Fundamentos de las Redes Neuronales Profundas

Fundamentos Matemáticos Básicos

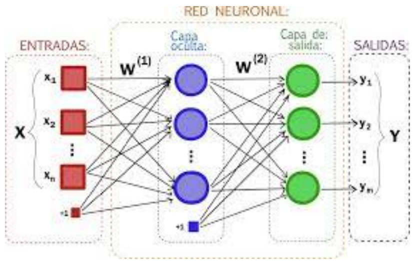
Clase 2

Dra. Wendy Aguilar

Modelos Generativos Profundos

UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

¿Cuáles son los fundamentos matemáticos básicos que necesito saber para crear modelos generativos profundos?



1. Vectores, Matrices y Tensores

numéricos

Son el lenguaje en el que las redes neuronales "piensan" y realizan cálculos.

Los bloques fundamentales para representar y manipular datos en DL son:

- Vectores
- Matrices
- Tensores

Vector

Definición:

Un vector es un arreglo unidimensional de números organizados en un orden específico. Puedes imaginarlo como una sola fila o una sola columna de números.

Cada número dentro del vector se denomina elemento o componente.

Representación:

Suelen denotarse con letras minúsculas *v*, a veces con una flecha encima $\vec{v}$.

En programación, típicamente se representan como listas o arreglos.

Propiedades:

Dimensión (o longitud): El número de elementos en el vector.

Orden: El orden de los elementos importa. Un vector [1,2] es diferente de [2,1].

Operaciones: Se pueden sumar, restar, multiplicar por un escalar, realizar el producto punto, y más.
Estas operaciones son fundamentales en los cálculos que realizan las redes neuronales.

1. Vectores, Matrices y Tensores

… Vector

Suma de vectores

Sean $\boldsymbol{u} = [u_1, u_2, \ldots, u_n]$ y $\boldsymbol{v} = [v_1, v_2, \ldots, v_n]$, la suma de vectores se define como:

$$u + v = [u_1 + v_1, u_2 + v_2, \ldots, u_n + v_n]$$

$\boldsymbol{u} = [2, 3, 18, 7]$, $\boldsymbol{v} = [4, 1, 10, 3] \Rightarrow u + v =$

1. Vectores, Matrices y Tensores

… Vector

Resta de vectores

Sean $u = [u_1, u_2, \ldots, u_n]$ y $v = [v_1, v_2, \ldots, v_n]$, la suma de vectores se define como:

$$u - v = [u_1 - v_1, u_2 - v_2, \ldots, u_n - v_n]$$

$u = [2, 3, 18, 7]$, $v = [4, 1, 10, 3] \Rightarrow u - v =$

1. Vectores, Matrices y Tensores

… Vector

Multiplicar por un escalar

Sea $\boldsymbol{\alpha}$ un escalar y $\boldsymbol{u} = [\boldsymbol{u_1}, \boldsymbol{u_2}, \ldots, \boldsymbol{u_n}]$, la multiplicación por un escalar se define como:

$$\alpha\boldsymbol{u} = [\alpha\boldsymbol{u_1}, \alpha\boldsymbol{u_2}, \alpha\boldsymbol{u_n}]$$

$\alpha = 10$, $\boldsymbol{u} = [2, 3, 18, 7] \Rightarrow \alpha u =$

1. Vectores, Matrices y Tensores

… Vector

Producto elemento a elemento (element-wise product)

Sean $u = [u_1, u_2, \ldots, u_n]$ *y* $v = [v_1, v_2, \ldots, v_n]$*, el producto punto se define como:*

$$u \odot v = [u_1 v_1, u_2 v_2\ , u_n v_n]$$

$u = [1, 2, 3]$, $v = [4, -5, 6] \Rightarrow u \odot v =$

1. Vectores, Matrices y Tensores

… Vector

Producto punto/producto interno/producto escalar

Sean $u = [u_1, u_2, \ldots, u_n]$ *y* $v = [v_1, v_2, \ldots, v_n]$*, el producto punto se define como:*

$$u \cdot v = u_1 v_1 + u_2 v_2 + \ldots + u_n v_n$$

$u = [1, 2, 3]$, $v = [4, -5, 6] \Rightarrow u \cdot v =$

1. Vectores, Matrices y Tensores

… Vector

Ejemplos de uso en DL:

1. Características de una muestra de datos: En aprendizaje automático, cada punto de datos suele describirse mediante un conjunto de características. Estas características pueden organizarse en un vector.

Python

```
x = [5.5, 72, 23]  # Represents a data sample with:
                   # - Height: 5.5 (in some unit)
                   # - Weight: 72 (in some unit)
                   # - Age: 23 (in years)
```

2. Embeddings (como vectores de palabras): En el Procesamiento de Lenguaje Natural (NLP), las palabras suelen representarse como vectores densos llamados *embeddings*. Estos *embeddings* capturan el significado semántico de las palabras, de modo que aquellas con significados similares están más cercanas entre sí en el espacio vectorial.

Python

```
word_embedding_of_cat = [0.2, -0.5, 1.3, 0.8, -0.1]
word_embedding_of_dog = [0.1, -0.3, 1.5, 0.7, 0.2]
```

Modelos Generativos Profundos

UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

1. Ejercicios de vectores

Modelos Generativos Profundos

UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

1. Ejercicios de vectores

$$u = [1,2,3], \quad v = [4,-1,2], \quad a = 8$$

$$u + v = [1+4,\ 2+(-1),\ 3+2] = \boxed{[5,\ 1,\ 5]}$$

$$u - v = [1-4,\ 2-(-1),\ 3-2] = \boxed{[-3,\ 3,\ 1]}$$

$$a \cdot u = 8 \cdot [1,2,3] = \boxed{[8,16,24]}$$

$$u \odot v = [1 \cdot 4,\ 2 \cdot (-1),\ 3 \cdot 2] = \boxed{[4,\ -2,\ 6]}$$

$$u \cdot v = (1)(4) + (2)(-1) + (3)(2) = 4 - 2 + 6 = \boxed{8}$$

1. Vectores, Matrices y Tensores

Matriz

Definición:

Una matriz es un arreglo bidimensional de números ordenados en renglones y columnas.

Es en esencia una colección de vectores apilados.

Representación:

Suelen escribirse con letras en mayúsculas, algunas veces en negritas (p. ej. $\vec{A}$).

En programación, se suelen representar como listas de listas o como arreglos bidimensionales.

Propiedades:

Dimensión: El número de renglones y columnas (p.ej, una matrix de 3x4 tiene 3 renglones y 4 columnas. .

Elementos : Cada número en la matriz es un elemento, identificado por su renglón y su columna.

Operaciones: Se pueden sumar, restar, multiplicar por un escalar, multiplicar dos matrices, obtener la transpuesta de una matriz, etc.

La multiplicación de matrices es fundamental en los cálculos que realiza una red neuronal, ya que esta operación permite que se propague de manera eficiente la información entre sus diferentes capas.

1. Vectores, Matrices y Tensores

... Matriz

Suma (resta) de matrices

Si A y B son matrices de tamaño $m \times n$, entonces su suma $A + B$ es otra matriz C de tamaño $m \times n$ tal que $C[i][j] = A[i][j] + B[i][j]$

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\mathbf{A} + \mathbf{B} =$$

1. Vectores, Matrices y Tensores

… Matriz

Multiplicación por un escalar

Si *A* es una matriz de tamaño *m×n* y *α* es un escalar, entonces *αA* es una matriz *C* de tamaño *m×n* tal que *C*[*i*][*j*] = *k* * *A*[*i*][*j*].

$$\mathbf{A} = \begin{bmatrix} 1 & -2 \\ 3 & 0 \end{bmatrix}, \quad \alpha = 3$$

$$\alpha\mathbf{A} =$$

1. Vectores, Matrices y Tensores

… Matriz

Multiplicación de matrices

Si *A* es de tamaño *m×n* y *B* es de tamaño *n×p*, el producto *AB* es una matriz *C* de tamaño *m×p* donde:

C[*i*][*j*] = Σ *A*[*i*][*k*] * *B*[*k*][*j*] *para k* = *1* a *n*.

$1\cdot1 + 2\cdot3 = 7$

$$\begin{pmatrix}\boxed{1 \quad 2}\\3 \quad 4\\5 \quad 6\end{pmatrix}\begin{pmatrix}\boxed{\begin{matrix}1\\3\end{matrix}} & \begin{matrix}2\\4\end{matrix} & \begin{matrix}3\\5\end{matrix}\end{pmatrix}=\begin{pmatrix}7&10&13\\15&22&29\\23&34&45\end{pmatrix}$$

mxn nxp mxp

$$\mathbf{A}=\begin{bmatrix}1&2\\3&4\end{bmatrix}, \quad \mathbf{B}=\begin{bmatrix}2&0\\1&2\end{bmatrix}$$

$$\mathbf{AB}=$$

$$A=\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix} \quad x=\begin{bmatrix}7\\8\\9\end{bmatrix}$$

$$Ax=$$

1. Vectores, Matrices y Tensores

… Matriz

Multiplicación de matrices elemento a elemento (element-wise product)

$$A = [a_{ij}] \in \mathbb{R}^{m \times n} \qquad B = [b_{ij}] \in \mathbb{R}^{m \times n}$$

$$A \odot B = [\, a_{ij} \cdot b_{ij} \,]_{m \times n}$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \odot \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} =$$

1. Vectores, Matrices y Tensores

… Matriz

Transpuesta

La transpuesta de una matriz A de tamaño $m \times n$ es una matriz A^T de tamaño $n \times m$ tal que $A^T[i][j]$ = $A[j][i]$.

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad \mathbf{A}^\top =$$

1. Vectores, Matrices y Tensores

… Matriz

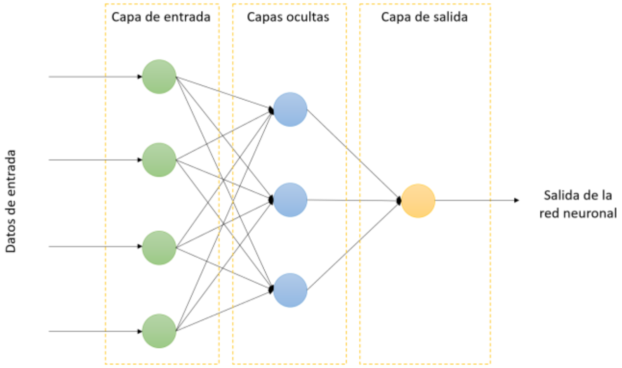
Ejemplos de uso en DL:

1. Un dataset (renglones:muestras, columnas:características): Una manera común de representar un dataset en ML es como una matriz en donde cada renglón corresponde a un solo ejemplo, y cada columna representa una característica específica.

Python

```
X = [[5.5, 72, 23],   # Sample 1: height, weight, age
    [5.1, 65, 27],   # Sample 2: height, weight, age
    [6.0, 80, 22]]   # Sample 3: height, weight, age
```

X es una matriz con 3 renglones (ejemplos) y 3 columnas (características).



$$W = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} & w_{1,4} \\ w_{2,1} & w_{2,2} & w_{2,3} & w_{2,4} \\ w_{3,1} & w_{3,2} & w_{3,3} & w_{3,4} \end{bmatrix}$$

2. Pesos en una capa de una red neuronal: Las conexiones entre neuronas en capas consecutivas de una red neuronal tienen pesos asociados. Estos pesos normalmente se organizan en una matriz.

Python

```
weights_layer1 = [[0.1, 0.2, -0.3, 0.05], # Weights for out-
put neuron 1
              [-0.2, 0.1, 0.4, -0.1], # Weights for output neu-
ron 2
              [0.3, -0.4, 0.2, 0.0]]  # Weights for output neu-
ron 3
```

Modelos Generativos Profundos

UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

2. Ejercicios matrices

Modelos Generativos Profundos

UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

2. Ejercicios matrices

1) $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad B = \begin{bmatrix} 2 & 0 \\ -1 & 3 \end{bmatrix}, \quad \alpha = 7$

$$A + B = \begin{bmatrix} 1+2 & 2+0 \\ 3+(-1) & 4+3 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 2 & 7 \end{bmatrix}$$

$$A - B = \begin{bmatrix} 1-2 & 2-0 \\ 3-(-1) & 4-3 \end{bmatrix} = \begin{bmatrix} -1 & 2 \\ 4 & 1 \end{bmatrix}$$

$$7A = \begin{bmatrix} 7\cdot 1 & 7\cdot 2 \\ 7\cdot 3 & 7\cdot 4 \end{bmatrix} = \begin{bmatrix} 7 & 14 \\ 21 & 28 \end{bmatrix}$$

$$AB = \begin{bmatrix} 1\cdot 2+2\cdot(-1) & 1\cdot 0+2\cdot 3 \\ 3\cdot 2+4\cdot(-1) & 3\cdot 0+4\cdot 3 \end{bmatrix} = \begin{bmatrix} 2-2 & 0+6 \\ 6-4 & 0+12 \end{bmatrix} = \begin{bmatrix} 0 & 6 \\ 2 & 12 \end{bmatrix}$$

$$A \odot B = \begin{bmatrix} 1\cdot 2 & 2\cdot 0 \\ 3\cdot(-1) & 4\cdot 3 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ -3 & 12 \end{bmatrix}$$

$$A^{\top} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$$

2. Ejercicios matrices

2) $A \in \mathbb{R}^{32\times 784}, \quad B \in \mathbb{R}^{784\times 256}, \quad C \in \mathbb{R}^{256\times 128}, \quad D \in \mathbb{R}^{128\times 10}.$

Se calcula

$(((AB)C)D)$

Determina las dimensiones de las matrices resultantes después de cada multiplicación.

Aplicando $(m \times k)\,(k \times n) = (m \times n)$:

AB: $(32 \times 784)(784 \times 256) \Rightarrow 32 \times 256.$

$(AB)C$: $(32 \times 256)(256 \times 128) \Rightarrow 32 \times 128$

$((AB)C)D$: $(32 \times 128)(128 \times 10) \Rightarrow 32 \times 10$

2. Ejercicios matrices

3) $P \in \mathbb{R}^{20\times 15}, \quad Q \in \mathbb{R}^{15\times 10}, \quad R \in \mathbb{R}^{8\times 5}.$

Verifica si las multiplicaciones indicadas son posibles. En caso de no serlo, identifica en qué paso ocurre el error y explica por qué.

La multiplicación falla en el **segundo paso** al intentar (20×10)(8×5)(20\times 10)(8\times 5)(20×10)(8×5) porque los tamaños internos (10 y 8) no coinciden.

1. Vectores, Matrices y Tensores

Tensor

Definición:

Es una generalización de los escalares (de 0 dimensiones), vectores (de 1 dimensión) y matrices (de 2 dimensiones) a dimensiones superiores (n-dimensionales).

Puede pensarse como un **arreglo multidimensional de números**.

Representación:

Son la estructura de datos fundamental en los marcos modernos de aprendizaje profundo como **TensorFlow** y **PyTorch**.

Estos frameworks ofrecen formas eficientes de crear, manipular y realizar cálculos sobre tensores.

Propiedades:

Rango (rank or order or number of axes): El número de dimensiones del tensor. Un escalar tiene rango 0, un vector tiene rango 1, una matriz tiene rango 2, un tensor en 3D tiene rango 3, etc.

Tipo de datos: El tipo de valores numéricos que se almacenan en un tensor (p.ej., enteros, flotantes, etc.) Los frameworks de DL normalmente usan números de punto flotante para los cálculos.

Operaciones: Soportan una amplia gama de operaciones matemáticas, muchas de las cuales están optimizadas para ser eficientes en GPUs.

Incluyen: operaciones elemento a elemento, multiplicación de matrices (generalizadas a altas dimensiones, reshaping, transpuesta, reducciones (como sumas y promedios a lo largo de los ejes), etc.

1. Vectores, Matrices y Tensores

Tensor

¿Por qué son importantes para DL?

1. Los datos y los pesos de las conexiones de las redes neuronales se almacenan en tensores.

- Todos los datos de entrada a una red neuronal (imágenes, texto, audio, etc.) se representa en tensores.
- Los parámetros que aprende la RN, tales como los pesos que conectan neuronas en capas diferentes y los términos *bias*, también se almacenan en tensores.

2. Las operaciones impulsan los cálculos neuronales.

- Los cálculos fundamentales dentro de las redes neuronales —como la propagación de la entrada a través de las capas y el cálculo de gradientes durante el entrenamiento— se realizan mediante operaciones con tensores.
- La multiplicación de matrices es esencial para transmitir información de una capa a la siguiente.
- Las operaciones elemento a elemento (como la aplicación de funciones de activación) también son fundamentales.

3. Comprenderlos te ayuda a depurar modelos y a optimizar el rendimiento:

- Depuración: Al construir y entrenar modelos de aprendizaje profundo, entender las formas (shapes) y los tipos de datos de tus tensores es esencial para identificar y corregir errores. Los desajustes de forma son una fuente común de fallos.

- Optimización de rendimiento: Conocer cómo se implementan las operaciones con tensores y cómo se organiza la información en memoria puede ayudarte a escribir código más eficiente.

  Los frameforks de DL suelen ofrecer funciones optimizadas para operaciones con tensores que aprovechan el procesamiento paralelo en GPUs.

  Comprender las dimensiones de los tensores también permite optimizar el uso de memoria y los tamaños de lote (batch sizes).

Modelos Generativos Profundos

Un enfoque desde la Creatividad computacional

3. Ejercicios de tensores

Modelos Generativos Profundos

UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

3. Ejercicios de tensores

$$A, B \in \mathbb{R}^{4\times 3\times 2}$$

$$A = \left[\begin{bmatrix}1 & 2\\3 & 4\\5 & 6\end{bmatrix}, \begin{bmatrix}7 & 8\\9 & 10\\11 & 12\end{bmatrix}, \begin{bmatrix}13 & 14\\15 & 16\\17 & 18\end{bmatrix}, \begin{bmatrix}19 & 20\\21 & 22\\23 & 24\end{bmatrix}\right]$$

$$B = \left[\begin{bmatrix}1 & 0\\0 & 1\\1 & 1\end{bmatrix}, \begin{bmatrix}0 & 1\\1 & 0\\0 & 1\end{bmatrix}, \begin{bmatrix}1 & 1\\1 & 1\\0 & 0\end{bmatrix}, \begin{bmatrix}0 & 1\\1 & 0\\1 & 0\end{bmatrix}\right]$$

$$C = A + B$$

$$C = \left[\begin{bmatrix}2 & 2\\3 & 5\\6 & 7\end{bmatrix}, \begin{bmatrix}7 & 9\\10 & 10\\11 & 13\end{bmatrix}, \begin{bmatrix}14 & 15\\16 & 17\\17 & 18\end{bmatrix}, \begin{bmatrix}19 & 21\\22 & 22\\24 & 24\end{bmatrix}\right]$$

$$C[0] = A[0] + B[0]$$

$$\begin{bmatrix}1+1 & 2+0\\3+0 & 4+1\\5+1 & 6+1\end{bmatrix} = \begin{bmatrix}2 & 2\\3 & 5\\6 & 7\end{bmatrix}$$

$$C[1] = A[1] + B[1]$$

$$\begin{bmatrix}7+0 & 8+1\\9+1 & 10+0\\11+0 & 12+1\end{bmatrix} = \begin{bmatrix}7 & 9\\10 & 10\\11 & 13\end{bmatrix}$$

$$C[2] = A[2] + B[2]$$

$$\begin{bmatrix}13+1 & 14+1\\15+1 & 16+1\\17+0 & 18+0\end{bmatrix} = \begin{bmatrix}14 & 15\\16 & 17\\17 & 18\end{bmatrix}$$

$$C[3] = A[3] + B[3]$$

$$\begin{bmatrix}19+0 & 20+1\\21+1 & 22+0\\23+1 & 24+0\end{bmatrix} = \begin{bmatrix}19 & 21\\22 & 22\\24 & 24\end{bmatrix}$$

Modelos Generativos Profundos

UN ENFOQUE DESDE LA CREATIVIDAD COMPUTACIONAL

3. Ejercicios de tensores

$$A, B \in \mathbb{R}^{4\times 3\times 2}$$

$$A = \left[\begin{bmatrix}1 & 2\\3 & 4\\5 & 6\end{bmatrix}, \begin{bmatrix}7 & 8\\9 & 10\\11 & 12\end{bmatrix}, \begin{bmatrix}13 & 14\\15 & 16\\17 & 18\end{bmatrix}, \begin{bmatrix}19 & 20\\21 & 22\\23 & 24\end{bmatrix}\right]$$

$$B = \left[\begin{bmatrix}1 & 0\\0 & 1\\1 & 1\end{bmatrix}, \begin{bmatrix}0 & 1\\1 & 0\\0 & 1\end{bmatrix}, \begin{bmatrix}1 & 1\\1 & 1\\0 & 0\end{bmatrix}, \begin{bmatrix}0 & 1\\1 & 0\\1 & 0\end{bmatrix}\right]$$

$$C = A \odot B$$

$$C = \left[\begin{bmatrix}1 & 0\\0 & 4\\5 & 6\end{bmatrix}, \begin{bmatrix}0 & 8\\9 & 0\\0 & 12\end{bmatrix}, \begin{bmatrix}13 & 14\\15 & 16\\0 & 0\end{bmatrix}, \begin{bmatrix}0 & 20\\21 & 0\\23 & 0\end{bmatrix}\right]$$

$$C[0] = A[0] \odot B[0]$$

$$\begin{bmatrix}1\cdot 1 & 2\cdot 0\\3\cdot 0 & 4\cdot 1\\5\cdot 1 & 6\cdot 1\end{bmatrix} = \begin{bmatrix}1 & 0\\0 & 4\\5 & 6\end{bmatrix}$$

$$C[1] = A[1] \odot B[1]$$

$$\begin{bmatrix}7\cdot 0 & 8\cdot 1\\9\cdot 1 & 10\cdot 0\\11\cdot 0 & 12\cdot 1\end{bmatrix} = \begin{bmatrix}0 & 8\\9 & 0\\0 & 12\end{bmatrix}$$

$$C[2] = A[2] \odot B[2]$$

$$\begin{bmatrix}13\cdot 1 & 14\cdot 1\\15\cdot 1 & 16\cdot 1\\17\cdot 0 & 18\cdot 0\end{bmatrix} = \begin{bmatrix}13 & 14\\15 & 16\\0 & 0\end{bmatrix}$$

$$C[3] = A[3] \odot B[3]$$

$$\begin{bmatrix}19\cdot 0 & 20\cdot 1\\21\cdot 1 & 22\cdot 0\\23\cdot 1 & 24\cdot 0\end{bmatrix} = \begin{bmatrix}0 & 20\\21 & 0\\23 & 0\end{bmatrix}$$

1. Vectores, Matrices y Tensores

numpy

```
import numpy as np
```

Creación de vectores, matrices y tensores en general

```
v = np.array([1, 2, 3])

A = np.array([[1, 2, 3], [4, 5, 6]])

T = np.array([
    [[ 1,  2,  3,  4],
     [ 5,  6,  7,  8],
     [ 9, 10, 11, 12]],

    [[13, 14, 15, 16],
     [17, 18, 19, 20],
     [21, 22, 23, 24]]
])
```

Operaciones Básicas con Vectores y Matrices

```
#Operaciones con vectores
alpha = 2
print('Suma:', u + v)
print('Resta:', u - v)
print('Multiplicación por escalar:', alpha * u)
print('Multiplicación elemento a elemento:', u*v)
print('Producto punto:', np.dot(u, v))

#Operaciones con matrices
alpha = 2
print('Suma A + B:\n', A + B)
print('Resta A - B:\n', A - B)
print('Multiplicación por escalar α·A:\n', alpha*A)
print('Multiplicación de matrices estándar AB:\n', np.dot(A,B))
print('Transpuesta de A:\n', A.T)
```

Tutorial_uso_tensores_con_numpy.ipynb
Ejercicios_vectores_matrices_tensores_con_Numpy1.ipynb
Ejercicios_vectores_matrices_tensores_con_Numpy2.ipynb