Exactitud	Precisión	Sensibilidad	F1-Score
EfficientNet-Bo Trans (Marca)	0.9505	0.9627	0.9501	0.9562
DenseNet-121 Trans (Marca-Modelo)	0.7962	0.7660	0.7769	0.7712

Tabla 10.7.: Rendimiento final de los modelos con aumento de datos en el conjunto de validación para EfficientNet-Bo en la especificidad Marca y para DenseNet-121 en la especificidad Marca-Modelo.

Una vez obtenidos los modelos, podemos observar en la [Tabla 10.8](#) las métricas obtenidas en cada modelo en el conjunto de test, tanto el modelo base como el de aumento de datos. Es especialmente interesante el comportamiento de DenseNet-121 con aumento de datos. Pese a tener bajos resultados en el conjunto de validación, ha mostrado una buena generalización en el conjunto de test, incluso superando al modelo base.

Modelo	Exactitud	Precisión	Sensibilidad	F1-Score
EfficientNet-Bo Base (Marca)	<b>0.9505</b>	<b>0.9462</b>	<b>0.9386</b>	<b>0.9423</b>
EfficientNet-Bo Trans (Marca)	0.9474	0.9319	0.9271	0.9294
DenseNet-121 Base (Marca-Modelo)	0.9185	0.9077	0.9126	0.9101
DenseNet-121 Trans (Marca-Modelo)	<b>0.9474</b>	<b>0.9319</b>	<b>0.9271</b>	<b>0.9294</b>

Tabla 10.8.: Rendimiento final de los modelos con y sin aumento de datos en el conjunto de test para EfficientNet-Bo en la especificidad Marca y para DenseNet-121 en la especificidad Marca-Modelo.

## 10.2. Análisis de la homología persistente

Una vez habiendo entrenado todos los modelos necesarios, procederemos analizando los resultados obtenidos en función de la homología persistente. Para ello, se ha calculado la persistencia total de una muestra de 128 instancias (debido al coste en memoria) del conjunto de test tras cada activación no lineal presente en la red evaluada. Las gráficas empleadas a lo largo de toda la sección registran tanto la persistencia total como la normalizada de la muestra frente a su posición relativa en su paso por la red expresada en porcentaje. Además, las figuras muestran la curva de regresión cuadrática calculada a partir de la persistencia con

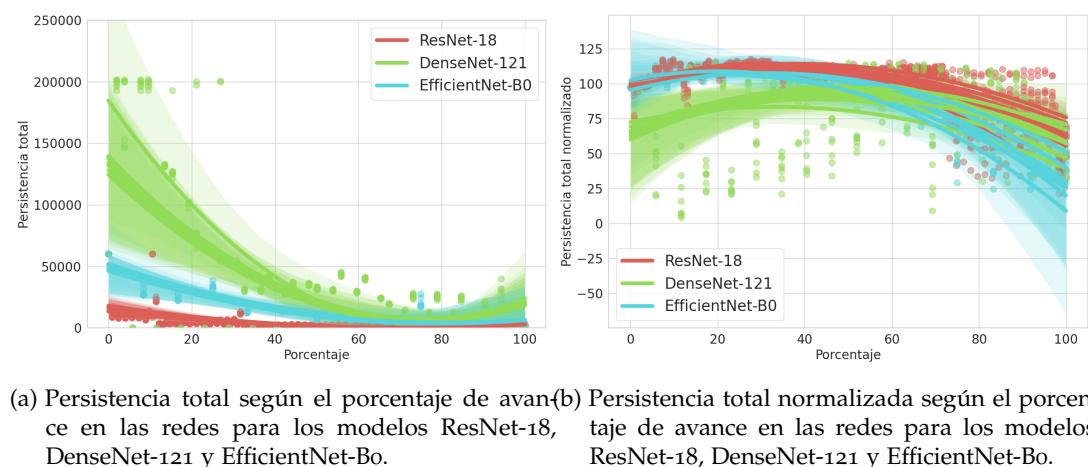
el fin de una visualización más clara de la tendencia de dichos valores. Las áreas sombreadas indican el intervalo de confianza del 95 %.

Las Secciones 10.2.1, 10.2.2 y 10.2.3 realizan un estudio comparativo en función de la arquitectura, optimizador y tamaño de lote descritos en el Capítulo 9. Posteriormente, en las Secciones 10.2.4, 10.2.5 y 10.2.6, el análisis se realiza a parir del mejor modelo para cada especificidad con el fin de analizar cómo afectan el aumento de datos, la granularidad del conjunto y la partición de datos escogida a la topología de estos.

### 10.2.1. Comparación según la arquitectura

**Especificidad Marca** Comencemos analizando la Figura 10.5a. La gráfica muestra una tendencia clara entre los modelos estudiados con los diferentes hiperparámetros: una disminución pronunciada de la persistencia total con un leve repunte en la etapa final de ejecución. Esto implica que las transformaciones que realiza el modelo sobre el conjunto de instancias (como los cambios en dimensionalidad y las funciones aplicadas sobre ellos) tienden tanto a disminuir la persistencia de las clases de homología como a reducir el número de ellos que se generan, claramente simplificando los datos. Un comportamiento interesante es el obtenido al final de las ejecuciones, donde los modelos complican la homología persistente de los datos de cara a la tarea de clasificación final, aumentando la persistencia de componentes conexas y otras características topológicas más complejas, con el fin de facilitar la separación de clases final.

Asimismo, la Figura 10.5b muestra una conclusiones similares con otra tendencia: la persistencia homológica de los datos crece durante las primeras fases de la inferencia, mientras que desciende de cara al final de la ejecución. Es decir, las clases de homología persistente tienden a ser más homogéneas en el punto medio de la ejecución, de forma que los datos están más dispersos y desordenados. De no ser así, las componentes conexas y clases de homología persistentes en dimensión 1 morirían en fases tempranas de la filtración, lo que daría una persistencia total normalizada más baja.

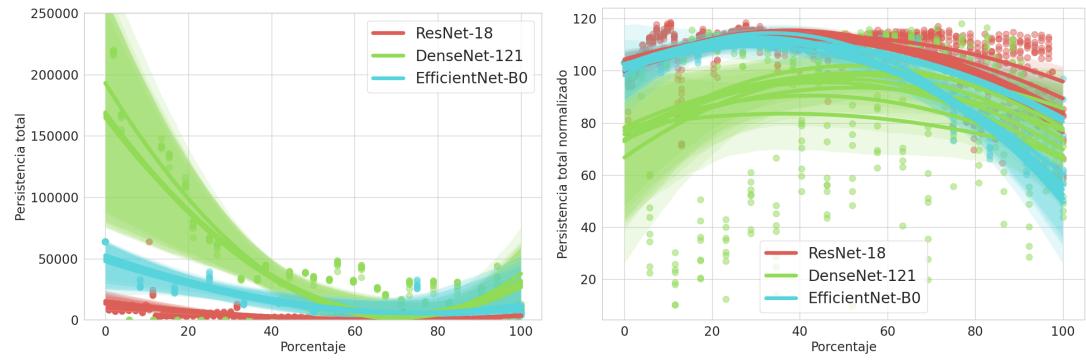


(a) Persistencia total según el porcentaje de avance en las redes para los modelos ResNet-18, DenseNet-121 y EfficientNet-Bo.

(b) Persistencia total normalizada según el porcentaje de avance en las redes para los modelos ResNet-18, DenseNet-121 y EfficientNet-Bo.

Figura 10.5.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) de diferentes arquitecturas de redes neuronales en función del porcentaje de avance de los datos a través de la red para la especificidad Marca-Modelo.

**Especificidad Marca-Modelo** Los resultados de las gráficas de la Figura 10.6 muestran una tendencia similar. En este caso, las tendencias del modelo DenseNet-121 en la Figura 10.6a muestran una pendiente más pronunciada, indicando transformaciones más agresivas en la homología persistente de los datos. Por otro lado, la Figura 10.6b muestra un comportamiento algo diferente respecto a la obtenida en la especificidad Marca, obteniendo unos valores de persistencia total normalizada notablemente superiores al final de las ejecuciones. Dicha consecuencia puede deberse al aumento en la granularidad de la clasificación, haciendo necesarias un mayor número de componentes conexas para facilitar dicha tarea.



(a) Persistencia total según el porcentaje de avance en las redes para los modelos ResNet-18, DenseNet-121 y EfficientNet-Bo.  
 (b) Persistencia total normalizada según el porcentaje de avance en las redes para los modelos ResNet-18, DenseNet-121 y EfficientNet-Bo.

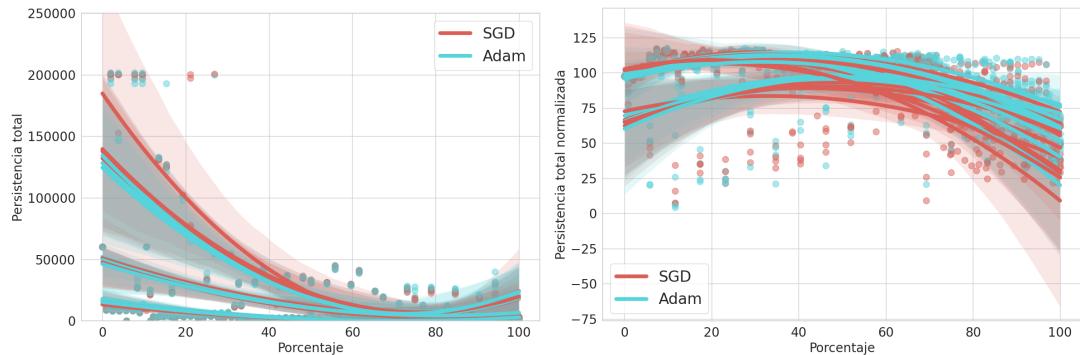
Figura 10.6.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) de diferentes arquitecturas de redes neuronales en función del porcentaje de avance de los datos a través de la red para la especificidad Marca-Modelo.

Es claro que las Figuras 10.5 y 10.6 indican que la arquitectura escogida es un factor determinante en las transformaciones que los datos sufren desde el punto de vista topológico. Todos los modelos entrenados con la misma arquitectura muestran evoluciones muy similares incluso para las distintas especificidades, donde si que se aprecia un desplazamiento vertical de la homología persistente en función de la granularidad de las clases del conjunto.

### 10.2.2. Comparación según el optimizador

**Especificidad Marca** A diferencia de la comparativa en función de la arquitectura, la Figura 10.7 no muestra patrones tan claros en cómo afecta el optimizador a la persistencia homológica. La Figura 10.7a muestra cómo los modelos que presentan una menor persistencia total al inicio, como ResNet-18, tienden a presentar una persistencia inicial todavía menor para el optimizador SGD. Sin embargo, esta tendencia empieza a cambiar cuando vamos pasando a modelos de mayor complejidad topológica como DenseNet-121, donde en general Adam parece mostrar una menor persistencia inicial. Por otro lado, la Figura 10.7b no parece mostrar ningún patrón que nos indique que la elección del optimizador sea relevante para la modificación de la homología persistente de los datos.

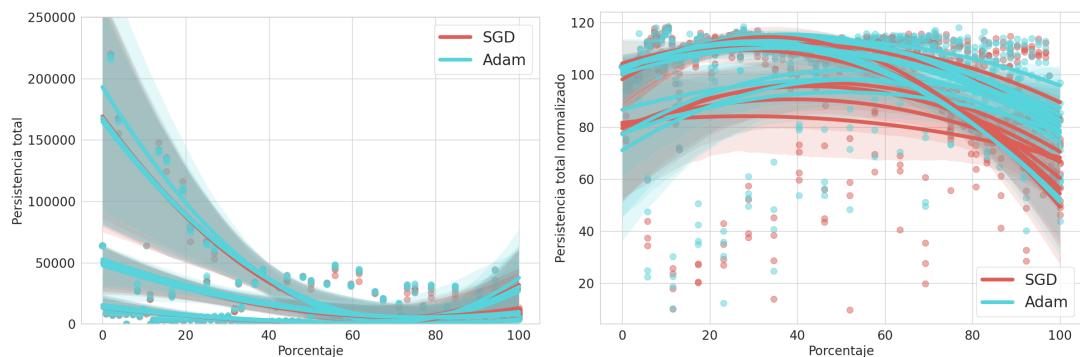
## 10.2. Análisis de la homología persistente



(a) Persistencia total según el porcentaje de avance en las redes para optimizadores SGD y Adam. (b) Persistencia total normalizada según el porcentaje de avance en las redes para SGD y Adam.

Figura 10.7.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) de diferentes optimizadores de redes neuronales en función del porcentaje de avance de los datos a través de la red para la especificidad Marca.

**Especificidad Marca-Modelo** De nuevo, los resultados obtenidos en la Figura 10.8 son poco esclarecedores acerca de la homología de los datos. No obstante, el aumento en el número de clases parece haber homogeneizado las diferencias observadas en la persistencia total, tal y como muestra la Figura 10.8a. Además, las tendencias en la Figura 10.8b muestran una mayor persistencia total normalizada al final de la inferencia para los modelos entrenados con Adam.



(a) Persistencia total normalizada según el porcentaje de avance en la red para optimizadores SGD y Adam. (b) Persistencia total normalizada según el porcentaje de avance en la red para optimizadores SGD y Adam.

Figura 10.8.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) de diferentes optimizadores de redes neuronales en función del porcentaje de avance de los datos a través de la red para la especificidad Marca-Modelo.

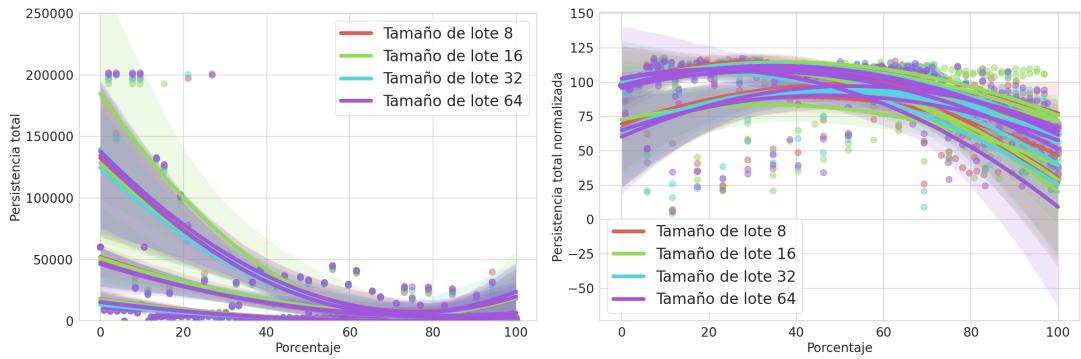
Los resultados recién vistos sobre las Figuras 10.7 y 10.8 parecen mostrar que el optimizador escogido (al menos, en el caso de los dos empleados) no es un factor especialmente relevante a la hora de modificar la «forma» de los datos. A pesar de ello, las tendencias observadas al inicio de la red en la persistencia total y al final de ella en la persistencia

## 10. Resultados experimentales

total normalizada muestran de manera débil ciertos patrones que podrían estudiarse en más profundidad.

### 10.2.3. Comparación según el tamaño de lote

**Especificidad Marca** Las gráficas de la Figura 10.9 no muestran ninguna influencia directa o significativa de la elección del tamaño de lote en el ámbito de la topología de los datos durante la etapa de entrenamiento. Las curvas de regresión se muestran muy entrelazadas y las nubes de puntos no muestran claras distinciones o agrupaciones.



- (a) Persistencia total según el porcentaje de avance en las redes entrenadas para diferentes tamaños de lote.  
(b) Persistencia total normalizada según el porcentaje de avance en las redes para diferentes tamaños de lote.

Figura 10.9.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para diferentes tamaños de lote en función del porcentaje de avance de los datos a través de la red para la especificidad Marca.

**Especificidad Marca-Modelo** La misma observación se obtiene para los modelos entrenados en la especificidad Marca-Modelo, donde las gráficas de la Figura 10.10 muestran un patrón bastante similar a las recién comentadas.

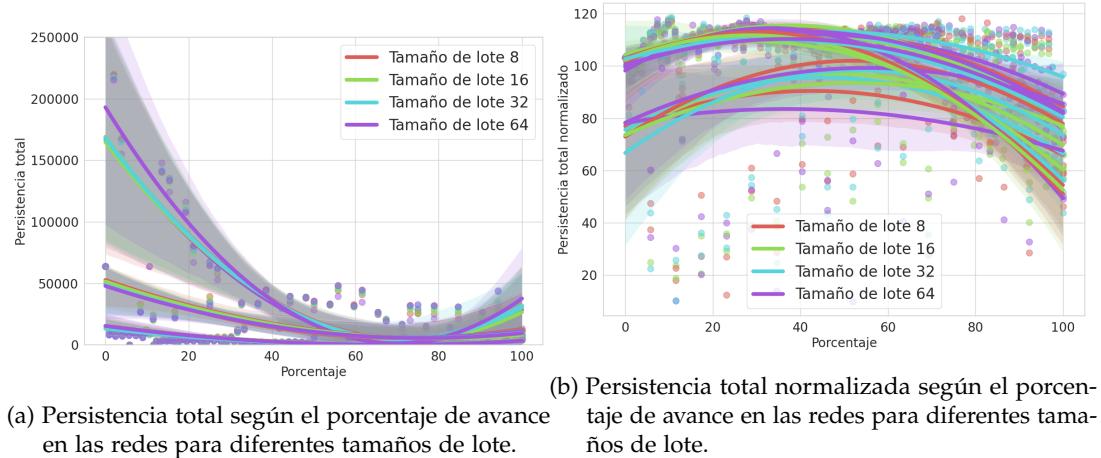


Figura 10.10.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para diferentes tamaños de lote en función del porcentaje de avance de los datos a través de la red para la especificidad Marca-Modelo.

Es curioso observar que esta elección de hiperparámetros no muestre alteraciones en la homología persistente de los datos de test estudiados. Este hecho podría implicar que los métodos de optimización empleados son poco sensibles al tamaño de lote escogido a la hora de inferir la variedad subyacente de los datos.

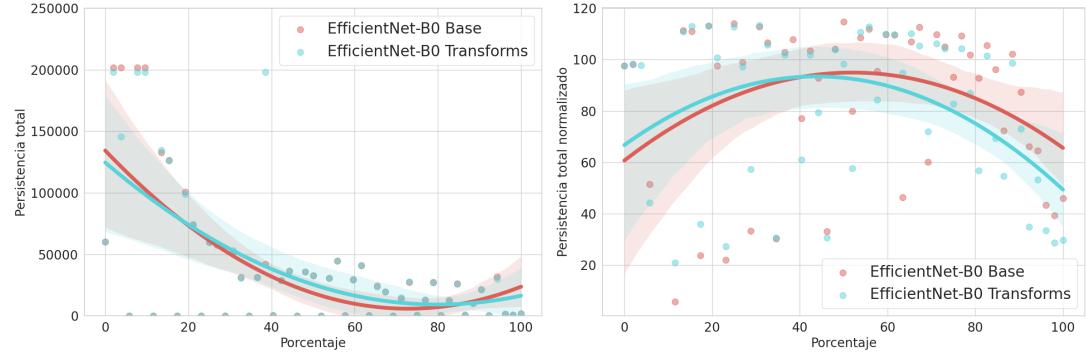
#### 10.2.4. Comparación en presencia de aumento de datos

A continuación trabajaremos sobre los modelos escogidos en la Subsección 10.1.1 para cada especificidad.

**Especificidad Marca: EfficientNet-BO** La Figura 10.11 muestra los resultados de persistencia total y persistencia total normalizada para el modelo base de EfficientNet-Bo y su variante con aumento de datos. Podemos observar que el modelo que obtuvo mejores métricas en el conjunto de test, el modelo base, presenta una persistencia total superior tanto al inicio como al final respecto al modelo con aumento de datos, mientras que es menor en el punto medio de la ejecución, tal y como se ve en la Figura 10.11a. Esto es, el modelo con mejores métricas muestra transformaciones más agresivas sobre la variedad subyacente de los datos.

En cuanto a la Figura 10.11b, vemos que el modelo con aumento de datos presenta una persistencia total normalizada inicial más alta que el modelo base y una final más baja. Además, vemos que el máximo de persistencia lo alcanza antes que el modelo base, mostrando una traslación del proceso de aumento de la complejidad en homología persistente a etapas más tempranas de la ejecución.

## 10. Resultados experimentales

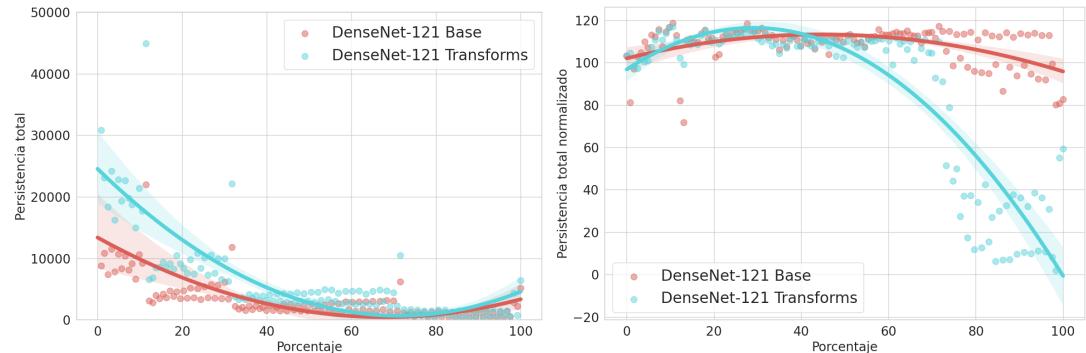


(a) Persistencia total según el porcentaje de avance en la red para el modelo base entrenado EfficientNet-Bo y su versión con aumento de datos.

(b) Persistencia total normalizada según el porcentaje de avance en la red para el modelo base entrenado EfficientNet-Bo y su versión con aumento de datos.

Figura 10.11.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para EfficientNet-Bo Base y EfficientNet-Bo Transforms en función del porcentaje de avance de los datos a través de la red para la especificidad Marca.

**Especificidad Marca-Modelo: DenseNet-121** Estas observaciones se ven reforzadas en la Figura 10.12. En particular, la Figura 10.12a muestra dichas observaciones de una manera más agresiva. Vemos que el mejor modelo, DenseNet-121 con aumento de datos, muestra una persistencia total mayor al inicio de la red y al final. Además, la Figura 10.12b muestra como el aumento de datos traslada de nuevo el máximo a momentos más tempranos de ejecución y una fuerte reducción de la complejidad total normalizada al final de la red.



(a) Persistencia total según el porcentaje de avance en la red para el modelo base entrenado EfficientNet-Bo y su versión con aumento de datos.

(b) Persistencia total normalizada según el porcentaje de avance en la red para el modelo base entrenado EfficientNet-Bo y su versión con aumento de datos.

Figura 10.12.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para EfficientNet-Bo Base y EfficientNet-Bo Transforms en función del porcentaje de avance de los datos a través de la red para la especificidad Marca-Modelo.

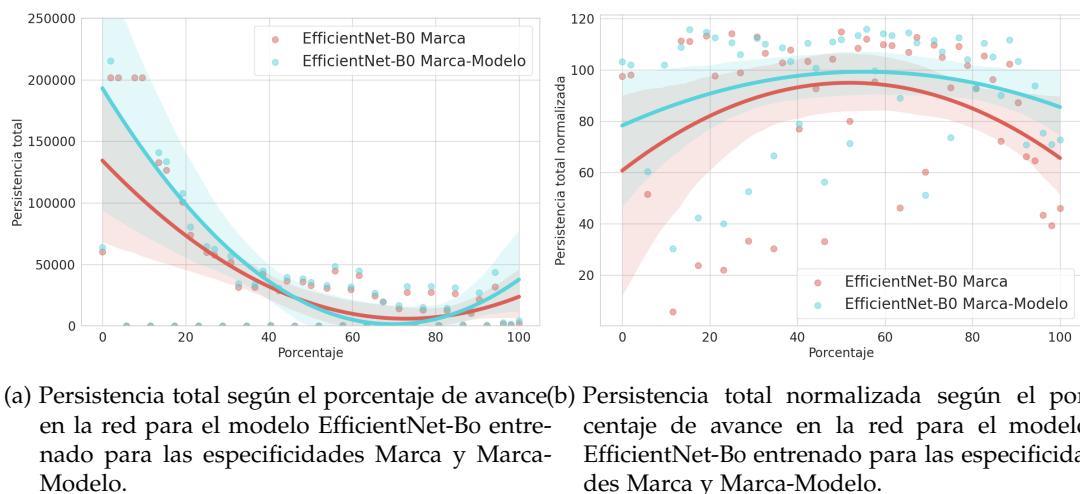
Los resultados recién vistos muestran patrones interesantes en el comportamiento del mo-

delo cuando realizamos aumento de datos: tiende a realizar modificaciones más intensas en fases anteriores de la red y simplificar las estructuras al final del modelo. Este comportamiento muestra que las componentes conexas al final de la red son más compactas y están mejor definidas, lo que lleva a una reducción de la complejidad topológica. No solo eso, si no que al tener en cuenta las clases de persistencia homológicas de dimensión 1, también se reducen el número de componentes conexas que forman bucles, lo que evita entrelazamientos entre éstas.

### 10.2.5. Comparación según la granularidad de las clases

**EfficientNet-BO** EfficientNet-Bo ha mostrado el mejor rendimiento en las métricas empleadas para la especificidad Marca. La persistencia total del modelo entrenado en Marca-Modelo en la Figura 10.13a muestra transformaciones más agresivas en la homología persistente de los datos. Aquí, en los extremos de la ejecución los datos presentan una persistencia total más alta y un mínimo inferior al del modelo entrenado para Marca.

Por su parte, la persistencia total normalizada muestra valores regularmente superiores para la especificidad Marca-Modelo a los de Marca (Figura 10.13b). Estos hechos parecen coherentes, pues a pesar de corresponderse con el mismo conjunto de datos, la necesidad de etiquetar una mayor variedad de clases implica un desglose más fino de las características de los datos y en consecuencia, una complejidad topológica mayor.



(a) Persistencia total según el porcentaje de avance en la red para el modelo EfficientNet-Bo entrenado para las especificidades Marca y Marca-Modelo.  
(b) Persistencia total normalizada según el porcentaje de avance en la red para el modelo EfficientNet-Bo entrenado para las especificidades Marca y Marca-Modelo.

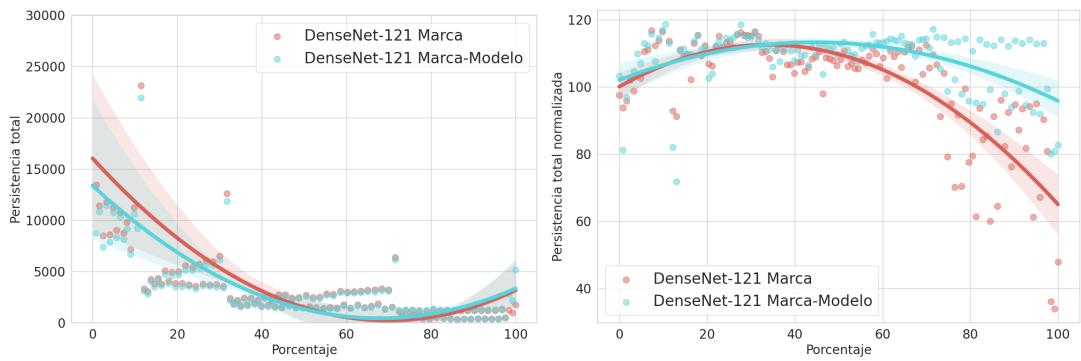
Figura 10.13.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para EfficientNet-Bo entrenado con SGD y un tamaño de lote 64 en función del porcentaje de avance de los datos a través de la red para las especificidades Marca y Marca-Modelo.

**DenseNet-121** En cuanto a DenseNet-121, vemos en la Figura 10.14a que ambas curvas se asemejan más que en el caso anterior. Es interesante ver como en el último instante de la red, el modelo entrenado para la especificidad Marca-Modelo presenta una subida de persistencia considerable, mientras que la de Marca es más modesta. Este hecho muestra claramente la necesidad del modelo de deshacer la simplificación realizada con el fin de separar los datos para la clasificación.

## 10. Resultados experimentales

Además, las muestras tomadas en los distintos instantes de la red muestran cuatro puntos con un aumento considerable de la persistencia total. Estos instantes coinciden con los puntos donde se encuentran las activaciones de transición entre los cuatro bloques densos que presenta DenseNet-121, mostrando cómo las conexiones residuales entre dichos bloques aumenta la complejidad topológica de los datos.

Por otro lado, la Figura 10.14b nos muestra cómo, de nuevo, la persistencia total normalizada presenta en general valores inferiores para la especificidad Marca. Otra observación relevante es la reducción de complejidad más progresiva y gradual que muestra el modelo de Marca en la segunda mitad de la inferencia. El hecho de que DenseNet-121 entrenado con una mayor granularidad requiera de mayor persistencia durante más tiempo puede deberse a la clara dificultad añadida por el aumento de clases.



(a) Persistencia total según el porcentaje de avance en la red para el modelo DenseNet-121 entrenado para las especificidades Marca y Marca-Modelo.

(b) Persistencia total normalizada según el porcentaje de avance en la red para el modelo DenseNet-121 entrenado para las especificidades Marca y Marca-Modelo.

Figura 10.14.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para DenseNet-121 entrenado con SGD y un tamaño de lote 32 en función del porcentaje de avance de los datos a través de la red para las especificidades Marca y Marca-Modelo.

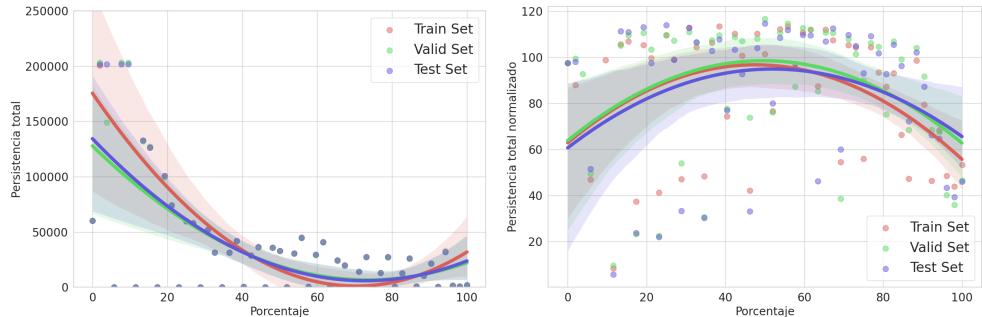
### 10.2.6. Comparación según el subconjunto de datos

Hasta ahora hemos estado viendo como distintos modelos con distintos hiperparámetros y granularidad durante la etapa de entrenamiento afectan a la variedad subyacente de los datos empleados. A continuación, compararemos cómo transforma un mismo modelo los propios datos en función del subconjunto al que pertenecen: entrenamiento, validación y test.

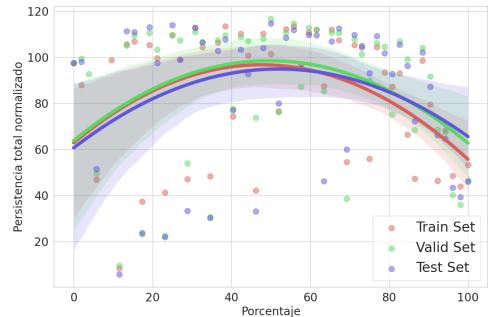
**Especificidad Marca: EfficientNet-BO** Lo primero que observamos en las Figuras 10.15a y 10.15c es una mayor persistencia total al inicio sobre el conjunto de entrenamiento respecto al de validación y test en ambos modelos. Por otro lado, en la Figura 10.15a, que muestra los resultados con mejores métricas, los conjuntos de validación y test muestran una persistencia total prácticamente idéntica, mientras que el modelo con aumento de datos presenta mayores discrepancias al respecto.

## 10.2. Análisis de la homología persistente

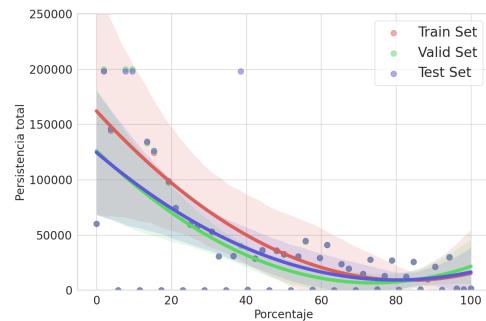
Por lo que se refiere a la persistencia total normalizada (Figuras 10.15b y 10.15d), observamos tendencias muy similares, donde las curvas de validación y test muestran un ajuste más fino al de entrenamiento para el modelo base que para el modelo con aumento de datos.



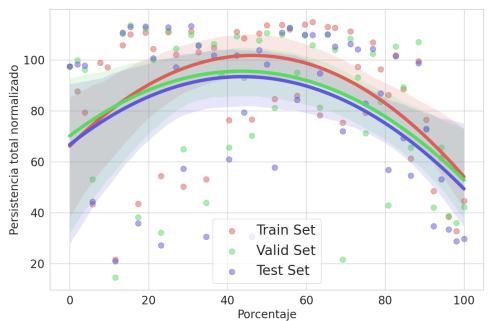
(a) Persistencia total según el porcentaje de avance en la red para los conjuntos de entrenamiento, validación y test.



(b) Persistencia total normalizada según el porcentaje de avance en la red para los conjuntos de entrenamiento, validación y test.



(c) Persistencia total según el porcentaje de avance en la red con aumento de datos para los conjuntos de entrenamiento, validación y test.



(d) Persistencia total normalizada según el porcentaje de avance en la red con aumento de datos para los conjuntos de entrenamiento, validación y test.

Figura 10.15.: Comparación de la persistencia total (a, c) y la persistencia total normalizada (b, d) para los conjuntos de entrenamiento, validación y test para la especificidad Marca. Las Figuras (a) y (b) representan el modelo base, mientras que las Figuras (c) y (d) representan el modelo con aumento de datos.

**Especificidad Marca-Modelo: DenseNet-121** Las Figuras 10.16a y 10.16c muestran un ajuste de la persistencia total prácticamente perfecto entre los tres subconjuntos de datos. Más aún, podemos ver cómo el modelo que ha presentado mejores métricas en el conjunto de test, es decir, el modelo con aumento de datos, presenta un ajuste mucho más fino entre los valores de persistencia total normalizados para los tres subconjuntos (Figura 10.16d). Es interesante observar como en esta ocasión, los puntos obtenidos se comportan de manera más homogénea independientemente del subconjunto escogido, a diferencia de lo observado en la Figura 10.16b.

## 10. Resultados experimentales

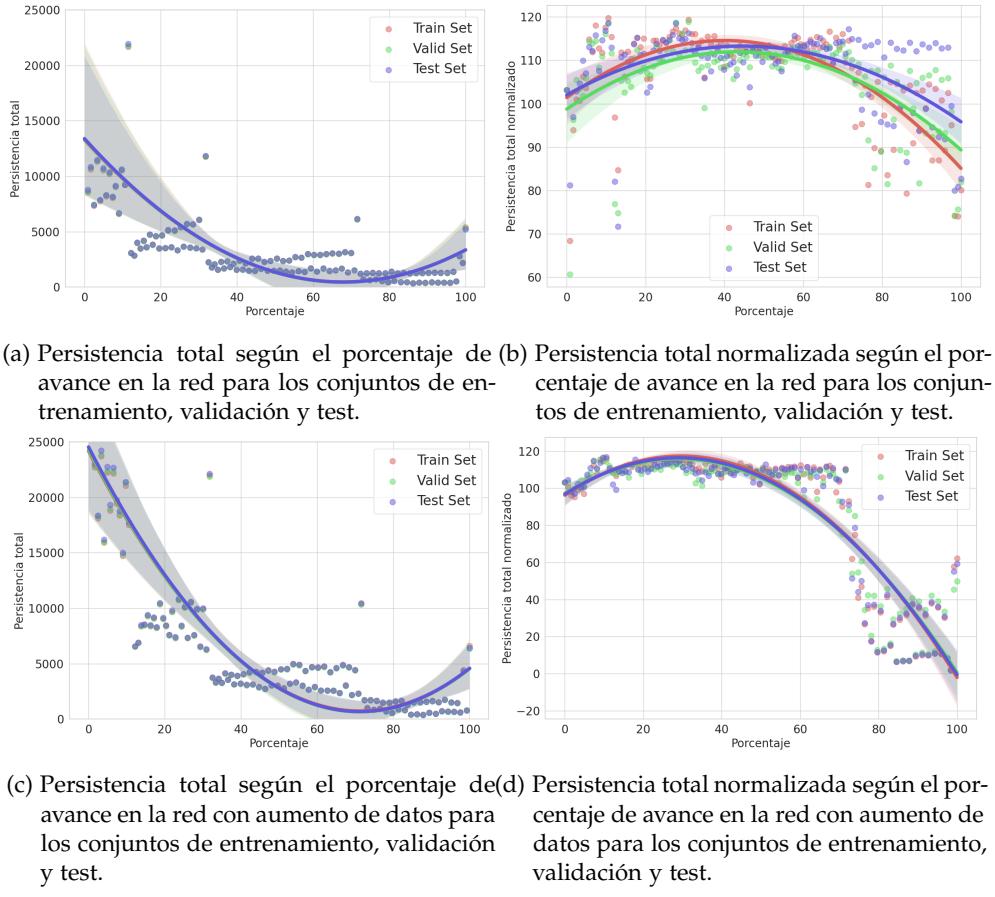


Figura 10.16.: Comparación de la persistencia total (a, c) y la persistencia total normalizada (b, d) para los conjuntos de entrenamiento, validación y test para la especificidad Marca-Modelo. Las Figuras (a) y (b) representan el modelo base, mientras que las Figuras (c) y (d) representan el modelo con aumento de datos.

Las Figuras 10.15 y 10.16 muestran cómo las redes mejor entrenadas obtienen una coincidencia mayor entre los distintos subconjuntos de datos empleados típicamente en el entrenamiento de las CNNs, lo que indica un mejor aprendizaje de la variedad subyacente de la que parten los datos.

### 10.2.7. Reflexiones sobre la topología en CNNs

El análisis recién realizado sobre cómo diferentes configuraciones de modelos de CNNs afectan las características topológicas de los datos ha mostrado patrones claros. Las arquitecturas estudiadas comienzan simplificando los datos, probablemente como consecuencia de los cambios de la dimensionalidad de las activaciones y para eliminar ruido o detalles innecesarios. Sin embargo, hacia el final del proceso de aprendizaje, la persistencia total aumenta nuevamente. Esta dinámica parece ser una estrategia para crear representaciones más ricas y detalladas que ayuden a diferenciar mejor entre las clases durante la clasificación de instancias.

El tipo de optimizador utilizado, como Adam o SGD, también juega un papel relevante, pero su impacto varía a lo largo del entrenamiento. Adam, que ajusta dinámicamente las tasas de aprendizaje, parece ser especialmente efectivo en las fases iniciales, facilitando una rápida reducción de la persistencia total. En cambio, SGD adopta un enfoque diferente, partiendo de una mayor complejidad y realizando transformaciones más agresivas en la topología de los datos.

Sorprendentemente, el tamaño del lote mostró tener un impacto mínimo en la homología persistente. Esto indica que los ajustes en la topología de los datos son bastante robustos a variaciones en el tamaño del lote, siendo este un factor a tener en cuenta a la hora de trabajar con la topología de los datos.

El aumento de datos es otro factor importante. Al entrenar modelos con una mayor variedad de datos durante el entrenamiento, se promueve una adaptación más temprana y efectiva a la complejidad relativa de los datos, lo que lleva a una menor necesidad de trabajar los datos de manera tan intensiva en etapas posteriores. Esto no solo ayuda a generalizar mejor sino también a simplificar eficazmente esa complejidad hacia el final del entrenamiento, preparando el modelo para tareas de clasificación más precisas.

La granularidad en la clasificación necesita que los modelos manejen y preserven una complejidad topológica mayor durante más tiempo, manteniendo las distinciones claras entre múltiples categorías. Al comparar cómo se comportan los modelos en diferentes subconjuntos de datos, es notable que aquellos con mejores métricas ajustan más precisamente sus características topológicas entre los conjuntos de entrenamiento, validación y prueba, mostrando un aprendizaje más consistente y profundo de la estructura subyacente de los datos.

## 10.3. Propuesta de mejora: regularización topológica

El análisis realizado sobre la homología persistente de los datos ha mostrado ciertos patrones y resultados que podrían aprovecharse durante el entrenamiento de las CNNs. En esta sección aplicamos las propuestas realizadas con el fin de confirmar nuestras hipótesis y hallar nuevas técnicas que mejoren y faciliten el aprendizaje a las CNNs.

### 10.3.1. Mejora en clasificación

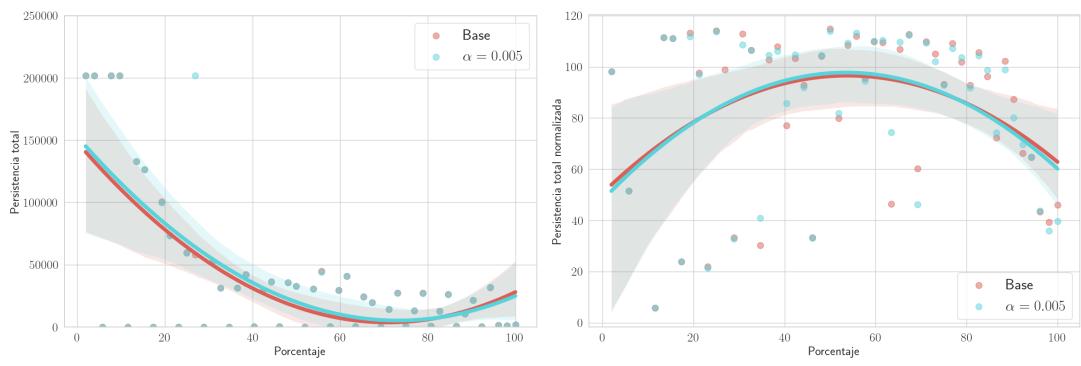
**Especificidad Marca: EfficientNet-BO** La propuesta de emplear el regularizador topológico para mejorar la tasa de clasificación en EfficientNet-Bo parece haber mostrado una leve mejoría respecto al modelo base y el modelo refinado sin emplear el regularizador ([Tabla 10.9](#)).

## 10. Resultados experimentales

Modelo	$\alpha$	Exactitud	Precisión	Sensibilidad	F1-Score
EfficientNet-Bo Base	-	0.9505	0.9462	0.9386	0.9423
EfficientNet-Bo Refinado	0.0	0.9598	0.9512	0.9445	0.9478
EfficientNet-Bo Refinado	0.001	0.9598	0.9496	0.9448	0.9472
EfficientNet-Bo Refinado	0.005	<b>0.9659</b>	<b>0.9545</b>	<b>0.9514</b>	<b>0.9529</b>
EfficientNet-Bo Refinado	0.01	0.9598	0.9517	0.9481	0.9499
EfficientNet-Bo Refinado	0.05	0.9536	0.9515	0.9412	0.9462
EfficientNet-Bo Refinado	0.1	0.9567	0.9475	0.9413	0.9444
EfficientNet-Bo Refinado	0.5	0.9505	0.9486	0.9369	0.9426
EfficientNet-Bo Refinado	1.0	0.9505	0.9453	0.9413	0.9432

Tabla 10.9.: Comparación de métricas tras un proceso de refinamiento en la misma especificidad Marca para distintos valores de  $\alpha$  en el término de regularización del modelo EfficientNet-Bo.

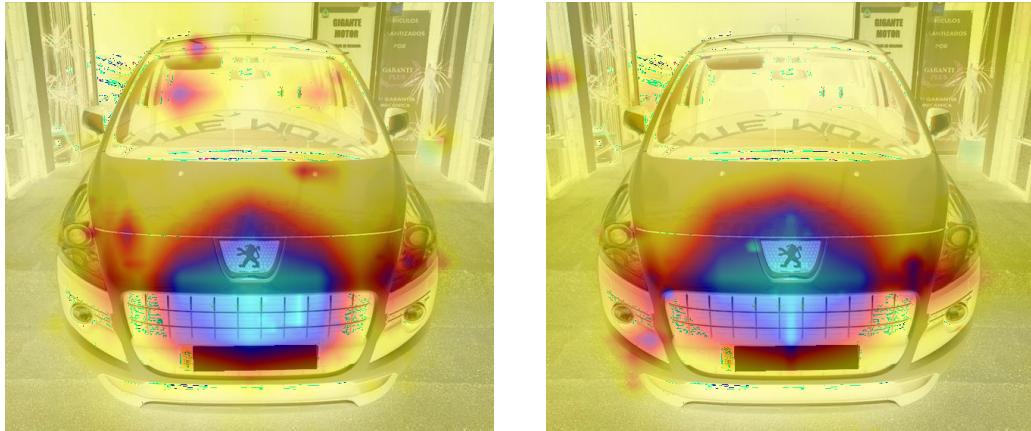
Observando la Figura 10.17a vemos que la persistencia total del modelo refinado sin regularizador es generalmente inferior a la del modelo regularizado. Sin embargo, en las etapas finales muestra un leve repunte frente al modelo sin regularizar. La persistencia total normalizada sigue mostrando un comportamiento similar en ambos modelos, siendo algo inferior en la salida del modelo regularizado (Figura 10.17b).



- (a) Persistencia total según el porcentaje de avance en la red para el modelo EfficientNet-Bo refinado para la especificidad Marca.  
(b) Persistencia total normalizada según el porcentaje de avance en la red para el modelo EfficientNet-Bo refinado para Marca.

Figura 10.17.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para EfficientNet-Bo refinado con SGD y un tamaño de lote 64 en función del porcentaje de avance de los datos a través de la red para las especificidad Marca con y sin regularizador.

El ejemplo que se ve en la Figura 10.18 muestra la diferencia entre las activaciones en ambas versiones. En la versión regularizada desaparecen ciertas activaciones pequeñas para centrarse principalmente en el logo del coche.



(a) Ejemplo mal clasificado para el modelo EfficientNet-Bo refinado para la especificidad Marca.

(b) Ejemplo bien clasificado para el modelo EfficientNet-Bo refinado con regularización para Marca.

Figura 10.18.: Comparación de clasificación para el caso sin regularizar (a) frente al regularizado (b) para EfficientNet-Bo.

Es interesante ver cómo pese a la gran similaridad que presentan ambas alternativas en sus código de barras, pequeñas diferencias logran marcar la diferencia ([Figura 10.19](#)). Estos detalles son más notables al 100 % de la ejecución, donde el modelo regularizado topológicamente muestra intervalos de persistencia de dimensión 1 más cortos.

**Especificidad Marca-Modelo: DenseNet-121** Las observaciones recién realizadas sobre los valores de las métricas se repiten con mayor fuerza para DenseNet-121 en Marca-Modelo ([Tabla 10.10](#)). Esta aportación, para el caso donde  $\alpha = 0.1$ , lleva a una mejora por encima del 2 % en la exactitud del modelo, mientras que F1-Score llega a aumentar hasta un 3 % sobre el modelo base. No sólo eso, sino que estos valores superan a los obtenidos a los del modelo refinado sin el término de regularización, mostrando su eficacia.

Modelo	$\alpha$	Exactitud	Precisión	Sensibilidad	F1-Score
DenseNet-121 Base	-	0.9185	0.9077	0.9126	0.9101
DenseNet-121 Refinado	0.0	0.9333	0.9220	0.9286	0.9253
DenseNet-121 Refinado	0.001	0.9333	0.9220	0.9286	0.9253
DenseNet-121 Refinado	0.005	0.9370	0.9269	0.9336	0.9302
DenseNet-121 Refinado	0.01	0.9333	0.9220	0.9286	0.9253
DenseNet-121 Refinado	0.05	0.9370	0.9284	0.9334	0.9308
DenseNet-121 Refinado	0.1	<b>0.9407</b>	<b>0.9384</b>	<b>0.9414</b>	<b>0.9399</b>
DenseNet-121 Refinado	0.5	0.9370	0.9297	0.9330	0.9313
DenseNet-121 Refinado	1.0	0.9000	0.8997	0.8945	0.8970

Tabla 10.10.: Comparación de métricas tras un proceso de refinamiento en la misma especificidad Marca-Modelos para distintos valores de  $\alpha$  en el término de regularización del modelo DenseNet-121.

## 10. Resultados experimentales

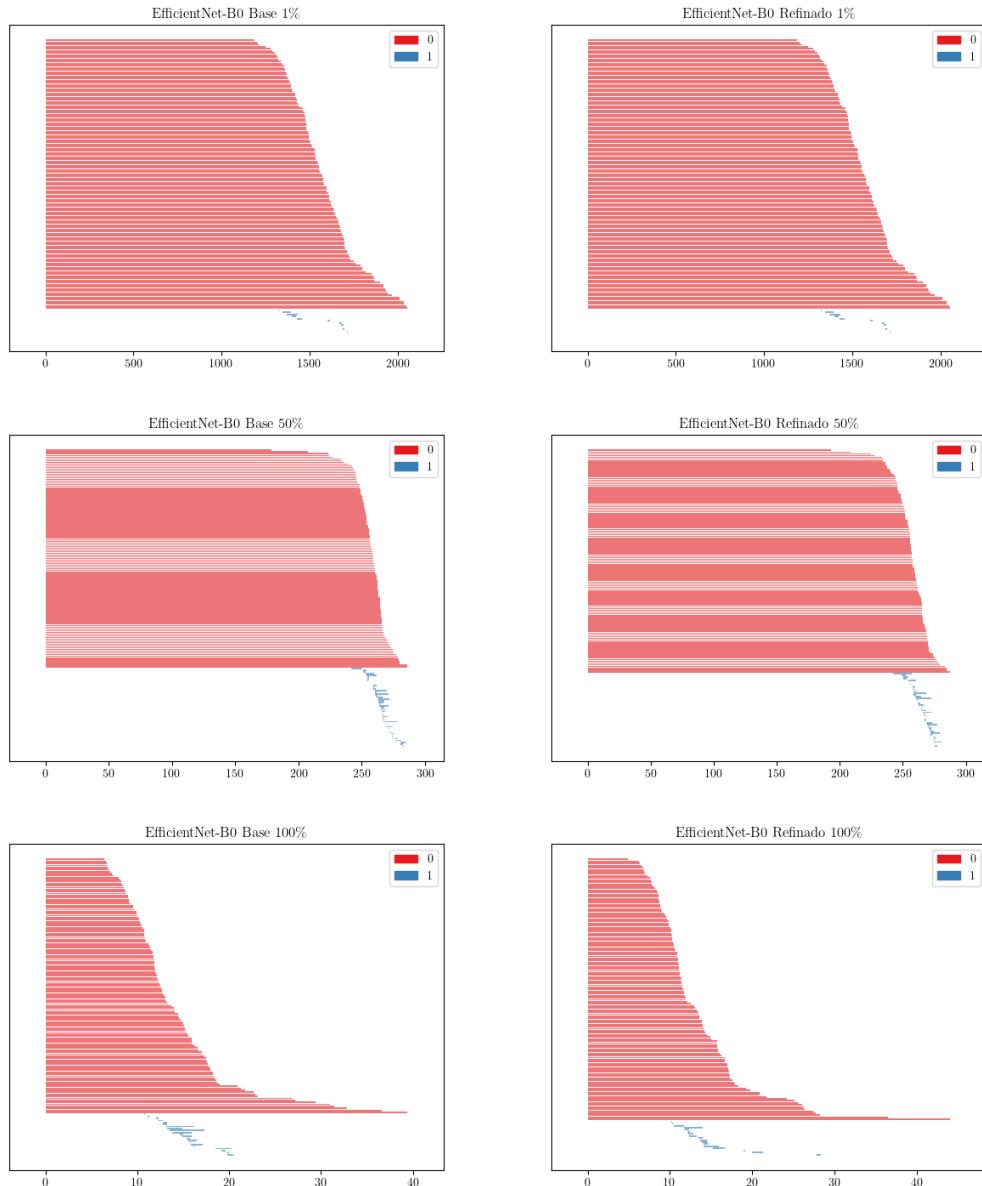
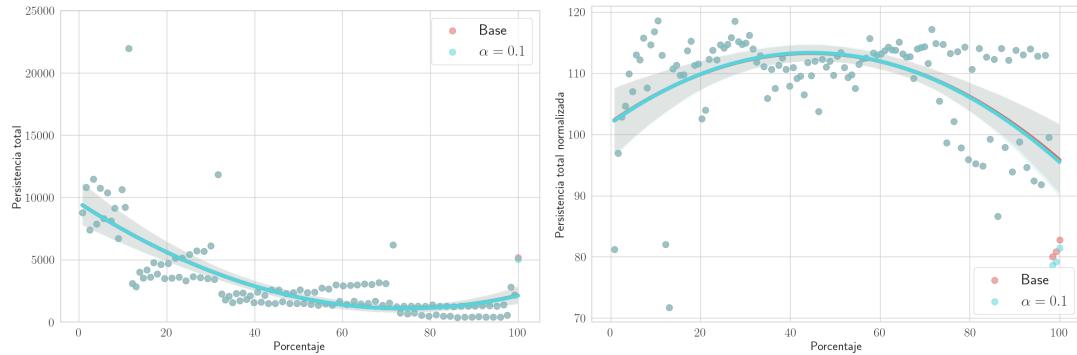


Figura 10.19.: Comparación de los códigos de barras del modelo EfficientNet-Bo para el refinamiento en la especificidad Marca sin regularización (izquierda) frente al modelo refinado con regularización (derecha). La figura muestra los códigos de barras para el conjunto de test tras avanzar un 1 % (arriba), un 50 % (centro) y el 100 % (abajo) de la red.

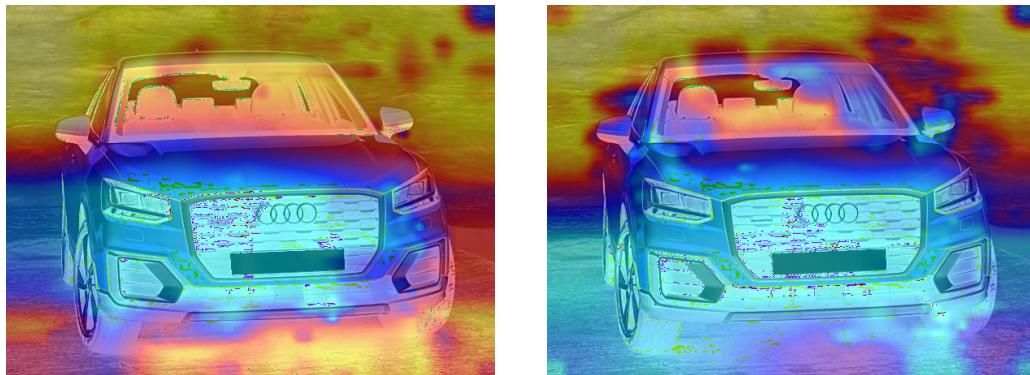
Las gráficas de persistencia ([Figura 10.20](#)) apenas muestran diferencias para ambas versiones, con la excepción de la [Figura 10.20b](#), donde el modelo regularizado muestra al final del proceso una persistencia total normalizada ligeramente inferior. Este hecho es especialmente curioso si observamos los códigos de barras obtenidos en la [Figura 10.22](#). Si bien los inter-

valos de persistencia son similares a comienzos y mediados de la inferencia, al final de la red observamos un menor número de componentes conexas persistentes y más duraderas. Es más, observamos también que aumentan el número y la longitud de clases de homología persistente de dimensión 1, lo que choca con las hipótesis planteadas. Un motivo podría deberse a que la importante reducción en el número de componentes conexas persistentes haya beneficiado más a la reducción de persistencia total normalizada, de forma que el sacrificio en homología persistente en dimensión 1 siga siendo una mejor opción en el cómputo total.



(a) Persistencia total normalizada según el porcentaje de avance en la red para el modelo DenseNet-121 refinado para Marca-Modelo.  
 (b) Persistencia total normalizada según el porcentaje de avance en la red para el modelo DenseNet-121 refinado para Marca-Modelo.

Figura 10.20.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para DenseNet-121 refinado con SGD y un tamaño de lote 32 en función del porcentaje de avance de los datos a través de la red para la especificidad Marca-Modelo con y sin regularizador.



(a) Ejemplo mal clasificado para el modelo DenseNet-121 refinado para la especificidad Marca-Modelo.

(b) Ejemplo bien clasificado para el modelo DenseNet-121 refinado con regularización para Marca-Modelo.

Figura 10.21.: Comparación de clasificación para el caso sin regularizar (a) frente al regularizado (b) para DenseNet-121 para la especificidad Marca-Modelo.

Esta vez, al tratar con una especificidad más fina, el logotipo de la marca no parece ser suficiente criterio para detectar también el modelo, ampliando el área de estudio (Figura 10.21).

## 10. Resultados experimentales

Al emplear el regularizador topológico vemos que el modelo se centra más en otros aspectos como los retrovisores o el techo del coche frente a la versión sin regularizar.

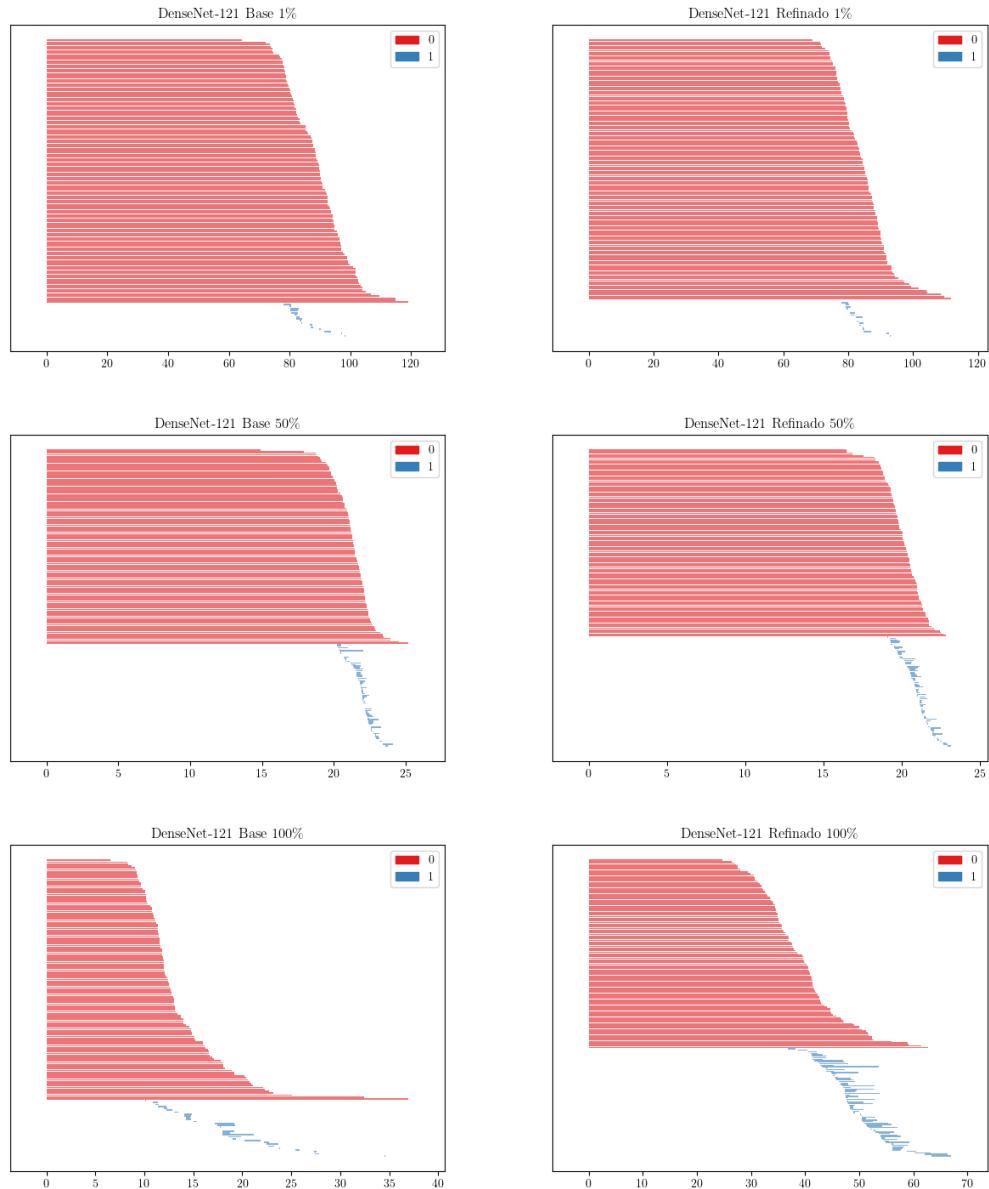


Figura 10.22.: Comparación de los códigos de barras del modelo DenseNet-121 para el refinamiento en la especificidad Marca-Modelo sin regularización (izquierda) frente al modelo refinado con regularización (derecha). La figura muestra los códigos de barras para el conjunto de test tras avanzar un 1 % (arriba), un 50 % (centro) y el 100 % (abajo) de la red.

### 10.3.2. Mejora en transferibilidad

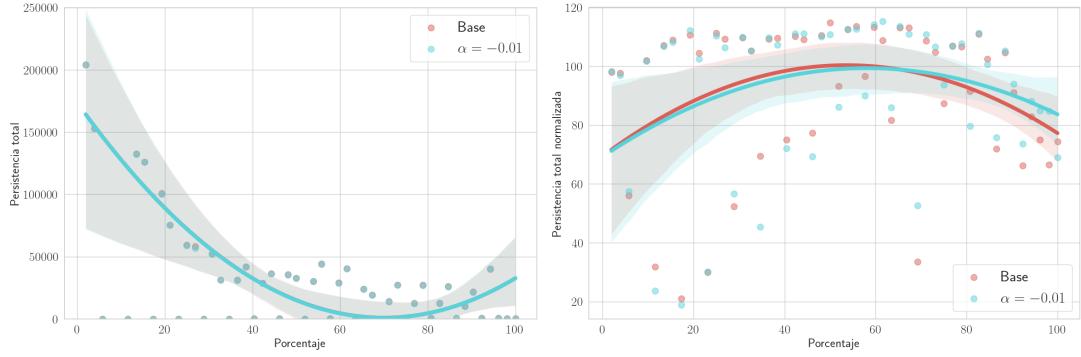
**Transferencia EfficientNet-BO: Marca → Marca-Modelo** La Tabla 10.11 muestra resultados interesantes en relación a cómo la topología afecta a la transferencia de conocimiento. Al aumentar la granularidad del conjunto de clases y la acción del regularizador topológico para aumentar la persistencia total normalizada de los datos, vemos que la capacidad de clasificación del modelo mejora notablemente frente a la alternativa sin regularizar. En el caso donde el valor  $\alpha = -0.01$ , el modelo llega a mejorar la exactitud del modelo más de un 4 % y el F1-Score más de un 5 % por encima de la transferencia base.

Modelo	$\alpha$	Exactitud	Precisión	Sensibilidad	F1-Score
EfficientNet-Bo Fine-Tune	0.0	0.8333	0.7724	0.8053	0.7884
EfficientNet-Bo Fine-Tune	-0.001	0.8407	0.7882	0.8162	0.8018
EfficientNet-Bo Fine-Tune	-0.005	0.8407	0.7959	0.8193	0.8074
EfficientNet-Bo Fine-Tune	-0.01	<b>0.8741</b>	<b>0.8276</b>	<b>0.8522</b>	<b>0.8397</b>
EfficientNet-Bo Fine-Tune	-0.05	0.7963	0.7379	0.7624	0.7499
EfficientNet-Bo Fine-Tune	-0.1	0.6000	0.5200	0.5631	0.5405
EfficientNet-Bo Fine-Tune	-0.5	0.0222	0.0006	0.0218	0.0011
EfficientNet-Bo Fine-Tune	-1.0	0.0148	0.0010	0.0103	0.0016

Tabla 10.11.: Comparación de métricas tras un proceso de *fine-tuning* del modelo EfficientNet-Bo desde la especificidad Marca hacia la especificidad Marca-Modelo para distintos valores de  $\alpha$  en el término de regularización.

Los resultados de las métricas se ven reforzados por las tendencias en la Figura 10.23. Aunque apenas veamos diferencias en la evolución de la persistencia total (Figura 10.23a), la persistencia total normalizada muestra unos valores mayores en la parte final respecto a la transferencia base (Figura 10.23b). Este hecho es resultado del esfuerzo en aumentar la complejidad topológica, mostrando una relación directa entre una clasificación más fina y la topología de los datos. Una diferencia que vemos respecto a los resultados obtenidos en la Figura 10.13b es la ausencia de una mayor complejidad en otras etapas de la red. Al estar considerando solamente la salida final de la red en el regularizador, el optimizador solo parece centrarse en complicar la topología de dichas salidas. En consecuencia, podría ser interesante plantear este concepto con activaciones de varias capas de la red.

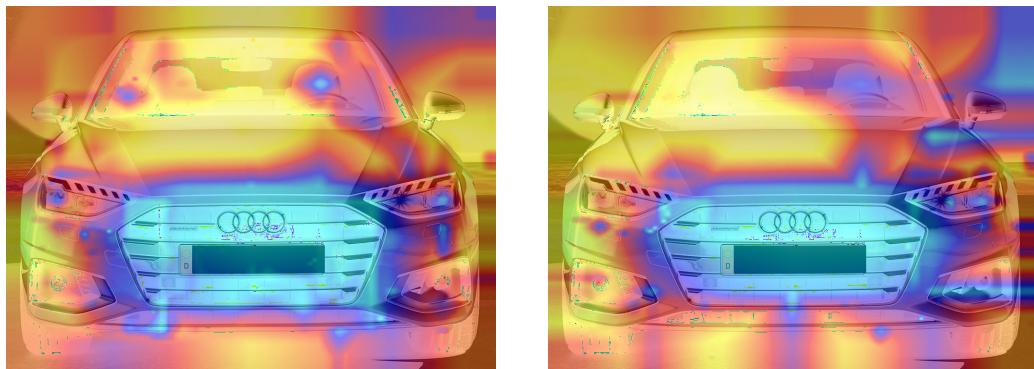
## 10. Resultados experimentales



(a) Persistencia total normalizada según el porcentaje de avance en la red para el modelo EfficientNet-Bo transferido a Marca-Modelo. (b) Persistencia total normalizada según el porcentaje de avance en la red para el modelo EfficientNet-Bo transferido a Marca-Modelo.

Figura 10.23.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para EfficientNet-Bo transferido con SGD y un tamaño de lote 32 en función del porcentaje de avance de los datos a través de la red desde la especificidad Marca a Marca-Modelo.

Si observamos la Figura 10.24, vemos que la transferencia hacia el mismo *dataset* con una clasificación más fina, el modelo necesita ampliar el área donde se enfoca. En particular, vemos que el modelo regularizado logra tener un enfoque más homogéneo en la parrilla del vehículo a cambio de introducir ciertos artefactos lineales.



(a) Ejemplo mal clasificado para el modelo EfficientNet-Bo transferido desde la especificidad Marca a Marca-Modelo. (b) Ejemplo bien clasificado para el modelo EfficientNet-Bo transferido desde la especificidad Marca a Marca-Modelo.

Figura 10.24.: Comparación de clasificación para el caso sin regularizar (a) frente al regularizado (b) para la transferencia de EfficientNet-Bo a Marca-Modelo.

Como consecuencia directa de aumentar la complejidad topológica de los datos, los códigos de barras en el final de la red regularizada muestra intervalos algo más numerosos y de longitud similar, de forma que disponemos de una mayor persistencia total normalizada (Figura 10.25).

### 10.3. Propuesta de mejora: regularización topológica

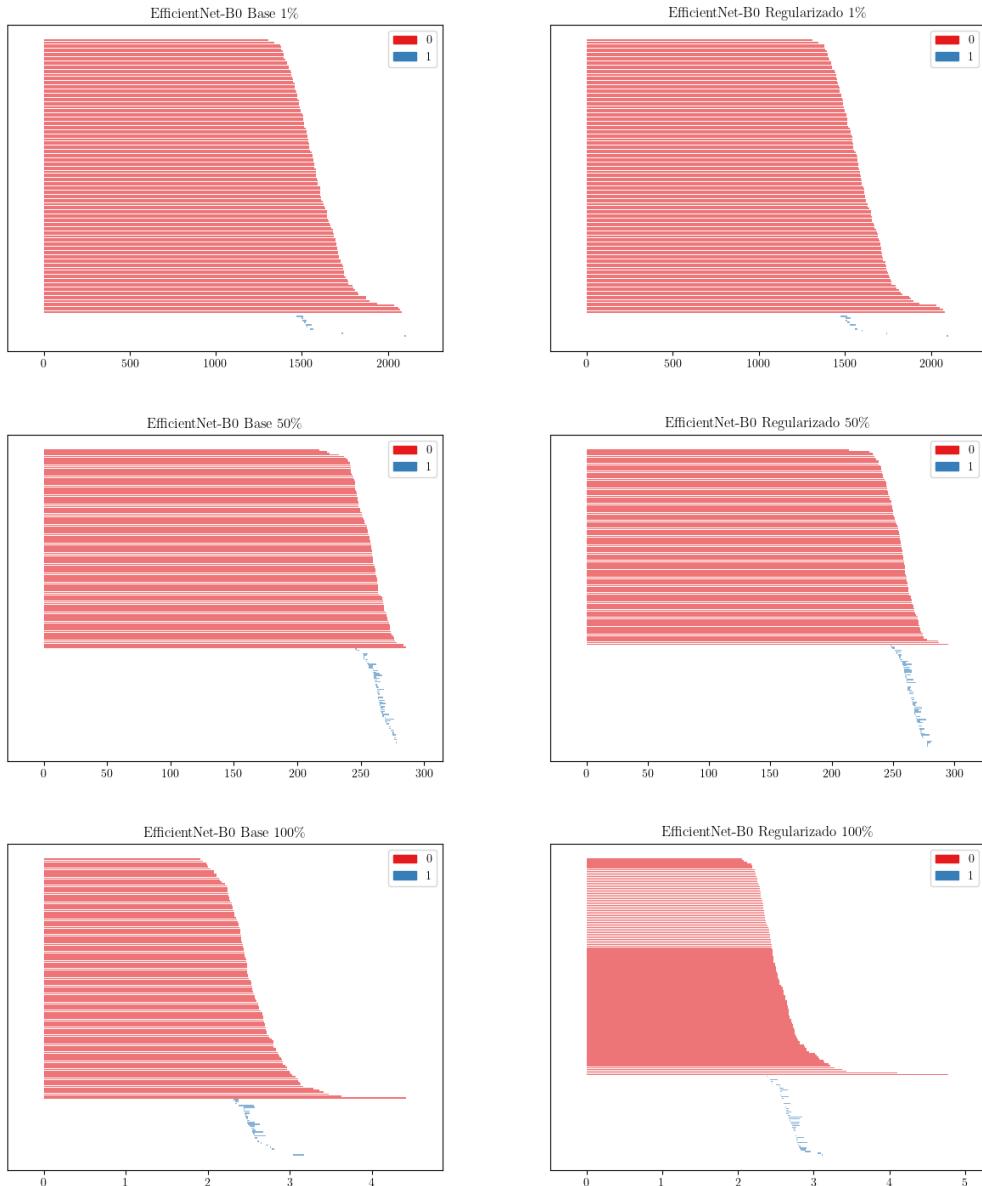


Figura 10.25.: Comparación de los códigos de barras del modelo EfficientNet-Bo para la transferencia de la especificidad Marca a Marca-Modelo sin regularización (izquierda) frente al modelo transferido con regularización (derecha). La figura muestra los códigos de barras para el conjunto de test tras avanzar un 1% (arriba), un 50 % (centro) y el 100 % (abajo) de la red.

**Transferencia Densenet-121: Marca-Modelo → Marca** Por su parte, la transferencia desde Marca-Modelo a Marca ha mostrado métricas mucho más altas que las vistas en la transferencia opuesta (Tabla 10.12). A pesar de la similitud entre los resultados obtenidos, vemos que los mejores resultados los han obtenido las variantes para los valores de  $\alpha$  iguales a 0.001 y 0.005.

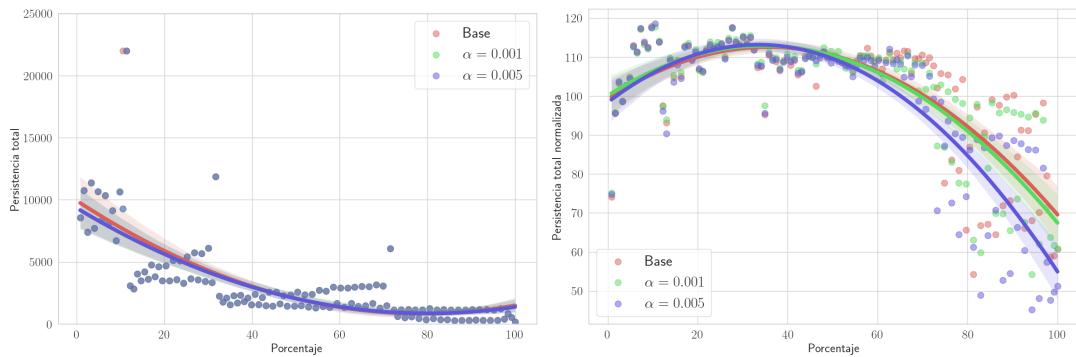
## 10. Resultados experimentales

Más aún, es interesante ver cómo alteraciones similares de la topología de los datos pueden llevar los resultados obtenidos a los mismos valores.

Modelo	$\alpha$	Exactitud	Precisión	Sensibilidad	F1-Score
DenseNet-121 Fine-Tune	0.0	0.9907	0.9816	0.9864	0.9840
DenseNet-121 Fine-Tune	0.001	<b>0.9938</b>	<b>0.9834</b>	<b>0.9882</b>	<b>0.9857</b>
DenseNet-121 Fine-Tune	0.005	<b>0.9938</b>	<b>0.9834</b>	<b>0.9882</b>	<b>0.9857</b>
DenseNet-121 Fine-Tune	0.01	0.9876	0.9747	0.9812	0.9779
DenseNet-121 Fine-Tune	0.05	0.9443	0.8895	0.9077	0.8983
DenseNet-121 Fine-Tune	0.1	0.8111	0.6628	0.7252	0.6919
DenseNet-121 Fine-Tune	0.5	0.0194	0.0738	0.0738	0.0287
DenseNet-121 Fine-Tune	1.0	0.0743	0.0048	0.0546	0.0086

Tabla 10.12.: Comparación de métricas tras un proceso de *fine-tuning* del modelo DenseNet-121 desde la especificidad Marca-Modelo hacia la especificidad Marca para distintos valores de  $\alpha$  en el término de regularización.

Pese a la poca diferencia entre métricas, los cambios en homología persistente muestran varios aspectos de interés. En esta ocasión, la simplificación de la persistencia total normalizada se ha reflejado en la persistencia total a través de una simplificación al comienzo de la red, tal y como muestra la Figura 10.26a. Por otra parte, la Figura 10.26b muestra claramente como el regularizador ha funcionado como se esperaba. Es curioso ver cómo para los casos donde  $\alpha = 0.001$  y  $\alpha = 0.005$ , cuyas métricas son idénticas, obtenemos diferencias en complejidad topológica tan notables.



- (a) Persistencia total según el porcentaje de avance en la red para el modelo DenseNet-121 transferido desde la especificidad Marca-Modelo a Marca.  
(b) Persistencia total normalizada según el porcentaje de avance en la red para el modelo DenseNet-121 transferido desde la especificidad Marca-Modelo a Marca.

Figura 10.26.: Comparación de la persistencia total (a) y la persistencia total normalizada (b) para DenseNet-121 transferido con SGD y un tamaño de lote 32 en función del porcentaje de avance de los datos a través de la red desde la especificidad Marca-Modelo a Marca.

Finalmente, los códigos de barras en la Figura 10.27 reflejan el comportamiento esperado en las capas finales, donde el modelo regularizado muestra intervalos de persistencia más cortos que su par (Figura 10.27). Pese al aumento de persistencia total en las capas iniciales, los códigos de barras no parecen dar una respuesta a esta cuestión.

### 10.3. Propuesta de mejora: regularización topológica

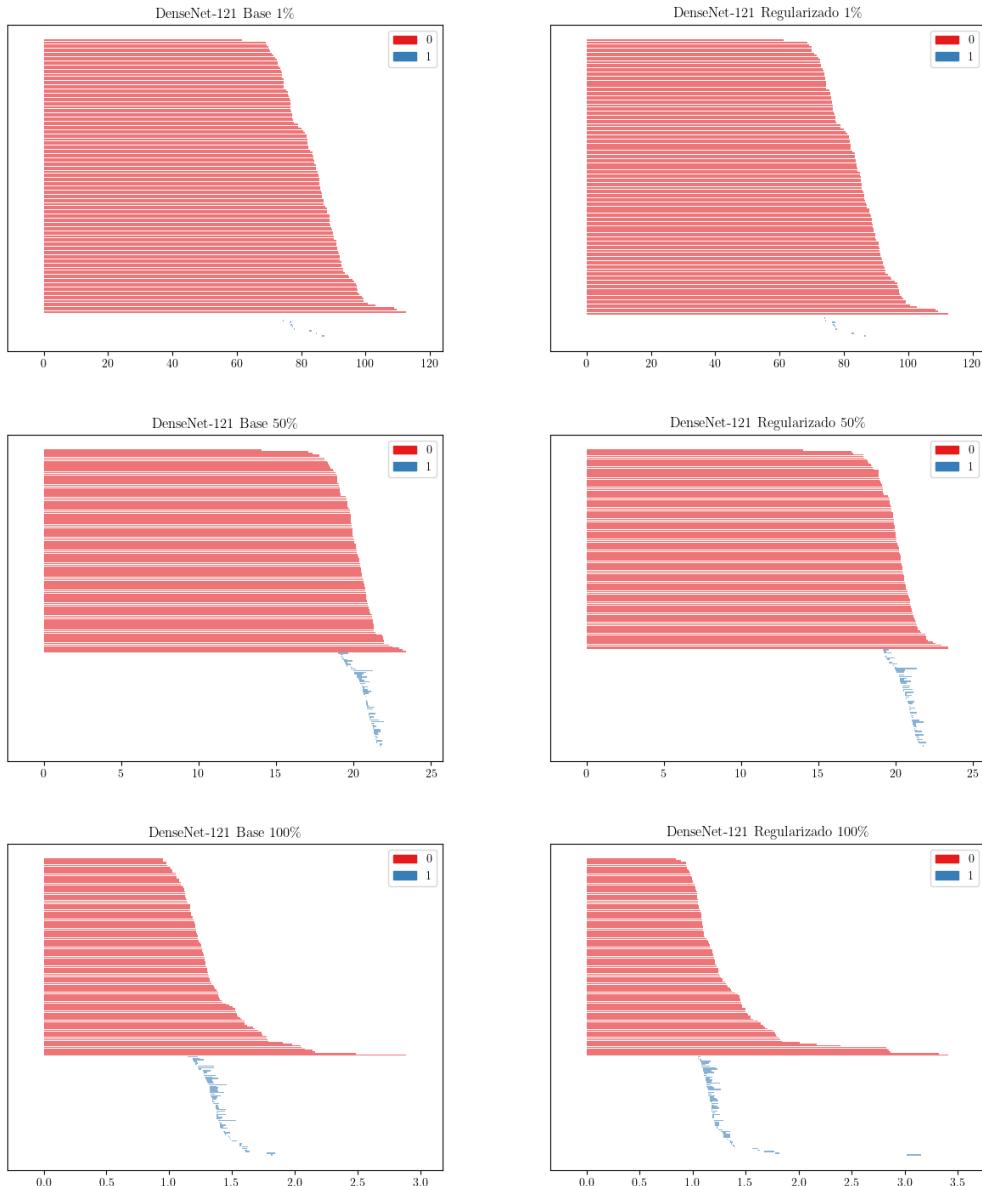
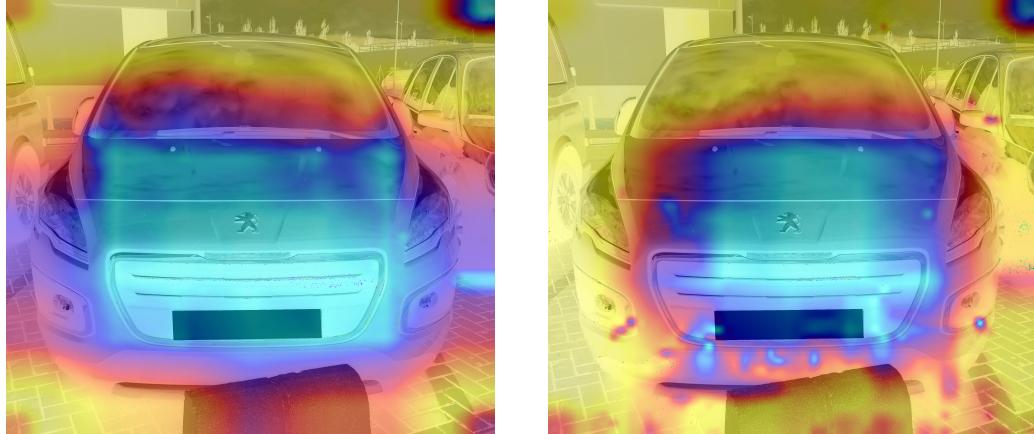


Figura 10.27.: Comparación de los códigos de barras del modelo DenseNet-121 para la transferencia de la especificidad Marca-Modelo a Marca sin regularización (izquierda) frente al modelo con regularización (derecha). La figura muestra los códigos de barras para el conjunto de test tras avanzar un 1 % (arriba), un 50 % (centro) y el 100 % (derecha) de la red.

A diferencia de la transferencia anterior, al reducir el número de clases vemos que el modelo focaliza su análisis en un área más concentrada (Figura 10.28). La Figura 10.28b muestra un área de enfoque más reducida que el caso sin regularizar, centrándose en las características más relevantes y evitando otras que podrían introducir ruido.



(a) Ejemplo mal clasificado para el modelo DenseNet-121 transferido desde la especificidad Marca-Modelo a Marca.

(b) Ejemplo bien clasificado para el modelo DenseNet-121 transferido desde la especificidad Marca-Modelo a Marca.

Figura 10.28.: Comparación de clasificación para el caso sin regularizar (a) frente al regularizado (b) para la transferencia de DenseNet-121 a Marca.

### 10.3.3. Impacto de la regularización topológica en CNNs

La implementación de un regularizador topológico en modelos como EfficientNet-Bo y DenseNet-121 ha demostrado ser prometedora, mejorando tanto la clasificación como la transferencia de conocimiento. Este enfoque ajusta la complejidad topológica de los datos, lo que parece reflejarse en la mejora general del rendimiento del modelo.

Para el modelo EfficientNet-Bo, el uso del regularizador topológico ha mostrado una mejora leve pero significativa en comparación con versiones sin regularización. Las observaciones realizadas indican que ajustar la topología, principalmente en etapas finales del aprendizaje, puede afinar la capacidad del modelo para diferenciar entre clases y por tanto haciendo más efectiva la clasificación final. Esto se ve en cómo el modelo regularizado presenta intervalos más cortos de persistencia en instantes más cercanos al final del proceso de inferencia.

En el caso de DenseNet-121, la aplicación del regularizador ha tenido un impacto considerable, optimizando de manera efectiva la estructura de los datos para mejorar el rendimiento del modelo. Los cambios sutiles en la homología al final de la red han demostrado ser importantes, reduciendo la cantidad de componentes conexas y aumentando la duración de las clases de homología persistente de dimensiones superiores, lo que sugiere una manipulación estratégica de la topología para beneficiar la tarea de clasificación.

En términos de transferibilidad, los estudios sobre EfficientNet-Bo y DenseNet-121 muestran que el regularizador topológico no solo mejora la clasificación en el dominio original, sino que también facilita la transferencia de conocimiento al menos hacia dominios de diferente granularidad. Esto se muestra en la mejora de la clasificación cuando se aumenta la granularidad de las clases, y cómo la complejidad topológica ajustada al final del proceso contribuye de manera notable a estos resultados. La relación entre una clasificación más precisa y la topología de los datos sugiere que enfocarse en optimizar la topología en las salidas finales puede ser una estrategia eficaz.

Sin embargo, la selección del valor de  $\alpha$  para el regularizador ha mostrado ser relevante y debe hacerse con cuidado. Un valor inadecuado puede llevar a resultados subóptimos,

### *10.3. Propuesta de mejora: regularización topológica*

indicando la sensibilidad de la técnica y la necesidad de entender profundamente cómo las modificaciones topológicas influencian el aprendizaje. Por este motivo, el estudio realizado sugiere escoger valores de  $\alpha$  entre 0.001 y 0.1.



**Parte IV.**

**Conclusiones y trabajo futuro**



## 11. Conclusión

Concluimos este trabajo habiendo alcanzado los objetivos que nos propusimos al inicio. Comenzamos revisando las herramientas algebraicas necesarias para comprender la homología simplicial, estudiando algunos de los principales resultados de este campo del conocimiento. Esto nos permitió establecer una base sólida desde la cual explorar en detalle la homología persistente.

Seguidamente, nos adentramos en el ámbito del aprendizaje profundo, con un enfoque particular en su aplicación a problemas de clasificación de imágenes. Analizamos las CNNs y las arquitecturas más empleadas en la actualidad, dando un repaso histórico de los hitos que nos han llevado hasta aquí. Esta exploración no solo reforzó nuestra comprensión de las técnicas modernas de aprendizaje automático, sino que también nos preparó para entender cómo estas tecnologías podrían estudiarse desde el punto de vista de la topología.

Habiendo establecido un marco teórico y técnico robusto, investigamos cómo el TDA, con un enfoque en la homología persistente, se integra y mejora las capacidades de las CNNs. Este minucioso análisis nos permitió ahondar en la comprensión de las CNNs, revelando nuevas perspectivas y metodologías de utilidad para el estudio de los modelos de aprendizaje profundo.

El núcleo de este trabajo se centró en la implementación y evaluación de un regularizador topológico diseñado para ajustar la complejidad topológica de los datos procesados por las CNNs. Los resultados obtenidos demostraron que la inclusión de este regularizador no solo mejora la precisión de la clasificación, sino que también potencia la capacidad de transferencia de conocimiento entre diferentes dominios de datos. Este hallazgo destaca la utilidad del TDA como un puente entre la topología algebraica y las aplicaciones prácticas en machine learning y visión por computadora.

Además, el estudio de la transferibilidad de los modelos con regularización topológica ofreció información sobre cómo las características topológicas ajustadas pueden afectar positivamente la generalización del modelo más allá de su conjunto de entrenamiento inicial. Esto se reflejó en una mejora en la clasificación cuando los modelos se enfrentaron a nuevas etiquetas de los datos, lo cual es de gran interés para la transferencia de conocimiento con recursos limitados.

Aunque el uso de la regularización topológica ha revelado algunas limitaciones, el campo del estudio de las CNNs mediante técnicas de TDA es relativamente nuevo y está demostrando tener un gran potencial. Este trabajo ha establecido una base sólida para comprender cómo las CNNs afectan la persistencia total y, de manera más específica, la persistencia total normalizada de los datos. Aún existe un amplio margen para explorar y descubrir cómo las CNNs transforman la estructura de los datos, ofreciendo oportunidades para afinar estas técnicas y mejorar la eficacia de los modelos de aprendizaje profundo.



## 12. Trabajo futuro

Nuestro trabajo ha dado un paso más en la comprensión de arquitecturas de aprendizaje profundo como lo son las CNNs. Dentro del ámbito de estudio de estos modelos mediante TDA, nuestro trabajo ha ido más allá de los estudios previos y ha profundizado en cómo afectan distintos componentes y mecanismos en las CNNs a la topología de los datos. Este estudio en profundidad ha sido posible gracias al empleo de la persistencia total normalizada, que ha mostrado ser una métrica con una gran capacidad expresiva en la comprensión de las transformaciones topológicas que realiza una CNN sobre sus datos.

Estos factores abren una nueva vía de exploración donde poder realizar estudios más exhaustivos y a distintos niveles de detalle. Empezando por análisis a más alto nivel, podrían compararse diferentes arquitecturas y conjuntos de datos para ofrecer conclusiones más robustas sobre cómo afectan a las transformaciones que realizan. Más aún, sería interesante estudiar cómo afecta cada paso individual del modelo a la topología de los datos, más allá de considerar las salidas de las activaciones no lineales.

Por otra parte, el término de regularización topológico ha mostrado ser efectivo y potencial de mejorar tanto la tasa de clasificación del modelo como su capacidad de transferibilidad. Sin embargo, su inestabilidad en la elección del hiperparámetro empleado puede ser un problema y sería interesante encontrar mecanismos más robustos para elegir correctamente este valor. Otra observación que sería interesante de analizar es la aparición de artefactos verticales y horizontales en algunos casos tras la regularización. Además, el regularizador propuesto se ha visto bastante limitado a la homología persistente en las capas finales del modelo, por lo que sería interesante explorar mejoras que consideren otras activaciones del modelo para solucionar el problema.

Como se puede ver, existe un gran margen de mejora en el trabajo propuesto y numerosas oportunidades para arrojar luz sobre la explicabilidad de modelos de aprendizaje profundo. El TDA ha mostrado la capacidad de ofrecer información relevante sobre cómo modifican los datos las CNNs, proponiendo nuevas herramientas que nos permitan afrontar los nuevos desafíos que nos esperan.



## Bibliografía

- [1] Anaconda software distribution, 2020.
- [2] A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.
- [3] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin. *Learning from data*, volume 4. AMLBook New York, 2012.
- [4] F. Agostinelli, S. McAleer, A. Shmakov, and P. Baldi. Solving the rubik’s cube with deep reinforcement learning and search. *Nature Machine Intelligence*, 1, 08 2019.
- [5] I. Aguilera-Martos, D. López, M. García-Barzana, J. Luengo, and F. Herrera. Time-series industrial anomaly dataset, 2022.
- [6] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate, 2016.
- [7] H. Bast, D. Delling, A. Goldberg, M. Müller-Hannemann, T. Pajor, P. Sanders, D. Wagner, and R. F. Werneck. Route planning in transportation networks. *Algorithm engineering: Selected results and surveys*, pages 19–80, 2016.
- [8] Y. Bengio, I. Goodfellow, and A. Courville. *Deep learning*, volume 1. MIT press Cambridge, MA, USA, 2017.
- [9] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.
- [10] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners, 2020.
- [11] G. Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.
- [12] A. Cauchy. Méthode générale pour la résolution des systemes d’équations simultanées. *Comp. Rend. Sci. Paris*, 25(1847):536–538, 1847.
- [13] F. Chazal, V. De Silva, M. Glisse, and S. Oudot. *The structure and stability of persistence modules*, volume 10. Springer, 2016.
- [14] F. Chazal and B. Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *Frontiers in artificial intelligence*, 4:108, 2021.
- [15] F. Chazal and B. Michel. An introduction to topological data analysis: Fundamental and practical aspects for data scientists. *Frontiers in Artificial Intelligence*, 4, 2021.
- [16] G. Chen, H. Yu, W. Dong, X. Sheng, X. Zhu, and H. Ding. Learning to navigate from simulation via spatial and semantic information synthesis with noise model embedding, 2019.
- [17] M. T. Chiu, X. Xu, Y. Wei, Z. Huang, A. G. Schwing, R. Brunner, H. Khachatrian, H. Karapetyan, I. Dozier, G. Rose, et al. Agriculture-vision: A large aerial image database for agricultural pattern analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2828–2838, 2020.
- [18] S. Chowdhury and F. Mémoli. Persistent path homology of directed networks. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1152–1169. SIAM, 2018.

## Bibliografía

- [19] S. Coary and M. Poor. How consumer-generated images shape important consumption outcomes in the food domain. *Journal of Consumer Marketing*, 33(1):1–8, 2016.
- [20] U. Cohen, S. Chung, D. D. Lee, and H. Sompolinsky. Separability and geometry of object manifolds in deep neural networks. *Nature communications*, 11(1):746, 2020.
- [21] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pages 263–271, 2005.
- [22] R. Corbet and M. Kerber. The representation theorem of persistence revisited and generalized. *Journal of Applied and Computational Topology*, 2(1–2):1–31, July 2018.
- [23] E. G. Dada, J. S. Bassi, H. Chiroma, A. O. Adetunmbi, O. E. Ajibawa, et al. Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, 5(6), 2019.
- [24] N. C. Dang, M. N. Moreno-García, and F. De la Prieta. Sentiment analysis based on deep learning: A comparative study. *Electronics*, 9(3):483, 2020.
- [25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.
- [26] Y. Deng, Z. Liu, and J. G. Korvink. Topology optimization on two-dimensional manifolds. *Computer Methods in Applied Mechanics and Engineering*, 364:112937, 2020.
- [27] T. K. Dey and Y. Wang. *Computational topology for data analysis*. Cambridge University Press, 2022.
- [28] V. Divol and W. Polonik. On the choice of weight functions for linear representations of persistence diagrams. *Journal of Applied and Computational Topology*, 3, 09 2019.
- [29] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.
- [30] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.
- [31] D. S. Dummit and R. M. Foote. *Abstract algebra*, volume 3. Wiley Hoboken, 2004.
- [32] V. Dumoulin and F. Visin. A guide to convolution arithmetic for deep learning. *ArXiv*, abs/1603.07285, 2016.
- [33] Edelsbrunner, Letscher, and Zomorodian. Topological persistence and simplification. *Discrete & computational geometry*, 28:511–533, 2002.
- [34] S. Eilenberg and S. MacLane. General theory of natural equivalences. *Transactions of the American Mathematical Society*, 58:231–294, 1945.
- [35] C. Fefferman, S. Mitter, and H. Narayanan. Testing the manifold hypothesis, 2013.
- [36] F.-G. Fernandez. Torchcam: class activation explorer. <https://github.com/frgfm/torch-cam>, March 2020.
- [37] P. Frosini. A distance for similarity classes of submanifolds of a euclidean space. *Bulletin of the Australian Mathematical Society*, 42(3):407–415, 1990.
- [38] Z. Fu, T. Z. Zhao, and C. Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation, 2024.
- [39] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.
- [40] A. Garin and G. Tauzin. A topologicalreading"lesson: Classification of mnist using tda. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1551–1556. IEEE, 2019.

- [41] Z. Ghahramani. Unsupervised learning. In *Summer school on machine learning*, pages 72–112. Springer, 2003.
- [42] J. C. Giarratano and G. Riley. *Expert systems: principles and programming*. Brooks/Cole Publishing Co., 1989.
- [43] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. Fernández del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585:357–362, 2020.
- [44] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.
- [45] A. Hatcher. *Algebraic Topology*. Algebraic Topology. Cambridge University Press, 2002.
- [46] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin, and H. Larochelle. Brain tumor segmentation with deep neural networks. *Medical Image Analysis*, 35:18–31, Jan. 2017.
- [47] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [48] G. Hinton, N. Srivastava, and K. Swersky. Lecture 6a overview of mini-batch gradient descent. *Coursera Lecture slides* <https://class.coursera.org/neuralnets-2012-001/lecture>, [Online], 2012.
- [49] M. S. Hossain, M. Al-Hammadi, and G. Muhammad. Automatic fruit classification using deep learning for industrial applications. *IEEE transactions on industrial informatics*, 15(2):1027–1034, 2018.
- [50] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.
- [51] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.
- [52] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90–95, 2007.
- [53] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.
- [54] C. M. Judd, G. H. McClelland, and C. S. Ryan. *Data analysis: A model comparison approach to regression, ANOVA, and beyond*. Routledge, 2017.
- [55] A. Kannan, P. P. Talukdar, N. Rasiwasia, and Q. Ke. Improving product classification using images. In *2011 IEEE 11th International Conference on Data Mining*, pages 310–319. IEEE, 2011.
- [56] J. Kiefer and J. Wolfowitz. Stochastic Estimation of the Maximum of a Regression Function. *The Annals of Mathematical Statistics*, 23(3):462 – 466, 1952.
- [57] S.-Y. Kim, T.-S. Jung, E.-H. Suh, and H.-S. Hwang. Customer segmentation and strategy development based on customer lifetime value: A case study. *Expert systems with applications*, 31(1):101–107, 2006.
- [58] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [59] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
- [60] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

## Bibliografía

- [61] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [62] J. Lee. *Introduction to topological manifolds*, volume 202. Springer Science & Business Media, 2010.
- [63] R. Lewis and C. Séquin. Generation of 3d building models from 2d architectural plans. *Computer-Aided Design*, 30(10):765–779, 1998.
- [64] J. Leygonie, S. Oudot, and U. Tillmann. A framework for differential calculus on persistence barcodes, 2021.
- [65] Z. Li, T. Müller, A. Evans, R. H. Taylor, M. Unberath, M.-Y. Liu, and C.-H. Lin. Neuralangelo: High-fidelity neural surface reconstruction, 2023.
- [66] S. Mac Lane. *Categories for the working mathematician*, volume 5. Springer Science & Business Media, 2013.
- [67] S. MacLane. *Homology*. Springer Science & Business Media, 2012.
- [68] G. Magai. Deep neural networks architectures from the perspective of manifold learning. In *2023 IEEE 6th International Conference on Pattern Recognition and Artificial Intelligence (PRAI)*, pages 1021–1031. IEEE, 2023.
- [69] W. McKinney et al. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX, 2010.
- [70] N. Morgan and H. Bourlard. Generalization and parameter estimation in feedforward nets: Some experiments. *Advances in neural information processing systems*, 2, 1989.
- [71] J. R. Munkres. *Elements of algebraic topology*. CRC press, 2018.
- [72] G. Naitzat, A. Zhitnikov, and L.-H. Lim. Topology of deep neural networks. *Journal of Machine Learning Research*, 21(184):1–40, 2020.
- [73] S. Nashif, M. R. Raihan, M. R. Islam, and M. H. Imam. Heart disease detection by using machine learning algorithms and a real-time cardiovascular health monitoring system. *World Journal of Engineering and Technology*, 6(4):854–873, 2018.
- [74] A. Y. Ng. Feature selection,  $L_1$  vs.  $L_2$  regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78, 2004.
- [75] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6(1), Aug. 2017.
- [76] P. O’Donovan, A. Agarwala, and A. Hertzmann. Learning layouts for single-page geographic designs. *IEEE Transactions on Visualization and Computer Graphics*, 20(8):1200–1213, 2014.
- [77] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- [78] F. Pérez-Hernández, S. Tabik, A. Lamas, R. Olmos, H. Fujita, and F. Herrera. Object detection binary classifiers methodology based on deep learning to identify small objects handled similarly: Application in video surveillance. *Knowledge-Based Systems*, 194:105590, 2020.
- [79] H. Pham, Z. Dai, Q. Xie, M.-T. Luong, and Q. V. Le. Meta pseudo labels, 2021.
- [80] C. Phua, V. Lee, K. Smith, and R. Gayler. A comprehensive survey of data mining-based fraud detection research. *arXiv preprint arXiv:1009.6119*, 2010.
- [81] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.
- [82] A. Rafael Ayala. *Elementos de la teoría de homología clásica*. Serie Ciencias / Universidad de Sevilla. Secretariado de Publicaciones, Universidad de Sevilla, 2003.

- [83] P. Ramachandran, B. Zoph, and Q. V. Le. Searching for activation functions, 2017.
- [84] V. Robins. Towards computing homology from finite approximations. In *Topology proceedings*, volume 24, pages 503–532, 1999.
- [85] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.
- [86] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.
- [87] M. Rucco, F. Castiglione, E. Merelli, and M. Pettini. Characterisation of the idiotypic immune network through persistent entropy. In *Proceedings of ECCS 2014: European conference on complex systems*, pages 117–128. Springer, 2016.
- [88] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.
- [89] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, England, third edition, 2016.
- [90] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2015.
- [91] P. Schwarz and A. Hein. Conception of a humanoid-robot-patient in education to train and practice. In *2023 IEEE 2nd German Education Conference (GECon)*, pages 1–5, 2023.
- [92] B. Schweinhart. Fractal dimension and the persistent homology of random geometric complexes, 2020.
- [93] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, Oct. 2019.
- [94] T. Serre, A. Oliva, and T. Poggio. A feedforward architecture accounts for rapid categorization. *Proceedings of the national academy of sciences*, 104(15):6424–6429, 2007.
- [95] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm, 2017.
- [96] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [97] K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfahl, P. Payne, M. Seneviratne, P. Gamble, C. Kelly, N. Scharli, A. Chowdhery, P. Mansfield, B. A. y Arcas, D. Webster, G. S. Corrado, Y. Matias, K. Chou, J. Gottweis, N. Tomasev, Y. Liu, A. Rajkomar, J. Barral, C. Semturs, A. Karthikesalingam, and V. Natarajan. Large language models encode clinical knowledge, 2022.
- [98] L. N. Smith. Cyclical learning rates for training neural networks, 2017.
- [99] N. Snavely. Scene reconstruction and visualization from internet photo collections: A survey. *IPSJ Transactions on Computer Vision and Applications*, 3:44–66, 2011.
- [100] H. SrivastavaN, A. Krizhevsky, et al. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929, 2014.
- [101] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [102] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
- [103] R. Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022.

## Bibliografía

- [104] M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.
- [105] M. Tan and Q. V. Le. Efficientnetv2: Smaller models and faster training, 2021.
- [106] H. Tang, Y. Chen, T. Wang, Y. Zhou, L. Zhao, Q. Gao, M. Du, T. Tan, X. Zhang, and T. Tong. Htc-net: A hybrid cnn-transformer framework for medical image segmentation. *Biomedical Signal Processing and Control*, 88:105605, 2024.
- [107] G. Tauzin, U. Lupo, L. Tunstall, J. B. Pérez, M. Caorsi, A. M. Medina-Mardones, A. Dassatti, and K. Hess. giotto-tda: A topological data analysis toolkit for machine learning and data exploration. *Journal of Machine Learning Research*, 22(39):1–6, 2021.
- [108] M. J. Thurley. Automated image segmentation and analysis of rock piles in an open-pit mine. In *2013 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–8. IEEE, 2013.
- [109] C. Tralie, N. Saul, and R. Bar-On. Ripser.py: A lean persistent homology library for python. *The Journal of Open Source Software*, 3(29):925, Sep 2018.
- [110] G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.
- [111] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [112] L. Venkataramana, D. Prasad, G. Varma, and C. Vishnusree. Identifying the probability of genetic mutations in lung cancer using predictive and prognostic biomarkers from histopathological images. *Medical Imaging Process & Technology*, 6, 12 2023.
- [113] D. J. Waibel, S. Atwell, M. Meier, C. Marr, and B. Rieck. Capturing shape information with multi-scale topological loss terms for 3d reconstruction. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 150–159. Springer, 2022.
- [114] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, 2022.
- [115] P.-Y. Wang, C.-T. Chen, J.-W. Su, T.-Y. Wang, and S.-H. Huang. Deep learning model for house price prediction using heterogeneous data analysis along with joint self-attention mechanism. *IEEE access*, 9:55244–55259, 2021.
- [116] Q. Wang, Y. Ma, K. Zhao, and Y. Tian. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, pages 1–26, 2020.
- [117] M. Waskom, O. Botvinnik, D. O’Kane, P. Hobson, S. Lukauskas, D. C. Gemperline, T. Augspurger, Y. Halchenko, J. B. Cole, J. Warmenhoven, J. de Ruiter, C. Pye, S. Hoyer, J. Vanderplas, S. Villalba, G. Kunter, E. Quintero, P. Bachant, M. Martin, K. Meyer, A. Miles, Y. Ram, T. Yarkoni, M. L. Williams, C. Fitzgerald, Brian, C. Fonnesbeck, A. Lee, and A. Qalieh. mwaskom/seaborn: v0.8.1 (september 2017), Sept. 2017.
- [118] C. Webb. Decomposition of graded modules. *Proceedings of the American Mathematical Society*, 94(4):565–571, 1985.
- [119] H. White. A reality check for data snooping. *Econometrica*, 68(5):1097–1126, 2000.
- [120] J. H. C. Whitehead. Combinatorial homotopy. I. *Bull. Amer. Math. Soc.*, 55:213–245, 1949.
- [121] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1):37–52, 1987. Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists.
- [122] A. Zomorodian and G. Carlsson. Computing persistent homology. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 347–356, 2004.