Facsímil 11

IA multimodal

Capítulo 03

CLIP y aprendizaje contrastivo: alinear texto e imagen

686f6c61 · 2026

Capítulo 03: CLIP y aprendizaje contrastivo: alinear texto e imagen

Qué deberías poder hacer al terminar

En el capítulo anterior vimos cómo una imagen se convierte en patches y tokens visuales. Ahora damos el salto que permite muchas experiencias que parecen naturales: buscar una imagen con una frase, clasificar imágenes con nombres de clases escritos en lenguaje natural o encontrar productos parecidos sin entrenar un clasificador específico para cada etiqueta.

La idea central de CLIP es sencilla de decir y muy potente de usar: entrenar un encoder de imagen y un encoder de texto para que la imagen y su descripción queden cerca en el mismo espacio vectorial.[1] Pero si lo dejamos en esa frase, suena a humo. Lo importante para ingeniería está en los pares, la matriz de similitud, la temperatura, los negativos y las métricas.

Al terminar este capítulo deberías poder hacer esto:

RESULTADO DE APRENDIZAJE	EVIDENCIA DE QUE LO SABES HACER
Explicar un entrenamiento contrastivo imagen-texto.	Puedes dibujar imagen encoder, texto encoder, matriz de similitudes y diagonal positiva.
Calcular una similitud coseno.	Sabes comparar embeddings normalizados sin convertirlo en magia semántica.
Entender la temperatura en el softmax.	Puedes explicar por qué una distribución se vuelve más o menos agresiva.
Leer `Recall@k` en retrieval multimodal.	No confundes ranking bonito con evaluación seria.
Detectar negativos duros.	Sabes buscar casos parecidos que rompen el sistema antes de producción.
Practicar el ranking en el cuaderno.	Lo abres en Google Colab, generas la matriz y justificas una decisión.

La frase central del capítulo es esta:

CLIP no aprende “la verdad” de una imagen. Aprende una geometría útil entre imágenes y textos, y esa geometría hay que evaluarla con casos reales.

La escena: buscar una silla sin tener la etiqueta exacta

Imagina un catálogo interno con miles de productos. Alguien busca: “silla negra de oficina con ruedas y respaldo alto”. En una base de datos perfecta, tendríamos esa descripción exacta en metadatos. Pero la vida suele ser menos limpia: unas fichas dicen “silla ergonómica”, otras “asiento oficina”, otras tienen foto buena y texto pobre.

Un sistema imagen-texto permite hacer algo útil: codificar la frase como vector, codificar las imágenes como vectores y ordenar por similitud. Si funciona bien, la silla correcta aparece arriba aunque la etiqueta exacta no exista.

Ahora cambia el dominio. En soporte, quieres encontrar capturas parecidas: “botón desactivado por campo obligatorio”. En documentación, quieres localizar una factura con tabla de importes. En industria, quieres buscar un manómetro con aguja en zona alta. Son tareas distintas, pero comparten una idea: **comparar representaciones de modalidades distintas**.

La trampa es creer que una búsqueda que “parece” acertar ya sirve. Puede fallar por fondo visual, texto ambiguo, clases mal redactadas, dominios no vistos, sesgos del dataset o descripciones demasiado pobres. Por eso este capítulo insiste en medir.

Volvemos al caso guía: recuperar casos parecidos

En la solicitud de beca bloqueada, CLIP o un modelo similar no debería decidir la resolución final. Sí puede ayudar a recuperar evidencias o casos parecidos:

CONSULTA	QUÉ PODRÍA RECUPERAR	QUÉ NO DEBE DECIDIR SOLO
“solicitud de beca bloqueada por documento pendiente”	Capturas parecidas de formularios bloqueados.	Si la política vigente permite enviar o no.
“documento de política de becas con fecha límite”	PDFs o páginas parecidas.	Si el expediente concreto cumple la cláusula.
“captura con botón desactivado y alerta roja”	Incidencias visualmente similares.	La causa operativa si la tabla de estados contradice la UI.

El patrón útil es: retrieval primero, decisión después. El ranking trae candidatos. El sistema completo debe leer evidencia, validar estado, citar fuente y bloquear si hay conflicto.

Lectura de ingeniería: CLIP como ranking, no como veredicto

CLIP y los modelos contrastivos son extraordinariamente útiles cuando el problema es encontrar candidatos. Esa palabra, “candidatos”, es la clave. Un ranking no es una decisión final. Si buscas “botón desactivado por documento pendiente” y recuperas cinco capturas parecidas, has reducido el espacio de búsqueda. Todavía no has demostrado que el caso actual tenga la misma causa, ni que la política sea la misma, ni que la persona pueda enviar la solicitud.

El aprendizaje contrastivo funciona porque empuja pares relacionados a estar cerca y pares no relacionados a estar lejos. Pero esa geometría hereda el mundo que vio durante entrenamiento: descripciones ruidosas, sesgos visuales, dominios sobrerrepresentados, textos incompletos y asociaciones que pueden no valer en tu producto. En un catálogo general puede funcionar muy bien. En documentos administrativos con formularios internos, quizá necesita evaluación privada y negativos duros diseñados a mano.

Qué hace bien y qué no debes pedirle

Un modelo CLIP-like sirve muy bien para **recuperar**: dame imágenes parecidas a este texto, textos parecidos a esta imagen o candidatos visuales cercanos a una clase descrita en lenguaje natural. Ese uso encaja con catálogos, buscadores internos, deduplicación visual, clasificación inicial y routing de documentos. Pero no conviene tratarlo como un verificador final. La similitud no entiende permisos, reglas de negocio, vigencia legal, estado operativo ni consecuencias.

En el caso de beca, un ranking puede traer capturas similares a “formulario bloqueado por documento pendiente”. Eso es útil para encontrar patrones y ejemplos. Pero la causa real debe comprobarse en fuentes más fuertes: tabla de estados, política vigente y región concreta de la UI. Si el primer resultado del ranking muestra otra beca, otra convocatoria o una alerta visual parecida con una causa distinta, una decisión automática sería frágil.

Los negativos duros son el examen de verdad

El aprendizaje contrastivo se ve bonito cuando los negativos son fáciles. “Gato” frente a “avión” separa bien. En ingeniería nos importan los negativos que se parecen demasiado: una factura con la misma plantilla pero moneda distinta, una captura con botón desactivado por otra razón, una página de política antigua, un producto visualmente parecido pero de categoría prohibida, una gráfica con la misma forma pero otro eje. Esos negativos duros revelan si el ranking distingue lo que el negocio necesita distinguir.

Por eso un dataset privado de evaluación no debería contener solo ejemplos correctos y errores obvios. Debe incluir confusiones esperables. En una práctica útil, el alumno debería poder abrir el reporte y decir: “el sistema recupera bien casos visualmente parecidos, pero confunde documentos administrativos con capturas de soporte cuando la descripción menciona beca; necesito metadatos o reranking”. Esa frase ya es ingeniería: identifica el fallo, no se limita a decir que “el modelo va regular”.

Cómo se usa como componente

Un ingeniero debería leer cada resultado de CLIP como una hipótesis: “esto se parece a la consulta”. La siguiente capa debe comprobar si la evidencia real aguanta. Si recuperas una factura parecida, necesitas leer campos. Si recuperas una página de política, necesitas sección y versión. Si recuperas una captura de UI, necesitas región y estado. Si recuperas un vídeo o una imagen con texto, necesitas tratar ese texto como dato, no como instrucción.

La pregunta práctica no es “¿CLIP acierta?”. Es “¿el ranking trae lo que necesito antes de gastar en una llamada más cara o tomar una decisión sensible?”. Por eso en producción miramos Recall@k, casos difíciles, cobertura por dominio y ejemplos donde el primer resultado parece plausible pero lleva a una conclusión equivocada. También miramos coste de indexado, frecuencia de refresco, idioma de las consultas, plantillas de prompt para clases y qué filtros deterministas se aplican antes o después del ranking.

Un diseño razonable suele parecerse a esto: filtros duros por permisos y tipo de documento; ranking contrastivo para traer candidatos; reranking o extracción especializada para comprobar evidencia; decisión con reglas o modelo más caro; y registro de la fuente usada. CLIP no desaparece en ese flujo. Ocupa su sitio correcto.

Qué problema resuelve CLIP

CLIP resuelve una forma de alineación: hacer que texto e imagen vivan en un espacio donde podamos compararlos. Esto encaja con la taxonomía de aprendizaje multimodal: representación, alineación y fusión son problemas distintos.[2]

En un sistema CLIP-like hay dos encoders:

ENCODER	ENTRADA	SALIDA
Encoder visual	Imagen ya preprocesada: patches, CNN features o tokens visuales.	Vector de imagen.
Encoder textual	Texto tokenizado: descripción, clase o prompt.	Vector de texto.

Después ambos vectores se comparan. Si una imagen y su texto corresponden, deberían estar cerca. Si no corresponden, deberían estar más lejos.

La consecuencia práctica es enorme:

USO	QUÉ PERMITE
Búsqueda texto->imagen	“Muéstrame capturas con botón desactivado”.
Búsqueda imagen->texto	“Qué descripción encaja mejor con esta imagen”.
Clasificación zero-shot	Comparar una imagen contra textos de clases: “foto de factura”, “foto de producto”, “captura de app”.
Detección de duplicados aproximados	Agrupar imágenes semánticamente parecidas.
Filtrado previo para VLM	Recuperar candidatos antes de pedir razonamiento más caro.

Pero CLIP no sustituye todo. No garantiza OCR preciso, no valida importes, no entiende permisos, no da evidencia espacial por sí solo y no convierte una búsqueda en una decisión de negocio. Es una pieza de representación y ranking.

Qué no resuelve CLIP

Esta sección es importante porque CLIP se presta a demos muy vistosas. Si buscas “factura con total”, puede encontrar una imagen parecida. Eso no significa que haya leído el total ni que pueda aprobar una operación.

NECESIDAD	POR QUÉ CLIP NO BASTA	PIEZA QUE SUELE FALTAR
Leer texto pequeño	El embedding puede capturar semántica general sin OCR fiable.	OCR, VLM con lectura, recorte de región o parser documental.
Citar una región exacta	Un vector global no dice necesariamente qué patch justificó la salida.	Grounding, bounding boxes, evidencia por región.
Validar importes	La similitud visual no recalcula sumas ni impuestos.	Reglas, extracción estructurada y validación numérica.
Saber estado operativo	La imagen puede estar desactualizada.	Consulta a tabla, API o sistema fuente.
Autorizar una acción	Recuperar una captura parecida no da permiso.	Política, aprobación y trazas.
Explicar causalidad	Cercanía vectorial no demuestra causa.	Modelo de tarea, evidencia y revisión.

En nuestro caso guía, CLIP puede recuperar capturas y políticas similares. La causa final de la beca bloqueada sale de combinar captura, PDF, tabla de estados y contrato de salida.

El mecanismo: pares positivos y negativos

Supón un batch con B pares imagen-texto:

$$\{(I_1, T_1), (I_2, T_2), \ldots, (I_B, T_B)\}$$

Cada imagen I_i tiene un texto positivo T_i. Dentro del mismo batch, los otros textos funcionan como negativos para esa imagen. Es decir, para I_1, el positivo es T_1, y $T_2, T_3, \ldots, T_B$ son negativos.

El esquema mínimo es:

1. Codificar imágenes.
2. Codificar textos.
3. Normalizar vectores.
4. Calcular matriz de similitud.
5. Empujar la diagonal hacia arriba.
6. Empujar el resto hacia abajo.

Hay una elegancia escondida en ese esquema, y conviene detenerse en ella. Nadie etiqueta a mano qué es un negativo: dentro de un batch, el texto correcto de una imagen es su positivo y todos los demás textos del batch funcionan, automáticamente, como sus negativos. Eso significa que un batch de B pares genera del orden de B negativos por imagen sin ningún coste de anotación, y que el tamaño del batch importa de verdad: con más pares a la vez, el modelo se enfrenta a más confusiones potenciales y se ve obligado a afinar el criterio. Es la diferencia entre un examen de dos opciones y uno de cien, cuantas más alternativas parecidas hay, más fino tiene que ser el límite que las separa.

No hace falta pensar todavía en millones de ejemplos. Con seis pares ya se entiende el mecanismo, y la clave es justamente esa: el modelo no recibe una clase cerrada como “silla”, sino muchos pares de imagen y texto, y aprende a separar correspondencias en vez de memorizar etiquetas. Por eso luego puede comparar una imagen contra una descripción que nunca vio escrita exactamente así.

Similitud coseno

La similitud coseno compara dirección entre vectores:

$$\mathrm{sim}(u, v) = \frac{u \cdot v}{\|u\|\|v\|}$$

SÍMBOLO	SIGNIFICADO	EN CLIP
u	Vector de una modalidad.	Embedding de imagen.
v	Vector de otra modalidad.	Embedding de texto.
$u \cdot v$	Producto escalar.	Aumenta si apuntan en direcciones parecidas.
$\|u\|$, $\|v\|$	Normas de los vectores.	Evitan que gane solo el vector más grande.

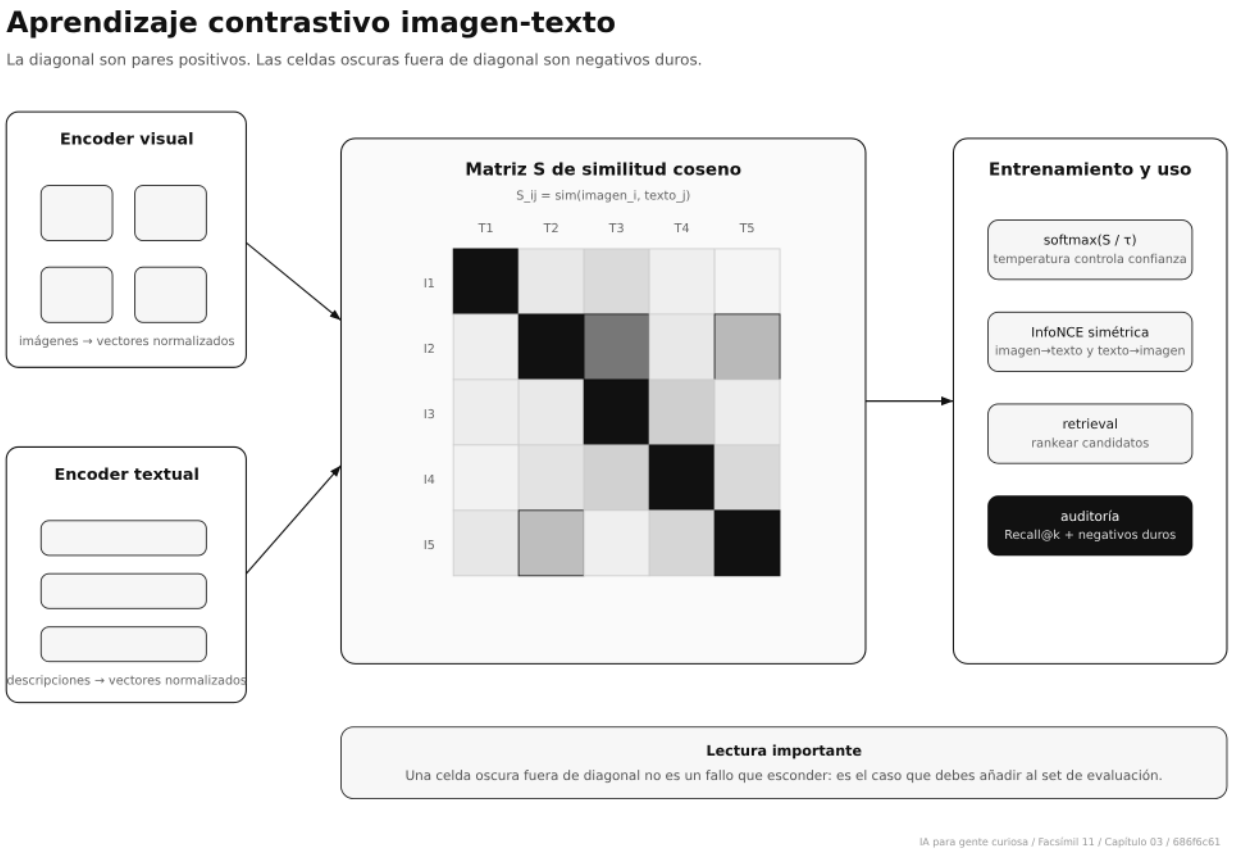
Cuando normalizamos embeddings, el producto escalar y el coseno quedan muy relacionados. En términos de ingeniería, esto hace que podamos indexar y rankear con operaciones vectoriales.

La matriz de similitudes para un batch de B pares es:

$$S_{ij} = \mathrm{sim}(f_{\text{img}}(I_i), f_{\text{text}}(T_j))$$

PIEZA	QUÉ SIGNIFICA
f_{img}	Encoder de imagen.
f_{text}	Encoder de texto.
S_{ij}	Similitud entre la imagen i y el texto j.
Diagonal S_{ii}	Pares positivos del batch.
Fuera de diagonal	Negativos del batch.

La diagonal debería ser alta. Las celdas altas fuera de diagonal son confusiones. Y esas confusiones son oro para evaluar.



Temperatura y softmax

La similitud sola no basta para entrenar. Necesitamos convertir puntuaciones en una distribución de probabilidad. Para una imagen I_i, comparamos contra todos los textos del batch y aplicamos softmax:

$$P(T_j \mid I_i) = \frac{\exp(S_{ij}/\tau)}{\sum_{k=1}^{B} \exp(S_{ik}/\tau)}$$

SÍMBOLO	SIGNIFICADO
S_{ij}	Similitud entre imagen i y texto j.
B	Tamaño del batch.
τ	Temperatura.
$P(T_j \mid I_i)$	Probabilidad asignada al texto j para la imagen i.

La temperatura τ es muy importante:

TEMPERATURA	EFECTO	RIESGO
Baja	Vuelve el softmax más concentrado: el top gana mucho.	Puede volverse demasiado confiado ante diferencias pequeñas.
Alta	Suaviza diferencias entre candidatos.	Puede separar peor positivos y negativos.

En producción no solemos tocar la temperatura de un CLIP entrenado como si fuera un dial mágico, pero sí conviene entender el concepto porque reaparece en decodificación, clasificación, calibración y ranking. Es la misma idea general que vimos al hablar de parámetros de generación: una distribución puede ser más puntiaguda o más suave.

Pérdida InfoNCE

La pérdida contrastiva busca que el positivo tenga más probabilidad que los negativos. Una forma de escribir la pérdida imagen->texto para un par i es:

$$\mathcal{L}_i^{I \to T} = -\log \frac{\exp(S_{ii}/\tau)}{\sum_{j=1}^{B} \exp(S_{ij}/\tau)}$$

Y de forma simétrica, texto->imagen:

$$\mathcal{L}_i^{T \to I} = -\log \frac{\exp(S_{ii}/\tau)}{\sum_{j=1}^{B} \exp(S_{ji}/\tau)}$$

La pérdida total suele promediar ambas direcciones:

$$\mathcal{L} = \frac{1}{2B} \sum_{i=1}^{B} \left(\mathcal{L}_i^{I \to T} + \mathcal{L}_i^{T \to I} \right)$$

Esta familia de objetivos conecta con aprendizaje contrastivo y con ideas como InfoNCE, popularizadas en representación contrastiva.[3]

No memorices la fórmula como si fuera decoración. Lee lo que obliga a hacer:

PARTE	LECTURA PRÁCTICA
Numerador $\exp(S_{ii}/\tau)$	Queremos subir el score del par correcto.
Denominador	Comparamos contra todos los candidatos del batch.
$-\log$	Penaliza mucho si el positivo recibe poca probabilidad.
Dirección simétrica	No basta imagen->texto; también queremos texto->imagen.

Conviene saber que el softmax de InfoNCE no es la única forma de escribir esta pérdida. Su denominador obliga a normalizar sobre todo el batch, lo que acopla cada par con todos los demás y se vuelve costoso cuando el lote es enorme. SigLIP propone una alternativa: en lugar de un softmax sobre el batch, usa una pérdida sigmoide que trata cada par imagen-texto de forma independiente, como un problema binario de "¿coinciden o no?".[4] El resultado práctico es que escala mejor a batches muy grandes y funciona bien incluso con lotes pequeños, sin necesitar la normalización global. La idea de fondo no cambia, acercar positivos y alejar negativos; cambia la contabilidad de cómo se comparan.

El espacio compartido y el modality gap

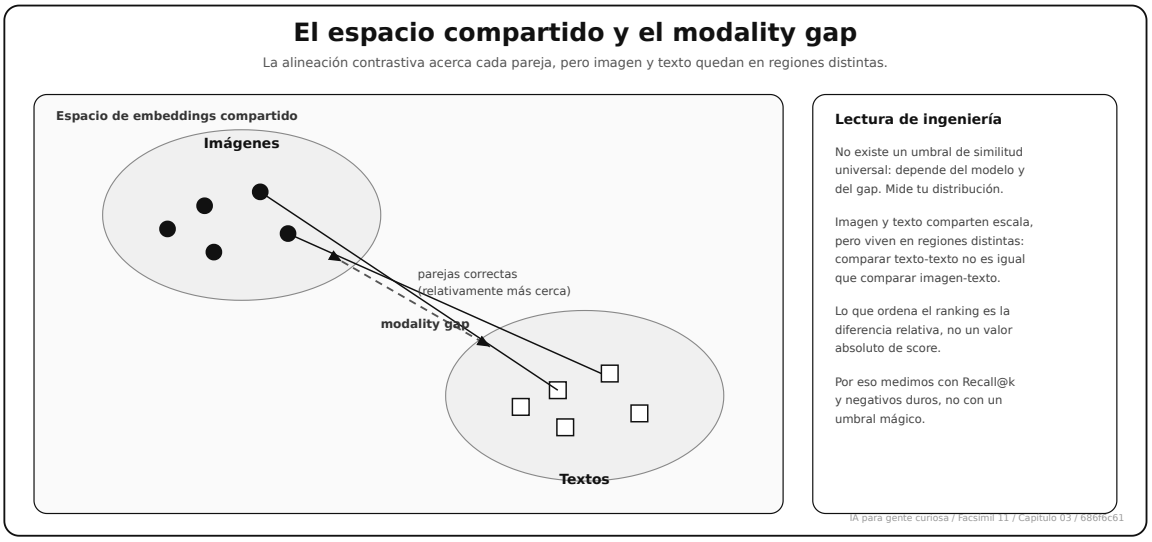
El entrenamiento contrastivo hace algo muy concreto con el espacio de embeddings: acerca cada imagen a su descripción y aleja las parejas que no van juntas. Visto desde fuera, parece que imagen y texto acaban "en el mismo sitio". En la práctica no es exactamente así, y conviene saberlo antes de fijar umbrales.

Cuando se miran los embeddings de un CLIP entrenado, las imágenes tienden a agruparse en una zona del espacio y los textos en otra, con una separación entre las dos modalidades que se conoce como *modality gap*.[5] La alineación no funde imagen y texto en un punto común; los coloca de forma que, dentro de la pareja correcta, la similitud sube más que con las parejas incorrectas. Lo que importa para rankear es la diferencia relativa, no un valor absoluto.

La consecuencia de ingeniería es directa:

SI CREES QUE...	LA REALIDAD ES...
Una similitud de 0,3 es "baja" en absoluto.	El umbral depende del modelo y del gap; mide tu distribución, no un número universal.
Imagen y texto son intercambiables en el espacio.	Comparten escala, pero viven en regiones distintas; comparar texto-texto e imagen-texto no es lo mismo.
Un score alto garantiza acierto.	El score ordena candidatos; el acierto se mide con casos reales y negativos duros.

Por eso, a lo largo del capítulo, insistimos en rankear y medir con `Recall@k` en vez de fijar un umbral mágico de similitud.



> El entrenamiento contrastivo acerca cada imagen a su descripción, pero las dos modalidades quedan en regiones separadas del espacio. Lo que decide el ranking es la cercanía relativa de la pareja, no un score absoluto.

Qué son los negativos duros

Un negativo duro es un ejemplo incorrecto que se parece mucho al positivo. Para catálogo, puede ser una silla negra muy parecida pero sin ruedas. Para facturas, una pizarra con números puede parecerse visualmente a una tabla. Para soporte, dos capturas de formularios pueden compartir layout aunque el error sea distinto.

Los negativos duros son incómodos y necesarios. Si tu evaluación solo tiene casos fáciles, el sistema parecerá mejor de lo que es.

TIPO DE NEGATIVO	EJEMPLO	QUÉ REVELA
Visualmente parecido	Dos productos casi iguales.	El embedding no separa atributos finos.
Textualmente parecido	“Factura con tabla” frente a “pizarra con columnas”.	El texto no contiene suficiente criterio.
Dominio parecido	Capturas de dos apps internas.	El modelo agrupa por estética, no por causa.
Fondo parecido	Fotos en el mismo entorno.	El dataset enseña correlaciones de fondo.
Clase incompleta	“Silla” cuando importa “silla con ruedas”.	La etiqueta es demasiado pobre.

Si encuentras un negativo duro, no lo escondas. Añádelo al set de evaluación y decide si necesitas mejor texto, metadatos, filtros, OCR, reranking o revisión humana.

Zero-shot: útil, pero no mágico

Una de las aportaciones prácticas de CLIP fue mostrar clasificación zero-shot con prompts de texto. En vez de entrenar un clasificador para cada clase, comparamos la imagen contra descripciones:

- “una foto de una factura”
- “una captura de una aplicación”
- “una foto de una silla de oficina”
- “un manómetro industrial”
- “una pizarra con planificación”

La clase con mayor similitud gana. Esto puede funcionar sorprendentemente bien, pero depende muchísimo de cómo escribas las clases, del dominio, del idioma, de la imagen y del dataset de entrenamiento.

DECISIÓN	QUÉ PROBARÍA
Redacción de clase	Comparar “factura” frente a “documento contable con tabla de importes”.
Idioma	Probar español, inglés o ambos si el modelo fue entrenado mayoritariamente en inglés.
Prompts plantilla	“una foto de ...”, “una captura de ...”, “un documento que contiene ...”.
Negativos explícitos	Añadir clases que suelen confundirse.
Métrica por slice	Ver si falla en capturas oscuras, documentos escaneados, productos parecidos o texto pequeño.

Zero-shot no significa “sin evaluación”. Significa “sin entrenamiento específico para esa tarea”. La evaluación sigue siendo obligatoria.

Datasets imagen-texto y ruido

CLIP se entrenó con pares imagen-texto a gran escala. El enfoque abrió una línea muy influyente: usar lenguaje natural como supervisión amplia para aprender representaciones visuales transferibles.[6] Datasets abiertos como LAION-5B muestran la escala y los retos de construir colecciones masivas imagen-texto.[7]

Aquí conviene ser serio, porque la calidad de esa supervisión decide lo que el modelo aprende. Un par tomado de la web puede tener un texto que no describe la imagen, sino el contexto donde apareció: una foto de una silla cuyo caption es “oferta de la semana”, o una captura de pantalla acompañada de “haz clic aquí”. Tomado de uno en uno parece inofensivo, pero cuando esos pares entran por millones el modelo no aprende lo que esperabas, aprende la correlación que de verdad había en los datos. Por eso conviene conocer los tipos de ruido que traen los pares web:

RUIDO	EJEMPLO	CONSECUENCIA
Texto incompleto	Imagen de producto con título genérico.	El modelo aprende señales vagas.
Texto incorrecto	Caption que no describe la imagen.	Se ensucia la alineación.
Sesgo de dominio	Muchas fotos de stock, pocas capturas internas.	Mal rendimiento en tu aplicación.
Idioma desigual	Mucho inglés, poco español técnico.	Prompts en español pueden requerir pruebas cuidadosas.
Correlaciones espurias	Fondo, marca de agua, estilo visual.	El ranking aprende atajos.

Por eso no basta decir “uso CLIP”. Hay que decir qué modelo, qué datos, qué idioma, qué dominio, qué métrica y qué fallos aceptas.

Cómo usarlo en un proyecto real

Un patrón razonable para una primera búsqueda multimodal:

1. Define la tarea: producto, documento, captura, incidencia, catálogo.
2. Crea un set pequeño de consultas reales.
3. Prepara imágenes reales o sintéticas representativas.
4. Calcula embeddings de imágenes.
5. Calcula embeddings de textos o consultas.
6. Rankea por similitud.
7. Mide `Recall@k`, errores y negativos duros.
8. Añade metadatos y filtros.
9. Decide si necesitas reranking, OCR, VLM o revisión humana.

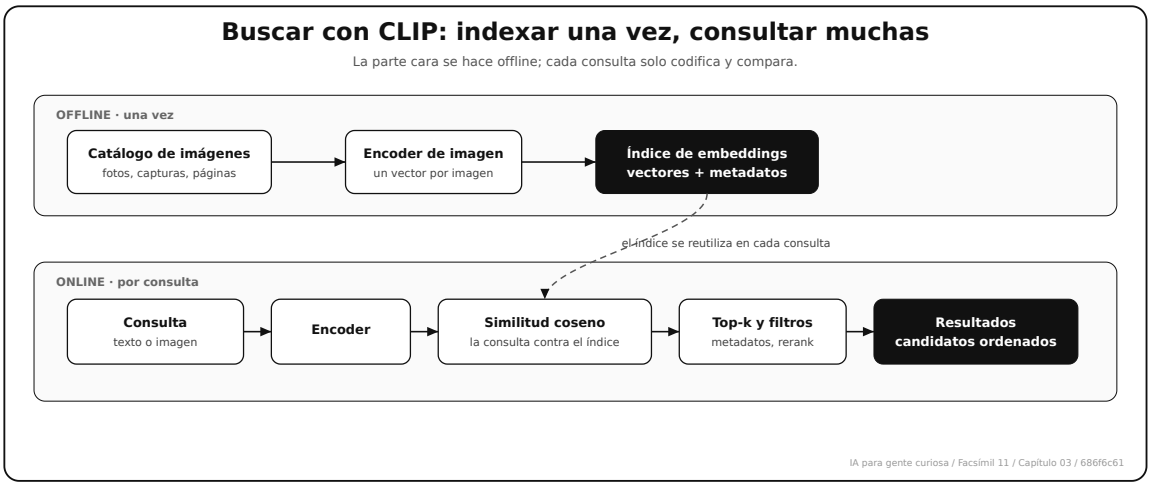
Ejemplo para catálogo:

CAPA	DECISIÓN
Imagen	Foto principal, fondo recortado, varias vistas si existen.
Texto	Nombre, descripción, atributos importantes.
Metadatos	Categoría, color, talla, disponibilidad, precio.
Ranking	Embedding multimodal + filtros de negocio.
Métrica	`Recall@10`, precisión por categoría, tasa de confusiones caras.
Revisión	Muestras de errores con negativos duros.

Ejemplo para soporte:

CAPA	DECISIÓN
Imagen	Captura de pantalla, recortes de alerta, modo claro/oscuro.
Texto	Descripción del usuario y etiquetas internas.
OCR	Extraer mensajes visibles si el texto pequeño decide.
Ranking	Recuperar incidencias parecidas.
Métrica	`Recall@5` de caso similar y utilidad para resolver.
Revisión	Confirmar antes de sugerir una acción irreversible.

Conviene separar lo que se hace una vez de lo que se hace en cada consulta. La parte cara, codificar el catálogo, se hace **offline** y se guarda en un índice; en **online**, solo codificas la consulta y la comparas contra ese índice. Esa separación es la que hace que la búsqueda escale a millones de imágenes sin recalcular nada.



Indexar el catálogo es caro y se hace una vez; cada consulta solo codifica la consulta y la compara contra el índice. Por eso CLIP escala a búsquedas grandes.

Métricas que no deberías saltarte

`Recall@k` mide si el elemento correcto aparece entre los k primeros resultados:

$$\text{Recall@k} = \frac{\text{consultas con positivo en top k}}{\text{número total de consultas}}$$

MÉTRICA	QUÉ RESPONDE	CUIDADO
`Recall@1`	¿El primero suele ser el correcto?	Muy exigente; útil si el usuario ve solo una respuesta.
`Recall@5`	¿El correcto aparece entre varias opciones?	Útil si hay interfaz de selección.
MRR	¿A qué distancia aparece el positivo?	Penaliza que el positivo baje en ranking.
Margen positivo	¿Cuánto separa el positivo del mejor negativo?	Margen bajo anticipa fragilidad.
Error por slice	¿Dónde falla?	Es lo que descubre problemas de dominio, idioma o formato.

No midas solo media global. Divide por categoría, idioma, resolución, fuente, tipo de imagen, presencia de texto, dominio y coste de error.

Cuando `Recall@1 = 1.0` no basta

Un set pequeño puede dar `Recall@1 = 1.0` y aun así no estar listo. Mira el margen. Si el positivo gana por muy poco, una imagen real un poco borrosa, una descripción más corta o una clase nueva pueden romper el ranking.

SEÑAL	LECTURA
Positivo top 1 con margen alto	Buen candidato, aunque hay que seguir midiendo por slices.
Positivo top 1 con margen bajo	Parece correcto, pero es frágil. Añade negativos duros.
Positivo en top 5 pero no top 1	Puede servir si la interfaz muestra varias opciones. No sirve para decisión automática.
Negativo visualmente muy cercano	Necesitas atributos, metadatos, OCR, filtros o reranking.
Error concentrado en un dominio	El problema no es “CLIP”; es distribución de datos y tarea.

En el laboratorio, al añadir `grant_form_blocked` y `grant_policy_pdf`, el sistema sigue pasando, pero el margen medio baja. Eso es bueno pedagógicamente: enseña que hacer el dataset más realista reduce la comodidad de la métrica y obliga a mirar errores.

En el día a día

CLIP y el contraste aparecen en cuanto alguien quiere «buscar por imagen» o «encontrar casos parecidos». Funciona sorprendentemente bien en la demo y luego falla en lo concreto: el buscador devuelve una silla que se parece pero no es la del catálogo, o confunde una captura de formulario con una de soporte porque ambas «se parecen» en el espacio de embeddings. El equipo descubre que un `Recall@1` del 100 % en cinco ejemplos no dice nada, y que el margen entre el positivo y el siguiente candidato importa más que el acierto.

El otro sitio donde reaparece es el coste: usar embeddings para recuperar candidatos baratos y reservar el modelo caro para los pocos mejores es uno de los patrones que más dinero ahorra. Pero exige medir de verdad la recuperación con `qrels` propios, no fiarse de que «el coseno es alto»: una cercanía alta significa que dos cosas están cerca en ese espacio, no que la respuesta sea correcta.

Y hay un tercer frente más sutil que descoloca a quien no lo conoce: el modality gap. Aunque entrenes imagen y texto para que queden cerca, en la práctica los embeddings de cada modalidad tienden a agruparse en zonas distintas del espacio, y eso sesga las comparaciones. Un equipo que lo ignora ajusta un umbral que funciona texto-texto y falla texto-imagen, o se

obsesiona con que una imagen «idéntica» no sale la primera. Saber que el gap existe evita perseguir durante días un bug que en realidad es geometría del espacio, no un fallo del código.

Por qué debería importarte

Saber que CLIP es ranking y no veredicto cambia cómo construyes y evalúas la búsqueda. Te ahorra el error más caro: tratar una similitud alta como una verdad. Imagina un buscador de incidencias que devuelve un caso «muy parecido» con coseno 0.9; si lo tratas como respuesta, cierras un ticket con la solución equivocada, y si lo tratas como candidato, lo verificas antes de actuar. Esa distinción, candidato frente a veredicto, es la que separa un buscador útil de uno peligroso. Quien lo entiende mide el margen, construye casos ambiguos a propósito, controla el modality gap y decide cuándo basta un embedding para recuperar y cuándo hace falta reranking, metadatos o revisión humana. Es la diferencia entre un sistema que impresiona en la demo y uno que aguanta cuando dos cosas se parecen pero solo una es la correcta.

Dónde volverá a aparecer

Este capítulo es la bisagra entre representación visual y sistemas multimodales completos.

CAPÍTULO FUTURO	QUÉ HEREDA
Capítulo 04	Los VLMs conectan representación visual con generación de lenguaje, pero CLIP enseña la alineación básica.
Capítulo 05	Document AI puede usar embeddings, pero necesita evidencia por campo y layout.
Capítulo 06	RAG multimodal usa retrieval sobre texto, imágenes, páginas, tablas o regiones.
Capítulo 10	Las métricas de retrieval reaparecen en evaluación multimodal.

Y conecta con facsímiles anteriores:

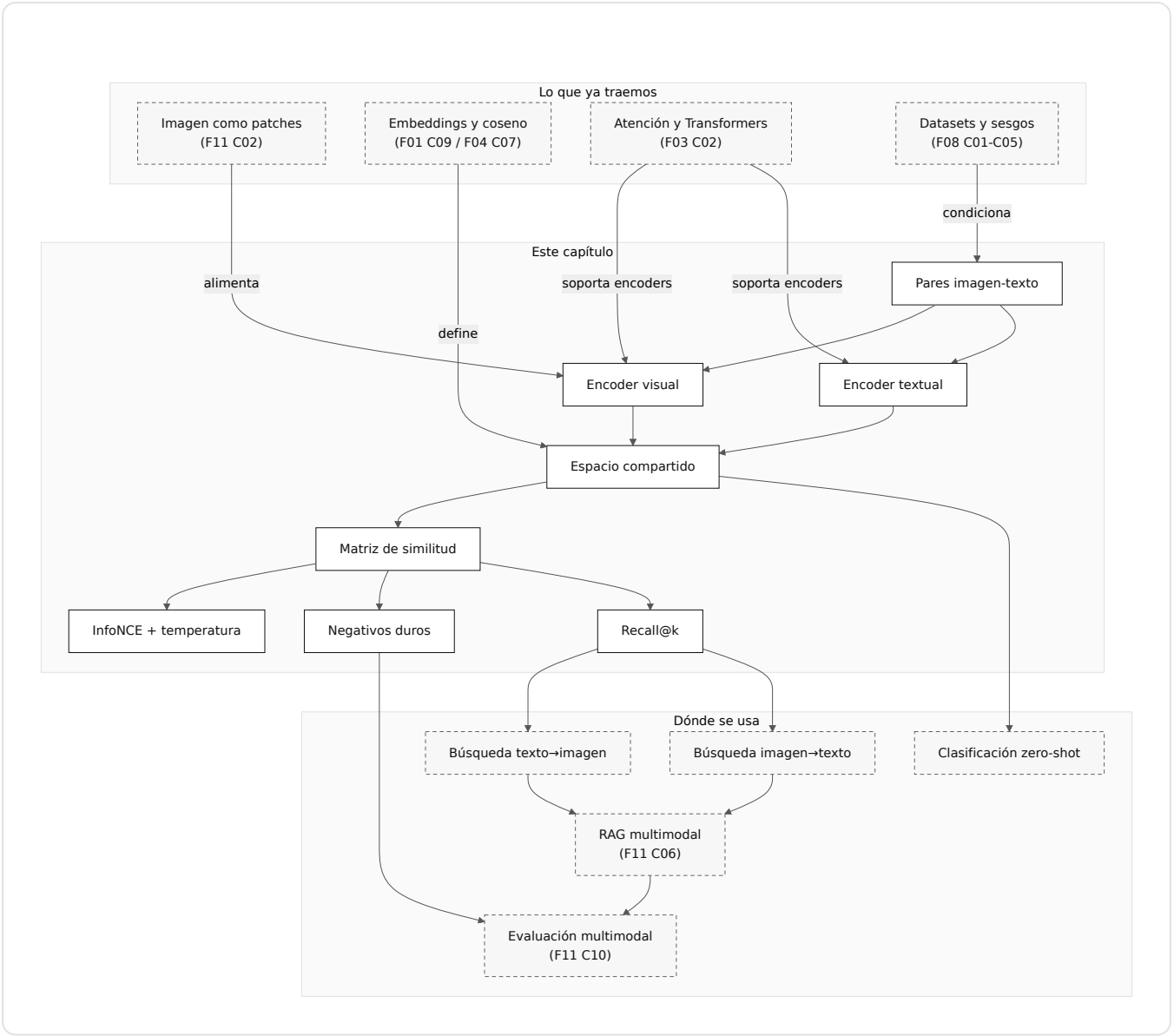
TEMA ANTERIOR	CONEXIÓN
Embeddings	Misma lógica vectorial, ahora cruzando modalidades.
RAG	Retrieval no siempre recupera texto; también puede recuperar imágenes o páginas.
Datasets	Los pares imagen-texto son datos de entrenamiento y evaluación con sesgos propios.
Evaluación	Un ranking debe medirse con casos reales, no con ejemplos bonitos.

Dónde solía tropezar yo

TROPIEZO	POR QUÉ ES UN PROBLEMA	ANTÍDOTO
Decir “CLIP entiende imágenes”	Es una frase demasiado grande para una herramienta de representación.	Habla de embeddings, similitud, ranking y evaluación.
No construir negativos duros	El sistema parece perfecto con casos fáciles.	Añade confusiones realistas al dataset.
Creer que zero-shot elimina el trabajo	Solo elimina entrenamiento específico, no evaluación.	Mide prompts, clases, idioma y slices.
Usar solo texto corto de clase	“Factura” puede ser peor que una descripción operativa.	Redacta clases como criterios, no como etiquetas pobres.
No combinar metadatos	El vector puede traer productos parecidos pero no disponibles o incorrectos.	Usa filtros, reglas de negocio y reranking.

Cómo encaja todo

Este mapa tiene tres zonas. A la izquierda están las representaciones que ya conocemos. En el centro está el aprendizaje contrastivo: pares, encoders, matriz y pérdida. A la derecha están los usos: búsqueda, clasificación zero-shot, RAG multimodal y evaluación.


Lo que ya traemos
Imagen como patches (F11 C02)
Embeddings y coseno (F01 C09 / F04 C07)
Atención y Transformers (F03 C02)
Datasets y sesgos (F08 C01-C05)
condiciona
Este capítulo
alimenta
soporta encoders
soporta encoders
Pares imagen-texto
define
Encoder visual
Encoder textual
Espacio compartido
Matriz de similitud
InfoNCE + temperatura
Negativos duros
Recall@k
Dónde se usa
Búsqueda texto→imagen
Búsqueda imagen→texto
Clasificación zero-shot
RAG multimodal (F11 C06)
Evaluación multimodal (F11 C10)

Vocabulario aprendido

TÉRMINO	DEFINICIÓN
Par positivo	Imagen y texto que deberían quedar cerca en el espacio compartido.
Negativo	Candidato incorrecto que debería quedar lejos del positivo.
Negativo duro	Negativo muy parecido que revela una confusión importante.
Espacio compartido	Espacio vectorial donde se comparan embeddings de imagen y texto.
Similitud coseno	Métrica que compara dirección entre vectores normalizados.
Temperatura	Parámetro que controla la suavidad del softmax.
InfoNCE	Pérdida contrastiva que favorece positivos frente a negativos.
Recall@k	Métrica de recuperación: positivo dentro de los k primeros.
Zero-shot	Uso sin entrenamiento específico para esa tarea, normalmente con prompts de clase.

Antes de pasar página

Antes de avanzar, comprueba que puedes responder estas preguntas:

- [ ] ¿Puedo explicar qué representa una celda S_{ij} en la matriz imagen-texto?
- [ ] ¿Puedo decir por qué la diagonal debería ser alta?
- [ ] ¿Puedo explicar qué hace la temperatura en el softmax?
- [ ] ¿Puedo leer la pérdida InfoNCE sin repetir la fórmula de memoria?
- [ ] ¿Puedo diseñar tres negativos duros para un catálogo, soporte o documentos?
- [ ] ¿Puedo medir `Recall@k` y explicar qué significa para una interfaz real?
- [ ] ¿Puedo ejecutar el cuaderno del facsímil y justificar una mejora del ranking?

En resumen

IDEA	QUÉ DEBERÍAS LLEVARTE
CLIP alinea modalidades.	Entrena embeddings de imagen y texto para que puedan compararse.
La matriz manda.	La diagonal son positivos; fuera de diagonal aparecen errores y negativos.
Temperatura no es decoración.	Controla cómo de concentrada es la distribución al entrenar.
`Recall@k` es obligatorio.	Un retrieval multimodal se evalúa por ranking, no por impresiones.
Zero-shot no elimina evaluación.	Solo evita entrenar un clasificador específico; los fallos siguen existiendo.
Los negativos duros son valiosos.	Son los casos que hacen que el sistema sea defendible.

Para saber más

Baltrušaitis, T., Ahuja, C. y Morency, L.-P. (2019). Multimodal Machine Learning: A Survey and Taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2), 423-443. https://doi.org/10.1109/TPAMI.2018.2798607

Dosovitskiy, A. y otros (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *International Conference on Learning Representations*. https://arxiv.org/abs/2010.11929

LeCun, Y., Bengio, Y. y Hinton, G. (2015). Deep learning. *Nature*, 521, 436-444. https://doi.org/10.1038/nature14539

Oord, A. van den, Li, Y. y Vinyals, O. (2018). Representation Learning with Contrastive Predictive Coding. https://arxiv.org/abs/1807.03748

Radford, A. y otros (2021). Learning Transferable Visual Models From Natural Language Supervision. *Proceedings of the 38th International Conference on Machine Learning*, 8748-8763. https://arxiv.org/abs/2103.00020

Schuhmann, C. y otros (2022). LAION-5B: An Open Large-Scale Dataset for Training Next Generation Image-Text Models. *Advances in Neural Information Processing Systems 35*. https://arxiv.org/abs/2210.08402

Vaswani, A. y otros (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems 30*. https://arxiv.org/abs/1706.03762

Notas

1. Radford, A. y otros (2021). Learning Transferable Visual Models From Natural Language Supervision. *Proceedings of the 38th International Conference on Machine Learning*, 8748-8763. https://arxiv.org/abs/2103.00020. CLIP popularizó el entrenamiento contrastivo imagen-texto a gran escala y mostró un uso muy práctico de clasificación zero-shot. ↩

2. Baltrušaitis, T., Ahuja, C. y Morency, L.-P. (2019). Multimodal Machine Learning: A Survey and Taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2), 423-443. https://doi.org/10.1109/TPAMI.2018.2798607. ↩

3. Oord, A. van den, Li, Y. y Vinyals, O. (2018). Representation Learning with Contrastive Predictive Coding. https://arxiv.org/abs/1807.03748. El trabajo formuló una pérdida contrastiva influyente para aprender representaciones separando positivos y negativos. ↩

4. Zhai, X. y otros (2023). *Sigmoid Loss for Language Image Pre-Training*. Proceedings of ICCV 2023. https://arxiv.org/abs/2303.15343. ↩

5. Liang, V. W., Zhang, Y., Kwon, Y., Yeung, S. y Zou, J. (2022). Mind the Gap: Understanding the Modality Gap in Multi-modal Contrastive Representation Learning. *Advances in Neural Information Processing Systems 35*. https://arxiv.org/abs/2203.02053. Muestra que en el espacio compartido las dos modalidades quedan en regiones separadas pese a la alineación contrastiva. ↩

6. Radford, A. y otros (2021). Learning Transferable Visual Models From Natural Language Supervision. *Proceedings of the 38th International Conference on Machine Learning*, 8748-8763. https://arxiv.org/abs/2103.00020. ↩

7. Schuhmann, C. y otros (2022). LAION-5B: An Open Large-Scale Dataset for Training Next Generation Image-Text Models. *Advances in Neural Information Processing Systems 35*. https://arxiv.org/abs/2210.08402. ↩