

# Tema 7.2

# Large Language Models

Deep Learning

Máster Oficial en Ingeniería Informática

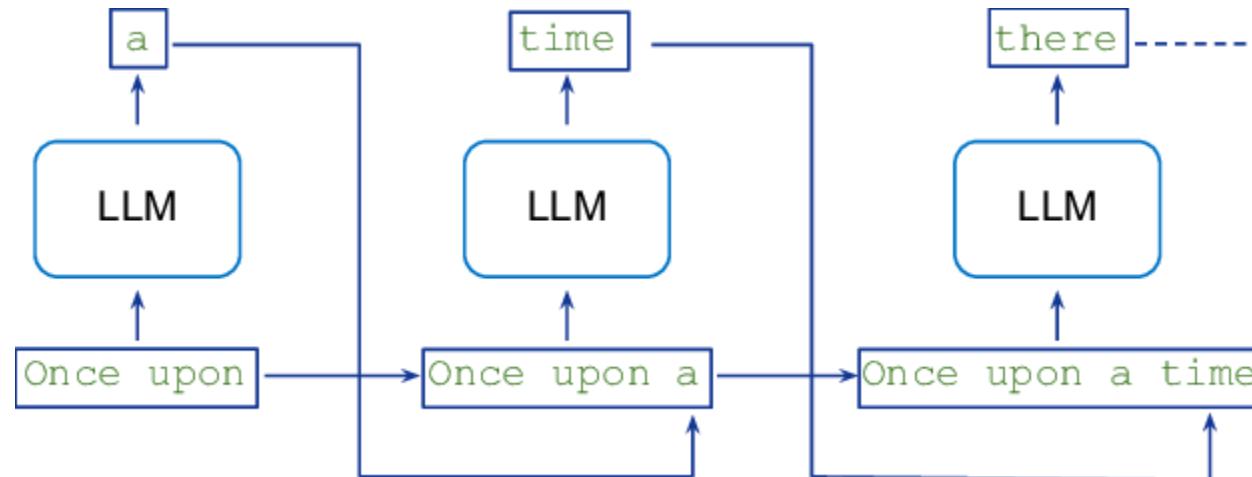
Universidad de Sevilla

# Contenido

- Tipos de LLMs
- Tokenizadores
- Transformers Actuales
- Proceso de entrenamiento
- Modelos de razonamiento

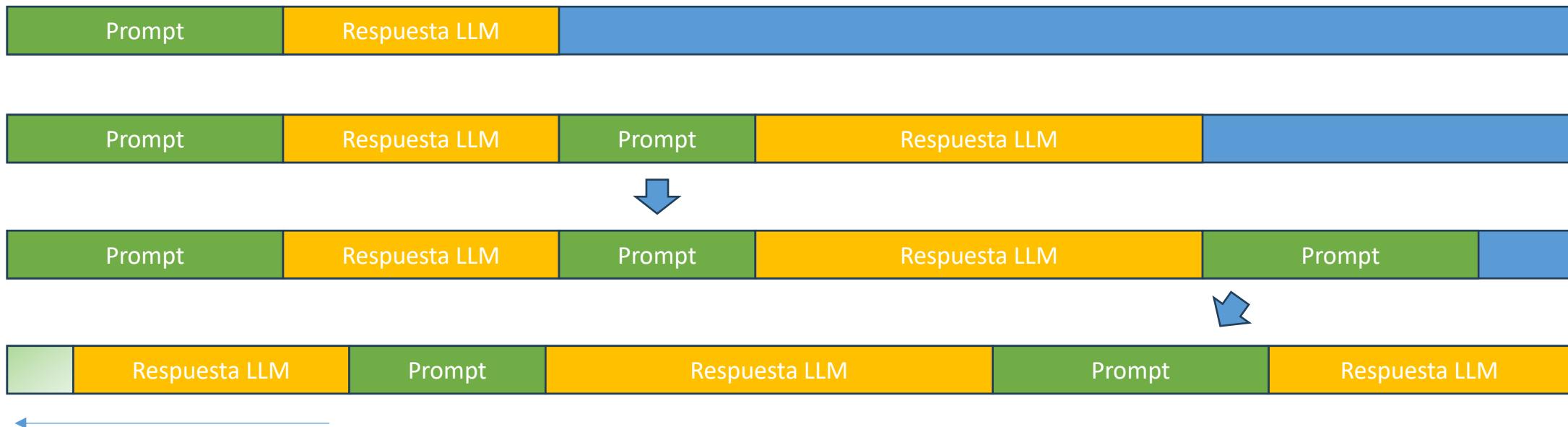
# Large Language Models

- LLMs son transformers de gran tamaño (billones-trillones de parámetros), trabajando de forma **autoregresiva**:



# Large Language Models

- **Ventana de Contexto:** la cantidad de tokens que admite el LLM:
  - Tu prompt de entrada + la respuesta generada por el LLM



# Tipos de LLMs

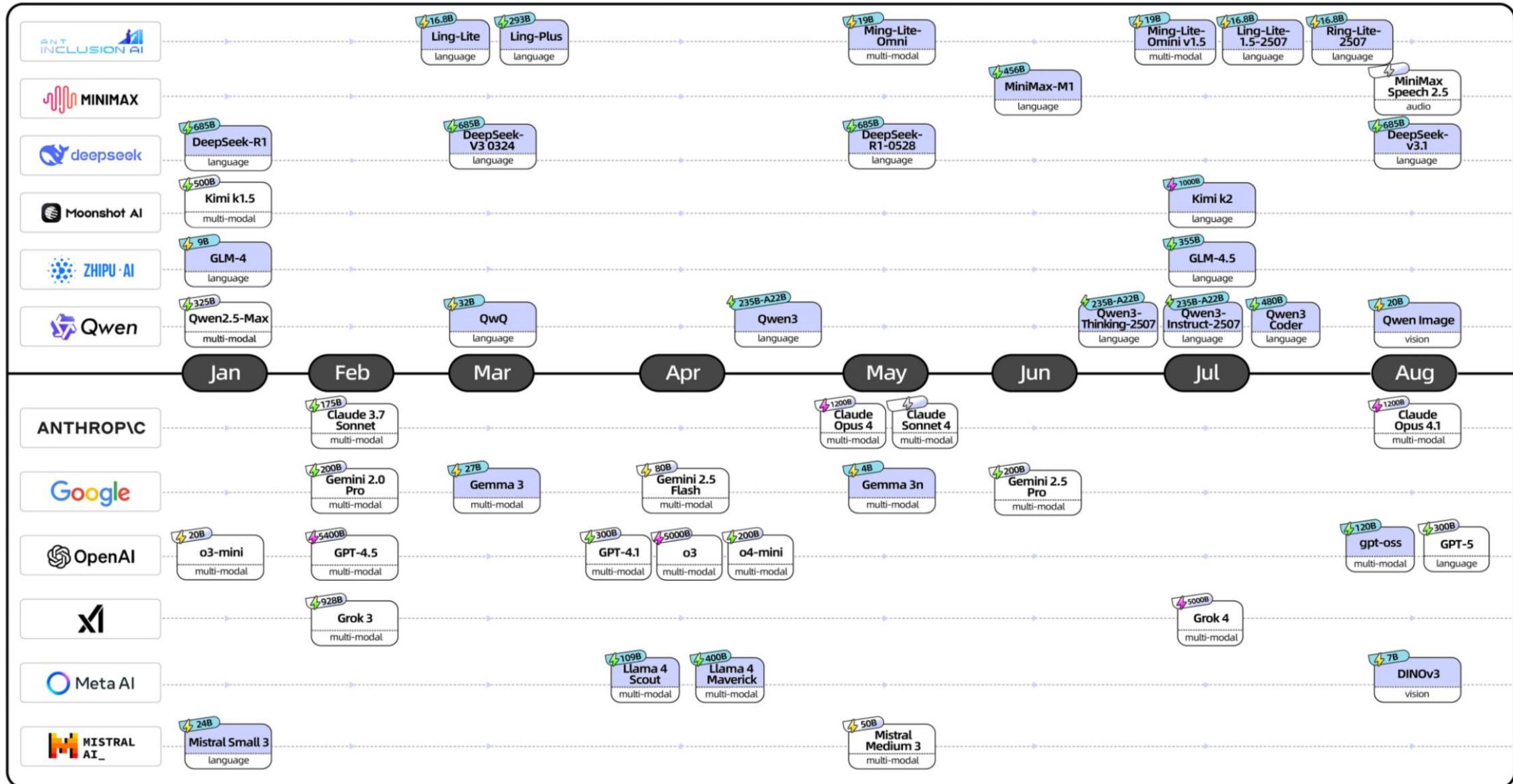
- **Privados:**

- Arquitectura y pesos **no públicos**. No posible hacer fine-tuning.
- El modelo se ejecuta en remoto a través de una **API**. Problemas **privacidad**:
  - Datos se usan para re-entrenar modelos!
- Costoso para proveedores. Modelos potentes de **pago**.

- **Abiertos:**

- Arquitectura y pesos son de dominio **público** (pero a veces licencias no comerciales!).
- Posibilidad de hacer **fine-tuning**.
- Requiere **recursos** propios para ejecutarlos. A veces también tienen API.

# Large Models Landscape 2025

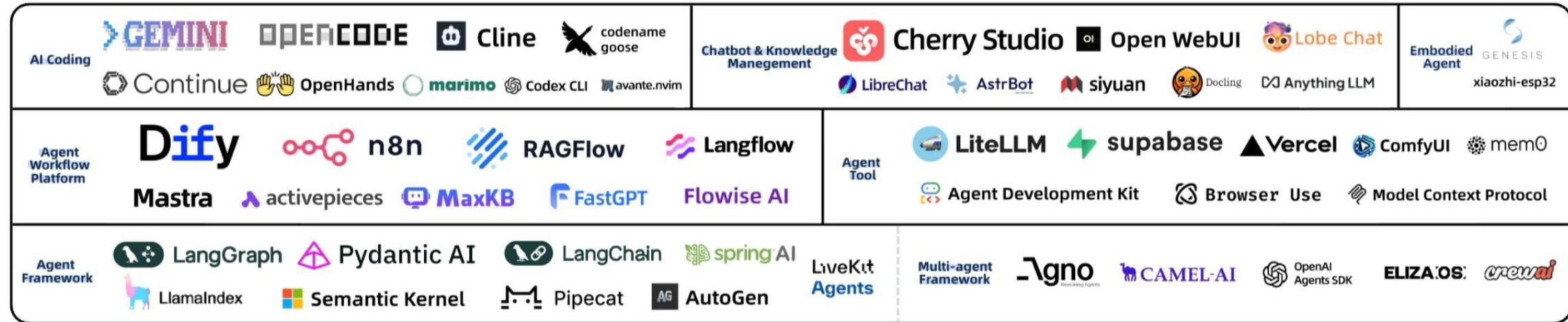


# Tipos de LLMs

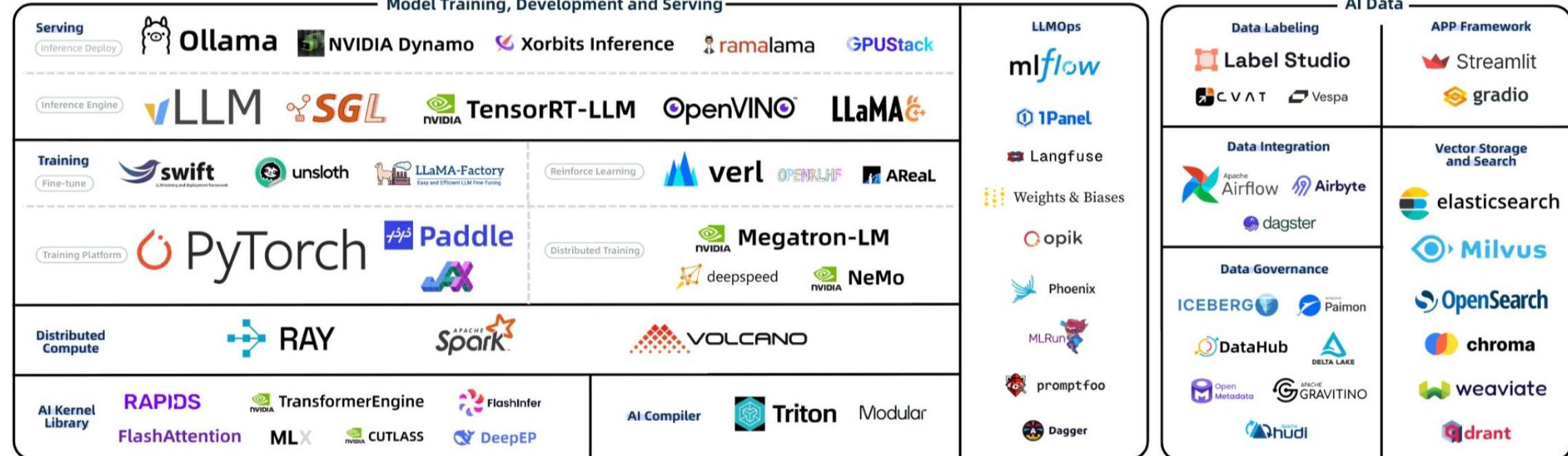
- Modelos **Privados** (2025):
  - ChatGPT 5.1 (OpenAI), Gemini 3 (Google), Grok 4 (xAI), Claude (Antropic)
- Modelos **Abiertos** (2025):
  - Grandes: Llama 4, DeepSeek 3.1, Qwen 3, Mixtral 3,
  - Pequeños: GPT OSS (OpenAI), Gemma 2 (Google), Grok 2.5 (xAI)
- **Plataformas** para ejecutar LLMs abiertos:
  - Colección de mayoría de modelos: Hugging Face Tranformers
  - App Escritorio: llm Studio, LangChain, llama.cpp...

# Open Source LLM Development Landscape

AI Agent

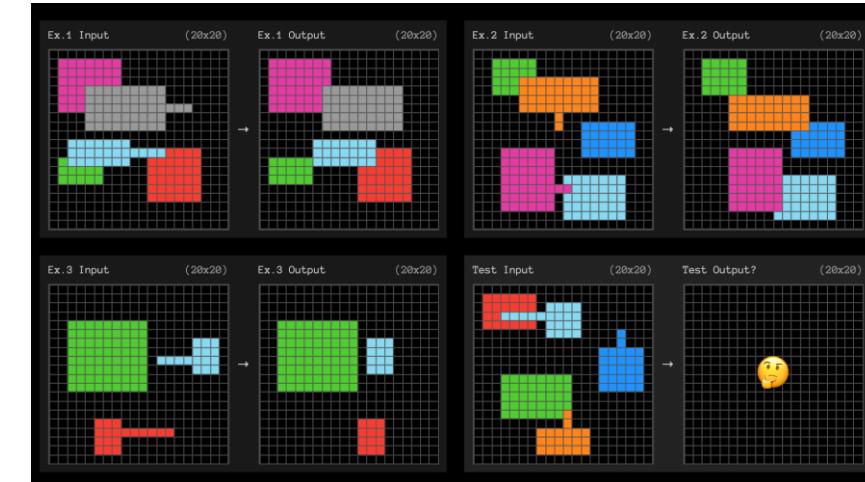


AI Infra



# Evaluación de LLMs

- [LMarena](#): usuarios valoran a ciegas LLMs de dos en dos en diferentes tareas (Texto, WebDev, Text-to-Image...)
- [ARC AGI](#): puzzles diseñados para evaluar capacidad razonamiento
- [Humanity's last exam](#): preguntas de nivel de doctorado.
- [HumanEval](#): problemas de programación para evaluar generación de Código.



# Transformers actuales

## Tokenizador

- Métodos:
  - Por **palabras**: WordPiece (BERT). Restringidos en vocabulario.
  - **Sub-palabras**: Más flexible que por palabras.
    - Agotador vs agotado (palabras), agot- -ador vs -ado vs -ante, ... (sub-palabras)
  - **Caracteres/Bytes**: P.ej. Byte Pair Encoding (BPE) - usado por modelos GPT
- Elegir tamaño vocabulario y tokens especiales.
- Se entrena sobre un dataset para obtener el mejor vocabulario
- Usados tanto en entrada como en la salida de modelos

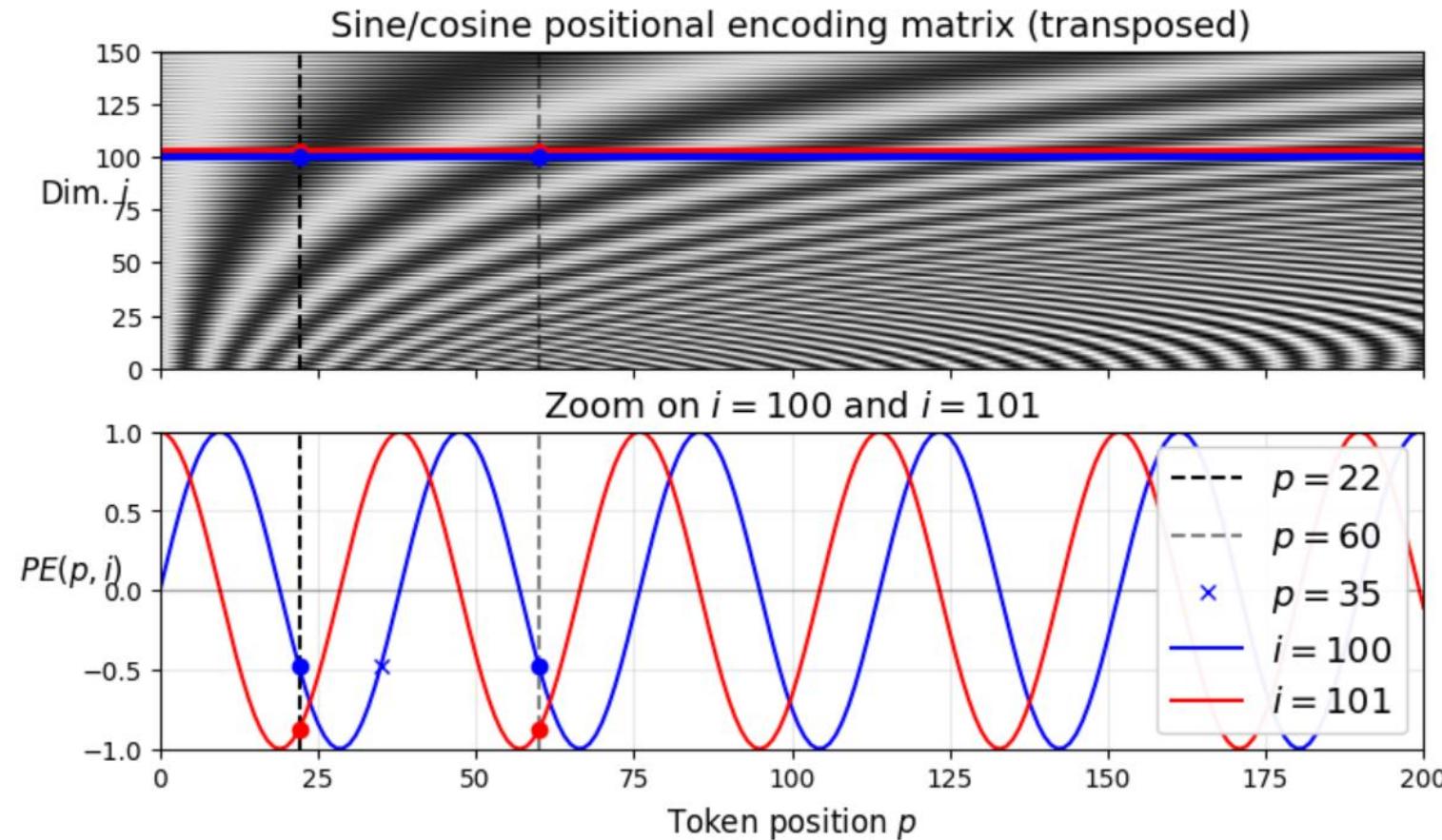
# Transformers actuales

## Codificación posicional

- En “Attention is all you need”, proponen codificación posicional fija en vez de entrenable: **codificación sinusoidal**
  - Menos parámetros por aprender, y ayuda al modelo aprender patrones con posición relativa.
  - Codificar la posición con 1,2,3,...N daría valores mucho más altos que en los embeddings
  - $PE [p, i] = \text{func} \left( \frac{p}{10000^{\frac{i}{embDim}}} \right)$ , donde  $\begin{cases} \text{func} = \text{seno, si } i \text{ es par} \\ \text{func} = \text{coseno, si } i \text{ es impar} \end{cases}$ 
    - $i$  se refiere a la componente  $i$ -ésima del embedding
    - $p$  es la posición del token en la secuencia
    - $embDim$  es el tamaño del embedding (mismo para posicional como para la palabra)

# Transformers actuales

## Codificación posicional



# Transformers actuales

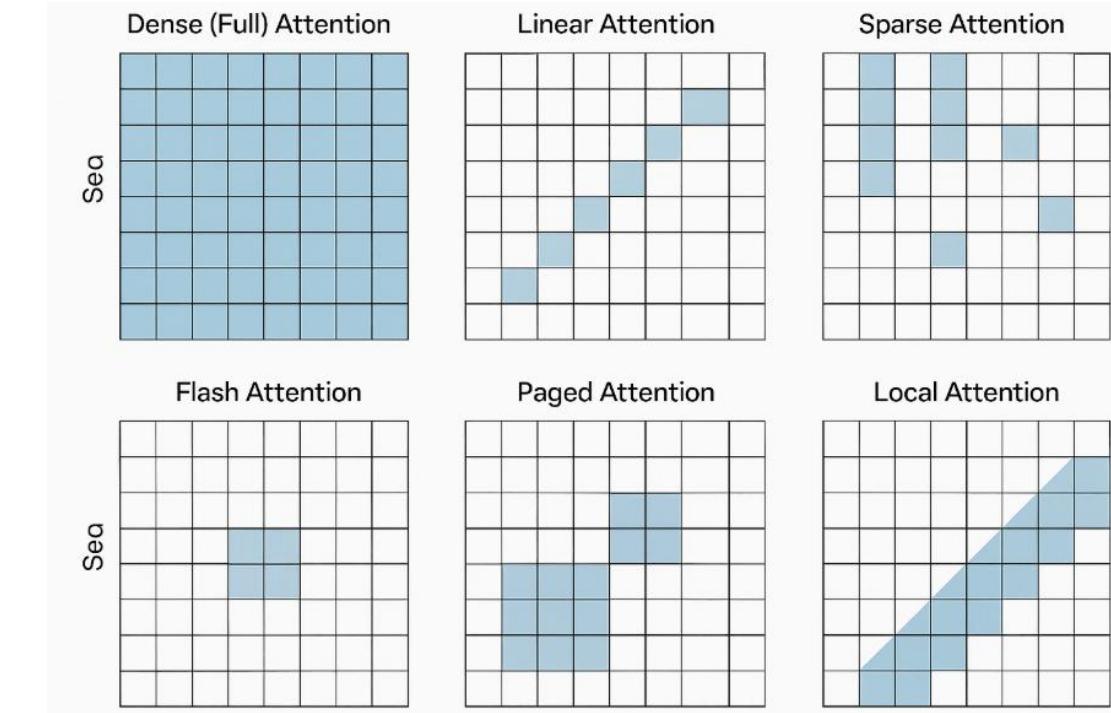
## Codificación posicional

- Otros codificadores posicionales modernos:
  - **RoPE** (Rotary Positional Embedding): codifica posición rotando los vectores Q y K, permitiendo que la atención dependa de la posición relativa entre los tokens.
  - **ALIBI** (Attention with Linear Biases): no usa codificaciones posicionales explícitas, sino que aplica un sesgo lineal a las puntuaciones de atención proporcional a la distancia.
- Arquitecturas **solo decoder**: actualmente, la mayoría de LLMs no utilizan el encoder, sino solo el decoder.
- **Caché Key-Value (KV cache)**: optimización que almacena los K y V calculados para tokens ya procesados, evitando recalcularlos.

# Transformers actuales

## Variantes de atención

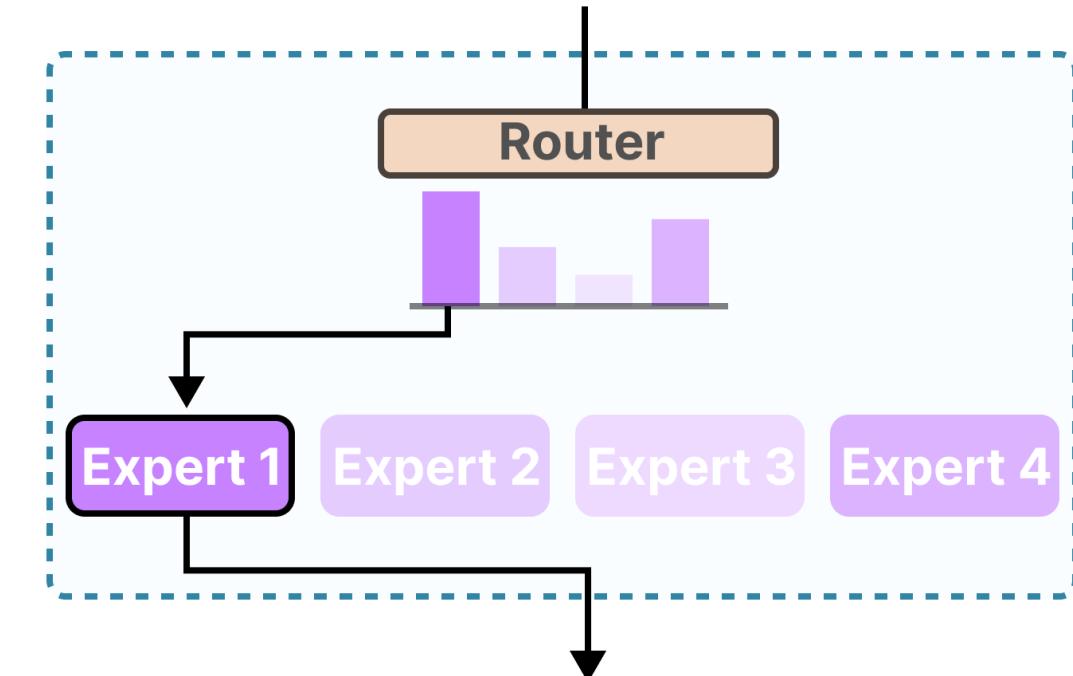
- La operación de atención no escala bien ante contextos muy grandes  $O(n^2)$ , ¿cómo aliviarlo?
  - **Sparse attention:** optimización algorítmica que reduce la complejidad de  $O(n^2)$  a  $O(n\sqrt{n})$  o incluso lineal  $O(n)$ , al calcular la atención solo para un subconjunto selecto de tokens (patrones fijos, ventanas o aleatorios), haciéndola más eficiente para secuencias muy largas.
  - **Flash attention:** optimización de implementación para GPU (en CUDA) que calcula la atención completa de  $O(n^2)$ , pero lo hace en bloques dentro de la memoria compartida (Shared Memory).



# Transformers actuales

## Mixture of Experts (MOE)

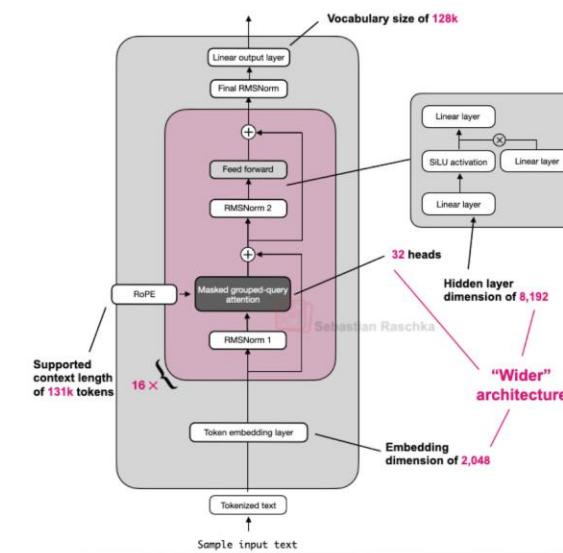
- Para cada *token* de entrada, un **Router** o *Gating Network* decide dinámicamente **cuál o cuáles de estos expertos** deben procesarlo.
- Permite entrenar modelos mucho más grandes, pero solo activando y computando un subconjunto muy pequeño por *token*, resultando en una **inferencia más rápida** a pesar del tamaño.



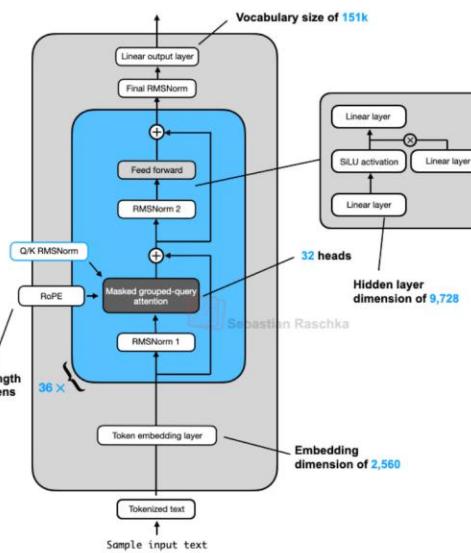


# Transformers actuales

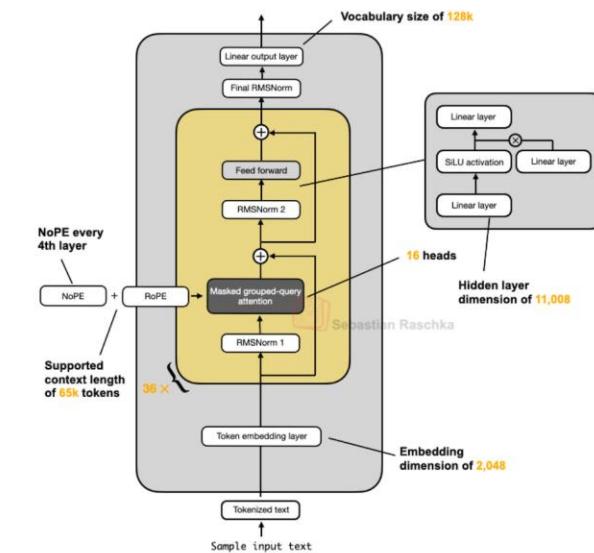
Llama 3.2 1B



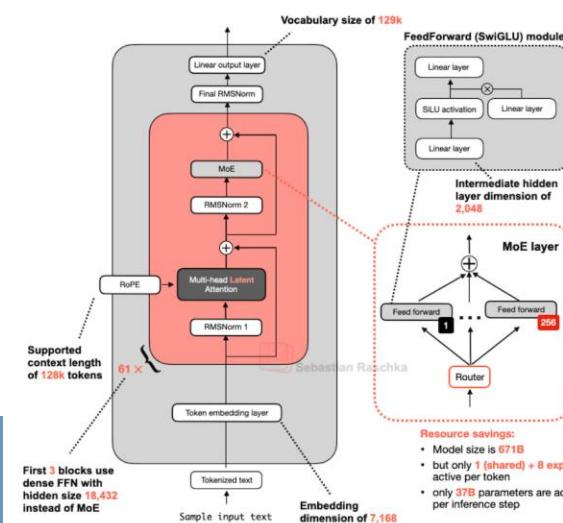
Qwen3 4B



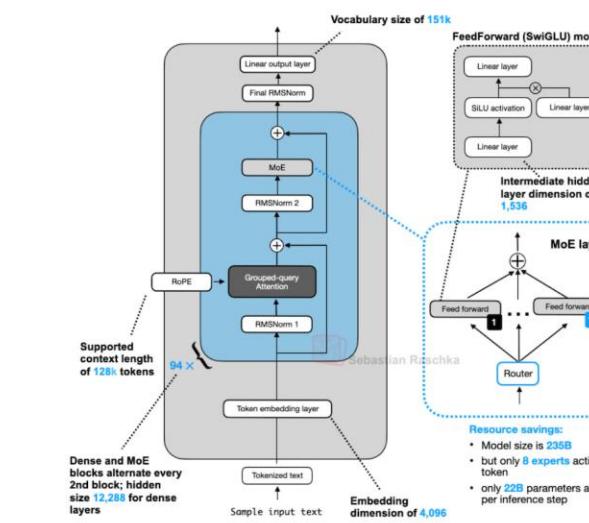
SmoLM3 3B



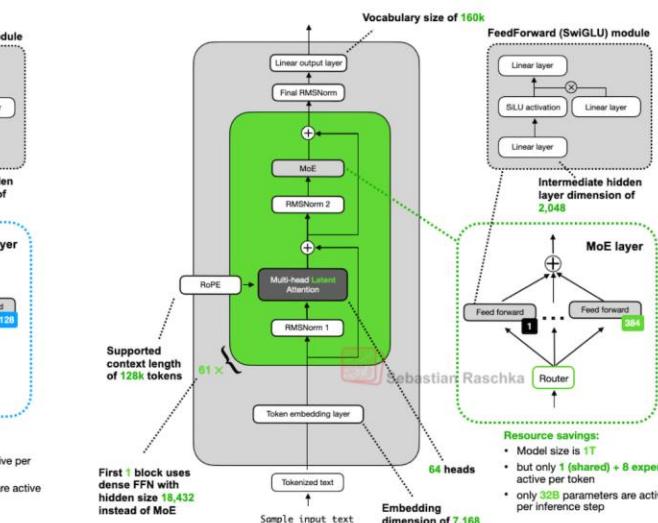
DeepSeek V3 (671B)



Qwen3 235B-A22B

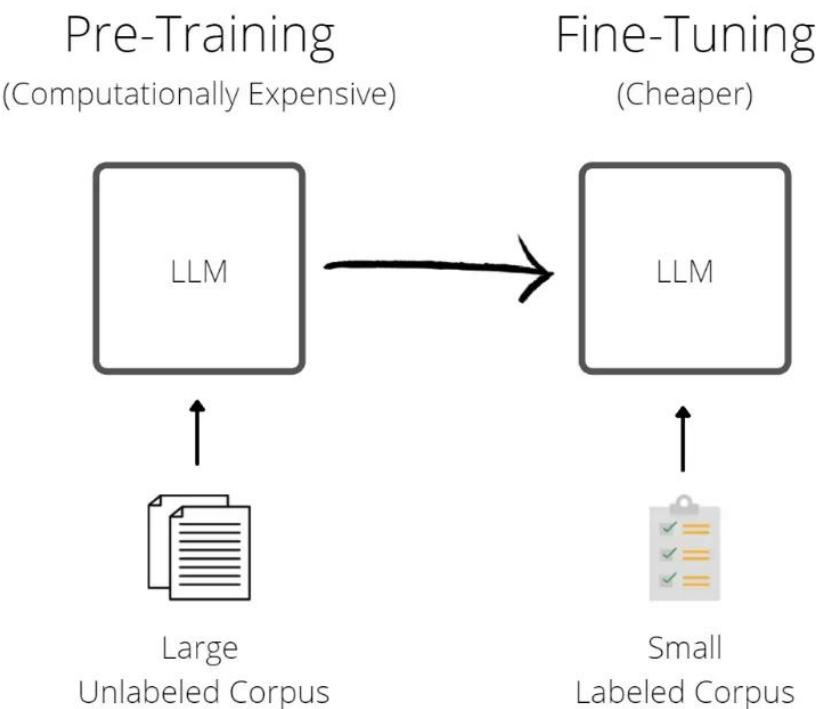


Kimi K2 (1 trillion)



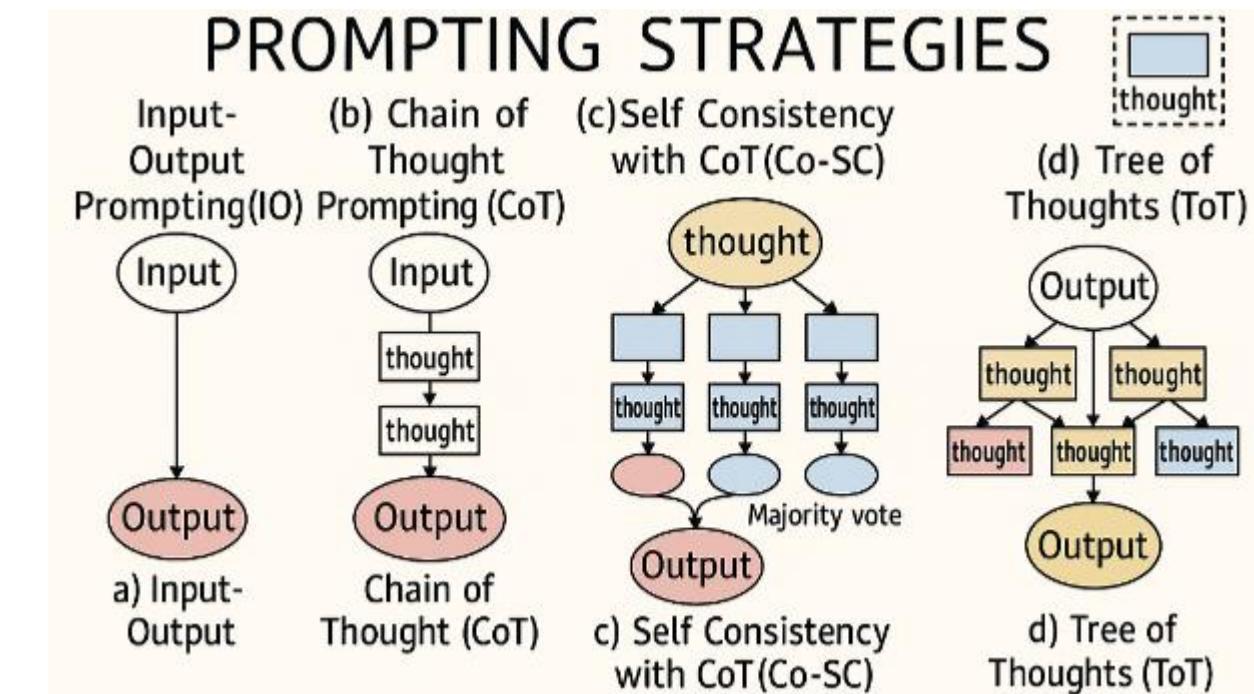
# Proceso de entrenamiento

- **Pre-entrenamiento:** fase donde se usa más datos con menos calidad
  - Semi-supervisado: eliminar palabra y predecirla, continuar frase, ...
- **Fine tuning:** adaptación a aplicaciones concretas
  - **Modelos instruct:** se han adaptado para seguir instrucciones y **conversar**.
  - **Reinforcement Learning with Human Feedback (RLHF):** Usa *feedback* humano para entrenar un Modelo de Recompensa, que luego se usa para ajustar el LLM (a través de aprendizaje por refuerzo) para que sus respuestas sean más útiles, honestas y seguras.



# Razonamiento

- **Chain-of-Thoughts (CoT)**: técnica que permite a un LLM generar pasos de razonamiento intermedios antes de llegar a una respuesta final
- Inspirado en cómo trabajamos los humanos: a tareas más complicadas tardamos más tiempo en contestar



# Recapitulación

- Qué es un LLM
- Qué LLMs hay actualmente
- Arquitecturas actuales de LLMs
- El proceso de entrenamiento
- Los modelos de razonamiento