



UNIVERSIDAD DE LA RIOJA

Facultad de Ciencia y Tecnología

TRABAJO FIN DE GRADO

Grado en Matemáticas

**Homología persistente como herramienta de análisis de
redes neuronales**

Realizado por:

José Manuel Ros Rodrigo

Tutelado por:

José Luis Ansorena Barasoain

Julio Rubio García

Logroño, julio, 2022

Resumen/Abstract

Resumen

En este trabajo fin de grado se realiza un análisis de una novedosa herramienta empleada en la depuración y explicabilidad de las redes neuronales artificiales: *la homología persistente*. Esta herramienta, publicada en un artículo de investigación del que haremos un análisis en profundidad, permite medir la complejidad del conocimiento que podrá obtener una red neuronal.

En primer lugar, para la comprensión de esta herramienta, presentamos un capítulo de preliminares: en él exponemos todos los fundamentos teóricos de la homología persistente y hacemos una breve introducción a las redes neuronales. En segundo lugar, nos centramos en el desarrollo del artículo de investigación, así como en los problemas encontrados durante el estudio del mismo. Finalmente, detallamos el código producido para la aplicación de la herramienta, y la empleamos sobre una colección de ejemplos.

Abstract

In this final degree project we will analyse a novel tool used in the debugging and explainability of artificial neural networks: *persistent homology*. This tool, published in a research article which we will analyse in depth, allows us to measure the complexity of the knowledge that a neural network will be able to obtain.

Firstly, in order to understand this tool, we present a preliminary chapter in which we set out all the theoretical foundations of the persistent homology and we do a brief introduction to artificial neural networks. Secondly, we focus on the development of the research paper, as well as on the problems encountered during the study of it. Finally, we detail the code developed in this project, and we use the tool on a collection of examples.

Índice general

Resumen/Abstract	III
1 Introducción	3
2 Preliminares	5
2.1 Complejos simpliciales	5
2.2 Homología. Homología persistente	7
2.3 Redes neuronales. Grafos subyacentes	19
3 ¿Es posible aplicar la homología persistente a las redes neuronales?	27
3.1 Descripción matemática	27
3.2 Interpretación global. Ejemplo	31
3.3 Interpretación local. Algoritmos y programas	37
3.4 Ejemplos	45
Conclusiones	59
Bibliografía	61

Capítulo 1

Introducción

Supongamos que nos entregan las imágenes 1.1 y nos solicitan que identifiquemos la figura geométrica de la que proceden.

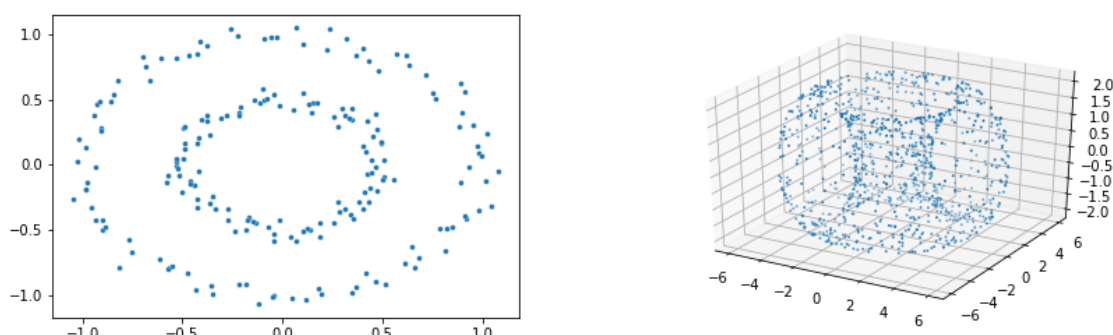


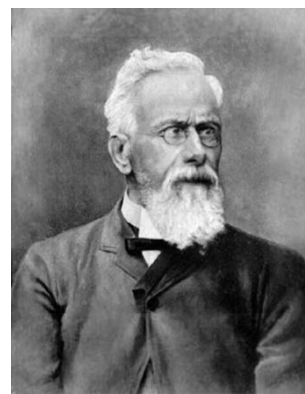
Figura 1.1: Imágenes misteriosas

Observando las imágenes podemos apreciar que la primera proviene de dos circunferencias concéntricas, mientras que la segunda proviene de un toro. Reflexionemos momentáneamente acerca de cómo hemos alcanzado esta conclusión.

Es probable que, de una manera inconsciente, hayamos analizado la forma de las imágenes, el número de agujeros que poseen y la dimensión de dichos agujeros. Este análisis es el que nos ha permitido encontrar las figuras de procedencia. Este proceso, que en 2 y 3 dimensiones nos puede resultar inmediato, en dimensiones superiores resulta más complicado, pues nuestra intuición comienza a fallar. Por ello, se hace necesario el uso de las matemáticas para vencer en este desafío a la intuición.

En matemáticas existe toda una rama relacionada con la idea de estudiar las propiedades invariantes (como los agujeros) de los objetos mediante el uso de técnicas algebraicas. Esta rama se conoce como topología algebraica y tiene su origen con el matemático italiano Enrico Betti (1823-1892). E. Betti, quien en un principio trabajó sobre la teoría de Galois, se hizo célebre por su estudio topológico de la conexión entre figuras de dimensiones altas (mayores que tres) y por caracterizar la igualdad entre superficies.

Más tarde llegaría Henri Poincaré (1854-1912) quien, inspirado por el trabajo de E. Betti, formalizaría toda esta teoría



Enrico Betti
Enrico Betti

en su famoso «Analysis Situs», acuñando el término *número de Betti* que precisamente representa el número de agujeros de un objeto. Esta teoría fue, en un comienzo, puramente topológica y para el siguiente avance significativo hubo que esperar a la llegada de la matemática alemana Emmy Noether (1882-1935), quien apreció la estructura de grupo subyacente a los ciclos definidos en la teoría previa. Esto originó una algebraización de esta teoría.



Emmy Noether

En los subsiguientes años esta teoría se desarrolla rápidamente de la mano de grandes matemáticos como Pável Alexandroff, Leopold Vietoris, Georges de Rham, y Eduard Čech entre otros. Finalmente, Samuel Eilenberg (1913-1998) y Saunders Mac Lane (1909-2005) axiomatizaron esta teoría y la relacionaron con su *teoría de categorías*.

Enmarcada en la topología algebraica tenemos la noción principal sobre la que versa el presente trabajo: *la homología persistente*. Este concepto surge en la década de los 90 y los 2000 de la mano de varios autores: Patrizio Frosini, Massimo Ferri, Vanessa Robins, y Herbert Edelsbrunner. Estos autores llegaron de manera independiente al concepto de persistencia que se desarrollaría en profundidad durante los siguientes 15 años.

Estos fundamentos teóricos nos permiten, entre otras cosas, el cálculo sistemático de los agujeros de un objeto en cualquier dimensión.

En el presente trabajo, los objetos sobre los que estudiaremos los agujeros serán las redes neuronales. Siendo más precisos, vamos a estudiar los agujeros de los grafos subyacentes a dichas redes. Como veremos, estos agujeros, más que simples vacíos dentro de los grafos, representarán de cierta manera el grado de conocimiento que podrá ser adquirido por la red. Por lo tanto, mediante el uso de la topología algebraica podremos extraer conclusiones valiosas acerca de las redes neuronales que habitualmente son vistas como «cajas negras».

En el capítulo 2 del presente trabajo haremos una introducción a los conceptos necesarios para el recuento de agujeros mencionado anteriormente. En el mismo capítulo haremos una breve introducción a las redes neuronales. Finalmente, en el capítulo 3, aplicaremos toda la teoría introducida y veremos una serie de ejemplos.

Capítulo 2

Preliminares

A lo largo de este capítulo vamos a presentar todas las nociones teóricas necesarias para el uso de la homología persistente en redes neuronales.

2.1 Complejos simpliciales

Comenzamos con el primer concepto fundamental de todo el trabajo, los *complejos simpliciales*. Esta noción admite dos enfoques diferentes, por lo que debemos distinguir entre dos definiciones relacionadas: los complejos simpliciales *abstractos* y los complejos simpliciales *geométricos*. Para el desarrollo de estas nociones seguiremos la guía proporcionada por [3, 15].

Partiendo del enfoque combinatorio, comenzamos definiendo los complejos simpliciales abstractos y algunas nociones relacionadas.

Definición 2.1.1. Un *complejo simplicial abstracto* es una colección \mathcal{V} , de subconjuntos no vacíos de un conjunto \mathcal{V}_0 , que verifica las siguientes propiedades:

1. Si $v \in \mathcal{V}_0$, entonces $\{v\} \in \mathcal{V}$.
2. Si $\sigma \in \mathcal{V}$ y $\tau \subset \sigma$, entonces $\tau \in \mathcal{V}$.

A los elementos de \mathcal{V} los llamaremos *símplices*; más concretamente: dado $\sigma \in \mathcal{V}$, diremos que σ tiene *dimensión* p , y que σ es un *p -símplice*, si $|\sigma| = p + 1$. Asimismo, definimos la *dimensión de* \mathcal{V} como el máximo de las dimensiones de sus símplices y denotaremos por \mathcal{V}_p a la colección de los p -símplices de \mathcal{V} .

Para los propósitos del presente trabajo consideraremos que \mathcal{V}_0 es finito, lo que supondrá que los complejos simpliciales definidos a partir de él también serán finitos, así como sus correspondientes símplices. Nótese que la definición de los p -símplices es coherente con \mathcal{V}_0 , es decir, los elementos de \mathcal{V}_0 los podemos considerar como 0-símplices.

En relación con el concepto de símplice y de dimensión surge la siguiente noción:

Definición 2.1.2. Sean σ y τ dos símplices de \mathcal{V} tales que $\tau \subset \sigma$. Entonces diremos que τ es una *cara* de σ . Además, si las dimensiones de σ y τ difieren por un número natural a , diremos que τ es una cara de σ de *codimensión* a .

Ahora que hemos definido los complejos simpliciales abstractos veamos un pequeño ejemplo para fijar ideas.

Ejemplo 2.1.1. Supongamos el siguiente complejo simplicial abstracto:

$$\mathcal{V} = \{\{a\}, \{b\}, \{c\}, \{d\}, \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{c, d\}, \{a, b, c\}, \{a, b, d\}, \\ \{a, c, d\}, \{a, b, c, d\}\}.$$

Así, tenemos que la dimensión de \mathcal{V} es 3. También observamos que el 3-símplice $\{a, b, c, d\}$ tiene por caras de codimensión 1 a los 2-símplices $\{a, b, c\}$, $\{a, b, d\}$ y $\{a, c, d\}$. En la Figura 2.1 ilustramos una representación geométrica de \mathcal{V} .

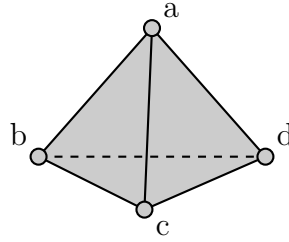


Figura 2.1: Representación geométrica del complejo simplicial \mathcal{V} .

La representación recogida en la Figura 2.1, en la que cada símplece corresponde con un poliedro regular (cada 0-símplice corresponde a un punto, cada 1-símplice a una arista, cada 2-símplice a un triángulo, cada 3-símplice a un tetraedro, etc.) es única salvo homeomorfismo. Observamos que interpretando \mathcal{V} como un subconjunto de \mathbb{R}^3 obtenemos un tetraedro. Esta idea motiva el otro enfoque de los complejos simpliciales: el enfoque geométrico. ◀

Siguiendo el enfoque geométrico es necesario que, antes de llegar a la definición de complejo simplicial geométrico, veamos unos conceptos previos relacionados con la propia definición.

Definición 2.1.3. Sean $\{u_0, u_1, \dots, u_k\} \subset \mathbb{R}^n$. Diremos que los $k+1$ puntos son *afínmente independientes* si los k vectores $u_1 - u_0, u_2 - u_0, \dots, u_k - u_0$ son linealmente independientes.

Sea $x \in \mathbb{R}^n$. Diremos que x es una *combinación afín* de $\{u_0, u_1, \dots, u_k\}$ si $\exists \lambda_0, \dots, \lambda_k$ tales que $x = \sum_{i=0}^k \lambda_i u_i$ y $\sum_{i=0}^k \lambda_i = 1$.

Definición 2.1.4. Sean $\{u_0, u_1, \dots, u_k\} \subset \mathbb{R}^n$ $k+1$ puntos afínmente independientes y $x = \sum_{i=0}^k \lambda_i u_i$ una combinación afín. Diremos que x es una *combinación convexa* de $\{u_0, u_1, \dots, u_k\}$ si $\{\lambda_0, \lambda_1, \dots, \lambda_k\}$ son no negativos.

Definimos la *clausura convexa* de $\{u_0, u_1, \dots, u_k\}$ como el conjunto de todas sus posibles combinaciones convexas.

Ahora que ya contamos con estas nociones previas pasamos a definir la pieza clave en la definición de complejo simplicial geométrico: el *símplice*.

Definición 2.1.5. Definimos un k -símplice como la clausura convexa de $k + 1$ puntos afínmente independientes. Lo denotaremos por $\sigma = \text{conv}\{u_0, u_1, \dots, u_k\}$, y diremos que la *dimensión de σ* es k . A los puntos u_0, \dots, u_k los llamamos *vértices* del símplex. Observamos que, al ser afínmente independientes, el conjunto de vértices de un símplex es único.

Definimos *cara* de σ como cualquier combinación convexa de un subconjunto no vacío de $\{u_0, u_1, \dots, u_k\}$. A la relación «ser cara de» la denotaremos por \leq .

Para los casos $k = 1, 2, 3$ diremos que σ es una arista, un triángulo o un tetraedro respectivamente.

Habiendo definido todos los conceptos previos necesarios pasamos a definir *complejo simplicial geométrico*.

Definición 2.1.6. Llamamos *complejo simplicial geométrico* a la colección finita de símplexes \mathcal{V} verificando las siguientes propiedades:

1. Si $\sigma \in \mathcal{V}$ y $\tau \leq \sigma$, entonces $\tau \in \mathcal{V}$,
2. Si $\sigma_1, \sigma_2 \in \mathcal{V}$, entonces $\sigma_1 \cap \sigma_2 = \emptyset$, o bien, $\sigma_1 \cap \sigma_2$ es una cara común a ambos.

La relación entre los complejos simpliciales abstractos y los geométricos viene dada por la construcción de la *realización geométrica* de un complejo simplicial abstracto, que es un complejo simplicial geométrico definido tal y como se ilustra en la Figura 2.1 del ejemplo 2.1.1 (para más detalles véase [3]). Siendo más precisos, los conjuntos de vértices de un símplex geométrico forman un símplex abstracto.

De aquí en adelante emplearemos la definición de complejo simplicial abstracto, pues es la más adecuada para el presente trabajo.

Ahora que ya hemos definido los objetos con los que vamos a trabajar, procedemos a definir las aplicaciones entre ellos.

Definición 2.1.7. Una *aplicación simplicial entre complejos simpliciales*, $f: \mathcal{V} \rightarrow \mathcal{V}'$, es una aplicación tal que $f(\sigma) = \{g(u_1), g(u_2), \dots, g(u_k)\} = \{v_1, v_2, \dots, v_k\}$; donde $g: \mathcal{V}_0 \rightarrow \mathcal{V}'_0$ es una aplicación entre 0-símplexes, $\sigma = \{u_1, u_2, \dots, u_k\} \in \mathcal{V}$ y $\{v_1, v_2, \dots, v_k\} \in \mathcal{V}'$.

Nota. Notemos que en la anterior definición la función g podría no ser inyectiva, esto provocaría que la función f no respetara las dimensiones de los símplexes. Este caso particular es muy interesante y veremos sus implicaciones en la siguiente sección.

2.2 Homología. Homología persistente

En la sección anterior hemos fijado el concepto de complejo simplicial, que nos será muy útil a lo largo de esta sección para desarrollar la noción de *espacio vectorial de homología*. A diferencia de como surgió el concepto de *espacio vectorial de homología* en la historia de las matemáticas, en el presente trabajo desarrollaremos primero la noción general para

luego reducir al caso particular de la *homología simplicial*. Para ello emplearemos la guía proporcionada por [3, 13, 4].

Comenzamos la sección con algunas definiciones básicas que serán necesarias para alcanzar la definición de los espacios vectoriales de homología.

Definición 2.2.1. Sea R un anillo. Definimos el R -módulo a izquierda sobre R como el conjunto M junto con las operaciones:

- Suma: $M \times M \rightarrow M, (x, y) \mapsto x + y$, y
- Producto por escalares: $R \times M \rightarrow M, (r, x) \mapsto rx$,

que satisfacen las siguientes propiedades:

1. La suma es asociativa, conmutativa, M contiene un elemento neutro para ella y todo elemento tiene opuesto. Es decir, $(M, +)$ es un grupo abeliano.
2. Para cualesquiera x, y de M y r, s de R :
 - (a) $(r + s)x = rx + sx$ (distributiva respecto a la suma de R).
 - (b) $(rs)x = r(sx)$ (asociativa).
 - (c) $r(x + y) = rx + ry$ (distributiva respecto a la suma de M).
 - (d) Si R es unitario, $1x = x$.

De manera análoga definimos el R -módulo a derecha. Si R es conmutativo, entonces el R -módulo a izquierda es el mismo que el R -módulo a derecha. En tal caso nos referiremos a él simplemente como R -módulo.

Nota. Notamos que un grupo abeliano se identifica de manera natural con un \mathbb{Z} -módulo. Asimismo, un módulo sobre un cuerpo no es otra cosa que un espacio vectorial.

Como es natural, a la noción de R -módulo le sigue la definición de R -submódulo.

Definición 2.2.2. Sea M un R -módulo. Definimos el R -submódulo de M como el subconjunto, no vacío, N de M tal que es cerrado para opuestos y para las operaciones heredadas de M . A la relación «ser submódulo de» la denotaremos por \leq .

Ahora que hemos definido los R -submódulos, vamos a definir los cocientes asociados.

Definición 2.2.3. Sea M un módulo sobre un anillo R y N un submódulo de M . Consideremos la relación de equivalencia en M dada por:

$$x \sim y \text{ si } x - y \in N,$$

con $x, y \in M$. Mediante esta relación de equivalencia definimos el conjunto cociente M/N . Ahora bien, mediante las operaciones suma y producto por escalares de M podemos inducir las correspondientes operaciones sobre M/N de la siguiente manera:

$$\begin{aligned} (x + N) + (y + N) &= (x + y) + N, \\ r(x + N) &= rx + N, \end{aligned}$$

donde $x, y \in M$ y $r \in R$. Así, al conjunto cociente M/N junto con las operaciones previas lo llamaremos *módulo cociente* de M sobre N .

Tras estas consideraciones básicas, comenzamos el camino que nos conducirá a la definición de los *espacios vectoriales de homología*. Empezamos el camino con la definición de *complejo de cadenas*.

Definición 2.2.4. Sea R un anillo. Decimos que un *complejo de cadenas* sobre R es un conjunto $\mathcal{C}_* = \{(C_p, \partial_p) \mid p \in \mathbb{Z}\}$ de R -módulos y R -homomorfismos $\{\partial_p: C_p \rightarrow C_{p-1} \mid p \in \mathbb{Z}\}$, que satisfacen la siguiente relación: $\partial_p \circ \partial_{p+1} = 0$. Se denota por $(\mathcal{C}_*, \partial)$ y a ∂ se le llama el *diferencial* del complejo.

Notemos que la relación $\partial_p \circ \partial_{p+1} = 0$ es equivalente a que $\text{im } \partial_{p+1} \leq \ker \partial_p$, $p \in \mathbb{Z}$. Es habitual pensar en \mathcal{C}_* como una sucesión infinita cuya representación es como sigue:

$$\cdots \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} C_{-1} \xrightarrow{\partial_{-1}} \cdots$$

Fijemos ahora nuestra atención en los R -submódulos de C_p : $\text{im } \partial_{p+1}$ y $\ker \partial_p$.

Definición 2.2.5. Sea $(\mathcal{C}_*, \partial)$ un complejo de cadenas sobre R anillo. Definimos los siguientes R -submódulos de C_p :

1. $Z_p(\mathcal{C}_*) = \ker \partial_p$. A sus elementos los llamaremos p -ciclos.
2. $B_p(\mathcal{C}_*) = \text{im } \partial_{p+1}$. A sus elementos los llamaremos p -bordes.

Ahora bien, en virtud de la relación de inclusión entre ambos submódulos y de las operaciones heredadas, podemos considerar el R -módulo cociente $Z_p(\mathcal{C}_*)/B_p(\mathcal{C}_*)$. Este cociente constituye una pieza fundamental del presente trabajo y merece una definición detallada:

Definición 2.2.6. Sea $(\mathcal{C}_*, \partial)$ un complejo de cadenas sobre R anillo. Definimos el p -ésimo R -módulo de homología de \mathcal{C}_* como el cociente:

$$H_p(\mathcal{C}_*) := Z_p(\mathcal{C}_*)/B_p(\mathcal{C}_*).$$

Ahora que ya tenemos una noción general de los R -módulos de homología, pasamos al caso particular que nos ocupa en el presente trabajo: *la homología simplicial*. El primer paso será definir el complejo de cadenas asociado a un complejo simplicial, tras esto, fijaremos los R -módulos y R -homomorfismos sobre los que finalmente calcularemos la homología. A continuación veamos la definición de los complejos de cadenas:

Definición 2.2.7. Sea R un anillo, \mathcal{V} un complejo simplicial, y $p \in \mathbb{N} \cup \{0\}$ tal que $p \leq \dim \mathcal{V}$. Una p -cadena es una suma formal de p -símplices de \mathcal{V} . Es decir, si c es una p -cadena, entonces $c = \sum a_i \sigma_i$, $\sigma_i \in \mathcal{V}$ y $a_i \in R$.

Con la noción de p -cadena, pasamos a la definición de los R -módulos que hemos mencionado anteriormente:

Definición 2.2.8. Sea R un anillo y \mathcal{V} un complejo simplicial abstracto. Definimos el R -módulo de p -cadenas de \mathcal{V} como el conjunto de todas las R -cadenas de \mathcal{V} , con la operación suma componente a componente con coeficientes en R , y el producto por escalares por elementos de R . Lo denotaremos por $C_p(\mathcal{V})$.

Naturalmente, una decisión que debemos tomar a la hora de calcular la homología simplicial es elegir el anillo de escalares sobre el que vamos a trabajar.

Lo habitual suele ser fijar una orientación para los símlices (por ejemplo, la dada por el orden lexicográfico de sus vértices) y escoger el anillo \mathbb{Z} . Otra opción, que ha demostrado dar información significativa en las aplicaciones, es escoger el anillo \mathbb{Z}_2 . Al tener dos elementos, el trabajo sobre él resulta especialmente sencillo, sobre todo desde un punto de vista computacional.

Observamos que con la segunda opción, una p -cadena puede verse como un conjunto de p -símlices y la suma de ellas consiste en hacer la unión de los conjuntos y eliminar los símlices repetidos un número par de veces. Además, la suma coincide con la resta, y el producto por escalares es nulo o la propia p -cadena.

Habiendo especificado los R -módulos que vamos a emplear, pasamos a definir los R -homomorfismos asociados. Para la definición de tales homomorfismos de manera general es necesario considerar un ordenamiento en el conjunto \mathcal{V}_0 .

Definición 2.2.9. Sea R un anillo, \mathcal{V} un complejo simplicial abstracto, $\sigma \in \mathcal{V}$ y $\sigma = \{u_0, \dots, u_p\}$ un p -símplice escogido de manera ordenada. Definimos el *operador borde para un símplice* como:

$$B_p(\sigma) = \sum_{j=0}^p (-1)^j \{u_0, \dots, \widehat{u_j}, \dots, u_p\}.$$

La suma anterior es una suma formal, donde $\widehat{u_j}$ indica que omitimos u_j . Ahora extendemos dicho operador para cadenas, en concreto:

$$\begin{aligned} \partial_p: C_p(\mathcal{V}) &\rightarrow C_{p-1}(\mathcal{V}) \\ c = \sum a_i \sigma_i &\mapsto \partial_p(c) = \sum a_i B_p(\sigma_i) \end{aligned} \tag{2.2.1}$$

donde $a_i \in R$. Notemos que, si escogemos $R = \mathbb{Z}_2$, no es necesaria la alternancia de signos ni el uso de coeficientes escalares. A ∂_p lo llamaremos *homomorfismo borde*.

Nota. Puesto que las p -cadenas son sumas formales, es decir, estamos considerando R -módulos libres sobre el conjunto de p -símlices, la aplicación borde está bien definida, y es lineal, o sea, un homomorfismo entre R -módulos.

Por lo tanto, ya tenemos casi construido nuestro complejo de cadenas asociado a un complejo simplicial. Nótese que en la definición 2.2.4 admitimos complejos de cadenas para cualquier índice $p \in \mathbb{Z}$, no obstante, tomaremos $C_p = \emptyset$ para todo $p \leq 0$ con $p \in \mathbb{Z}$. Tras esta consideración, sólo resta enunciar el siguiente resultado:

Teorema 1. Sea ∂_p definida como en la expresión 2.2.1. Entonces para todo $p \in \mathbb{N} \cup \{0\}$, $\partial_p \circ \partial_{p+1} = 0$.

Demostración. Sea $c \in C_{p+1}(\mathcal{V})$ y, sin pérdida de generalidad (en virtud de la definición 2.2.1), supongamos que c es un elemento de la base del R -módulo libre de cadenas $C_{p+1}(\mathcal{V})$. Veamos que $\partial_p(\partial_{p+1}(c)) = 0$.

En efecto, notemos que, si $p \geq 1$ (si $p = 0$ es trivial), $\partial_p(\partial_{p+1}(c))$ posee $\binom{p+2}{p} = (p+2)(p+1)$ sumandos, y cada uno de ellos es el producto de un escalar por un $(p-1)$ -símplice. Sea τ uno de ellos, y sean u_i, u_j los 0-símplices presentes en el $(p+1)$ -símplice de c pero no en τ . Supondremos también que u_i, u_j ocupan las posiciones i y j respectivamente en el $(p+1)$ -símplice de c escrito de manera ordenada.

Observamos que τ aparece en dos sumandos de $\partial_p(\partial_{p+1}(c))$ con las expresiones opuestas:

$$(-1)^i(-1)^j\tau, (-1)^{j-1}(-1)^i\tau.$$

La primera expresión se obtiene al omitir el elemento en la posición j y luego el de la posición i del $(p+1)$ -símplice de c (véase la definición 2.2.1); la segunda expresión se obtiene al omitir el elemento en la posición i , con lo que se adelanta una posición u_j , y luego el de la posición $j - 1$.

Notemos que al ser expresiones opuestas la suma es nula y se concluye el resultado. \square

Nota. Coloquialmente, diremos que «el borde del borde es vacío». Notemos que el teorema anterior denota lo significativo de la elección del anillo sobre el que se toman los coeficientes, pues la demostración sería distinta y los cálculos posteriores se complican. Veremos este hecho en los siguientes ejemplos.

Veamos un ejemplo que ilustre el teorema anterior, es decir, que «el borde del borde es vacío». En él, como hemos sugerido, tomaremos como anillo el cuerpo \mathbb{Z}_2 .

Ejemplo 2.2.1. Supongamos el complejo simplicial \mathcal{V} del ejemplo anterior y $\sigma = \{a, b, c\} \in \mathcal{V}$.

Así pues, tendremos $c \in C_2(\mathcal{V})$, con $c = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ la cadena asociada a σ en $C_2(\mathcal{V})$. Ahora expresamos las aplicaciones ∂_2 y ∂_1 en forma matricial:

$$\partial_2 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad y \quad \partial_1 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Ahora, teniendo en cuenta que estamos operando en un cuerpo de característica 2,

hacemos $\partial_1(\partial_2(c))$:

$$\begin{aligned}\partial_1(\partial_2(c)) &= \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \cdot \left(\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right) = \\ &= \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \vec{0}.\end{aligned}$$

Hemos comprobado que, en efecto, «el borde del borde» de c es 0. Para comprobarlo para cualquier vector bastará observar que:

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

◀

Tras el resultado anterior y las construcciones previas, hemos acabado con la construcción del complejo de cadenas asociado a un complejo simplicial. En consecuencia, ya estamos en posición de definir los *R-módulos de homología*.

Definición 2.2.10. Sea R un anillo, $p \in \mathbb{N} \cup \{0\}$ y \mathcal{V} , un complejo simplicial. Definimos el *p-ésimo R-módulo de homología simplicial* del complejo simplicial \mathcal{V} como el R -módulo cociente $\ker \partial_p / \operatorname{im} \partial_{p+1}$, donde ∂_p está definida como en la expresión 2.2.1. Lo denotaremos por $H_p(\mathcal{V})$.

La definición de R -módulos de homología tiene sentido para cualquier anillo unitario y conmutativo R , pero se puede decir más sobre su estructura si exigimos a R propiedades adicionales. En particular, si R es un dominio de ideales principales, el teorema de estructura de R -módulos finitamente generados (véase [4]) asegura que cada R -módulo de homología tendrá una parte libre (su rango) y una parte de torsión. Puesto que este estudio general alargaría en exceso esta memoria, particularizamos a partir de este punto los coeficientes para que sean el cuerpo con dos elementos \mathbb{Z}_2 , lo que es suficiente para las aplicaciones de las siguientes secciones. En este nuevo contexto, los R -módulos de homología, que ahora serán \mathbb{Z}_2 -espacios vectoriales, quedan caracterizados por su dimensión, que se denomina *número de Betti* (en honor al matemático del mismo nombre, y que también puede ser definido en el caso general como diferencia entre rangos).

Definición 2.2.11. Sea $H_p(\mathcal{V})$ el p -ésimo \mathbb{Z}_2 -espacio vectorial de homología de un complejo simplicial \mathcal{V} . A su dimensión, $\dim H_p(\mathcal{V}) = \dim \ker \partial_p - \dim \operatorname{im} \partial_{p+1}$, la llamaremos p -ésimo número de Betti, y la denotaremos por $\beta_p(\mathcal{V})$.

Intuitivamente, los p -ciclos que no son p -bordes representan agujeros p -dimensionales. Por lo tanto, $\beta_p(\mathcal{V})$ representa el número de p -agujeros de \mathcal{V} . Además, notemos que si $\dim \mathcal{V} = n$, entonces $\forall p > n \ H_p(\mathcal{V}) = [0]$, pues $\mathcal{V}_p = \emptyset$.

Ahora veamos un ejemplo (propuesto en [15]) en el que calculamos los números de Betti dado un complejo simplicial abstracto.

Ejemplo 2.2.2. Supongamos el siguiente complejo simplicial abstracto:

$$\mathcal{V} = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{c, d\}, \{a, b, c\}\}.$$

Construimos la secuencia de espacios de cadenas asociados:

$$\emptyset \longrightarrow C_2(\mathcal{V}) \xrightarrow{\partial_2} C_1(\mathcal{V}) \xrightarrow{\partial_1} C_0(\mathcal{V}) \xrightarrow{\partial_0} \emptyset.$$

Calculamos ∂_2 y ∂_1 , y los expresamos de manera matricial:

$$\partial_2 = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \quad \partial_1 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Ahora para calcular $\beta_0(\mathcal{V})$ y $\beta_1(\mathcal{V})$ bastará calcular el rango de las anteriores matrices. Observamos que:

$$\partial_1 = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ \boxed{1} & 0 & 0 & 1 & 0 \\ 0 & \boxed{1} & 0 & 1 & 1 \\ 0 & 0 & \boxed{1} & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Con lo que, $\dim \operatorname{im} \partial_1 = 3$ y $\dim \ker \partial_1 = 2$, y por lo tanto, $\beta_1(\mathcal{V}) = 1$ y $\beta_0(\mathcal{V}) = 2$. La representación gráfica de \mathcal{V} en \mathbb{R}^2 puede verse en la Figura 2.2.

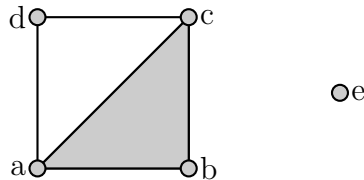


Figura 2.2: Representación geométrica del complejo simplicial \mathcal{V} .

Tal y como ya hemos comentado, los números de Betti nos cuentan los agujeros p -dimensionales. En este caso, si observamos la representación anterior, vemos que tenemos

un agujero 1-dimensional y dos componentes conexas, que se corresponde con los números de Betti que hemos calculado.

Notemos que, al igual que en el ejemplo anterior, si trabajamos sobre otro cuerpo que no sea \mathbb{Z}_2 , el cálculo de las matrices y los rangos sería más costoso. ◀

Al igual que hicimos para los complejos simpliciales, veamos cómo podemos definir morfismos, de manera más general, entre los objetos que estamos manejando.

Consideremos una aplicación entre complejos simpliciales, $f: \mathcal{V} \rightarrow \mathcal{V}'$. Aplicando el mismo razonamiento que para la definición del homomorfismo borde, tenemos que f induce un homomorfismo entre R -módulos de cadenas:

$$\begin{aligned} \overline{f_p}: C_p(\mathcal{V}) &\rightarrow C_p(\mathcal{V}') \\ c = \sum_{\sigma \in \mathcal{V}_p} \sigma &\mapsto \overline{f_p}(c) = \sum_{f(\sigma) \in \mathcal{V}'_p} f(\sigma). \end{aligned} \quad (2.2.2)$$

Además, tal f nos permite construir la secuencia:

$$\begin{array}{ccccccccccccccc} \emptyset & \rightarrow & C_p(\mathcal{V}) & \xrightarrow{\partial_R} & C_{p-1}(\mathcal{V}) & \xrightarrow{\partial_{p-1}} & \cdots & \xrightarrow{\partial_2} & C_1(\mathcal{V}) & \xrightarrow{\partial_1} & C_0(\mathcal{V}) & \xrightarrow{\partial_0} & \emptyset \\ & & \downarrow \overline{f_p} & & \downarrow \overline{f_{p-1}} & & & & \downarrow \overline{f_1} & & \downarrow \overline{f_0} & & \\ \emptyset & \rightarrow & C_p(\mathcal{V}') & \xrightarrow{\partial'_p} & C_{p-1}(\mathcal{V}') & \xrightarrow{\partial'_{p-1}} & \cdots & \xrightarrow{\partial'_2} & C_1(\mathcal{V}') & \xrightarrow{\partial'_1} & C_0(\mathcal{V}') & \xrightarrow{\partial'_0} & \emptyset. \end{array} \quad (2.2.3)$$

De esta secuencia observamos que:

$$\overline{f_{p-1}} \circ \partial_p = \partial'_p \circ \overline{f_p}.$$

En consecuencia, $\overline{f_p}$ induce un homomorfismo entre R -módulos de homología:

$$\begin{aligned} f_p: H_p(\mathcal{V}) &\rightarrow H_p(\mathcal{V}') \\ [c] &\mapsto [\overline{f_p}(c)]. \end{aligned}$$

Concluimos que, dada una aplicación f entre complejos simpliciales, siempre es posible asociarle una aplicación f_p entre grupos de homología.

Nota. Esta propiedad es muy importante, de hecho, se conoce como *funtorialidad* y pertenece al ámbito de la teoría de categorías que queda fuera del alcance del presente trabajo.

Nota. Si consideramos el caso particular $f = \iota$ (la inclusión) tendremos que f induce una inclusión entre los complejos de cadenas asociados a \mathcal{V} y \mathcal{V}' de manera natural. Por supuesto, esta propiedad ya no se hereda al considerar las aplicaciones entre R -módulos de homología. Es decir, ι_p no es necesariamente una aplicación inyectiva. Muy al contrario, la posibilidad de que no lo sea juega un papel fundamental en el estudio de la homología persistente, que estudiaremos a continuación.

Si bien los espacios vectoriales de homología de un complejo simplicial abstracto nos aportan mucha información acerca de sus características topológicas, esta información tiene bastante margen de mejora pues no nos dice nada de la variable «tiempo», es decir, sobre el carácter dinámico de la evolución de un complejo simplicial si este evolucionara en el tiempo. ¿Cómo introducimos la noción de tiempo en un complejo simplicial abstracto? Esta pregunta motiva la siguiente definición:

Definición 2.2.12. Sea \mathcal{V} un complejo simplicial abstracto finito. Consideremos la secuencia $\mathcal{V}^1 \subset \mathcal{V}^2 \subset \dots \subset \mathcal{V}^{k-1} \subset \mathcal{V}^k = \mathcal{V}$ de subcomplejos simpliciales cualesquiera de \mathcal{V} . A \mathcal{V} junto con su secuencia de subcomplejos simpliciales encajados lo llamaremos *complejo simplicial filtrado*.

Esta noción nos habilita la variable «tiempo», pues nos permite preguntarnos en que momento de la secuencia aparecerá una cierta característica topológica, y cuanto «tiempo» sobrevivirá dicha característica.

Existen varias maneras de construir la secuencia de complejos simpliciales, por ejemplo, empleando el *complejo simplicial de Čech*. Su construcción se realiza de la siguiente manera:

Sea \mathcal{V} un complejo simplicial y \mathcal{U} un cubrimiento de \mathcal{V} . Los p -símplices del complejo simplicial de Čech vendrán dados por la intersección no vacía de $p + 1$ conjuntos de \mathcal{U} .

Lo interesante de este método es que si \mathcal{U} verifica ciertas condiciones, el *Teorema del nervio* garantiza que el complejo de Čech recupera la homología de \mathcal{V} (para más detalles véase [5]). ¿Cómo podemos capturar y visualizar esta nueva información? Empleando la *homología persistente*.

Nota. En el presente trabajo nos ceñiremos a la homología persistente definida sobre un complejo simplicial filtrado. No obstante, cabe destacar que esta noción tiene una generalización conocida como *módulo de persistencia* que se define sobre un conjunto parcialmente ordenado.

Definición 2.2.13. Sea $\mathcal{V}^1 \subset \mathcal{V}^2 \subset \dots \subset \mathcal{V}^{k-1} \subset \mathcal{V}^k = \mathcal{V}$ un complejo simplicial filtrado. Definimos los *p -ésimos espacios vectoriales de homología persistente* como las imágenes de los homomorfismos inducidos por la inclusión, $H_p^{i,j} = \text{im } f_p^{i,j}$, con $0 \leq i \leq j \leq k$.

A su dimensión, $\dim H_p^{i,j}$, la llamaremos *p -ésimo número de Betti persistente* y la denotaremos por $\beta_p^{i,j}$.

Los homomorfismos $f_p^{i,j}$ los definimos siguiendo la idea dada por la funtorialidad. Es decir, tendremos el diagrama:

$$\begin{array}{ccccccccccccccc}
 \emptyset & \rightarrow & C_p(\mathcal{V}^1) & \xrightarrow{\partial_p^1} & C_{p-1}(\mathcal{V}^1) & \xrightarrow{\partial_{p-1}^1} & \dots & \xrightarrow{\partial_2^1} & C_1(\mathcal{V}^1) & \xrightarrow{\partial_1^1} & C_0(\mathcal{V}^1) & \xrightarrow{\partial_0^1} & \emptyset \\
 & & \downarrow \overline{f_p^{1,2}} & & \downarrow \overline{f_{p-1}^{1,2}} & & & & \downarrow \overline{f_1^{1,2}} & & \downarrow \overline{f_0^{1,2}} & & \\
 \emptyset & \rightarrow & C_p(\mathcal{V}^2) & \xrightarrow{\partial_p^2} & C_{p-1}(\mathcal{V}^2) & \xrightarrow{\partial_{p-1}^2} & \dots & \xrightarrow{\partial_2^2} & C_1(\mathcal{V}^2) & \xrightarrow{\partial_1^2} & C_0(\mathcal{V}^2) & \xrightarrow{\partial_0^2} & \emptyset \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 \emptyset & \rightarrow & C_p(\mathcal{V}^{k-1}) & \xrightarrow{\partial_p^{k-1}} & C_{p-1}(\mathcal{V}^{k-1}) & \xrightarrow{\partial_{p-1}^{k-1}} & \dots & \xrightarrow{\partial_2^{k-1}} & C_1(\mathcal{V}^{k-1}) & \xrightarrow{\partial_1^{k-1}} & C_0(\mathcal{V}^{k-1}) & \xrightarrow{\partial_0^{k-1}} & \emptyset \\
 & & \downarrow \overline{f_p^{k-1,k}} & & \downarrow \overline{f_{p-1}^{k-1,k}} & & & & \downarrow \overline{f_1^{k-1,k}} & & \downarrow \overline{f_0^{k-1,k}} & & \\
 \emptyset & \rightarrow & C_p(\mathcal{V}^k) & \xrightarrow{\partial_p^k} & C_{p-1}(\mathcal{V}^k) & \xrightarrow{\partial_{p-1}^k} & \dots & \xrightarrow{\partial_2^k} & C_1(\mathcal{V}^k) & \xrightarrow{\partial_1^k} & C_0(\mathcal{V}^k) & \xrightarrow{\partial_0^k} & \emptyset.
 \end{array}$$

Donde los homomorfismos $\overline{f_p^{i,j}}$ entre espacios vectoriales de cadenas vienen inducidos por el homomorfismo inclusión $f^{i,j}: \mathcal{V}^i \hookrightarrow \mathcal{V}^j$ con $0 \leq i \leq j \leq k$, como en la expresión

2.2.2. Ahora, aplicando el mismo razonamiento que en la expresión 2.2.3 definimos los homomorfismos entre espacios vectoriales de homología $f_p^{i,j}$.

Ahora que ya tenemos una herramienta que nos captura las características topológicas junto con la variable «tiempo» en un complejo simplicial abstracto filtrado, necesitamos una manera gráfica de visualizar esta información. Para ello, emplearemos los *diagramas de barras* y los *diagramas de persistencia*.

Para construir los diagramas de barras elegimos, para cada i , una base de cada espacio vectorial $H_p^{i,i+1}$, que representamos en el diagrama mediante $\beta_p^{i,i+1}$ puntos. Entonces, conectamos los $\beta_p^{i,i+1}$ puntos de la i -ésima filtración con los $\beta_p^{i+1,i+2}$ puntos de la filtración $i+1$ atendiendo al siguiente criterio: unimos los puntos a de la filtración i y b de la filtración $i+1$, si la clase del elemento que genera a a es preimagen por $f_p^{i,i+1}$ de la clase del elemento que genera a b . Si la clase del elemento que genera a a es enviada a 0 por $f_p^{i,i+1}$, dibujaremos una línea que sale de a en la filtración i hasta la filtración $i+1$. En este caso, diremos que la clase del elemento que genera a a *muerde* en la filtración $i+1$. Si la preimagen de la clase del elemento que genera a a por $f_p^{i-1,i}$ es el 0, diremos que la clase *nace* en la filtración i .

Para los diagramas de persistencia, dibujaremos $\sum_p \beta_p^{i,j}$ puntos cuyas coordenadas en \mathbb{R}^2 vendrán dadas por su filtración de nacimiento y de muerte en ese orden.

Nota. Estos diagramas dependen de la elección de la base de los espacios vectoriales subyacentes. Una mala elección nos llevará a diagramas ilegibles. No obstante, la existencia de una «buena» base está garantizada (véase [20]) y la coherencia de los diagramas también (véase [2]). Adicionalmente, debemos destacar que en las aplicaciones el cálculo de estos diagramas y la elección de una base apropiada se ha realizado de manera automática (véase la Sección 3.3).

Veamos un ejemplo (propuesto en [15]) ilustrativo de estos conceptos:

Ejemplo 2.2.3. Consideremos el siguiente complejo simplicial filtrado: $\mathcal{V}^1 \subset \mathcal{V}^2 \subset \mathcal{V}^3 \subset \mathcal{V}^4 = \mathcal{V}$. Donde:

- $\mathcal{V}^1 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{a, b\}\}.$
- $\mathcal{V}^2 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{a, b\}, \{a, c\}, \{b, c\}\}.$
- $\mathcal{V}^3 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{a, b\}, \{a, c\}, \{a, e\}, \{b, c\}, \{b, f\}, \{c, f\}, \{c, e\}, \{a, b, c\}\}.$
- $\mathcal{V}^4 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{a, b\}, \{a, c\}, \{a, e\}, \{b, c\}, \{b, f\}, \{c, f\}, \{c, e\}, \{a, b, c\}, \{a, c, e\}\}.$

En la Figura 2.3 podemos ver su representación gráfica en \mathbb{R}^2 . En la Figura 2.4 podemos ver los diagramas de barras correspondientes.

Como podemos observar en la Figura 2.4 diagrama izquierdo (grado 0), tenemos inicialmente (filtración 1 que corresponde al dibujo de \mathcal{V}^1) tres clases de equivalencia que son, en orden ascendente en el diagrama, las clases $[d]$, $[a+b]$, y $[c]$. Como ya hemos visto,

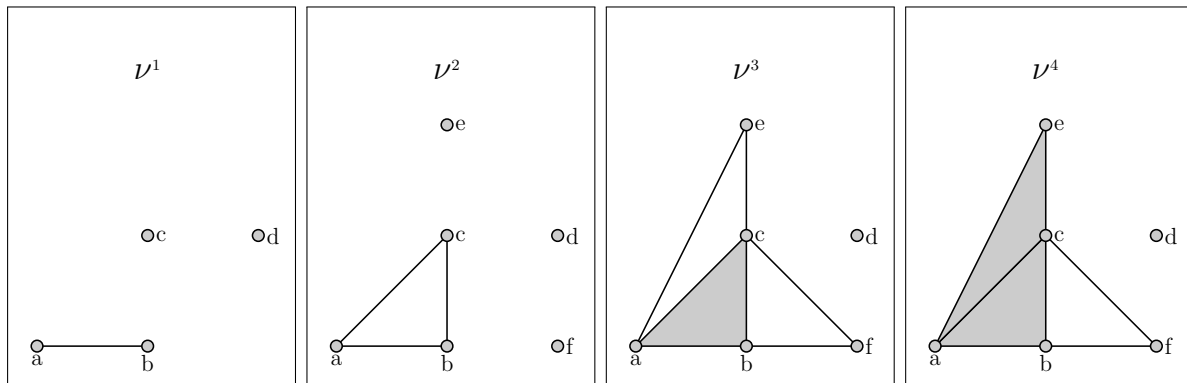


Figura 2.3: Representación gráfica de los pasos en la filtración de \mathcal{V} .

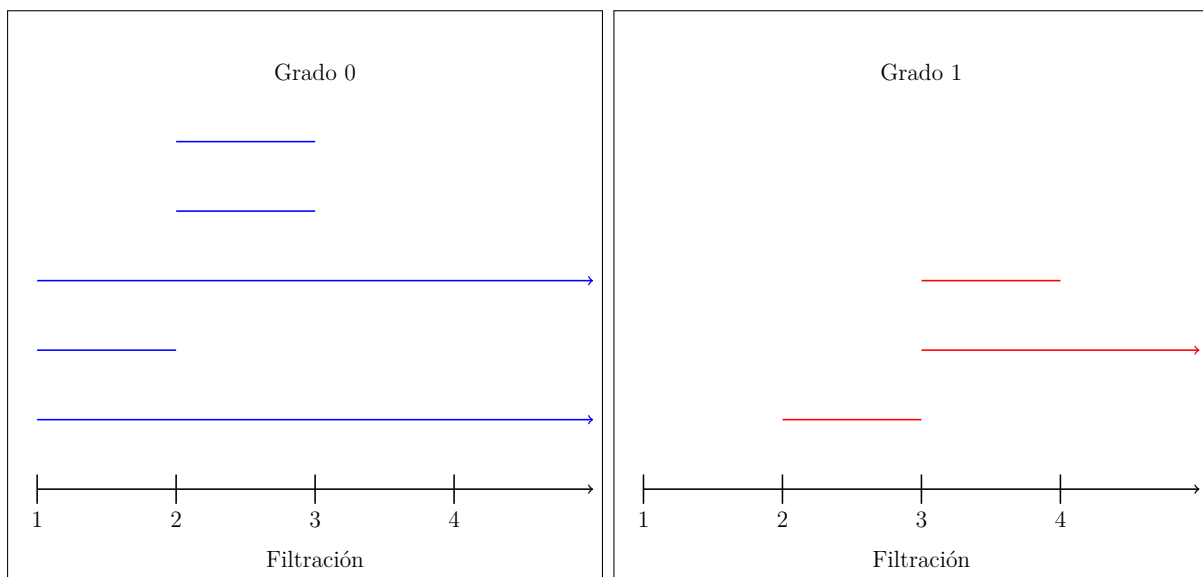


Figura 2.4: Diagramas de barras asociados a la filtración de \mathcal{V} .

esta representación se corresponde con las componentes conexas de \mathcal{V}^1 . A medida que avanzamos en las filtraciones podemos ver cómo la clase $[a + b]$ muere en la filtración 2 (que se corresponde con el dibujo de \mathcal{V}^2), nacen dos clases más ($[f]$ y $[e]$ en orden ascendente en el diagrama), ambas clases mueren en la filtración 3 (véase el dibujo de \mathcal{V}^3), y finalmente, las clases $[d]$ y $[c]$ persisten.

Observemos ahora la Figura 2.4 diagrama derecho (grado 1). Como se puede ver, en la filtración 1 no tenemos ninguna clase de equivalencia, tendremos que esperar a la filtración 2 en la que nace la clase $[ab + bc + ac]$. Esta clase muere en la siguiente filtración, pero nacen otras dos: la clase $[bc + cf + bf]$ y la clase $[ac + ae + ce]$. Esta última morirá en la filtración 4, no obstante, la clase $[bc + cf + bf]$ persiste. Como ya mencionamos, estas clases de equivalencia se corresponden con agujeros 2-dimensionales, cuya evolución podemos ir viendo en los dibujos anteriores que representan los pasos en la filtración del complejo simplicial \mathcal{V} .

De los anteriores diagramas podemos extraer información topológica muy importante acerca de nuestro complejo simplicial: vemos que presenta dos componentes conexas y un agujero 2-dimensional. Esta información nos la aporta la homología persistente, ya que las clases de equivalencia que persisten representan características topológicas, mientras que las que mueren en alguna filtración representan ruido.

Por último, en la Figura 2.5 podemos visualizar los diagramas de persistencia.

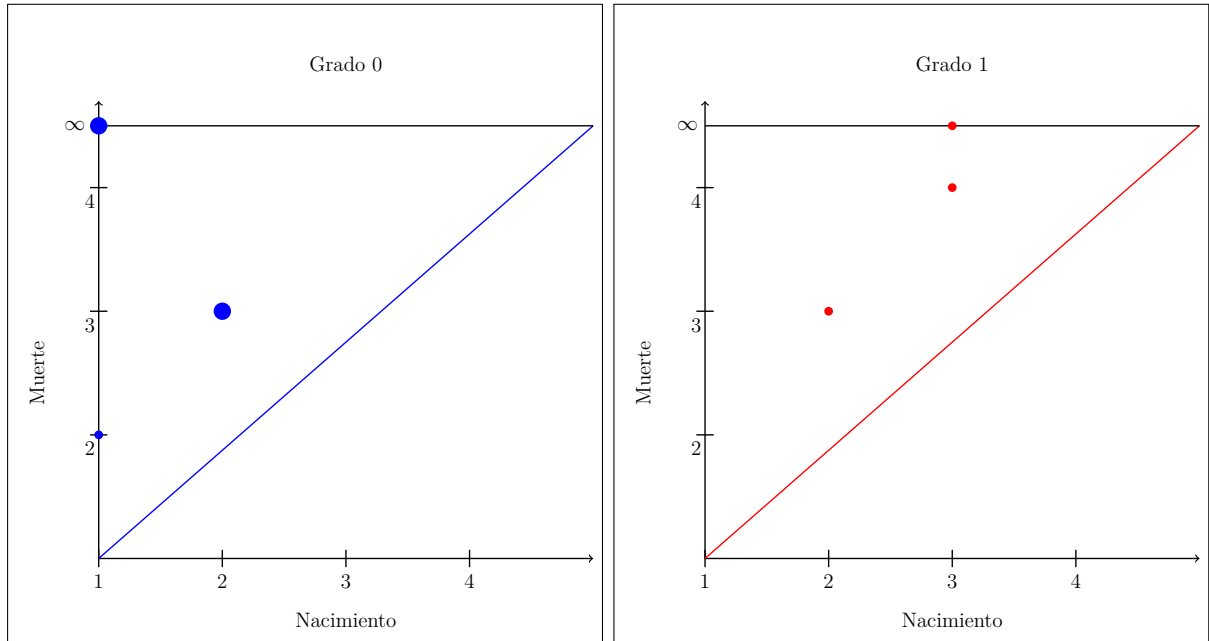


Figura 2.5: Diagramas de persistencia asociados a la filtración de \mathcal{V} .

En este caso, si nos fijamos en la Figura 2.5 diagrama izquierdo (grado 0), vemos que hay una clase de equivalencia ($[a + b]$) que nace en la filtración 1 y muere en la filtración 2, dos clases ($[f]$ y $[e]$) que nacen en la filtración 2 y mueren en la filtración 3, y dos clases que nacen en la filtración 1 y persisten. Para representar esta persistencia se ha dibujado

en el diagrama una línea horizontal que simboliza el infinito. Como se puede ver en los diagramas, para remarcar que hay varias clases que comparten nacimiento y muerte se han señalizado con un punto grueso.

Si nos fijamos ahora en la Figura 2.5 diagrama derecho (grado 1), observamos cómo hay una clase de equivalencia $([ab + bc + ac])$ que nace en la filtración 2 y muere en la filtración 3, una clase $([ac + ae + ce])$ que nace en la filtración 3 y muere en la filtración 4, y una clase $([bc + cf + bf])$ que nace en la filtración 3 y persiste. ◀

2.3 Redes neuronales. Grafos subyacentes

Ahora cambiamos completamente de tema: nos alejamos momentáneamente de la topología algebraica y nos acercamos al mundo de las ciencias de la información. En esta sección haremos una introducción hacia el otro concepto fundamental del presente trabajo: *las redes neuronales artificiales*. A diferencia de la sección anterior, en la presente sección haremos una aproximación un poco menos formal de las nociones que se introducen. Para el desarrollo de esta sección emplearemos la guía proporcionada por [6, 8, 7].

La historia del *aprendizaje profundo* y de las redes neuronales es muy amplia y queda fuera del alcance del presente trabajo. No obstante, conviene reseñar que, aunque los términos aprendizaje profundo y red neuronal nos puedan parecer algo recientes, provienen de una disciplina con un largo recorrido temporal. El origen de esta disciplina data de los años 40 de la mano de Warren McCulloch y Walter Pitts, quienes inspirados en el cerebro humano propusieron el primer modelo de red neuronal artificial. Tras ellos, en los años 50, Frank Rosenblatt propuso el primer modelo del que más tarde surgieron las redes neuronales modernas: *el perceptrón*. En esa misma época se comenzó a usar el algoritmo del *descenso del gradiente*, que se emplea para optimizar de manera automática los modelos de redes neuronales. Tras esto se alcanzaron multitud de hitos importantes, como la victoria en 1997 de la computadora Deep Blue ante el campeón mundial de ajedrez Gari Kaspárov, que hicieron avanzar la disciplina hasta nuestros días. Actualmente, gracias a la potencia de cómputo moderna la inteligencia artificial tiene multitud de aplicaciones: la conducción autónoma, la generación de imágenes realistas, la traducción automática, la detección de tumores, etc. No obstante, cabe reseñar que, al igual que con otras líneas de investigación, la inteligencia artificial (y en particular el aprendizaje profundo) sufrió varios estancamientos y contratiempos hasta llegar a nuestros días («inviernos de la IA»).

Supongamos que queremos que nuestro ordenador nos escriba con palabras un número entre 0 y 9, dado dicho número en cifras. Aparentemente es una tarea sencilla, bastará con desarrollar un programa que a partir de una entrada (la cifra) nos produzca una salida (la palabra) de entre las nueve posibilidades; resumidamente, unas pocas sentencias condicionales bastarán para obtener el resultado esperado.

Pensemos ahora en la misma tarea, pero esta vez le proporcionaremos como entrada una imagen de la cifra manuscrita. Ahora el problema es significativamente más difícil. La aproximación que propone el aprendizaje automático es dejar que el ordenador «aprenda» a hacer esa tarea. La idea intuitiva es desarrollar un programa que tome dos entradas: la

imagen y unos parámetros; y que a partir de estas entradas proporcione una salida. Tras esto, ajustar automáticamente los parámetros para mejorar el desempeño del programa. Iterando este proceso de predicción-ajuste, llegaremos a proporcionar la salida correcta. Una vez que hemos encontrado los parámetros correctos, podemos abandonar este proceso de ajuste y tendremos un programa clásico que a partir de unas entradas produce unas salidas.

Teniendo clara la idea intuitiva, vamos a ver su desarrollo hasta llegar a las redes neuronales.

Supongamos que queremos modelar la relación lineal entre una variable objetivo (y) y p variables independientes (x_1, \dots, x_p). Este modelo se conoce como regresión lineal y puede ser expresado como:

$$\hat{y} = w_1x_1 + w_2x_2 + \dots + w_px_p + b,$$

donde \hat{y} es valor estimado de y ; w_1, w_2, \dots, w_p son los pesos que indican la importancia de cada variable independiente para el modelo; y b es el término independiente. Aquí los parámetros w_1, w_2, \dots, w_p y el término independiente b toman valores en \mathbb{R} . Volviendo a la terminología introducida por la idea intuitiva, \hat{y} es nuestra salida, las variables independientes son nuestras entradas, y w_1, w_2, \dots, w_p son los parámetros a ajustar.

Para realizar este ajuste será necesario tener una métrica que nos indique si un conjunto de parámetros es mejor que otro. Para ello, usaremos el error cuadrático medio cuya expresión es la siguiente:

$$MSE(w_1, w_2, \dots, w_p, b) = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2,$$

donde n es el número de datos (cada dato es un vector de p entradas), y_i es la salida real y \hat{y}_i es la salida estimada por el modelo. Nuestro objetivo es el de minimizar $MSE(w_1, w_2, \dots, w_p, b)$ para que el modelo sea el mejor posible. Para minimizar esta función será necesario ir ajustando los parámetros correctamente. Para ello se emplea el algoritmo conocido como *el descenso del gradiente*. A continuación, veamos cómo funciona este algoritmo.

Supongamos una función continua y suave. Por ser ella continua sabemos que mapea puntos «ceranos» a puntos «ceranos», y por ser suave, sabemos que podemos aproximar valores «ceranos» a un punto de la función por una función lineal cuya pendiente sea la diferencial de la función. Este resultado se conoce como el *teorema de Taylor* (un enunciado más riguroso, así como su demostración, puede verse en [1]). El gradiente de una función en un punto nos indica cuanto aumenta o disminuye el valor de dicha función ante un pequeño cambio en su entrada. Lo que haremos será «movernos un poco» en la dirección marcada por el gradiente pero en sentido opuesto, pues nuestro objetivo es el de minimizar la función. Iterando este proceso, iremos dando pequeños «pasos» minimizando $MSE(w_1, w_2, \dots, w_p, b)$. En concreto, la actualización de los parámetros es como sigue:

$$\begin{aligned} w_i^{siguiente} &= w_i^{actual} - \alpha \frac{\partial MSE(w,b)}{\partial w_i} \\ b^{siguiente} &= b^{actual} - \alpha \frac{\partial MSE(w,b)}{\partial b} \end{aligned} \quad ,$$

donde α es el ratio de aprendizaje: un hiperparámetro que tenemos que elegir. Este paso de actualización se conoce como *propagación hacia atrás*, y el paso del cálculo de \hat{y} se conoce como *propagación hacia adelante*. Una vez hemos logrado la configuración óptima de los parámetros ya podemos emplear este modelo para la predicción de y .

Nota. Notemos que el proceso presentado logra teóricamente la convergencia al mínimo local de $MSE(w_1, w_2, \dots, w_p, b)$. Para converger de manera teórica al mínimo global son necesarias técnicas más avanzadas cuya descripción queda fuera del alcance del presente trabajo. Para más información véase [6].

Desafortunadamente, el modelo de regresión lineal es muy limitado: supone una relación lineal entre la variable objetivo y las variables independientes. Se hace necesaria una generalización que conseguiremos introduciendo una función no lineal a la regresión lineal. El primer modelo conocido con esta definición es el *perceptrón*. Este modelo se expresa de la siguiente manera:

$$\hat{y} = r(w_1x_1 + w_2x_2 + \dots + w_px_p + b),$$

donde, en general, $r(x)$ es una función no lineal que se conoce como *función de activación* y el resto de variables son como en la regresión lineal. En el caso particular del perceptrón, $r(x)$ está definida de la siguiente manera:

$$r(x) = \begin{cases} 1 & \text{si } x > 0 \\ 0 & \text{si } x \leq 0 \end{cases} \quad (2.3.1)$$

Observamos que la función $r(x)$ no es continua, por lo que no podemos aplicar el algoritmo del descenso del gradiente para la actualización de los parámetros. En su lugar se utiliza la siguiente fórmula:

$$w_i^{\text{siguiente}} = w_i^{\text{actual}} + \alpha(y - \hat{y})x_i$$

donde α es el ratio de aprendizaje, y y es la salida esperada. Obtenemos así un modelo más general que no presupone una relación lineal entre la variable objetivo y las variables independientes.

La función $r(x)$ puede ser considerada como un hiperparámetro, es decir, puede ser instanciada a una función no lineal distinta de la considerada anteriormente. Podemos, por ejemplo, considerar la función sigmoide, la tangente hiperbólica o la función ReLU (véase [6]). Dado que habitualmente se utiliza la función ReLU, vamos a detenernos brevemente a analizar su uso. La función ReLU está definida de la siguiente manera:

$$ReLU(x) = \begin{cases} x & \text{si } x > 0 \\ 0 & \text{si } x \leq 0 \end{cases} \quad (2.3.2)$$

Observamos que la función ReLU es continua pero no suave, pues no es diferenciable en $x = 0$. Por lo tanto, teóricamente no podremos asegurar que \hat{y} lo sea (y por tanto MSE tampoco) y no podremos utilizar el algoritmo del descenso del gradiente. Sin embargo, por cuestiones prácticas, este incidente no se tiene en cuenta ya que habitualmente no

se alcanza el mínimo local de \hat{y} , luego a priori, no hay problemas en que la función no sea diferenciable en dicho punto. Adicionalmente, cabe destacar que la mayoría de los algoritmos empleados para el cálculo de la derivada en el descenso del gradiente sólo calculan una de las derivadas laterales.

A fin de facilitar el entendimiento de nuestros siguientes pasos, en la Figura 2.6 representamos gráficamente la estructura subyacente del perceptrón.

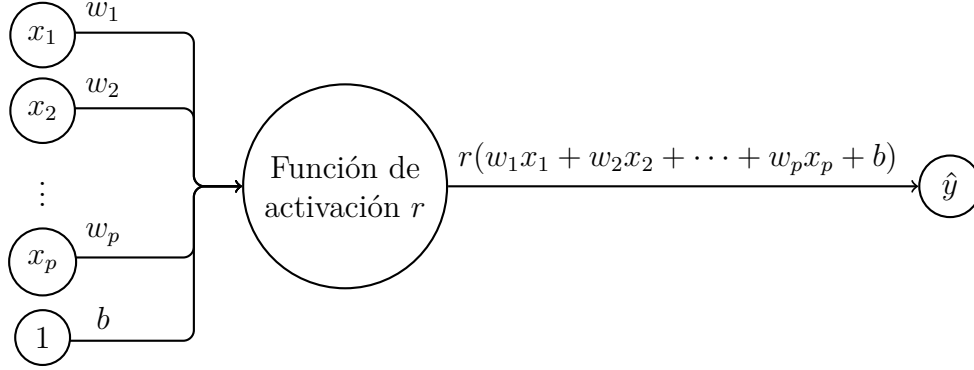


Figura 2.6: Representación gráfica de un perceptrón.

Esta representación gráfica y las limitaciones del perceptrón (para más detalles véase [11]), motivan el siguiente paso en la construcción de las *redes neuronales*. Este paso consiste en añadir más perceptrones en paralelo (nodos) y en serie (capas), consiguiendo lo que se conoce como *perceptrón multicapa*.

El perceptrón multicapa es un caso particular de arquitectura de una red neuronal. En esta arquitectura las conexiones entre nodos (neuronas) se realizan únicamente hacia adelante, no se permite la conexión entre nodos de la misma capa, y la función de activación es $r(x)$ definida en 2.3.1. Para los propósitos del presente trabajo, la función de activación no es relevante; pero sí lo es la dirección de las conexiones entre los nodos. Por lo tanto, nos ceñiremos al caso particular de las redes neuronales cuyas conexiones se realizan hacia adelante y sin conexiones dentro de la misma capa, estas redes neuronales se conocen como *redes neuronales prealimentadas*.

Nota. Aunque puede parecer que nos estamos restringiendo a un caso muy particular de las redes neuronales, el *teorema de aproximación universal* nos garantiza que una red de estas características puede aproximar cualquier función Borel-medible cuyo dominio tenga dimensión finita (para más información véase [9]).

En el presente trabajo nos interesa especialmente la estructura combinatoria asociada a las redes neuronales prealimentadas, es decir, los grafos subyacentes. En este sentido, veamos las siguientes definiciones:

Definición 2.3.1. Un *grafo* $G = (V, E)$ consiste en un conjunto finito V , cuyos elementos reciben el nombre de *vértices*, y un conjunto E de pares de elementos de V , cuyos elementos se conocen como *aristas*. Si $\{u, v\}$ es una arista de G , se dice que los vértices u y v son *adyacentes* y llamaremos a u y a v *extremos* de la arista.

Para el caso las redes neuronales prealimentadas necesitamos un caso particular de grafo:

Definición 2.3.2. Un *grafo dirigido* es un grafo en el que se asigna un orden a los extremos de cada arista. Las aristas dirigidas se denotan (u, v) , y nótese que $(u, v) \neq (v, u)$. En el caso de una arista dirigida (u, v) , se dice que u es el *origen* y al vértice v se le llama *destino* de la arista. Un grafo dirigido se dice *acíclico* si para cada vértice u , no existe una sucesión de aristas cuyo origen y destino sea u .

Así pues, vamos a estudiar los grafos dirigidos acíclicos subyacentes a las redes neuronales prealimentadas. No obstante, dadas las características de las redes neuronales prealimentadas, también podemos aproximarnos a sus grafos subyacentes desde otro punto de vista.

Como ya hemos visto, las neuronas de una red neuronal prealimentada están distribuidas por capas. Además, podemos identificar claramente estas capas, y ya hemos visto que, en el caso que nos ocupa, el flujo de información de la red se produce en un único sentido. Estas propiedades nos inducen a pensar, de una manera natural, en un orden sobre las neuronas de la red. En este sentido, veamos la otra aproximación a los grafos subyacentes de una red neuronal prealimentada:

Definición 2.3.3. Sea $G = (V, E)$ un grafo. Diremos que G es *grafo nivelado* si tomando una partición adecuada de V en subconjuntos L_1, \dots, L_h , se cumple que si $(u, v) \in E$ con $u \in L_i$ y $v \in L_j$, entonces $j = i + 1$. A los conjuntos L_1, \dots, L_h se los conoce como *niveles* y a h como *altura*.

Nota. La condición $j = i + 1$ de la definición anterior puede relajarse a $j > i$. Esta relajación implicaría que en nuestra red podrían haber saltos de nivel. Hemos optado por la primera condición pues es suficiente para los propósitos del presente trabajo.

Observamos que, la definición de grafo nivelado generalizado (admitiendo saltos de más de un nivel) y la definición de grafo dirigido acíclico son equivalentes. En efecto, un vértice u de un grafo dirigido acíclico está en el nivel L_i cuando la sucesión de aristas más larga cuyo destino es u tiene i vértices. Con lo que, siempre que tengamos una red neuronal prealimentada podremos identificar de manera natural su grafo nivelado subyacente, y por tanto, su grafo dirigido acíclico asociado.

Notemos que de la definición 2.3.3 se deduce naturalmente un orden parcial sobre los vértices del grafo: si $j > i$, $u \in L_i$ y $v \in L_j$, entonces $v > u$. Así pues, este será el orden que consideraremos para nuestras próximas definiciones y resultados.

Recapitulando, dada una red neuronal, le asociaremos un grafo dirigido acíclico (nivelado) sobre el que aplicaremos la teoría de homología vista anteriormente. A fin de facilitar la comprensión, veamos un ejemplo:

Ejemplo 2.3.1. Supondremos una red neuronal prealimentada de 2 capas y 13 neuronas. Su grafo asociado puede verse en la Figura 2.7.

Hemos etiquetado las neuronas (vértices) en orden ascendente desde la primera neurona de la capa de salida hasta la última de la capa de entrada. En las próximas representaciones seguiremos usando esta notación. Nótese que los cálculos y los resultados a

los que llegaremos en este trabajo son independientes de esta notación, seguiremos este convenio pues es el elegido en [19].

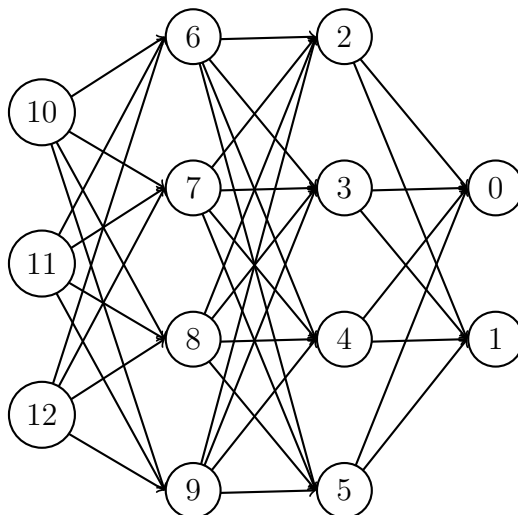


Figura 2.7: Grafo dirigido acíclico (nivelado) asociado a una red neuronal prealimentada de 13 neuronas y 2 capas.



Habiendo fijado los conceptos de red neuronal prealimentada y de su grafo asociado, no parece evidente la aplicación de la homología persistente a estas nociones. Lo que haremos será, dada una red neuronal prealimentada, construir el complejo simplicial asociado a su grafo dirigido acíclico (DAG por sus siglas en inglés); a partir del complejo simplicial construiremos un complejo simplicial filtrado; y sobre éste consideraremos la homología persistente. En esta sección ilustraremos únicamente el primer paso.

Si bien existen varias maneras de construir un complejo simplicial a partir de un DAG (véase [10]), en el presente trabajo desarrollaremos la más simple de todas ellas: la construcción del complejo simplicial *clique*. Esta construcción, aunque simple, nos servirá de inspiración para la construcción que realizaremos más adelante.

Definición 2.3.4. Definimos el *complejo simplicial clique* asociado a un grafo a partir de la construcción de sus p -símplices, $p \in \mathbb{N} \cup \{0\}$. Los p -símplices de un complejo simplicial clique están contruidos de la siguiente manera: el conjunto de vértices $\{u_0, \dots, u_p\}$ es un p -símplice si y sólo si cada par de vértices está conectado por una arista. Por simplicidad del lenguaje, nos referiremos a un complejo simplicial clique como clique.

Nota. Aunque la definición de clique es sencilla, el cálculo de un clique a partir de un grafo es un problema complejo. De hecho, este problema se conoce como *el problema del clique* y es NP-completo. Para más detalles véase [12].

Nótese que, por ser un complejo simplicial, un clique es cerrado por subconjuntos, esto es, cualquier subcomplejo simplicial de un clique es un clique. Veamos un ejemplo:

Ejemplo 2.3.2. Vamos a construir el clique asociado al ejemplo anterior. Lo denotaremos por \mathcal{V} . Atendiendo a la definición de clique y a la Figura 2.7 tendremos que:

$$\begin{aligned} \mathcal{V} = \{ & \{0\}, \{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{9\}, \{10\}, \{11\}, \{12\}, \{0, 2\}, \{0, 3\}, \\ & \{0, 4\}, \{0, 5\}, \{1, 2\}, \{1, 3\}, \{1, 4\}, \{1, 5\}, \{2, 6\}, \{2, 7\}, \{2, 8\}, \{2, 9\}, \{3, 6\}, \{3, 7\}, \{3, 8\}, \\ & \{3, 9\}, \{4, 6\}, \{4, 7\}, \{4, 8\}, \{4, 9\}, \{5, 6\}, \{5, 7\}, \{5, 8\}, \{5, 9\}, \{6, 10\}, \{6, 11\}, \{6, 12\}, \\ & \{7, 10\}, \{7, 11\}, \{7, 12\}, \{8, 10\}, \{8, 11\}, \{8, 12\}, \{9, 10\}, \{9, 11\}, \{9, 12\} \}. \end{aligned}$$

Observamos que \mathcal{V} es un clique muy particular, pues está compuesto únicamente por vértices (0-símplices) y aristas (1-símplices). Su representación gráfica puede verse en la Figura 2.8.

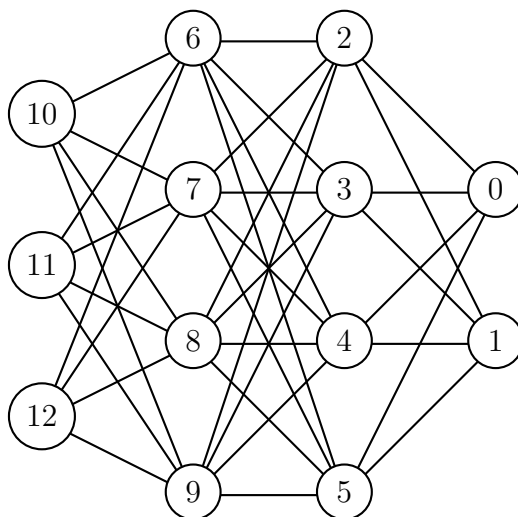


Figura 2.8: Representación gráfica del clique asociado a una red neuronal prealimentada de 13 neuronas y 2 capas.



El ejemplo anterior pone de manifiesto que considerar el clique asociado a un grafo por niveles de una red neuronal prealimentada no es suficiente, pues el resultado es un clique trivial que nos aporta muy poca información sobre la red. Reflexionando sobre ello, se hace notar que debemos hacer una extensión mediante una cierta relación de transitividad.

No obstante, estamos adelantando acontecimientos pues en el siguiente capítulo veremos detalladamente el proceso necesario para la aplicación de la homología persistente a una red neuronal prealimentada. Así pues, finalizamos este capítulo de preliminares en el que se han expuesto todas las nociones teóricas necesarias para la comprensión del artículo sobre el que versa el presente trabajo.

Capítulo 3

¿Es posible aplicar la homología persistente a las redes neuronales?

La respuesta a la pregunta del título es afirmativa, sin embargo, a lo largo de este capítulo iremos viendo que esta respuesta tiene sus matices. En este capítulo nos vamos a centrar en el desarrollo del artículo [19] sobre el que está basado el presente trabajo. La motivación principal de las técnicas que expondremos es esclarecer el funcionamiento interno de las redes neuronales prealimentadas, que habitualmente son vistas como «cajas negras». Mediante el uso de estas técnicas podremos extraer información relevante acerca de la complejidad y el grado de aprendizaje de las redes neuronales prealimentadas.

3.1 Descripción matemática

Partimos de la base del capítulo anterior, en el que se hizo notoria la necesidad de extender la noción de clique asociado a una red neuronal mediante una cierta idea de transitividad. Adicionalmente, debemos construir el complejo simplicial filtrado para poder aplicar la homología persistente.

Como bien sabemos, dada una red neuronal prealimentada, es posible asociarle un DAG. Además, aunque en el capítulo anterior no lo mencionamos, a este grafo podemos asociarle unos pesos positivos o negativos que serán los parámetros de la red neuronal prealimentada. Pensando de esta manera, tendremos un DAG con pesos w_{ij} , con w_{ij} el parámetro de la red entre u_i y u_j . Notemos que $w_{ij} = 0$ si y sólo si u_i y u_j no están conectadas.

Observamos que, en cierto modo, un peso positivo indica una predisposición para pasar de u_i a u_j , mientras que un peso negativo indica una obstrucción. Puesto que la complejidad de la red la medimos en términos de esa predisposición de pasar de una neurona a otra, en la siguiente definición obviaremos los pesos negativos. Para ello, tomamos la función parte positiva del peso asociado: $w_{ij}^+ := \max\{w_{ij}, 0\}$. Asimismo, adoptamos el axioma de que la predisposición de pasar de una neurona u_i a otra u_j hay que contextualizarla en el sentido de compararla con la predisposición de pasar de u_i a otra neurona de destino u_k . En cierto sentido «probabilizamos» nuestros pesos. Con esto en mente, damos la siguiente definición:

Definición 3.1.1. Definimos la *importancia* de u_i (salida) para u_j (llegada) como:

$$R_{ij} = \begin{cases} 1 & \text{si } i = j \\ w_{ij}^+ / \sum_{k, k \neq j} w_{kj}^+ & \text{si } i \neq j \end{cases}$$

Observamos que la importancia de una neurona para sí misma es de 1, y la importancia entre neuronas distintas es la proporción del peso entre ellas con respecto al resto de pesos de la neurona de llegada. Además, es claro que la importancia entre dos neuronas conectadas es un valor entre 0 y 1.

La noción de importancia entre neuronas nos habilita la construcción de un complejo simplicial filtrado a partir del DAG asociado a la red. Sin embargo, teniendo en mente la idea de la transitividad, todavía tenemos que extender la definición de la importancia entre neuronas para aquellas que no estén directamente conectadas. Así pues, damos la siguiente definición:

Definición 3.1.2. Consideremos las neuronas u_0 y u_2 conectadas por el camino: $u_2 \rightarrow u_1 \rightarrow u_0$, la importancia de u_2 para u_0 es, según el camino entre ellas, $R_{21} \cdot R_{10}$. Por lo tanto, definimos la *importancia extendida* entre neuronas, y la denotamos por \overline{R}_{ij} , como:

$$\overline{R}_{ij} = \max\{R_{u_i u_{m_1}} \cdots R_{u_{m_n} u_j} \mid (u_i, u_{m_1}, \dots, u_{m_n}, u_j) \in C_{ij}\}$$

donde C_{ij} denota el conjunto de todos los posibles caminos de u_i a u_j .

Nota. Nótese que es posible definir \overline{R}_{ij} considerando funciones distintas al máximo, tales como la suma o la suma cuadrática. El autor ([19]) justifica la elección del máximo por eficiencia computacional. Además, notemos que el máximo tiene una ventaja evidente: mantiene los números de importancia entre 0 y 1.

Como ya mencionamos en el capítulo anterior, numeraremos los nodos (vértices) de un DAG con pesos, asociado a una red neuronal prealimentada, en orden ascendente, desde las neuronas de llegada hasta las de salida. Veamos, a continuación, un ejemplo sencillo para interiorizar estas definiciones que serán clave a lo largo de este capítulo.

Ejemplo 3.1.1. Supondremos la representación de una red neuronal, que puede verse en la Figura 3.1, con sus correspondientes pesos.

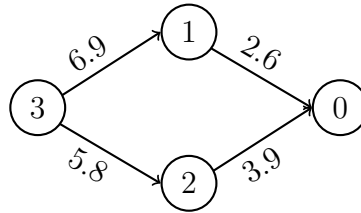


Figura 3.1: Representación de una red neuronal de 4 neuronas y 1 capa.

Tal y como vemos, las neuronas ya han sido ordenadas de manera correcta. Además, en este caso, todos los pesos son positivos, por lo que no nos tenemos que preocupar escoger la parte positiva. Vamos a calcular algunas importancias entre neuronas:

$$\begin{aligned} R_{31} &= \frac{6.9}{6.9} = 1 & R_{32} &= \frac{5.8}{5.8} = 1 \\ R_{10} &= \frac{2.6}{6.5} = 0.4 & R_{20} &= \frac{3.9}{6.5} = 0.6 \\ \overline{R}_{30} &= \max\{R_{31} \cdot R_{10}, R_{32} \cdot R_{20}\} = R_{32} \cdot R_{20} = 0.6 \end{aligned}$$



El ejemplo anterior pone de manifiesto la intuición detrás de la definición de la importancia entre neuronas: lo que hace es medir la aportación de la neurona emisora, u_i , con respecto al resto de neuronas emisoras de u_j .

La definición de importancia extendida nos habilita la construcción del complejo simplicial filtrado asociado a un DAG con pesos de una manera muy natural. El primer paso será definir el complejo simplicial asociado, pues no vamos a usar un clique. Esencialmente, vamos a definir umbrales de importancia para determinar los complejos simpliciales asociados a una red neuronal.

Definición 3.1.3. Sea $0 \leq t \leq 1$ un parámetro real. Definimos los p -símplices asociados a un DAG con pesos a partir del conjunto de vértices del DAG (que por simplificar notación denotaremos por \mathcal{V}_0) como sigue:

$$\mathcal{V}_p^t = \begin{cases} \mathcal{V}_0 & \text{si } p = 0 \\ \{\{u_{a_0}, \dots, u_{a_p}\} \mid u_{a_i} \in \mathcal{V}_0, \overline{R_{a_i a_j}} \geq t, \forall a_i > a_j\} & \text{si } p \geq 1 \end{cases}$$

donde cada u_{a_i} pertenece al nivel a_i del DAG y $a_0 < \dots < a_i < \dots < a_p$.

Ahora que ya tenemos definidos los p -símplices, vamos con la construcción del complejo simplicial asociado. Para ello damos el siguiente resultado:

Proposición 1. Sea $\mathcal{V}_0 = \{u_0, \dots, u_n\}$ un conjunto finito, y $\{w_{ij}\}_{0 \leq j \leq i \leq n}$ un conjunto de números reales. Sea $\overline{R_{ij}}$ la importancia entre neuronas definida en 3.1.2, y \mathcal{V}_p^t los p -símplices definidos en 3.1.3 con t parámetro real entre 0 y 1. Entonces $\mathcal{V}^t = \bigcup_{s=0}^{s=n} \mathcal{V}_s^t$ es un complejo simplicial abstracto.

Demostración. Supongamos hipótesis generales.

Para probar que \mathcal{V}^t es un complejo simplicial abstracto debemos ver:

1. $u \in \mathcal{V}_0$ implica que $\{u\} \in \mathcal{V}^t$
2. $\sigma \in \mathcal{V}^t$ y $\tau \subset \sigma$ implica que $\tau \in \mathcal{V}^t$

Notemos que la primera propiedad se deduce inmediatamente de la definición 3.1.3 y de la definición de \mathcal{V}^t .

Así pues, vamos a probar la segunda propiedad: $\sigma = \{u_{a_0}, \dots, u_{a_p}\} \in \mathcal{V}^t$ implica que $\overline{R_{a_i a_j}} \geq t, \forall a_i \geq a_j$. Ahora sea $\tau \subset \sigma$, entonces, $\tau = \{u_{b_0}, \dots, u_{b_q}\}$, y como $\{b_0, \dots, b_q\} \subset \{a_0, \dots, a_p\}$, se tendrá que $\overline{R_{b_i b_j}} \geq t, \forall b_i \geq b_j$. El resultado se sigue inmediatamente. \square

Ahora que ya tenemos construido nuestro complejo simplicial \mathcal{V} procedemos con la construcción del complejo simplicial filtrado. Para ello, una vez más, damos el siguiente resultado:

Proposición 2. Sea $(t_i)_{i=1}^n$ una sucesión, monótona decreciente, de números reales entre 1 y 0, e indexada sobre los naturales. Entonces $\mathcal{V}_0 = \emptyset$ y $\mathcal{V}_i = \mathcal{V}^{t_i}$ con $1 \leq i \leq n$, es un complejo simplicial filtrado.

Demostración. Supongamos hipótesis generales.

Por la proposición anterior, sabemos que \mathcal{V}^{t_n} es un complejo simplicial. Ahora bien, $t_i > t_j$ implica que $\mathcal{V}_p^{t_i} \subset \mathcal{V}_p^{t_j}$ por la definición 3.1.3. Entonces $\emptyset = \mathcal{V}_0 \subset \mathcal{V}_1 \subset \dots \subset \mathcal{V}_n = \mathcal{V}^{t_n}$. Se sigue inmediatamente el resultado. \square

Hasta ahora, hemos conseguido construir el complejo simplicial filtrado asociado a un DAG con pesos, con lo que podemos pensar que ya estamos listos para aplicar la homología persistente y extraer conclusiones. No obstante, nuestro análisis del artículo de investigación [19] nos lleva a considerar una forma diferente de definir complejos simpliciales a partir de DAG con pesos.

Si leemos detenidamente las definiciones de la presente sección, observamos que la definición 3.1.2 tiene un pequeño defecto: no se especifica el significado exacto de C_{ij} . Este pequeño defecto, que en un principio puede pasar desapercibido, supone que haya al menos dos maneras distintas de calcular los p -símplices dado un DAG con pesos.

Esta disyuntiva da lugar a las dos interpretaciones posibles sobre el cálculo de los p -símplices, las hemos bautizado como: la interpretación *local* y la interpretación *global*. De los mensajes que hemos podido intercambiar con los autores de [19], hemos constatado que los cálculos que realizan se corresponden con la interpretación que hemos bautizado como *local*.

Veamos un pequeño ejemplo que ilustre la diferencia entre ambas interpretaciones.

Ejemplo 3.1.2. Supondremos la representación de una red neuronal con las importancias entre neuronas ya calculadas que puede verse en la Figura 3.2.

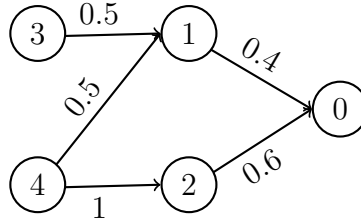


Figura 3.2: Representación de una red neuronal de 5 neuronas y 1 capa.

Vamos calcular $\mathcal{V}_2^{0.4}$ para ver las diferencias entre las dos interpretaciones. En primer lugar, listamos los posibles 2-símplices:

$$\{4, 2, 0\}, \{4, 1, 0\}, \{3, 1, 0\}$$

- Interpretación local

Como la importancia entre 4-2, 2-0 y 4-0 es mayor que 0.4, entonces $\{4, 2, 0\} \in \mathcal{V}_2^{0.4}$, donde importancia entre 4-0 viene dada por:

$$\overline{R_{40}} = \max\{R_{42} \cdot R_{20}\} = R_{42} \cdot R_{20} = 0.6$$

Siguiendo la misma regla tenemos que $\{4, 1, 0\}, \{3, 1, 0\} \notin \mathcal{V}_2^{0.4}$. Por lo tanto, $\mathcal{V}_2^{0.4} = \{\{4, 2, 0\}\}$.

- Interpretación global

Razonando igual que antes, $\{4, 2, 0\} \in \mathcal{V}_2^{0.4}$. Ahora bien, como la importancia entre 4-1 y 1-0 es mayor o igual que 0.4, y

$$\overline{R_{40}} = \max\{R_{42} \cdot R_{20}, R_{41} \cdot R_{10}\} = R_{42} \cdot R_{20} = 0.6$$

Entonces $\{4, 1, 0\} \in \mathcal{V}_2^{0.4}$. Sin embargo, $\{3, 2, 0\} \notin \mathcal{V}_2^{0.4}$ y así, $\mathcal{V}_2^{0.4} = \{\{4, 2, 0\}, \{4, 1, 0\}\}$



El ejemplo anterior pone de manifiesto la principal diferencia entre ambas interpretaciones: en la primera, el máximo se calcula sobre los caminos que aparecen en el p -símplice; en la segunda, el máximo se calcula sobre todos los caminos posibles entre el vértice de origen y de destino.

Nota. Mientras que la interpretación global es consistente para los 1-símplices (aristas), la interpretación local no lo es. Es decir, si seguimos el razonamiento local, no podremos calcular una arista entre dos vértices no conectados, pues no existirá camino local sobre el que tomar el máximo. Este hecho imposibilitaría el cálculo del complejo simplicial asociado a la red. Por lo tanto, para el cálculo en la interpretación local se calcula el producto a través de cada camino, y se adopta el criterio de que cuando ese producto alcanza el umbral establecido, el símplice formado por los nodos de ese camino y todos sus subsímplices pertenecen a nuestro complejo simplicial. Notamos que esta interpretación permite construir los complejos simpliciales sin realizar el cálculo de $\overline{R_{ij}}$ que introducen los autores de [19].

A lo largo de las siguientes secciones veremos más en detalle la diferencia entre ambas interpretaciones y cómo afecta esta diferencia al cálculo de la homología persistente.

3.2 Interpretación global. Ejemplo

A lo largo de esta sección desarrollaremos el cálculo de la homología persistente en el ejemplo propuesto en [19] mediante la interpretación que hemos bautizado como global. Comenzamos con esta interpretación pues es la que hicimos en un primer momento y es la que consideramos natural.

En primer lugar, vamos a detenernos momentáneamente a analizar la interpretación global. Si bien esta interpretación no es la original, como ya mencionamos, esta interpretación es consistente para el cálculo de los 1-símplices, y en general, para cualesquiera dos neuronas no conectadas directamente. Adicionalmente, esta interpretación es coherente con la definición de clique y con la definición 3.1.2, donde C_{ij} se considera globalmente: se toma el DAG inicial y se le calcula el cierre transitivo. Sobre este nuevo DAG se realizan el resto de cálculos.

No obstante, aunque esta interpretación nos pueda parecer más coherente desde el punto de vista teórico, habrá que realizar una serie de experimentos y comparativas para

validar la información obtenida siguiendo esta interpretación. A continuación describimos, de manera general, los pasos que hemos seguido para el cálculo de los símlices bajo esta interpretación:

- 1) Se calculan recursivamente todos los posibles caminos para un vértice de origen.
- 2) Se calcula la importancia de cada camino, es decir, la importancia entre el vértice de origen y de destino de la siguiente manera: se van multiplicando las importancias entre cada par de vértices; y si dos caminos comparten un vértice intermedio, entonces la importancia hasta ese vértice será la máxima producida por ambos caminos hasta ese vértice. La importancia obtenida tras esta multiplicación será la importancia del camino, y por ser un número entre 0 y 1, todas las importancias intermedias serán mayores o iguales que la importancia del camino.
- 3) Habiendo calculado las importancias de cada camino, se descartan aquellos caminos que no superan el umbral fijado.
- 4) Como cada camino constituye un símplex maximal, se calculan todos los subsímlices asociados al camino.

Aunque no lo hemos mencionado, un DAG con pesos no es más que un DAG al que le asociamos una matriz de pesos (recordemos que, en el caso que nos ocupa, estos pesos vienen dados por la importancia extendida entre neuronas). Sobre esta matriz de pesos y sobre una lista de adyacencia se siguen una serie de pasos para el cálculo del complejo simplicial filtrado. Veremos más en detalle estos pasos en la siguiente sección.

Habiendo descrito de manera general la interpretación global, vamos a aplicarla al ejemplo propuesto en [19], en el que vamos a calcular los números de Betti y vamos a ver los diagramas correspondientes.

Ejemplo 3.2.1. Supondremos la representación de una red neuronal con las importancias entre neuronas ya calculadas que puede verse en la Figura 3.3.

Dada la anterior representación, tenemos la matriz de pesos asociada:

$$M = \begin{pmatrix} 1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 \\ 0.3 & 1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 \\ 0.1 & 0 & 1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & 0 \\ 0.6 & 0 & 0 & 1 & 0 & \dots & \dots & \dots & \dots & \dots & 0 \\ 0 & 0.2 & 0 & 0 & 1 & 0 & \dots & \dots & \dots & \dots & 0 \\ 0 & 0.8 & 0.2 & 0 & 0 & 1 & 0 & \dots & \dots & \dots & 0 \\ 0 & 0 & 0.8 & 0.7 & 0 & 0 & 1 & 0 & \dots & \dots & 0 \\ 0 & 0 & 0 & 0.3 & 0 & 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0.2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.8 & 0.7 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.3 & 1 & 0 & 0 & 1 \end{pmatrix}$$

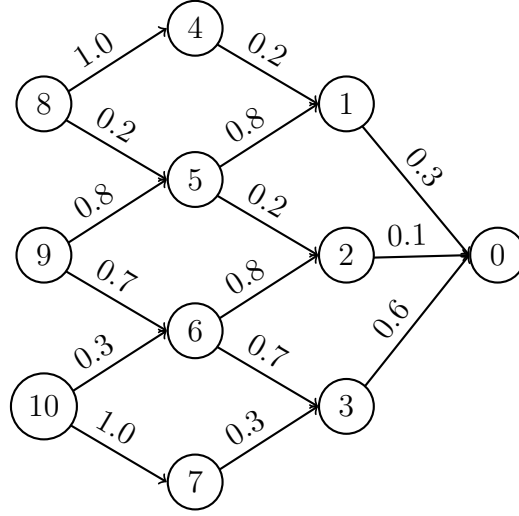


Figura 3.3: Representación de una red neuronal de 11 neuronas y 2 capas.

donde cada entrada de la matriz, a_{ij} , es la importancia de la neurona i para la neurona j ($i \geq j$).

Ahora vamos con el cálculo del complejo simplicial filtrado asociado. Para ello vamos a ilustrar unos cuantos pasos en la filtración con los correspondientes números de Betti asociados. Tras esto, mostraremos los complejos simpliciales asociados a cada filtración. Podemos ver estos pasos en la Figura 3.4.

Para una mayor claridad, se ha omitido el dibujo de algunos símlices. No obstante, damos una lista completa de los complejos simpliciales de cada filtración:

- Filtración 0 ($t=1.0$):

$$\begin{aligned} - \mathcal{V}_0^{1.0} &= \{\{10\}, \{9\}, \{8\}, \{7\}, \{6\}, \{5\}, \{4\}, \{3\}, \{2\}, \{1\}, \{0\}\} \\ - \mathcal{V}_1^{1.0} &= \{\{10, 7\}, \{8, 4\}\} \end{aligned}$$

A partir de ahora omitiremos \mathcal{V}_0 pues se mantiene igual para todos los pasos en la filtración.

- Filtración 3 ($t=0.7$):

$$- \mathcal{V}_1^{0.7} = \{\{10, 7\}, \{9, 6\}, \{9, 5\}, \{8, 4\}, \{6, 3\}, \{6, 2\}, \{5, 1\}\}$$

- Filtración 4 ($t=0.6$):

$$\begin{aligned} - \mathcal{V}_1^{0.6} &= \{\{10, 7\}, \{9, 6\}, \{9, 5\}, \{9, 1\}, \{8, 4\}, \{6, 3\}, \{6, 2\}, \{5, 1\}, \{3, 0\}\} \\ - \mathcal{V}_2^{0.6} &= \{\{9, 5, 1\}\} \end{aligned}$$

- Filtración 6 ($t=0.4$):

$$- \mathcal{V}_1^{0.4} = \{\{10, 7\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{8, 4\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 1\}, \{3, 0\}\}$$

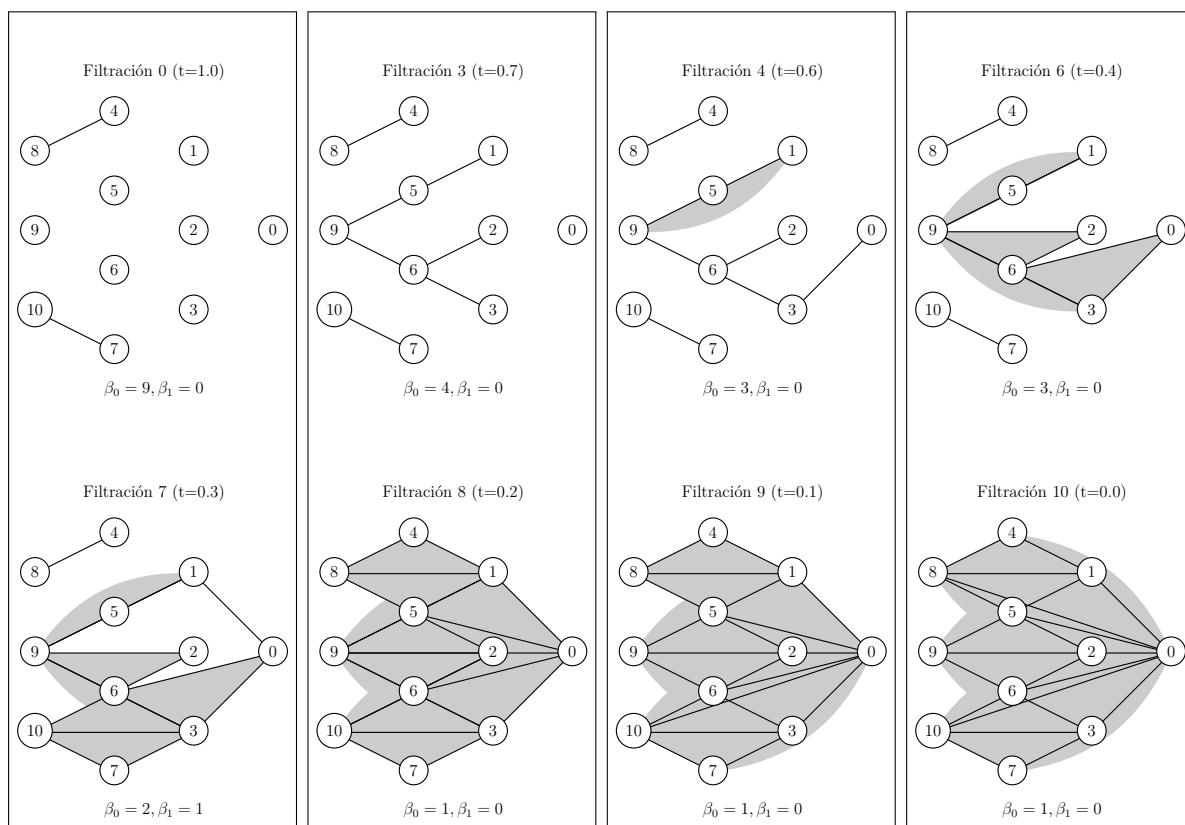


Figura 3.4: Pasos en la filtración del complejo simplicial asociado a la red y números de Betti correspondientes.

- $\mathcal{V}_2^{0.4} = \{\{9, 6, 3\}, \{9, 6, 2\}, \{9, 5, 1\}, \{6, 3, 0\}\}$
- Filtración 7 (t=0.3):
 - $\mathcal{V}_1^{0.3} = \{\{10, 7\}, \{10, 6\}, \{10, 3\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{8, 4\}, \{7, 3\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 1\}, \{3, 0\}, \{1, 0\}\}$
 - $\mathcal{V}_2^{0.3} = \{\{10, 7, 3\}, \{10, 6, 3\}, \{9, 6, 3\}, \{9, 6, 2\}, \{9, 5, 1\}, \{6, 3, 0\}\}$
- Filtración 8 (t=0.2):
 - $\mathcal{V}_1^{0.2} = \{\{10, 7\}, \{10, 6\}, \{10, 3\}, \{10, 2\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{9, 0\}, \{8, 5\}, \{8, 4\}, \{8, 1\}, \{7, 3\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 2\}, \{5, 1\}, \{5, 0\}, \{4, 1\}, \{3, 0\}, \{1, 0\}\}$
 - $\mathcal{V}_2^{0.2} = \{\{10, 7, 3\}, \{10, 6, 3\}, \{10, 6, 2\}, \{9, 6, 3\}, \{9, 6, 2\}, \{9, 6, 0\}, \{9, 5, 2\}, \{9, 5, 1\}, \{9, 5, 0\}, \{9, 3, 0\}, \{9, 1, 0\}, \{8, 5, 1\}, \{8, 4, 1\}, \{6, 3, 0\}, \{5, 1, 0\}\}$
 - $\mathcal{V}_3^{0.2} = \{\{9, 6, 3, 0\}, \{9, 5, 1, 0\}\}$
- Filtración 9 (t=0.1):
 - $\mathcal{V}_1^{0.1} = \{\{10, 7\}, \{10, 6\}, \{10, 3\}, \{10, 2\}, \{10, 0\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{9, 0\}, \{8, 5\}, \{8, 4\}, \{8, 1\}, \{7, 3\}, \{7, 0\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 2\}, \{5, 1\}, \{5, 0\}, \{4, 1\}, \{3, 0\}, \{2, 0\}, \{1, 0\}\}$
 - $\mathcal{V}_2^{0.1} = \{\{10, 7, 3\}, \{10, 7, 0\}, \{10, 6, 3\}, \{10, 6, 2\}, \{10, 6, 0\}, \{10, 3, 0\}, \{10, 2, 0\}, \{9, 6, 3\}, \{9, 6, 2\}, \{9, 6, 0\}, \{9, 5, 2\}, \{9, 5, 1\}, \{9, 5, 0\}, \{9, 3, 0\}, \{9, 2, 0\}, \{9, 1, 0\}, \{8, 5, 1\}, \{8, 4, 1\}, \{7, 3, 0\}, \{6, 3, 0\}, \{6, 2, 0\}, \{5, 2, 0\}, \{5, 1, 0\}\}$
 - $\mathcal{V}_3^{0.1} = \{\{10, 7, 3, 0\}, \{10, 6, 3, 0\}, \{10, 6, 2, 0\}, \{9, 6, 3, 0\}, \{9, 6, 2, 0\}, \{9, 5, 2, 0\}, \{9, 5, 1, 0\}\}$
- Filtración 10 (t=0.0):
 - $\mathcal{V}_1^{0.0} = \{\{10, 7\}, \{10, 6\}, \{10, 3\}, \{10, 2\}, \{10, 0\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{9, 0\}, \{8, 5\}, \{8, 4\}, \{8, 2\}, \{8, 1\}, \{8, 0\}, \{7, 3\}, \{7, 0\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 2\}, \{5, 1\}, \{5, 0\}, \{4, 1\}, \{4, 0\}, \{3, 0\}, \{2, 0\}, \{1, 0\}\}$
 - $\mathcal{V}_2^{0.0} = \{\{10, 7, 3\}, \{10, 7, 0\}, \{10, 6, 3\}, \{10, 6, 2\}, \{10, 6, 0\}, \{10, 3, 0\}, \{10, 2, 0\}, \{9, 6, 3\}, \{9, 6, 2\}, \{9, 6, 0\}, \{9, 5, 2\}, \{9, 5, 1\}, \{9, 5, 0\}, \{9, 3, 0\}, \{9, 2, 0\}, \{9, 1, 0\}, \{8, 5, 2\}, \{8, 5, 1\}, \{8, 5, 0\}, \{8, 4, 1\}, \{8, 4, 0\}, \{8, 2, 0\}, \{8, 1, 0\}, \{7, 3, 0\}, \{6, 3, 0\}, \{6, 2, 0\}, \{5, 2, 0\}, \{5, 1, 0\}, \{4, 1, 0\}\}$
 - $\mathcal{V}_3^{0.0} = \{\{10, 7, 3, 0\}, \{10, 6, 3, 0\}, \{10, 6, 2, 0\}, \{9, 6, 3, 0\}, \{9, 6, 2, 0\}, \{9, 5, 2, 0\}, \{9, 5, 1, 0\}, \{8, 5, 2, 0\}, \{8, 5, 1, 0\}, \{8, 4, 1, 0\}\}$

Para el cálculo de las anteriores listas se han empleado una serie de algoritmos y programas que detallaremos en la Sección 3.3.

Finalmente, los diagramas de barras y persistencia asociados al complejo simplicial filtrado representado en la Figura 3.4 pueden verse en la Figura 3.5.

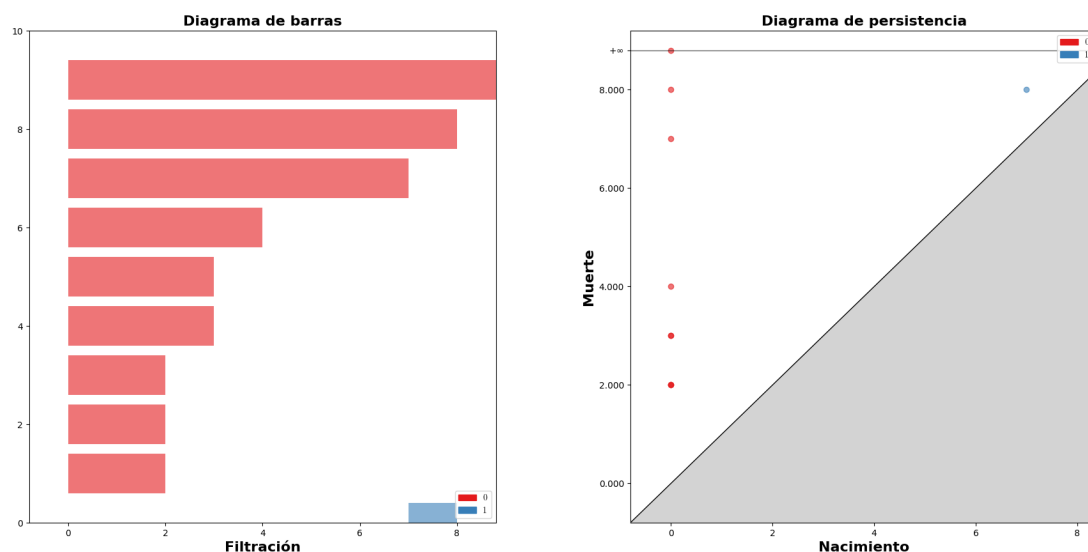


Figura 3.5: Diagramas de barras y persistencia asociados a la red 3.3.

Observamos que estos diagramas son ligeramente distintos que los presentados en el capítulo de preliminares. Esto es debido a que para el cálculo de éstos se ha empleado una librería que los produce automáticamente. Veremos esto en más detalle en la Sección 3.3.

Si observamos el diagrama izquierdo (diagrama de barras), vemos que en la filtración 0 nacen 9 clases de equivalencia (componentes conexas), y que a lo largo de las siguientes filtraciones van muriendo hasta que sólo queda una que persiste. Podemos seguir esta evolución si nos vamos fijando en el número β_0 en la Figura 3.4. También podemos observar que en la filtración 7 nace una clase de equivalencia (agujero 2-dimensional) y muere en la filtración 8. La evolución de esta clase la podemos seguir viendo la Figura 3.4 y fijándonos en el número β_1 .

Observando ahora el diagrama derecho (diagrama de persistencia) vemos la misma información pero con otra representación. En este caso se aprecian 6 puntos rojos que nacen en la filtración 0 y van muriendo a lo largo de las siguientes filtraciones, excepto uno que persiste. Estos puntos se corresponden con las componentes conexas de la filtración. A diferencia de los diagramas presentados en el capítulo de preliminares, en este caso, cuando varias clases nacen y mueren en la misma filtración se superponen sus puntos correspondientes en el diagrama dando lugar a puntos más nítidos en el diagrama. También observamos un único punto azul que se corresponde con la clase de equivalencia que da lugar a un agujero 2-dimensional, esta clase nace en la filtración 7 y muere en la filtración 8. ◀

En el ejemplo anterior, podemos apreciar que, aunque esta interpretación sea más coherente con la teoría desarrollada, la información que nos aporta sobre la red es un tanto

escasa, pues los agujeros p -dimensionales colapsan muy rápido. Esto se ve claramente reflejado en la Figura 3.5. Por ello, será conveniente hacer una serie de experimentos con el fin de probar el desempeño de esta herramienta.

3.3 Interpretación local. Algoritmos y programas

En esta sección veremos cómo funciona la interpretación que propone el autor (que se corresponde con la que hemos bautizado como local) mediante su desempeño en el mismo ejemplo que hemos visto en la sección anterior. Además, aprovecharemos para describir más detalladamente los algoritmos y programas que hemos empleado para el cálculo automático de la homología persistente en este ejemplo. Siguiendo esta interpretación, hemos realizado el cálculo de los símlices mediante estos pasos:

- 1) Se calculan recursivamente todos los posibles caminos para un vértice de origen.
- 2) Se calcula la importancia de cada camino multiplicando la importancia entre cada par vértices, sin tener en cuenta si existe otro camino que comparta vértices.
- 3) Al igual que en la interpretación global, se descartan aquellos caminos que no superan el umbral fijado.
- 4) Se calculan todos los subsímlices asociados a cada camino.

A continuación veamos el ejemplo tal y cómo lo propone el autor en [19]:

Ejemplo 3.3.1. Supondremos la red que puede verse en la Figura 3.3. Como en el ejemplo anterior ya hemos visto una representación de la misma y su matriz asociada, esta vez los omitimos.

Vamos directamente con el cálculo del complejo simplicial filtrado asociado. Al igual que antes, vamos a ilustrar unos cuantos pasos en la filtración con los correspondientes números de Betti asociados. Tras esto, mostraremos los complejos simpliciales asociados a cada filtración. La ilustración de los pasos en la filtración puede verse en la Figura 3.6.

Para una mayor claridad, se ha omitido el dibujo de algunos símlices. Al igual que antes, damos una lista completa de los complejos simpliciales de cada filtración:

- Filtración 0 ($t=1.0$):

$$\begin{aligned} - \mathcal{V}_0^{1.0} &= \{\{10\}, \{9\}, \{8\}, \{7\}, \{6\}, \{5\}, \{4\}, \{3\}, \{2\}, \{1\}, \{0\}\} \\ - \mathcal{V}_1^{1.0} &= \{\{10, 7\}, \{8, 4\}\} \end{aligned}$$

A partir de ahora omitiremos \mathcal{V}_0 pues se mantiene igual para todos los pasos en la filtración.

- Filtración 3 ($t=0.7$):

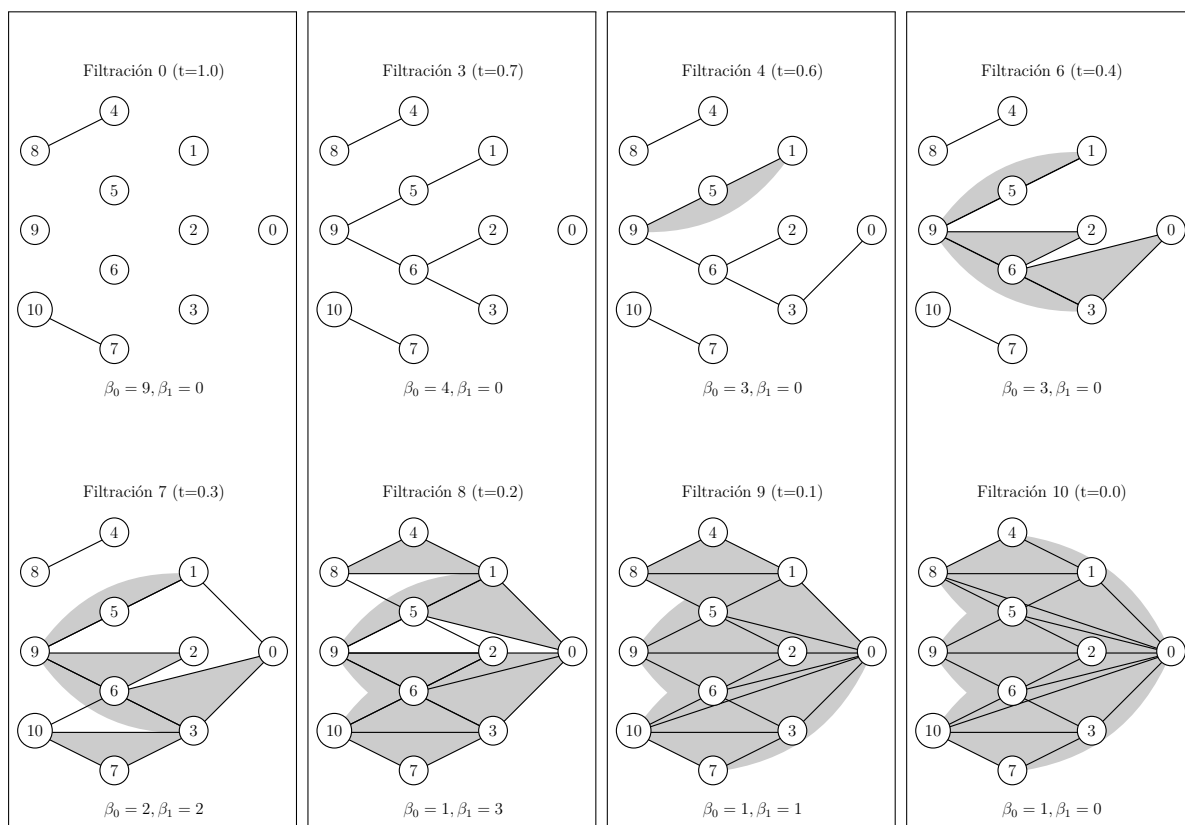


Figura 3.6: Pasos en la filtración del complejo simplicial asociado a la red y números de Betti correspondientes.

- $\mathcal{V}_1^{0.7} = \{\{10, 7\}, \{9, 6\}, \{9, 5\}, \{8, 4\}, \{6, 3\}, \{6, 2\}, \{5, 1\}\}$
- Filtración 4 (t=0.6):
 - $\mathcal{V}_1^{0.6} = \{\{10, 7\}, \{9, 6\}, \{9, 5\}, \{9, 1\}, \{8, 4\}, \{6, 3\}, \{6, 2\}, \{5, 1\}, \{3, 0\}\}$
 - $\mathcal{V}_2^{0.6} = \{\{9, 5, 1\}\}$
- Filtración 6 (t=0.4):
 - $\mathcal{V}_1^{0.4} = \{\{10, 7\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{8, 4\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 1\}, \{3, 0\}\}$
 - $\mathcal{V}_2^{0.4} = \{\{9, 6, 3\}, \{9, 6, 2\}, \{9, 5, 1\}, \{6, 3, 0\}\}$
- Filtración 7 (t=0.3):
 - $\mathcal{V}_1^{0.3} = \{\{10, 7\}, \{10, 6\}, \{10, 3\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{8, 4\}, \{7, 3\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 1\}, \{3, 0\}, \{1, 0\}\}$
 - $\mathcal{V}_2^{0.3} = \{\{10, 7, 3\}, \{9, 6, 3\}, \{9, 6, 2\}, \{9, 5, 1\}, \{6, 3, 0\}\}$
- Filtración 8 (t=0.2):
 - $\mathcal{V}_1^{0.2} = \{\{10, 7\}, \{10, 6\}, \{10, 3\}, \{10, 2\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{9, 0\}, \{8, 5\}, \{8, 4\}, \{8, 1\}, \{7, 3\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 2\}, \{5, 1\}, \{5, 0\}, \{4, 1\}, \{3, 0\}, \{1, 0\}\}$
 - $\mathcal{V}_2^{0.2} = \{\{10, 7, 3\}, \{10, 6, 3\}, \{10, 6, 2\}, \{9, 6, 3\}, \{9, 6, 2\}, \{9, 6, 0\}, \{9, 5, 1\}, \{9, 3, 0\}, \{8, 4, 1\}, \{6, 3, 0\}, \{5, 1, 0\}\}$
 - $\mathcal{V}_3^{0.2} = \{\{9, 6, 3, 0\}\}$
- Filtración 9 (t=0.1):
 - $\mathcal{V}_1^{0.1} = \{\{10, 7\}, \{10, 6\}, \{10, 3\}, \{10, 2\}, \{10, 0\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{9, 0\}, \{8, 5\}, \{8, 4\}, \{8, 1\}, \{7, 3\}, \{7, 0\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 2\}, \{5, 1\}, \{5, 0\}, \{4, 1\}, \{3, 0\}, \{2, 0\}, \{1, 0\}\}$
 - $\mathcal{V}_2^{0.1} = \{\{10, 7, 3\}, \{10, 7, 0\}, \{10, 6, 3\}, \{10, 6, 2\}, \{10, 6, 0\}, \{10, 3, 0\}, \{9, 6, 3\}, \{9, 6, 2\}, \{9, 6, 0\}, \{9, 5, 2\}, \{9, 5, 1\}, \{9, 5, 0\}, \{9, 3, 0\}, \{9, 1, 0\}, \{8, 5, 1\}, \{8, 4, 1\}, \{7, 3, 0\}, \{6, 3, 0\}, \{5, 1, 0\}\}$
 - $\mathcal{V}_3^{0.1} = \{\{10, 7, 3, 0\}, \{10, 6, 3, 0\}, \{9, 6, 3, 0\}, \{9, 5, 1, 0\}\}$
- Filtración 10 (t=0.0):
 - $\mathcal{V}_1^{0.0} = \{\{10, 7\}, \{10, 6\}, \{10, 3\}, \{10, 2\}, \{10, 0\}, \{9, 6\}, \{9, 5\}, \{9, 3\}, \{9, 2\}, \{9, 1\}, \{9, 0\}, \{8, 5\}, \{8, 4\}, \{8, 2\}, \{8, 1\}, \{8, 0\}, \{7, 3\}, \{7, 0\}, \{6, 3\}, \{6, 2\}, \{6, 0\}, \{5, 2\}, \{5, 1\}, \{5, 0\}, \{4, 1\}, \{4, 0\}, \{3, 0\}, \{2, 0\}, \{1, 0\}\}$

- $\mathcal{V}_2^{0,0} = \{\{10, 7, 3\}, \{10, 7, 0\}, \{10, 6, 3\}, \{10, 6, 2\}, \{10, 6, 0\}, \{10, 3, 0\}, \{10, 2, 0\}, \{9, 6, 3\}, \{9, 6, 2\}, \{9, 6, 0\}, \{9, 5, 2\}, \{9, 5, 1\}, \{9, 5, 0\}, \{9, 3, 0\}, \{9, 2, 0\}, \{9, 1, 0\}, \{8, 5, 2\}, \{8, 5, 1\}, \{8, 5, 0\}, \{8, 4, 1\}, \{8, 4, 0\}, \{8, 2, 0\}, \{8, 1, 0\}, \{7, 3, 0\}, \{6, 3, 0\}, \{6, 2, 0\}, \{5, 2, 0\}, \{5, 1, 0\}, \{4, 1, 0\}\}$
- $\mathcal{V}_3^{0,0} = \{\{10, 7, 3, 0\}, \{10, 6, 3, 0\}, \{10, 6, 2, 0\}, \{9, 6, 3, 0\}, \{9, 6, 2, 0\}, \{9, 5, 2, 0\}, \{9, 5, 1, 0\}, \{8, 5, 2, 0\}, \{8, 5, 1, 0\}, \{8, 4, 1, 0\}\}$

Finalmente, podemos ver los diagramas de barras y persistencia asociados al complejo simplicial filtrado representado en la Figura 3.6 en la Figura 3.7.

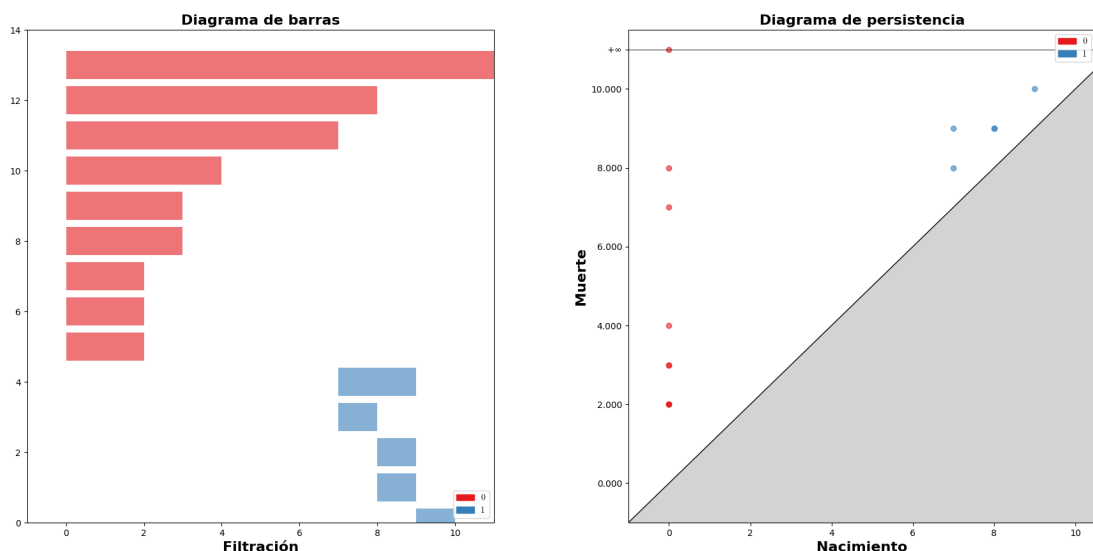


Figura 3.7: Diagramas de barras y persistencia asociados a la red 3.3.

Nótese que en esta interpretación, a diferencia de la anterior, tenemos agujeros p -dimensionales ocultos, como el que aparece en la Figura 3.6 en la filtración 9.

Observamos que, en este caso, los diagramas obtenidos son muy diferentes a los de la sección anterior. Si bien obtenemos las mismas clases de equivalencia de la homología de grado 0 (componentes conexas), vemos que hemos obtenido un resultado muy distinto para las clases de equivalencia de la homología de grado 1 (agujeros 2-dimensionales). En este caso hemos obtenido 5 clases de equivalencia mientras que en el caso anterior sólo teníamos 1. El hecho de que tengamos las mismas clases de equivalencia de la homología de grado 0 se debe a que, como ya hemos comentado, el cálculo de las aristas en esta interpretación se realiza siguiendo el razonamiento global, lo que resulta un tanto incoherente con el cálculo del resto de los p -símplices.

Cabe reseñar que este ejemplo reproduce fielmente el propuesto por el autor en [19], aunque habiendo solucionado alguna errata presente en los dibujos del autor. ◀

Del ejemplo anterior el autor extrae unas conclusiones muy importantes. Por una parte, observa que si las neuronas de entrada se conectan directamente a las de salida, el conocimiento de la red será «pobre» ya que será equivalente a la detección de patrones. Por otra parte, el incremento del número de Betti β_1 indica que la red determina la neurona de llegada por combinación de las neuronas de salida. De este modo, supone que el aumento de β_1 refleja la complejidad del conocimiento adquirido por la red. Por lo tanto, mediante la homología persistente es posible medir la complejidad del conocimiento adquirido por la red.

Este supuesto puede tener especial sentido en el caso de las redes neuronales convolucionales (CNN). Aunque una descripción detallada de las mismas queda fuera del alcance del presente trabajo, en estas redes la información se presenta de manera jerarquizada, es decir, las neuronas de las capas iniciales detectan patrones simples como líneas horizontales y verticales, y a medida que se avanza en las capas, estos patrones se combinan para representar objetos más complejos, por ejemplo una rueda de un coche. Estas redes son muy utilizadas en el ámbito de la visión por computador.

Habiendo presentado detalladamente el ejemplo que propone el autor, vamos a describir todos los algoritmos y programas que hemos empleado para el cálculo automático de los complejos simpliciales asociados, los números de Betti y los diagramas correspondientes.

Para el desarrollo de los algoritmos y programas se ha empleado el lenguaje *Python* ([18]) y el programa *Neovim* como entorno de desarrollo. Hemos elegido Python como lenguaje de programación ya que es un lenguaje de alto nivel (lo que lo hace bastante sencillo) y el código producido es de fácil lectura. Este lenguaje es de código abierto y es muy empleado en el campo de la inteligencia artificial. No obstante, este lenguaje tiene el inconveniente de ser poco eficiente ya que se trata de un lenguaje interpretado. Para solucionarlo, hemos empleado librerías que ofrecen una interfaz en Python pero que están programadas en lenguajes más eficientes como C++. Estas librerías son *GUDHI* ([17]) para el cálculo de los diagramas, y *Dionysus* ([14]) para el cálculo de los números de Betti.

Nota. Todo el código desarrollado para este trabajo se encuentra disponible en: <https://github.com/joros244/TFGMath2022.git>

En cuanto a los algoritmos empleados para el cálculo de los complejos simpliciales, nos hemos basado en los algoritmos propuestos en [16] y los hemos adaptado a nuestras necesidades particulares. Veamos tales algoritmos en detalle

En primer lugar, se define un objeto grafo dirigido acíclico (GDA). Este objeto representa un grafo dirigido acíclico mediante una lista de adyacencia. Tras esto, se define un objeto grafo dirigido acíclico con pesos (GDAP). Este objeto consta de un objeto grafo dirigido acíclico y de una matriz de pesos asociada, esta matriz es la original dada por la red neuronal y define completamente las aristas del objeto grafo asociado. La definición del objeto la podemos ver en el BC 3.1.

Ahora debemos distinguir entre los algoritmos empleados para el cálculo local y global. Empezaremos con éste último.

```

#Un GDAP es un GDA con una matriz asociada. La matriz determina las
#aristas y los vértices del GDA asociado.

def __init__(self, M):
    self.G = GDA.Grafo_dirigido_aciclico(len(M))
    self.M = M
    for i in range(0, len(self.M)):
        for j in range(i):
            if( self.M[i][j] > 0 ):
                self.G.crea_arista(i, j)

```

BC 3.1: Definición del objeto GDAP.

Para el cálculo global hemos empleado la función que podemos ver en el BC 3.2.

Aquí la función «filtración» es común a ambas interpretaciones y su funcionamiento es el siguiente: recorre la matriz asociada al grafo, si la entrada a_{ij} supera el umbral t pasa a la siguiente, en caso contrario la sustituye por 0; tras esto devuelve el GDAP asociado a la nueva matriz. La función «clausura transitiva» calcula la clausura transitiva de un GDA y la devuelve. La función «busca pesos», común a ambas interpretaciones, es especialmente importante ya que hace una búsqueda en profundidad sobre la matriz para calcular el peso apropiado. Podemos ver su definición en el BC 3.3.

Notemos que este algoritmo es el clave dentro de toda esta implementación, y además, es recursivo, lo que lo hace bastante ineficiente. Esta observación nos lleva a cuestionar la aplicabilidad de estos algoritmos a casos reales en los que el número de neuronas de la red es muy grande. No obstante, para el presente trabajo nos ha resultado suficiente con esta implementación.

Finalizamos la implementación de la interpretación global con una última función que nos devuelve la lista de los símlices del complejo simplicial. Podemos ver esta función en el BC 3.4.

Veamos ahora la implementación de los algoritmos asociados a la interpretación local. Como ya hemos descrito los algoritmos comunes a ambas interpretaciones, en este caso sólo son necesarias dos funciones. La primera de estas funciones calcula los símlices maximales y su definición la podemos ver en el BC 3.5. La definición de la segunda de estas funciones, responsable del cálculo de la lista de los símlices, puede verse en el BC 3.6.

Como ya hemos comentado, los algoritmos aquí expuestos son bastante ineficientes al ser recursivos. Aunque hemos tratado de optimizarlos en la medida de lo posible, no hemos conseguido una mejora significativa en el tiempo de ejecución. Aunque para los ejemplos del presente trabajo nos ha bastado con esta implementación, se queda como trabajo futuro la optimización de la presente implementación.

```

#Devuelve una lista de los símlices maximales de cada vértice si al
#menos uno de ellos pasa el filtro t. Recorre la lista de vértices
#adyacentes a uno dado y en caso de que no haya adyacencia directa,
#calcula los caminos indirectos y los añade. Si la adyacencia directa
#pasa el filtro, la añade.
def __calcula_simplices_global(self,t)->List[List[int]]:
    simp=[]
    #Necesario para recorrer los adyacentes. OPTIMIZACIÓN
    P=self.filtracion(t)
    T=P.clausura_transitiva()
    for i in range(len(self.M)):
        for j in T.G.adj[i]:
            if(self.M[i][j]==0):
                p=[]
                self.busca_pesos(i,j,p)
                if(len(p)>0):
                    m=max([v[0] for v in p])
                    if(m>=t):
                        l = [k[1] for k in p]
                        simp.extend(l)
                        simp.append([i,j])
            elif (self.M[i][j]>=t):
                simp.append([i,j])
    return simp

```

BC 3.2: Función para el cálculo de símlices maximales (global).

```

#Almacena en p una lista de tuplas donde el primer argumento es el
#posible peso entre origen(o) y destino(d); y el segundo argumento es
#el camino que produce dicho peso.
def busca_pesos(self, o, d, p=[], q=1, c=[]):
    #BÚSQUEDA EN PROFUNDIDAD
    if(o==d):
        c.append(o)
        p.append((q.__round__(4),c))
        q=1
        c=[]
    else:
        for j in range(d,o):
            if(self.M[o][j]!=0):
                #NOTA: Podríamos optimizarlo para que si q<t pase
                #al siguiente vértice. SÓLO VÁLIDO PARA V.LOCAL
                self.busca_pesos(j,d,p,q*self.M[o][j],c+[o])

```

BC 3.3: Función de búsqueda de pesos.

```

#Devuelve un lista con el complejo simplicial asociado al GDAP.
#V.GLOBAL
def get_simplex_global(self, t : float)->List[List[int]]:
    simplices=self.__calcula_simplices_global(t)

    #En este caso hay que añadir los vértices
    simplices.extend([[i] for i in range(len(self.M))])

    #Ahora añadimos los subsimplices de los calculados
    for s in simplices:
        for L in range(3,len(s)+1):
            for sub in it.combinations(s,L):
                if(list(sub) not in simplices):
                    simplices.append(list(sub))

    #Limpiamos aquellos simplices con aristas irreales
    tr=[s for s in simplices if len(s)>2]
    for e in tr:
        if(any([list(sub) not in simplices for sub in
                it.combinations(e,len(e)-1)])):
            simplices.remove(e)

    return simplices

```

BC 3.4: Función para el cálculo de todos los simplices (global).

```

#Devuelve una lista de los simplices maximales de cada vértice que
#pasan el filtro t. Lo que hace es recorrer la lista de vértices
#adyacentes y en caso de que no haya adyacencia directa, calcula los
#caminos indirectos.
def __calcula_simplices(self,t)->List[List[int]]:
    simp=[]
    #Necesario para recorrer los adyacentes. OPTIMIZACIÓN
    P=self.filtracion(t)
    T=P.clausura_transitiva()
    for i in range(len(self.M)):
        for j in T.G.adj[i]:
            if(self.M[i][j]==0):
                p=[]
                self.busca_pesos(i,j,p)
                if(len(p)>0):
                    l = [k[1] for k in p if k[0]>=t]
                    simp.extend(l)
            elif (self.M[i][j]>=t):
                simp.append([i,j])
    return simp

```

BC 3.5: Función para el cálculo de simplices maximales (local).


```

#Devuelve un lista con el complejo simplicial asociado al GDAP.
#V.LOCAL
def get_simplex_local(self, t : float)->List[List[int]]:
    simplices=self.__calcula_simplices(t)

    simplices.extend([[i] for i in range(len(self.M))])

    #Hemos calculado simplices "maximales", nos faltan sus
    #subsimplices
    for s in simplices:
        for L in range(2,len(s)+1):
            for sub in it.combinations(s,L):
                if(list(sub) not in simplices):
                    simplices.append(list(sub))
    return simplices

```

BC 3.6: Función para el cálculo de todos los simplices (local).

3.4 Ejemplos

En las secciones anteriores hemos reproducido fielmente los resultados que presenta el autor en [19]. En esta sección vamos a proponer una serie de ejemplos para observar más detalladamente el funcionamiento de la homología persistente en casos concretos.

Estos ejemplos van a estar centrados en comprobar cómo responde la homología persistente ante un cambio en la distribución de los pesos de la red. Es decir, vamos a ver un ejemplo con una distribución homogénea de los pesos, un ejemplo con una distribución extrema, y un ejemplo con una distribución intermedia. El objetivo de estos ejemplos es desarrollar una mejor intuición en el uso de la herramienta que hemos presentado en este trabajo, así como validar las hipótesis que se presentan en [19]. Adicionalmente, para un mayor interés de estos ejemplos, vamos a incluir los cálculos para las dos interpretaciones que hemos visto en este capítulo.

Para que la comparativa entre las distribuciones tenga sentido vamos a considerar fija la arquitectura de la red en los tres ejemplos. La arquitectura escogida puede verse en la Figura 3.8.

Aunque una arquitectura de mayor tamaño pudiera resultar más enriquecedora en cuanto a las conclusiones obtenidas de su estudio, motivados por la preocupación por la eficiencia de los algoritmos presentados en la sección anterior, hemos decidido estudiar esta arquitectura más reducida. No obstante, esta decisión no es demasiado restrictiva ya que los resultados que obtengamos podrán extrapolarse a redes más grandes. Para una mayor legibilidad de los ejemplos hemos decidido omitir la lista de los simplices de cada filtración que incluíamos en ejemplos anteriores. Para acceder a las listas de los simplices véase <https://github.com/joros244/TFGMATH2022.git>.

Vamos a comenzar con el ejemplo que presenta una distribución homogénea de los pesos.

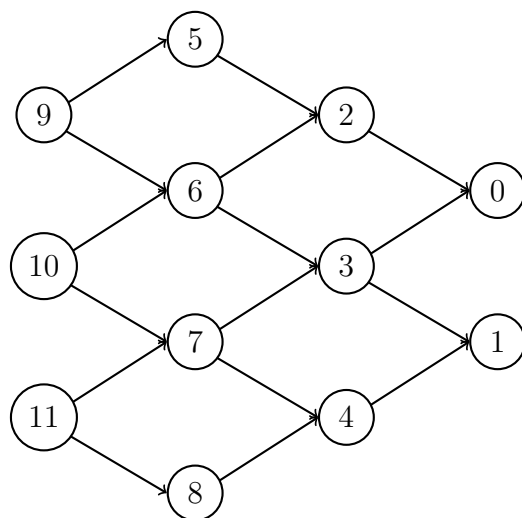


Figura 3.8: Representación de una red neuronal de 12 neuronas y 2 capas.

Ejemplo 3.4.1. Distribución homogénea. En un primer lugar, supondremos que la red estudiada posee una distribución ciertamente homogénea de los pesos. Esta distribución da lugar la representación de la red, con las importancias entre neuronas ya calculadas, que puede verse en la Figura 3.9.

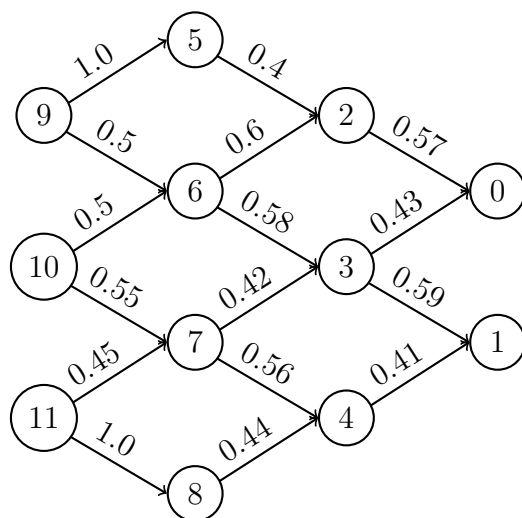


Figura 3.9: Representación de la red 3.8 con una distribución homogénea de las importancias entre neuronas.

Procedemos ahora con el cálculo del complejo simplicial asociado. Al igual que en los ejemplos que hemos presentado anteriormente, vamos a ilustrar unos cuantos pasos en la filtración con los correspondientes números de Betti asociados. Esta ilustración puede verse en la Figura 3.10 y en la Figura 3.12. También añadimos los diagramas de barras y de persistencia que pueden verse en la Figura 3.11 y en la Figura 3.13.

Si observamos detenidamente los pasos en la filtración del complejo simplicial en la

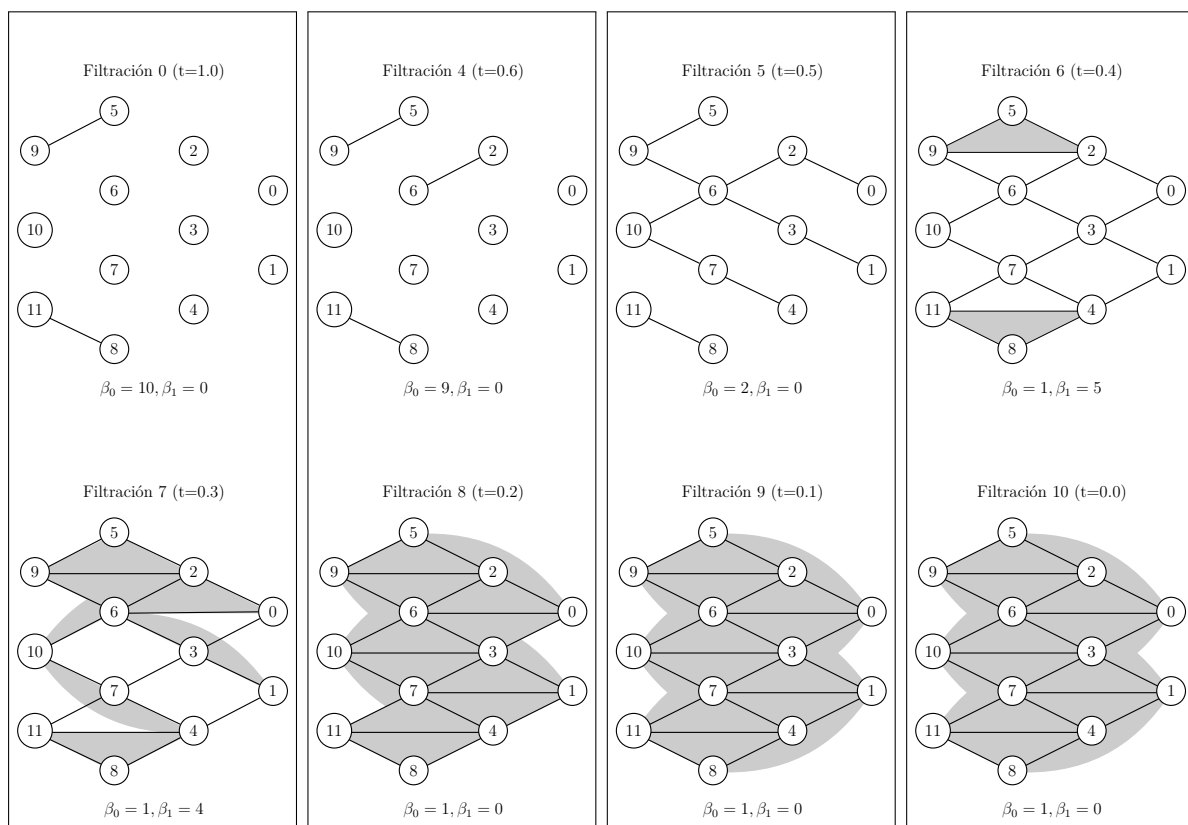


Figura 3.10: Pasos en la filtración del complejo simplicial asociado a 3.9 y números de Betti correspondientes (interpretación local).

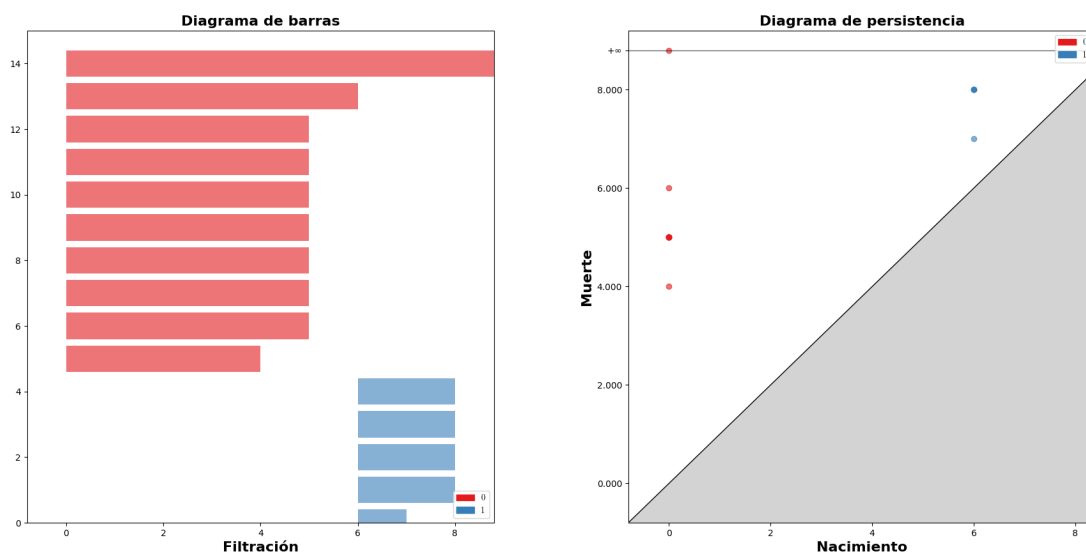


Figura 3.11: Diagrama de barras (izquierda) y diagrama de persistencia (derecha) asociados a la filtración ilustrada en 3.10 (interpretación local).

Figura 3.10 y en la Figura 3.12, apreciamos que la diferencia entre ambas interpretaciones se produce en las filtraciones 6 y 7, donde los números de Betti β_1 valen 4 en la interpretación local, y 2 en la interpretación global. También podemos apreciar esta distinción en los diagramas de barras correspondientes que podemos encontrar en la Figura 3.11 y la Figura 3.13. Como ya habíamos notado anteriormente, esta disparidad nos muestra que la información recogida por la interpretación global es escasa, lo que nos induce a pensar que esta interpretación, a pesar de su solidez teórica, proporciona menos información. No obstante, antes de tomar cualquier decisión en este aspecto, debemos ver su desempeño ante los cambios en las distribuciones de los siguientes ejemplos. ◀

Veamos ahora el ejemplo correspondiente a una distribución intermedia de los pesos de la red.

Ejemplo 3.4.2. Distribución intermedia. Supondremos que la red estudiada posee una distribución heterogénea de los pesos. Esta distribución, aunque heterogénea, no presentará desigualdades muy significativas entre los pesos de la red. Esta distribución dará lugar a la representación de la red, con las importancias entre neuronas ya calculadas, que puede verse en la Figura 3.14.

Procedemos ahora con el cálculo del complejo simplicial asociado. Como ya hemos mencionado anteriormente, vamos a ilustrar unos cuantos pasos en la filtración con los correspondientes números de Betti asociados. Esta ilustración puede verse en la Figura 3.15 y en la Figura 3.17. También añadimos los diagramas de barras y de persistencia asociados que pueden verse en la Figura 3.16 y en la Figura 3.18.

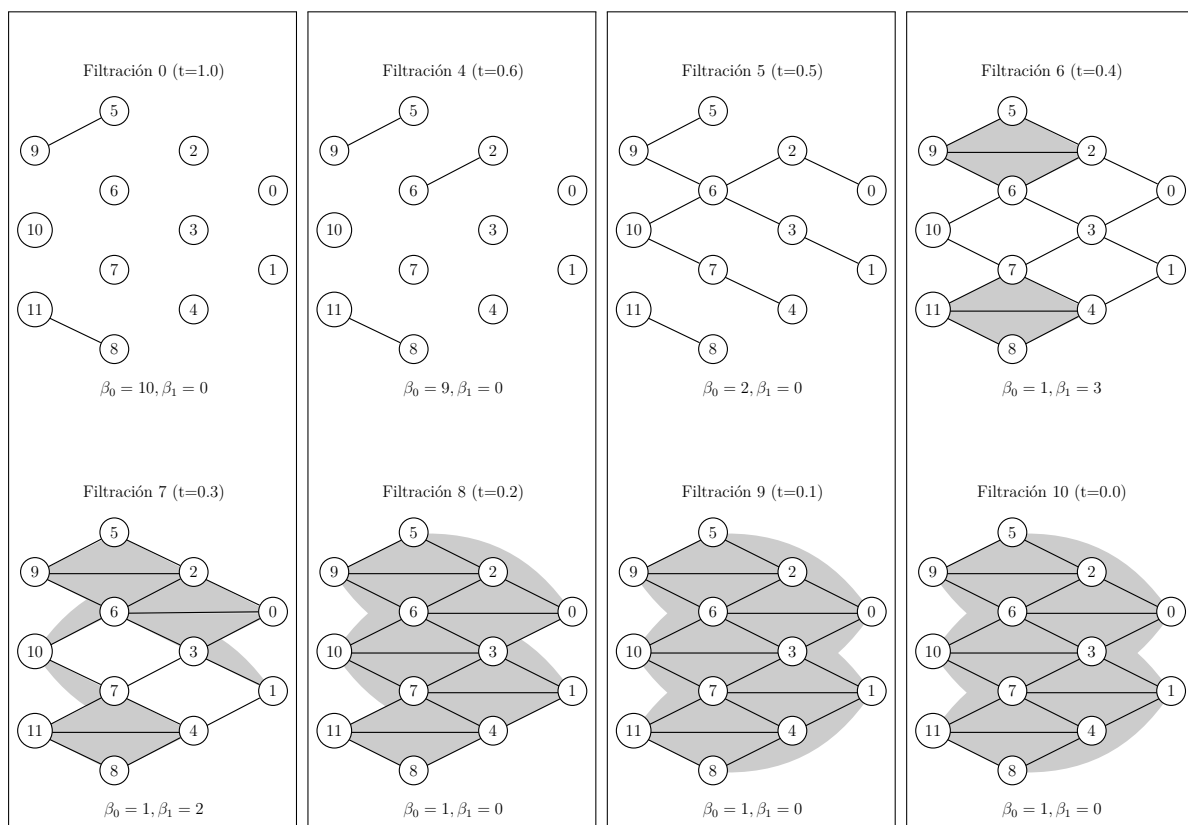


Figura 3.12: Pasos en la filtración del complejo simplicial asociado a 3.9 y números de Betti correspondientes (interpretación global).

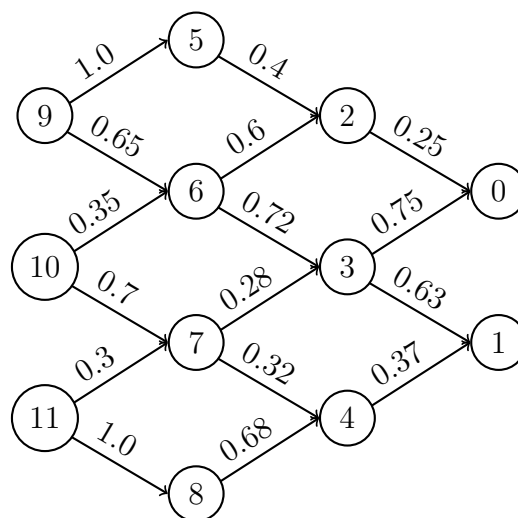
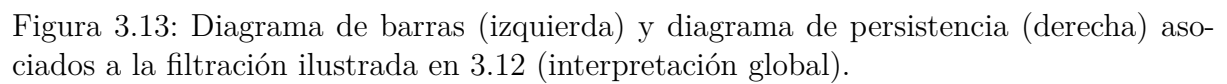


Figura 3.14: Representación de la red 3.8 con una distribución intermedia de las importancias entre neuronas.

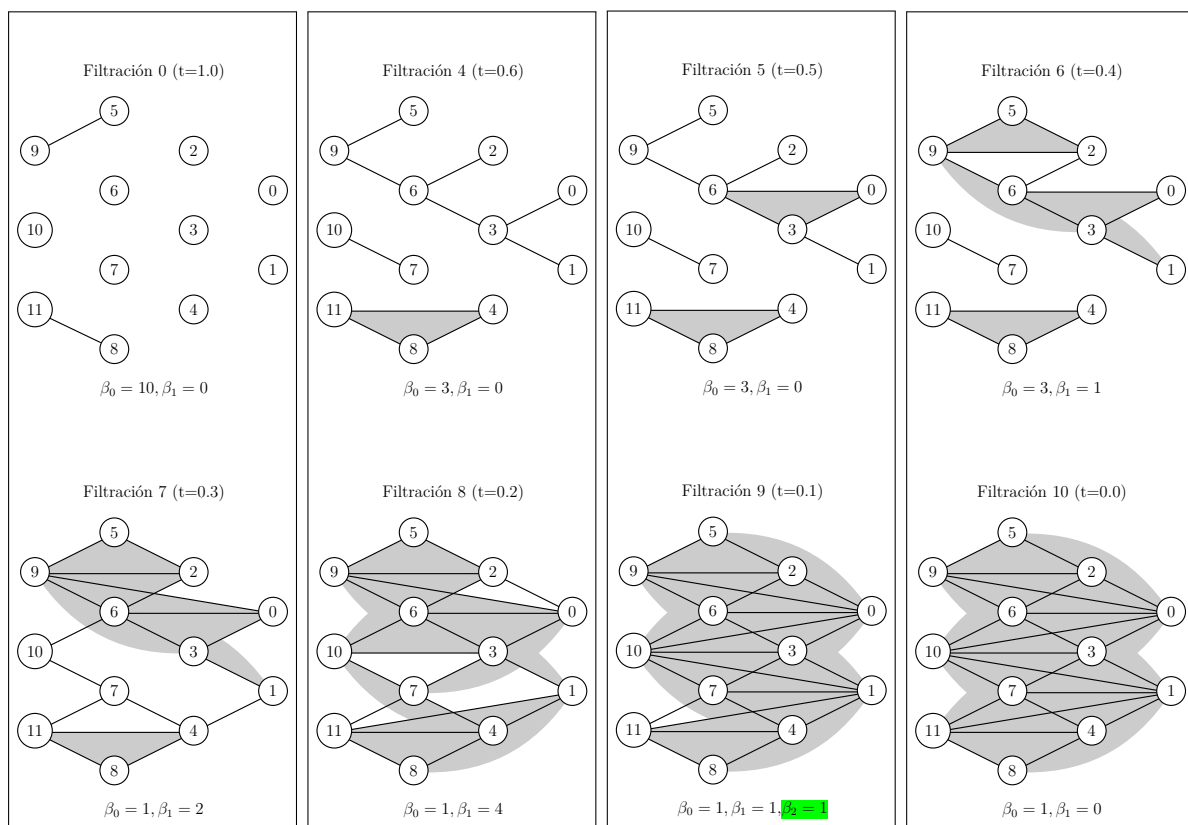


Figura 3.15: Pasos en la filtración del complejo simplicial asociado a 3.14 y números de Betti correspondientes (interpretación local).

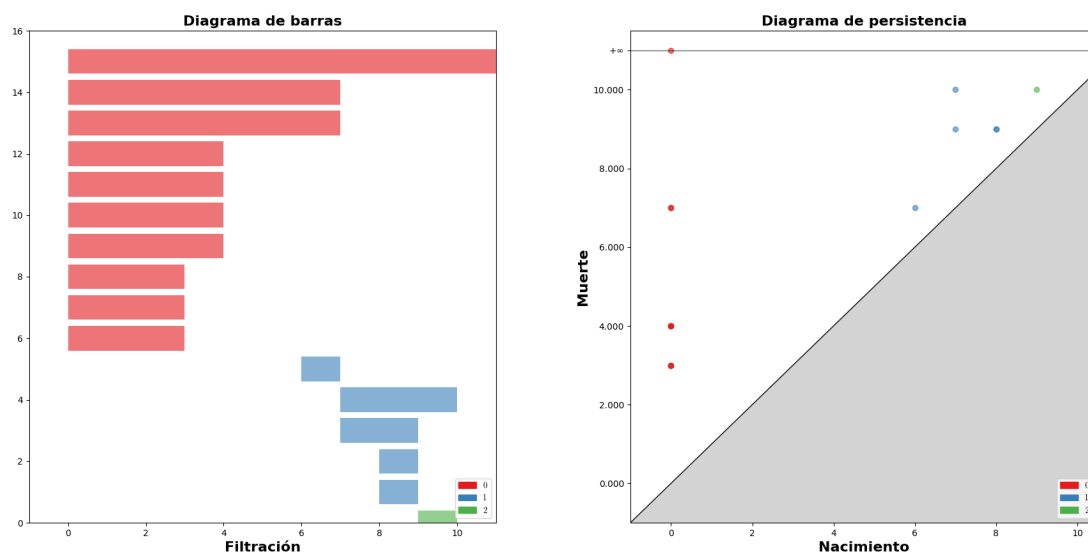


Figura 3.16: Diagrama de barras (izquierda) y diagrama de persistencia (derecha) asociados a la filtración ilustrada en 3.15 (interpretación local).

Si observamos detenidamente los pasos en la filtración del complejo simplicial en la Figura 3.15 y en la Figura 3.17, apreciamos una clara diferencia entre ambas interpretaciones que se produce en la filtración 6 y se mantiene hasta la última filtración. Esta disparidad es especialmente significativa en la filtración 9: mientras que en la interpretación local tenemos que $\beta_2 = 1$, en la interpretación global tenemos que $\beta_2 = 0$. Es decir, con la interpretación local estamos capturando un vacío de dimensión 2 que se nos escapa con la interpretación global. Este caso no nos había aparecido anteriormente y merece la pena detenerse momentáneamente a hacer una reflexión sobre ello.

Conceptualmente, $\beta_2 = 1$ se ha producido porque existe un agujero entre 3-símplices, es decir, existe un agujero entre tetraedros en la filtración 9 de la la Figura 3.15. De manera intuitiva, esto nos dice que la red ha necesitado, equitativamente, la salida de 3 neuronas para determinar la neurona de llegada. Siguiendo el razonamiento hecho en [19], esto mostraría un aumento en el grado de complejidad del conocimiento adquirido por la red.

No obstante, en este ejemplo en concreto, dicha clase de homología nace y muere en un breve lapso de filtraciones, lo que nos induce a pensar que simplemente se trata de ruido. Esta consideración es fácilmente observable en el diagrama de persistencia de la Figura 3.16, en el que las clases próximas a la diagonal las podemos considerar como ruido.

Adicionalmente, podemos observar que la interpretación local nos está permitiendo visualizar un número mayor de agujeros de dimensión 1. Esto puede verse en la filtración 8 de la Figura 3.15 y la Figura 3.17.

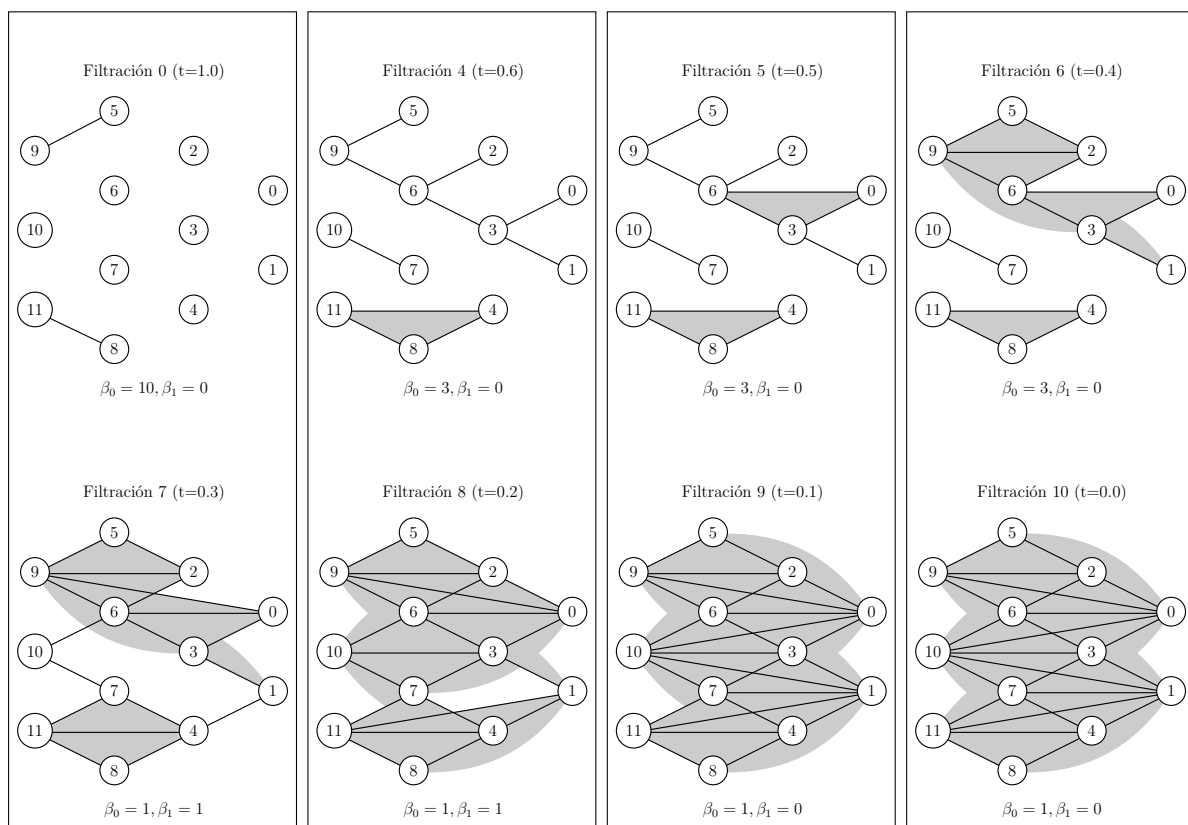


Figura 3.17: Pasos en la filtración del complejo simplicial asociado a 3.14 y números de Betti correspondientes (interpretación global).

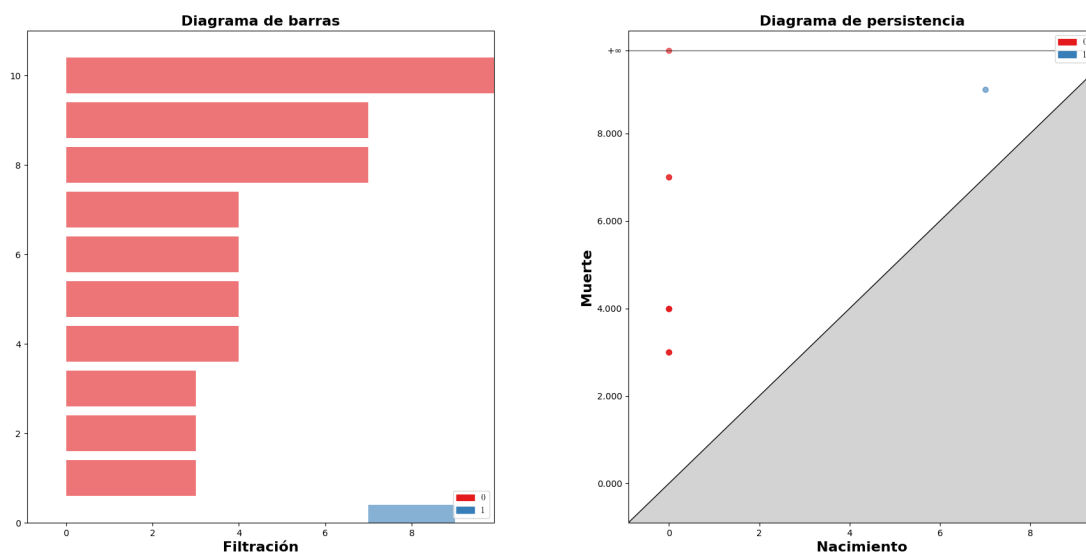


Figura 3.18: Diagrama de barras (izquierda) y diagrama de persistencia (derecha) asociados a la filtración ilustrada en 3.17 (interpretación global).

Todas estas disparidades quedan todavía mejor reflejadas en la comparativa de la Figura 3.11 con la Figura 3.18. ◀

Por último, veamos el ejemplo correspondiente a una distribución extrema de los pesos de la red.

Ejemplo 3.4.3. Distribución extrema. Supondremos que la red estudiada posee una distribución heterogénea de los pesos. Esta distribución, presentará desigualdades muy significativas entre los pesos de la red. Esta distribución dará lugar a la representación de la red, con las importancias entre neuronas ya calculadas, que puede verse en la Figura 3.19.

Procedemos ahora con el cálculo del complejo simplicial asociado. Como ya hemos mencionado anteriormente, vamos a ilustrar unos cuantos pasos en la filtración con los correspondientes números de Betti asociados. Esta ilustración puede verse en la Figura 3.20 y en la Figura 3.22. También añadimos los diagramas de barras y de persistencia asociados que pueden verse en la Figura 3.21 y en la Figura 3.23.

En este último caso, si observamos detenidamente los pasos en la filtración del complejo simplicial en la Figura 3.15 y en la Figura 3.17, podemos ver que no hay diferencia alguna entre ambas interpretaciones. Tampoco podemos apreciar ninguna diferencia entre los diagramas que mostramos en la Figura 3.21 y en la Figura 3.23.

Esta concordancia, que inicialmente puede parecer sorprendente, queda explicada por la disparidad en la distribución de los pesos, y por tanto, de las importancias de la red.

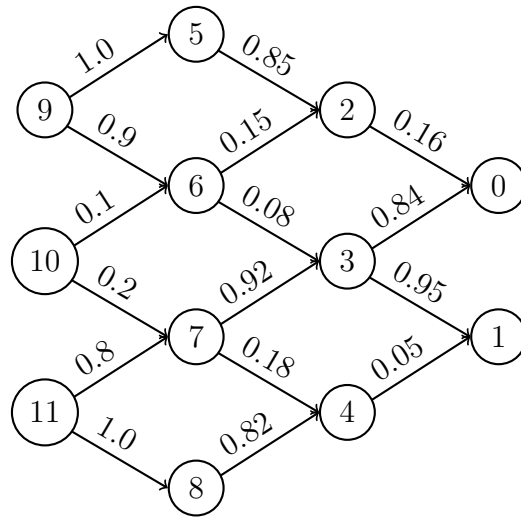


Figura 3.19: Representación de la red 3.8 con una distribución extrema de las importancias entre neuronas.

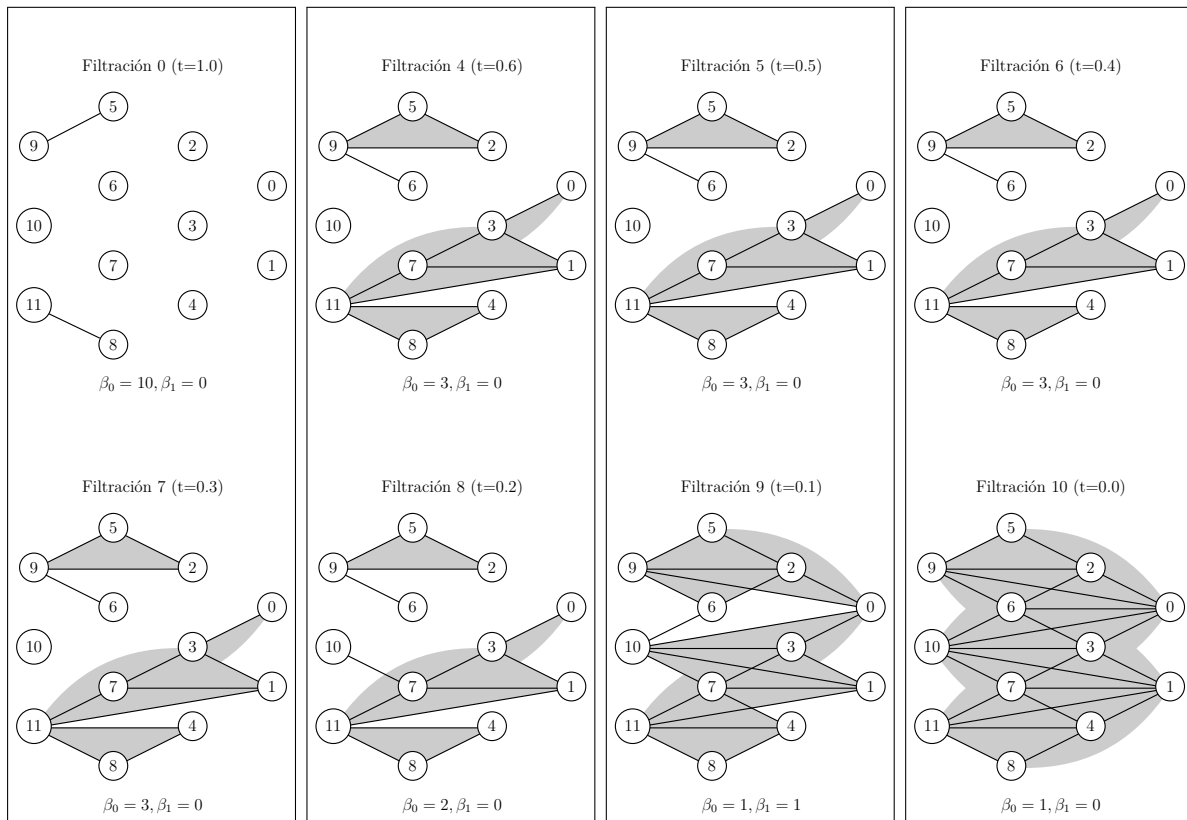


Figura 3.20: Pasos en la filtración del complejo simplicial asociado a 3.19 y números de Betti correspondientes (interpretación local).

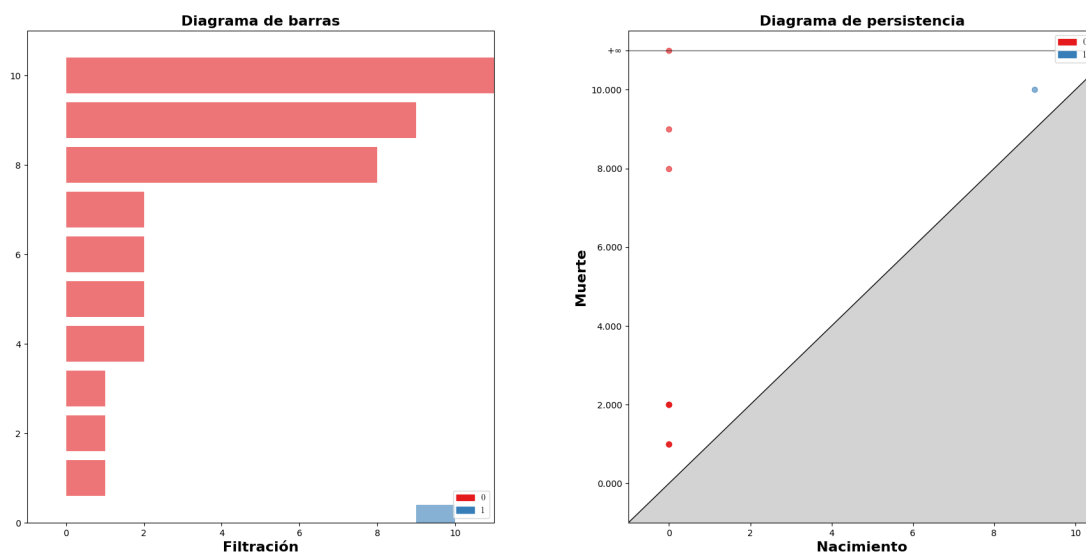


Figura 3.21: Diagrama de barras (izquierda) y diagrama de persistencia (derecha) asociados a la filtración ilustrada en 3.20 (interpretación local).

Esta gran desviación provoca que los complejos simpliciales asociados a cada filtración se mantengan constantes a lo largo del proceso de filtración, y que al final de este proceso cambien rápidamente.

No obstante, si quisiéramos profundizar en las diferencias entre ambas interpretaciones aplicadas a este ejemplo, únicamente tendríamos que cambiar la granularidad de las filtraciones, esto es, modificar el valor del parámetro que regula la construcción del complejo simplicial (véase la definición 3.1.3). Como nuestro objetivo es de comparar el funcionamiento de la homología persistente en función de la distribución de los pesos de la red, mantendremos la granularidad que hemos empleado en los ejemplos anteriores. ◀

A la vista de los anteriores ejemplos, discutimos a continuación el funcionamiento de la homología persistente en función de la distribución de los pesos de la red, así como del desempeño de ambas interpretaciones a lo largo de dichos ejemplos.

En lo referente a la distribución de los pesos, observamos que la homología persistente nos aporta información de mayor calidad en distribuciones homogéneas. Con esto nos referimos a que, en aquellas distribuciones más equitativas, la homología persistente refleja un mayor número de combinaciones entre neuronas. Además, estas combinaciones persisten a lo largo de un mayor número de filtraciones que en otras distribuciones menos equitativas. Este resultado, aunque esperable, nos sugiere que la homología persistente puede constituir una sólida herramienta para el análisis de redes neuronales. Ahora bien, para argumentar que la homología persistente realmente refleja el grado de complejidad adquirido por las redes (tal y como teoriza el autor en [19]) son necesarios una serie

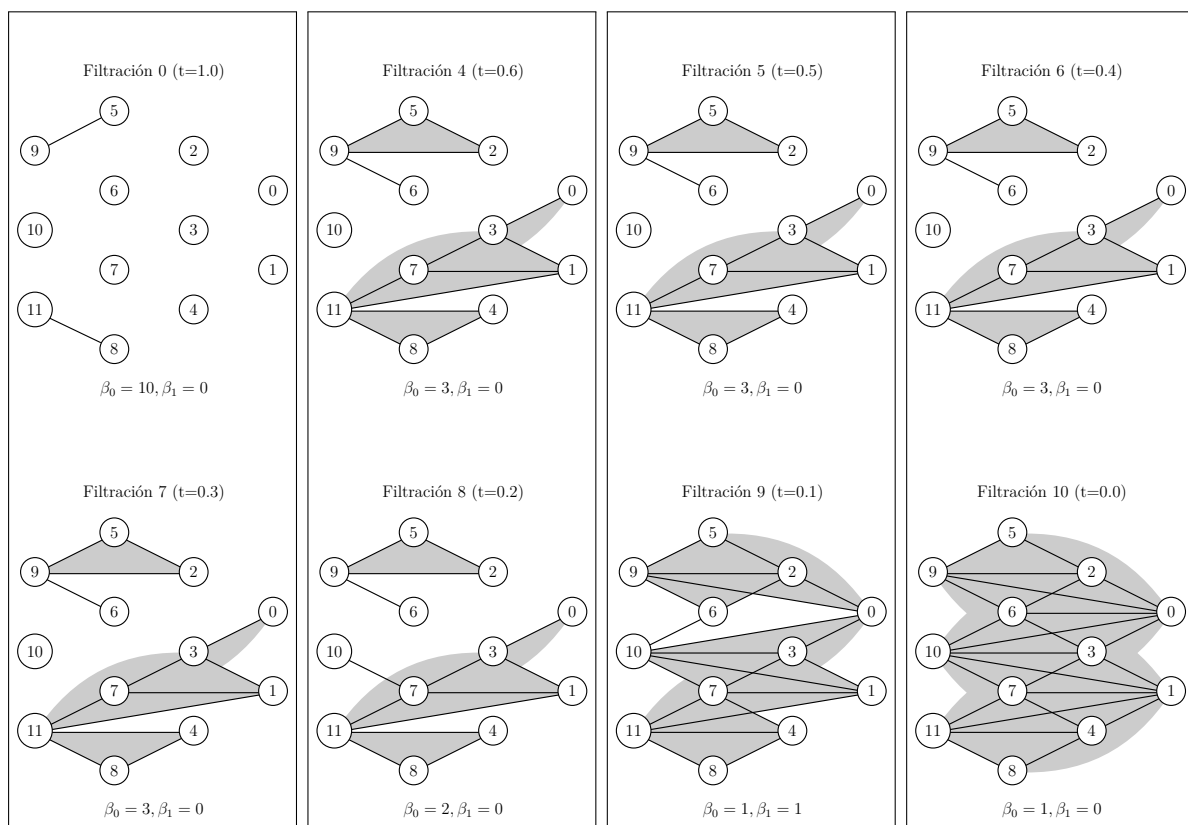


Figura 3.22: Pasos en la filtración del complejo simplicial asociado a 3.19 y números de Betti correspondientes (interpretación global).

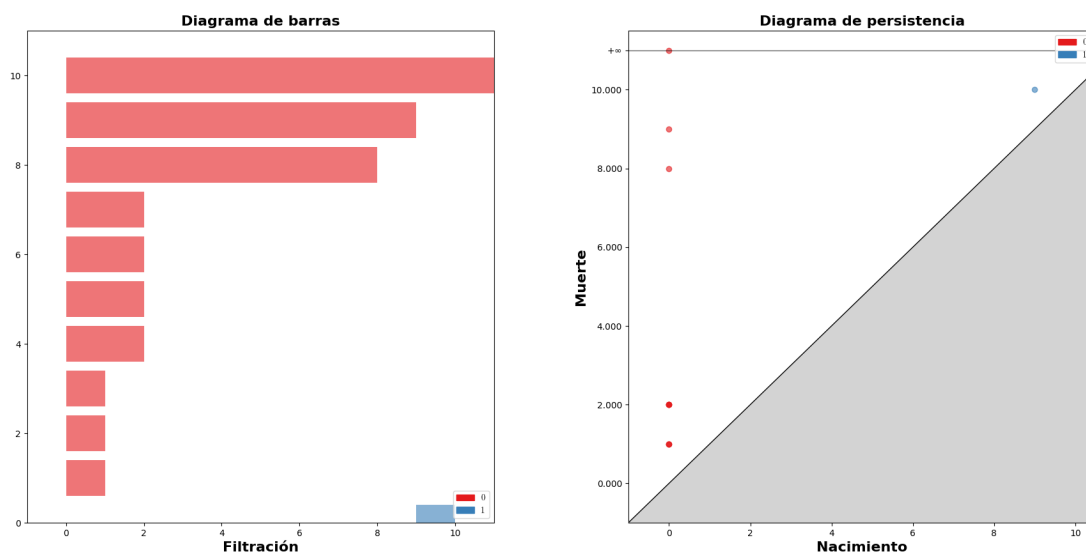


Figura 3.23: Diagrama de barras (izquierda) y diagrama de persistencia (derecha) asociados a la filtración ilustrada en 3.22 (interpretación global).

de experimentos con redes de un tamaño mayor a las expuestas en el presente trabajo. Desafortunadamente, la realización de estos experimentos queda fuera del alcance de este trabajo.

En cuanto al desempeño de ambas interpretaciones, podemos observar que, en líneas generales, la interpretación local (que se corresponde con la original del autor) captura un número mayor de clases de homología, lo que le permite aportar una mayor información sobre las redes estudiadas. Por otra parte, como ya hemos comentado anteriormente, esta ventaja puede ser a la vez un inconveniente. Esto se debe a que, al tener un mayor grado de sensibilidad, la interpretación local captura un número mayor de clases de homología que representan ruido, ensuciando así los diagramas obtenidos. Al igual que con la distribución de los pesos, son necesarios más experimentos para profundizar en las diferencias entre ambas interpretaciones antes de descartar cualquiera de ellas.

Conclusiones

A lo largo de este trabajo fin de grado he expuesto las dos interpretaciones del artículo [19] que, con la inestimable ayuda de mis tutores, he podido analizar en profundidad. Por desgracia, finalizo esta memoria sin decantarme por ninguna de las dos, pues ambas tienen sus ventajas e inconvenientes, y la realización de los experimentos necesarios hubiera alargado excesivamente esta memoria.

La inteligencia artificial es un tema que siempre me ha parecido fascinante y este trabajo me ha permitido explorar una dimensión del tema que me resultaba completamente desconocida: las matemáticas subyacentes. Con estas matemáticas he tenido la ocasión de descubrir la topología algebraica. Desconocía totalmente esta disciplina y la he encontrado de una belleza sublime, pues combina dos de las ramas que me parecen más interesantes del grado: la topología y el álgebra. Además, la realización de este trabajo ha requerido la revisión de multitud de conceptos de estas materias, así como de Teoría de Grafos y Análisis Real; y ha sido en esta revisión y en la investigación posterior donde he podido apreciar la importancia de los conceptos que he aprendido en el transcurso de esta titulación.

A título personal, creo que como estudiante de la doble titulación el tema de este trabajo se ha ajustado perfectamente a mi formación, y me gustaría expresar mi más sincera gratitud con el departamento de Matemáticas y Computación, y con mis tutores José Luis y Julio en particular, por brindarme la oportunidad de trabajar en un tema que me ha resultado apasionante de principio a fin, ya que combina a la perfección matemáticas con inteligencia artificial. He disfrutado verdaderamente con la labor de investigación realizada en este trabajo y espero tener la oportunidad de repetir esta experiencia en el futuro.

Bibliografía

- [1] Bello Hernández, Manuel: *Apuntes de Cálculo diferencial en varias variables*, 2019.
- [2] Cavanna, Nicholas J. and Donald R. Sheehy: *The generalized persistent nerve theorem*, 2018. URL <https://arxiv.org/abs/1807.07920>.
- [3] Edelsbrunner, Herbert and John Harer: *Computational Topology - an Introduction*. American Mathematical Society, 2010, ISBN 978-0-8218-4925-5.
- [4] Elduque Palomo, Alberto Carlos: *Introduction to algebra*, 2017.
- [5] Ghrist, Robert: *Elementary Applied Topology*. CreateSpace Independent Publishing Platform, 2014, ISBN 978-1502880857.
- [6] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville: *Deep Learning*. MIT Press, 2016. URL <http://www.deeplearningbook.org>.
- [7] Gutiérrez Jiménez, José Manuel y Víctor Lanchares Barrasa: *Elementos de matemática discreta*. Universidad de La Rioja. Servicio de Publicaciones, 2010, ISBN 978-84-693-6451-2.
- [8] Heras Vicente, Jónathan: *Notas del curso de inteligencia artificial*, 2022.
- [9] Hornik, Kurt, Maxwell Stinchcombe, and Halbert White: *Multilayer feedforward networks are universal approximators*. Neural Networks, 2(5):359–366, 1989, ISSN 0893-6080. URL <https://www.sciencedirect.com/science/article/pii/0893608089900208>.
- [10] Jonsson, Jakob: *Simplicial Complexes of Graphs*. Springer Berlin, Heidelberg, January 2008, ISBN 978-3-540-75858-7.
- [11] Kalyanakrishnan, Shivaram: *The perceptron learning algorithm and its convergence*, 2017. URL <https://www.cse.iitb.ac.in/~shivaram/teaching/old/cs344+386-s2017/resources/classnote-1.pdf>.
- [12] Karp, Richard M.: *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972, ISBN 978-1-4684-2001-2. https://doi.org/10.1007/978-1-4684-2001-2_9.
- [13] Macho Stadler, Marta: *De la homología a la cohomología: teoremas de dualidad*, 2006.
- [14] Morozov, Dmitriy, May 2021. URL <https://mrzv.org/software/dionysus2/>.
- [15] Otter, Nina, Mason A Porter, Ulrike Tillmann, Peter Grindrod, and Heather A Harrington: *A roadmap for the computation of persistent homology*. EPJ Data Science, 6(1), aug 2017. URL <https://doi.org/10.1140/2Fepjds%2Fs13688-017-0109-5>.

- [16] Sedgewick, Robert and Kevin Wayne: *Algorithms, 4th Edition*. Addison-Wesley, 2011, ISBN 978-0-321-57351-3.
- [17] The GUDHI Project: *GUDHI User and Reference Manual*. GUDHI Editorial Board, 2015. URL <http://gudhi.gforge.inria.fr/doc/latest/>.
- [18] Van Rossum, Guido and Fred L. Drake: *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009, ISBN 1441412697.
- [19] Watanabe, Satoru and Hayato Yamana: *Topological measurement of deep neural networks using persistent homology*, Jan 2022. ISSN 1573-7470. URL <https://doi.org/10.1007/s10472-021-09761-3>.
- [20] Zomorodian, Afra and Gunnar Carlsson: *Computing persistent homology*. Discrete & Computational Geometry, 33(2):249–274, Feb 2005, ISSN 1432-0444. URL <https://doi.org/10.1007/s00454-004-1146-y>.