

King of Dumbs

el rey de los tontos

Nivel: ♠ Técnico

¿Cómo convierte un ordenador una foto en números?

Una foto son millones de números, y casi todo lo que la IA hace con ella es multiplicar y sumar. Vectores, producto escalar y matrices en tres niveles, con dos widgets.

Español Publicado 17 de octubre de 2026



Escanea para abrir la lección interactiva

<https://kingofdumbs.com/es/lecciones/vectores-y-matrices>

Qué vas a aprender

- Una foto es una lista de números: vectores y dimensión
- El producto escalar: multiplicar y sumar, y cuánto apuntan igual dos flechas
- Una matriz mueve el plano: sus columnas, el área y las dimensiones que quedan
- Multiplicar matrices: encadenar cambios, por qué importa el orden y qué hace una red neuronal

Símbolos que aparecen

E_a subíndice: la etiqueta de abajo (♦♠) · Σ sumatorio: súmalos todos (♠) · x_i índice i : el elemento número i (♠)
· $\approx$ aproximadamente igual (♦) · $\leq \geq$ menor o igual / mayor o igual (♠) · $|x|$ valor absoluto: sin signo (♠) · ·
punto: multiplicar (o producto escalar) (♦♠) · σ sigma: depende de la lección (♠) · θ theta: un ángulo (casi siempre) (♠) · λ lambda: un número que estira (♠) · $\|x\|$ norma: la longitud de la flecha (♠) · x^T traspuesta: tumbar la columna (♠) · $\in$ pertenece a (♠) · $\mathbb{R} \mathbb{Z}$ conjuntos de números (♠)

Haces una foto a tu gato y, si buscas “gato” en la galería del móvil, muchas veces la encuentra. Pero un ordenador no ve gatos: lo único que sabe hacer es sumar y multiplicar números, muy deprisa.

La pregunta tonta de esta lección es: **¿cómo convierte un ordenador una foto en números?** Y, ya puestos, ¿qué hace con ellos? La respuesta cabe en tres ideas que suenan a examen, vector, producto escalar y matriz, y en una sola cuenta: multiplicar y sumar.

Elige tu carta: ♣ si empiezas de cero, ♦ si no te asusta una fórmula sencilla, ♠ si quieres la notación. Puedes cambiarla en cualquier momento.

Una foto es una lista de números

♦ → ♠ Del intermedio al técnico

En ♦ decíamos	En ♠ se escribe
vector	$x \in \mathbb{R}^n$, una lista de n números reales
componentes	$x = (x_1, \dots, x_n)$
longitud	norma euclídea $\ x\ = \sqrt{(\sum_i x_i^2)}$
dimensión	$\dim \mathbb{R}^n = n$: los vectores de una base, $e_1, \dots, e_n$
canal	C; en RGB, $C = 3$, enteros de 8 bits en $\{0, \dots, 255\}$
alto × ancho × 3	tensor $H \times W \times C$ (HWC); torchvision usa $C \times H \times W$
se pone toda en fila	aplanar (flatten): $x \in \mathbb{R}^n$ con $n = H \cdot W \cdot C$

$$x \in \mathbb{R}^n, \quad x = (x_1, \dots, x_n)$$

$$\|x\| = \sqrt{(\sum_i x_i^2)}$$

$$\text{foto} \in \{0, \dots, 255\}^{(H \times W \times C)}, \quad C = 3$$

$$\text{ToTensor}: H \times W \times C \rightarrow C \times H \times W, \quad /255$$

$$n = H \cdot W \cdot C = 4000 \cdot 3000 \cdot 3 = 3,6 \times 10^7$$

$$\dim \mathbb{R}^n = n \quad (\text{base } e_1, \dots, e_n)$$

Una imagen RGB de 8 bits es un tensor de enteros en $\{0, \dots, 255\}^{(H \times W \times C)}$ con $C = 3$: Pillow 13.0 describe el modo «RGB (3x8-bit pixels, true color)» y dice que «an 8-bit pixel has a range of 0-255». torchvision 0.29 lo convierte con ToTensor, de $(H \times W \times C)$ en $[0, 255]$ a un tensor de coma flotante de forma $(C \times H \times W)$ en $[0, 1]$. Aplanado, es un vector de $\mathbb{R}^n$ con $n = H \cdot W \cdot C$; para 4000×3000 , $n = 3,6 \times 10^7$.

La dimensión de un espacio es el número de vectores de cualquier base (Austin, *Understanding Linear Algebra*, def. 3.5.6). La base canónica $e_1, \dots, e_n$ tiene n vectores, así que $\dim \mathbb{R}^n = n$: el número de componentes. La norma $\|x\| = \sqrt{(\sum_i x_i^2)}$ es Pitágoras en n dimensiones.



El producto escalar: multiplicar y sumar

♦ → ♠ Del intermedio al técnico

En ♦ decíamos	En ♠ se escribe
producto escalar	$a \cdot b = \sum_i a_i b_i$ (también $\langle a, b \rangle$ o $a^T b$)
36 millones de multiplicaciones	coste $O(n)$: n productos y $n - 1$ sumas
longitud de $a \times$ longitud de $b \times$ coseno del ángulo	$a \cdot b = \ a\ \ b\ \cos \theta$, con $\theta \in [0, \pi]$
El coseno vale 1 si apuntan igual, 0 si forman un ángulo recto y -1 si son opuestas	$-1 \leq \cos \theta \leq 1$, es decir, $ a \cdot b \leq \ a\ \ b\ $ (Cauchy-Schwarz)
perpendiculares	ortogonales: $a \cdot b = 0$ ($a \perp b$)
similitud coseno	$\cos \theta = a \cdot b / (\ a\ \ b\)$
la flecha se alarga pero no gira	$\cos(x, \lambda x) = 1$ para todo $\lambda > 0$

$$a \cdot b = \sum_i a_i b_i = \|a\| \|b\| \cos \theta$$

$$\cos \theta = a \cdot b / (\|a\| \|b\|) \in [-1, 1]$$

$$a \perp b \Leftrightarrow a \cdot b = 0$$

$$|a \cdot b| \leq \|a\| \|b\| \quad (\text{Cauchy-Schwarz})$$

$$\cos(x, 1, 2 \cdot x) = 1: \text{ más brillo, misma dirección}$$

$$\text{coste: } n \text{ productos, } n - 1 \text{ sumas: } O(n)$$

En $\mathbb{R}^n$, $a \cdot b = \sum_i a_i b_i$. OpenStax (*Calculus Volume 3*, 2.3) demuestra con la ley de los cosenos que $a \cdot b = \|a\| \|b\| \cos \theta$ (teorema 2.4), y que dos vectores no nulos son ortogonales si y solo si $a \cdot b = 0$ (teorema 2.5). En más de tres dimensiones se va al revés: el ángulo se define por $\cos \theta = a \cdot b / (\|a\| \|b\|)$, y la desigualdad de Cauchy-Schwarz, $|a \cdot b| \leq \|a\| \|b\|$, asegura que ese cociente cae en $[-1, 1]$.

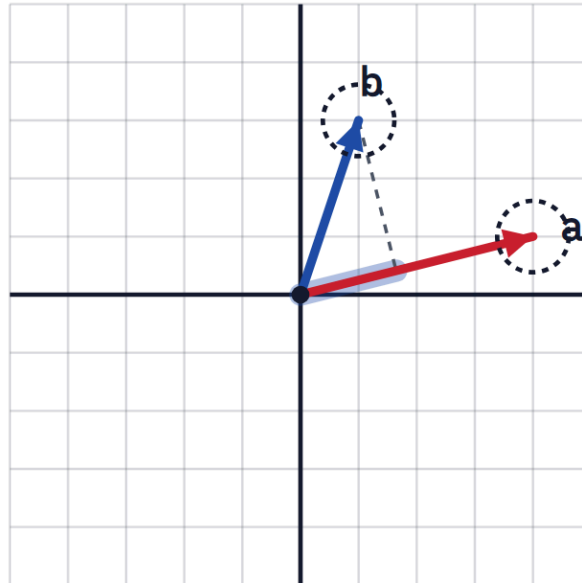
PyTorch 2.14 calcula la similitud coseno con `torch.nn.functional.cosine_similarity`, que divide entre $\max(\|x_1\| \|x_2\|, \epsilon)$ para no dividir entre cero. Como $\cos(x, \lambda x) = 1$ para $\lambda > 0$, la similitud coseno no distingue una foto de otra más clara o más oscura que solo multiplique sus valores. El coste es lineal: n productos y $n - 1$ sumas.

El Rey de los Tontos dice:

Multiplicar y sumar. No hay más. Con eso comparo precios, flechas y fotos, y aún me sobra tiempo para la siesta.

Prueba tú: dos flechas

Arrastra las puntas de las flechas, o muévelas con los deslizadores o con las flechas del teclado. Busca una posición en la que el producto escalar valga 0: las flechas quedarán en ángulo recto. La franja azul es la sombra de b sobre a: se alarga cuanto más apuntan hacia el mismo lado.



a, horizontal

a, vertical 1

b, horizontal

b, vertical 3

Producto escalar $a \cdot b$ $4 \cdot 1 + 1 \cdot 3 = 7$

Longitudes $\|a\| = 4,12 \cdot \|b\| = 3,16$

Ángulo $57,5^\circ$

Coseno (parecido de dirección) $0,54$

$a = (4, 1)$, $b = (1, 3)$, $a \cdot b = 7$. Apuntan hacia el mismo lado: el producto escalar es positivo.



Widget interactivo. Pruébalo en la web: <https://kingofdumbs.com/es/lecciones/vectores-y-matrices>

Una matriz mueve el plano

♦ → ♠ Del intermedio al técnico

En ♦ decíamos	En ♠ se escribe
matriz	$A \in \mathbb{R}^{(m \times n)}$, entradas a_{ij} (fila i , columna j)
cada fila	$(Ax)_i = \sum_j a_{ij}x_j = (\text{fila } i \text{ de } A) \cdot x$
la primera columna es adónde va $(1, 0)$ y la segunda, adónde va $(0, 1)$	$A = [Ae_1 \ Ae_2]$: la columna j es $T(e_j)$
cualquier punto es una mezcla de esos dos	linealidad: $A(c_1e_1 + c_2e_2) = c_1Ae_1 + c_2Ae_2$
tuerce	cizalla (shear), por ejemplo $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$
$ad - bc$	$\det A$, el área con signo del paralelogramo de las columnas
aplasta	$\det A = 0$: A no es invertible; el rango baja y la imagen es una recta
como en un espejo	$\det A < 0$: cambia la orientación
girar	rotación $R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$
de qué punto de la original sale	usa A^{-1} : recorre la salida y busca en la entrada
(más un desplazamiento)	transformación afín $x \mapsto Ax + t$

$$(Ax)_i = \sum_j a_{ij} x_j$$

$$A = [Ae_1 \ Ae_2], \quad A(cu + v) = cAu + Av$$

$$R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$$

$\det A = ad - bc$: área con signo

$\det A = 0 \Leftrightarrow$ no invertible $\Leftrightarrow$ rango < 2

$\begin{pmatrix} 1 & -1 \\ 2 & -2 \end{pmatrix}$: $\det 0$, rango 1, una recta

Una matriz A de $m \times n$ define $T(x) = Ax$ de $\mathbb{R}^n$ en $\mathbb{R}^m$, con $(Ax)_i = \sum_j a_{ij}x_j$. Sus columnas son las imágenes de la base: $A = [T(e_1) \ \cdots \ T(e_n)]$ (Austin, prop. 2.5.8). Como T es lineal, $T(\sum_j c_j e_j) = \sum_j c_j T(e_j)$, y con las columnas se sabe todo.

En 2×2 , la rotación de ángulo θ es $(\cos \theta \ -\sin \theta / \sin \theta \ \cos \theta)$, y $(1 \ 1 / 0 \ 1)$ es una cizalla (Austin, 2.6). $\det A = ad - bc$ es el área con signo del paralelogramo de las columnas (3.4), y A es invertible si y solo si $\det A \neq 0$ (prop. 3.4.12). Con $\det A = 0$ baja el rango: la imagen $\text{Col}(A)$ tiene dimensión $\text{rank } A$ (prop. 3.5.12), y con $(1 \ -1 / 2 \ -2)$ es una recta.

Pillow 13.0 (`ImageTransform.AffineTransform`) toma «the first two rows from the inverse of an affine transform matrix»: para cada píxel de salida (x, y) , lee la entrada en $(ax + by + c, dx + ey + f)$. Usa la inversa porque recorre la foto de salida, y el desplazamiento (c, f) hace la transformación afín, no lineal.



Prueba tú: deforma la cuadrícula

Mueve los cuatro números de la matriz o prueba los botones. La flecha roja es la primera columna, adónde va $(1, 0)$; la azul, la segunda. Busca una matriz que aplaste la cuadrícula en una recta: el área se hará 0. Y mira qué le pasa a la F con el espejo.



Multiplicar matrices: el orden importa

♦ → ♠ Del intermedio al técnico

En ♦ decíamos	En ♠ se escribe
multiplicar dos matrices	$(AB)_{ij} = \sum_k a_{ik} b_{kj}$
una fila de la primera por una columna de la segunda	(fila i de A) · (columna j de B)
una de $m \times k$ por una de $k \times n$ da una de $m \times n$	$(m \times k)(k \times n) \rightarrow m \times n$: las dimensiones interiores coinciden
AB	composición: $A(Bx) = (AB)x$; si $T \mapsto A$ y $S \mapsto B$, $S \circ T \mapsto BA$
AB no suele ser igual que BA	no conmutativa; sí asociativa: $(AB)C = A(BC)$
capa	capa lineal (<code>nn.Linear</code>): $y = xA^T + b$
pesos	$A \in \mathbb{R}^{(\text{out} \times \text{in})}$, aprendida; b, el sesgo
10 millones de multiplicaciones	$\text{out} \cdot \text{in} = 10^3 \cdot 10^4 = 10^7$ multiplicaciones-suma (FMA) por foto
un paso que no es una multiplicación	activación no lineal σ ; sin ella, $A_2(A_1x) = (A_2A_1)x$ es una sola capa
muchas fotos a la vez	lote (batch) de N: un GEMM de $(M \times K)(K \times N)$, $2 \cdot M \cdot N \cdot K$ FLOP

$$(AB)_{ij} = \sum_k a_{ik} b_{kj}$$

$$(m \times k)(k \times n) \rightarrow m \times n; \quad AB \neq BA$$

$$x \mapsto A(Bx) = (AB)x$$

$$\text{coste: } m \cdot n \cdot k \text{ FMA} = 2 \cdot m \cdot n \cdot k \text{ FLOP}$$

$$\text{nn.Linear: } y = xA^T + b, \quad A: (\text{out}, \text{in})$$

$$2 \cdot 1000 \cdot 64 \cdot 10000 = 1,28 \times 10^9 \text{ FLOP}$$

$(AB)_{ij} = \sum_k a_{ik} b_{kj}$, definido si las dimensiones interiores coinciden (OpenStax, *College Algebra 2e*, 7.5). Es asociativo y distributivo, pero no conmutativo. Si $T(x) = Ax$ y $S(x) = Bx$, $S \circ T$ tiene matriz BA (Austin, prop. 2.5.11): la que se aplica primero va a la derecha.

Cada una de las $m \cdot n$ entradas es un producto escalar de k términos: $m \cdot n \cdot k$ multiplicaciones-suma (FMA), es decir, $2 \cdot m \cdot n \cdot k$ FLOP. NVIDIA lo cuenta igual en su guía de GEMM, donde cada entrada es «a dot-product of K-element vectors». Para la capa de $1000 \times 10\,000$ con un lote de 64 fotos: $2 \cdot 1000 \cdot 64 \cdot 10\,000 = 1,28 \times 10^9$ FLOP (cálculo propio).

PyTorch 2.14 escribe la capa lineal como $y = xA^T + b$, con un ejemplo por fila y el peso A de forma (out_features, in_features). Por el sesgo b es afín, no lineal. NVIDIA pone los pesos en A y las activaciones en B , un ejemplo por columna: es la misma cuenta traspuesta. Sin una σ no lineal entre capas, la asociatividad haría que varias capas seguidas fueran una sola.

El Rey de los Tontos dice:

Yo no veo gatos: veo 36 millones de números. Lo difícil es saber por qué matrices multiplicarlos.



Reto final

1. Una foto en color de 100×100 píxeles. ¿Cuántos números guarda el ordenador al abrirla?

- a. 100
- b. 10 000, uno por píxel
- c. 30 000: tres por píxel, rojo, verde y azul

2. Dos flechas forman un ángulo recto. ¿Cuánto vale su producto escalar?

- a. 1
- b. 0
- c. Depende de lo largas que sean

3. Giras una foto 90° y luego la estiras a lo ancho. ¿Sale lo mismo que si la estiras primero y la giras después?

- a. Sí: multiplicar da lo mismo en cualquier orden
- b. No: con matrices, el orden importa
- c. Solo si la foto es cuadrada



En tres frases

1. Para un ordenador, una foto es una lista de números (tres por píxel, de 0 a 255): un vector, cuya dimensión es cuántos números tiene.
2. El producto escalar multiplica dos listas término a término y suma: da un total, como en la cuenta del frutero, y dice cuánto apuntan hacia el mismo lado dos flechas (0 si son perpendiculares).
3. Una matriz es una tabla que mueve cada punto del plano, y multiplicar matrices encadena esos cambios, con el orden importando: es la cuenta que una red neuronal hace millones de veces con cada foto.



Fuentes

- Vectores, longitud y producto escalar: OpenStax, *Calculus Volume 3*, [2.1](#) y [2.3](#) (el frutero, teoremas 2.4 y 2.5). Matrices y su producto: OpenStax, *College Algebra 2e*, [7.5](#).
- Transformaciones, determinante y dimensión: D. Austin, *Understanding Linear Algebra*, [2.2](#), [2.5](#), [2.6](#), [3.4](#) y [3.5](#).
- Píxeles y transformaciones de imágenes: Pillow 13.0, [Concepts](#) y [ImageTransform](#). De la foto al tensor: torchvision 0.29, [ToTensor](#).
- Redes neuronales: PyTorch 2.14, [nn.Linear](#) y [cosine_similarity](#). NVIDIA, [Matrix Multiplication Background](#) y [Linear/Fully-Connected Layers](#).

Cómo está hecha esta lección: investigada y redactada con ayuda de Claude a partir de las fuentes de arriba, consultadas el 17 de octubre de 2026. Las cifras sin enlace propio son cálculos a partir de sus definiciones. Los ejemplos (la foto de 4000 × 3000, las dos tiendas, la capa de 1000 salidas) son inventados. Los dibujos son propios.

Soluciones

1. **c)** 30 000: tres por píxel, rojo, verde y azul. Eso es. $100 \times 100 \times 3$: un vector de dimensión 30 000.
2. **b)** 0. Bien. Perpendiculares, producto escalar 0, sean como sean de largas.
3. **b)** No: con matrices, el orden importa. Exacto. Como el calcetín y el zapato.

Lección de King of Dumbs, kingofdumbs.com. Versión para leer sin conexión; la original, con widgets que se tocan, está en: <https://kingofdumbs.com/es/lecciones/vectores-y-matrices>

Actualizada el 17 de octubre de 2026.