Deep Learning para Business Analytics

Día 09 · Tema 3 — El Transformer entero

Atención multi-head, bloque encoder y por qué dominó la IA moderna.

Eduardo C. Garrido-Merchán

Profesor Propio Adjunto, Métodos Cuantitativos, ICADE

ecgarrido@comillas.edu

Madrid, Día 09

ejemplos del día

Lo que vamos a cubrir hoy.

1. Self-attention con números

2. Multi-head attention

3. Un bloque Transformer completo

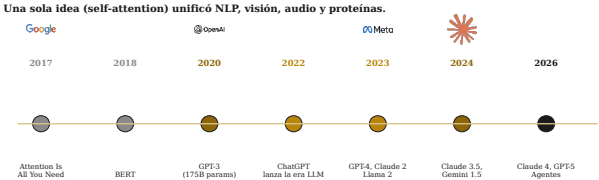
4. Por qué dominó la IA moderna

Al final del día: sabes pasar self-attention paso a paso, justificar multi-head, y dibujar un bloque Transformer de memoria.

Parte 1 / 4

Self-attention con números

⚡🌍 ChatGPT, Claude, Gemini: todos son Transformers.



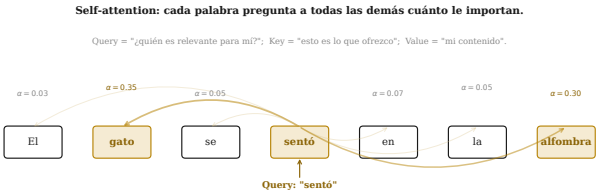
Una sola idea. Self-attention, propuesta en 2017, unificó NLP, visión, audio y proteínas. Hoy la usan miles de millones de personas sin saberlo.

La línea temporal. 2017: Attention Is All You Need. 2018: BERT. 2020: GPT-3. 2022: ChatGPT lanza la era LLM. 2024–26: Claude 4, GPT-5, agentes autónomos.

El impacto. Cada modelo de la lista (GPT-4, Claude, Gemini, Llama) es un Transformer. La arquitectura no ha cambiado en lo fundamental desde 2017; lo que escala es el tamaño y los datos.

Idea clave. Esta arquitectura cambió el mundo. Hoy la aprendes de principio a fin.

🔍 💬 Self-attention = cada palabra pregunta a todas las demás cuánto le importan.



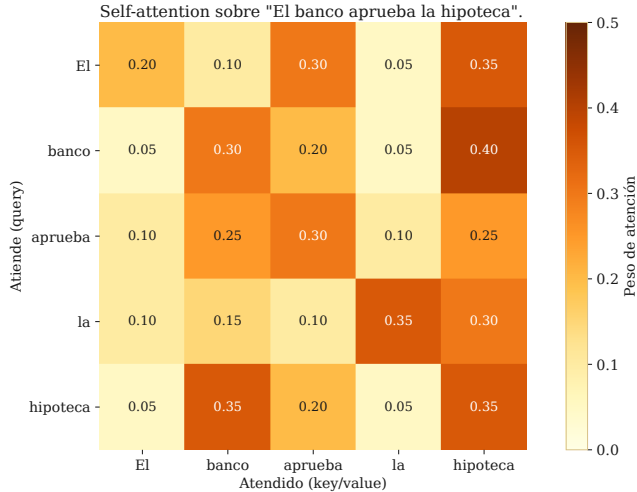
La frase. "El gato se sentó en la alfombra." Miramos desde "sentó" (la query).

Pesos. "sentó" atiende con fuerza a "gato" ($\alpha = 0{,}35$) y a "alfombra" ($\alpha = 0{,}30$): el sujeto y el lugar. Los artículos reciben peso casi nulo.

Q, K, V. Query = "¿quién es relevante para mí?". Key = "esto es lo que ofrezco". Value = "este es mi contenido". El producto Q·K dice cuánto; V dice qué.

Idea clave. Self-attention deja que cada posición mire directamente a todas las demás en una sola operación.

🧠 Self-attention sobre una frase real.



🔍 Self-attention, paso a paso.

Self-attention en cinco pasos sobre 4 tokens × 4 dim.

Setup	$X = \{x_1, x_2, x_3, x_4\} \in \mathbb{R}^{4\times 4}$ embeddings de 4 tokens.
1. Proyecciones	$Q = XW^Q, \quad K = XW^K, \quad V = XW^V, \quad W^* \in \mathbb{R}^{4\times 4}$
2. Scores	$S = QK^\top/\sqrt{d} \in \mathbb{R}^{4\times 4}$
3. Softmax	$\alpha_{ij} = e^{s_{ij}}/\sum_k e^{s_{ik}}$
4. Salida	$z_i = \sum_j \alpha_{ij} \cdot v_j \in \mathbb{R}^4$
5. Multi-head	Repetir con 4 conjuntos W^Q, W^K, W^V, concatenar, multiplicar por W^O.

🧠 Por qué la atención no necesita memoria explícita.

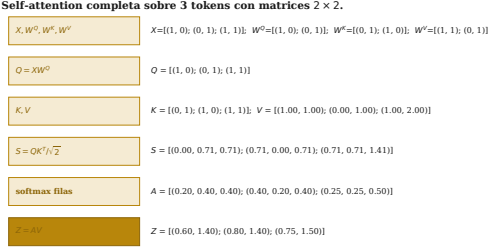
En la RNN. Para que el token 5 sepa del token 1 tienen que pasar 4 multiplicaciones recurrentes. La información se diluye.

En self-attention 🔍. Todos los tokens tienen acceso directo a todos los demás en una sola operación matricial. La distancia "temporal"no importa.

Coste. La atención es $O(T^2)$ en memoria y tiempo. Para $T = 8\,000$ tokens ya empieza a doler, y motivó las alternativas 🚀 que veremos mañana.

Idea clave. Self-attention compra calidad pagando memoria cuadrática.

🔍 Self-attention completa sobre 3 tokens con matrices 2×2.



Setup. X: 3 tokens, dim 2. $W^Q = I$, W^K permuta columnas, $W^V = \left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)$.

Paso 1. $Q = XW^Q$, $K = XW^K$, $V = XW^V$.

Paso 2. Scores: $S = QK^\top / \sqrt{2}$.

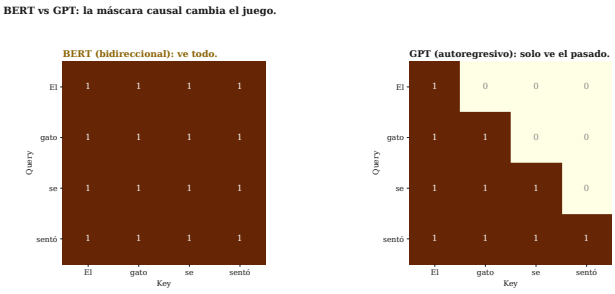
Paso 3. Softmax por filas $\longrightarrow$ matriz A de pesos.

Paso 4. Salida $Z = AV$. Cada fila de Z es la combinación ponderada de los values según los pesos de atención.

Comprobación. Las filas de A suman 1. La dimensión de Z coincide con la de V: (3×2).

Idea clave. Lleva la misma lógica Q/K/V del Día 07, pero ahora Q, K y V salen del propio input: self-attention.

🎭🤖 BERT vs GPT: la máscara causal cambia el juego.



BERT (bidireccional). Cada token ve a todos los demás. La matriz de atención es completa. Ideal para clasificación, NER y comprensión.

GPT (autoregresivo). Cada token solo ve los anteriores. La matriz de atención es triangular inferior (máscara causal). Ideal para generación de texto.

Encoder-decoder. T5, BART: el encoder es bidireccional y el decoder autoregresivo. Cross-attention conecta ambos.

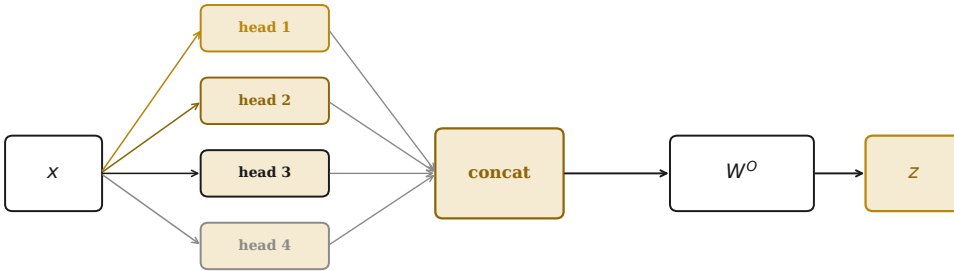
Idea clave. La misma operación (self-attention) produce modelos radicalmente distintos solo cambiando la máscara.

Parte 2 / 4

Multi-head attention

📚 Multi-head: cuatro miradas en paralelo + proyección final.

Multi-head attention: cuatro "miradas" en paralelo + proyección final.



📚 Por qué multi-head.

Idea. Cada cabeza 🧠 ve la frase desde una perspectiva distinta: una atiende a sintaxis, otra a co-referencia, otra a entidades, otra a tiempo.

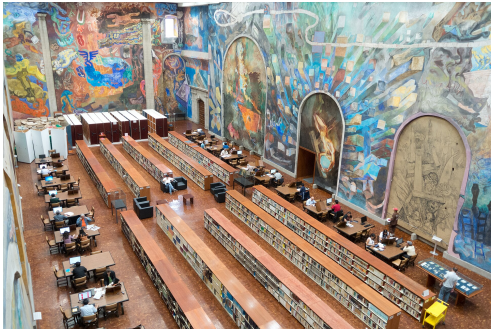
Matemática. Si la dimensión del modelo es $d_{model} = 512$ y tienes $h = 8$ cabezas, cada cabeza tiene $d_k = d_{model}/h = 64$. El coste total no cambia.

Cómo se inspecciona 🔍. En los notebooks veremos heatmaps por cabeza para entender qué aprende cada una. Una cabeza concreta atiende, por ejemplo, a la concordancia sujeto-verbo en español.

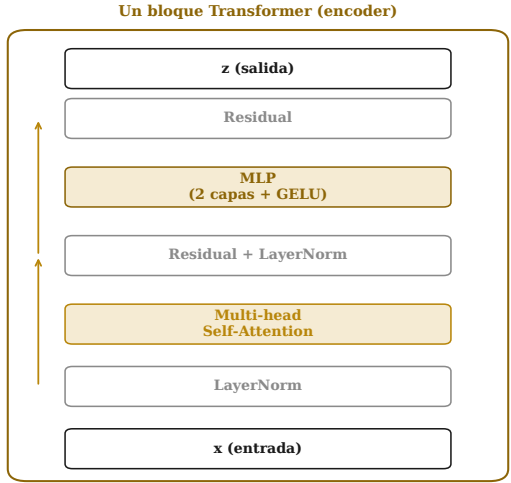
Idea clave. Multi-head es paralelización 🚀 barata de capacidad expresiva.

Parte 3 / 4

Un bloque Transformer completo



⚡ Un bloque encoder: atención + residual + ⚙️ MLP + residual.



⚡ Concepto → aplicación: tres formas del Transformer.

Forma	Modelo emblemático	Tarea típica	Familia
Solo encoder	BERT, RoBERTa, DeBERTa.	Clasificación, NER.	Bidireccional.
Solo decoder	GPT-3/4, Llama, Mistral.	Generación de texto.	Autoregresivo.
Encoder-decoder	T5, BART, MarianMT.	Traducción, resumen.	Seq2seq.
Visión	ViT, Swin, ConvNeXt.	Clasificación imágenes.	Patch embeddings.
Multimodal	CLIP, Flamingo, GPT-4V.	Imagen + texto.	Mixto.

Idea clave. La misma arquitectura cambia de tarea con sólo el régimen de entrenamiento.

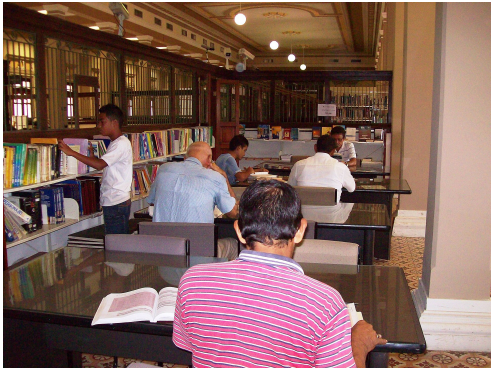
🚀 Receta exprés: usar un Transformer ⚡ pre-entrenado para clasificar.

```
from transformers import AutoTokenizer, AutoModelForSequenceClassification

tok = AutoTokenizer.from_pretrained("PlanTL-GOB-ES/roberta-base-bne")
model = AutoModelForSequenceClassification.from_pretrained(
    "PlanTL-GOB-ES/roberta-base-bne", num_labels=3)

# fine-tunear con Trainer (igual que Día 05) sobre tu CSV
# (queja_cliente, etiqueta) ; 3 clases: factura, técnico, comercial
```

Idea clave. En español, PlanTL-GOB-ES publica modelos open-source con licencia favorable; úsalos siempre que sea posible.

Parte 4 / 4

Por qué dominó la IA moderna



⚡ El Transformer ha desplazado al SOTA en seis dominios 🚀.

El Transformer ha sustituido a su predecesor en seis dominios distintos.

Ámbito	Antes	Modelo actual	Año salto
Traducción	RNN encoder-decoder	Transformer (Vaswani et al.)	2017
Lenguaje genérico	LSTM bidireccional	BERT, GPT-3 y descendientes	2018–2020
Visión	ResNet, EfficientNet	ViT y SAM	2020
Audio	RNN espectrogramas	Whisper, AudioLM	2022
Series temporales	ARIMA, LSTM	PatchTST, TimeGPT	2023
Robótica	Control clásico	RT-2 (vision-language-action)	2023

Lo que el alumno se lleva a casa de este día.

Concepto 1. 🧠 Self-attention deja que cada token mire a todos los demás directamente, sin pasar por una cadena recurrente.

Concepto 2. 📚 Multi-head es paralelizar varias atenciones con dimensión reducida; el coste total se mantiene.

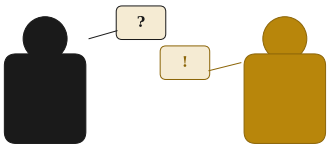
Concepto 3. ⚙️ Un bloque Transformer es atención + residual + MLP + residual, con LayerNorms entre medias.

Concepto 4. ⚡ La misma arquitectura sirve para texto, visión, audio, series y robótica con sólo cambiar tokenización y entrenamiento.

Para la práctica. Fine-tunear un RoBERTa en español sobre quejas de clientes; producir matriz de confusión y AUC por clase.

Próximo día. Día 10 (Tema 3.3): alternativas y sucesores del Transformer 🚀.

Pausa para debatir: Si el Transformer sirve para todo, ¿qué papel queda a CNN y RNN?



La pregunta. ViT supera a ResNet, los Transformer hacen forecast mejor que LSTM. ¿Es razonable enseñar CNN y RNN en 2026?

Posición A. Sí, son base conceptual y siguen siendo eficientes en edge.

Posición B. No, son legado; mejor dedicar tiempo a SSM y MoE.

Reglas. 5 minutos, dos turnos de palabra por bando, móviles boca abajo. Yo modero, no participo.

Hasta el Día 10.

Mamba, RWKV, MoE: cuando el Transformer se queda corto.

Eduardo C. Garrido-Merchán

ecgarrido@comillas.edu