ntic master
revolucionamos la comunicación

UNIVERSIDAD COMPLUTENSE MADRID

# LLMs

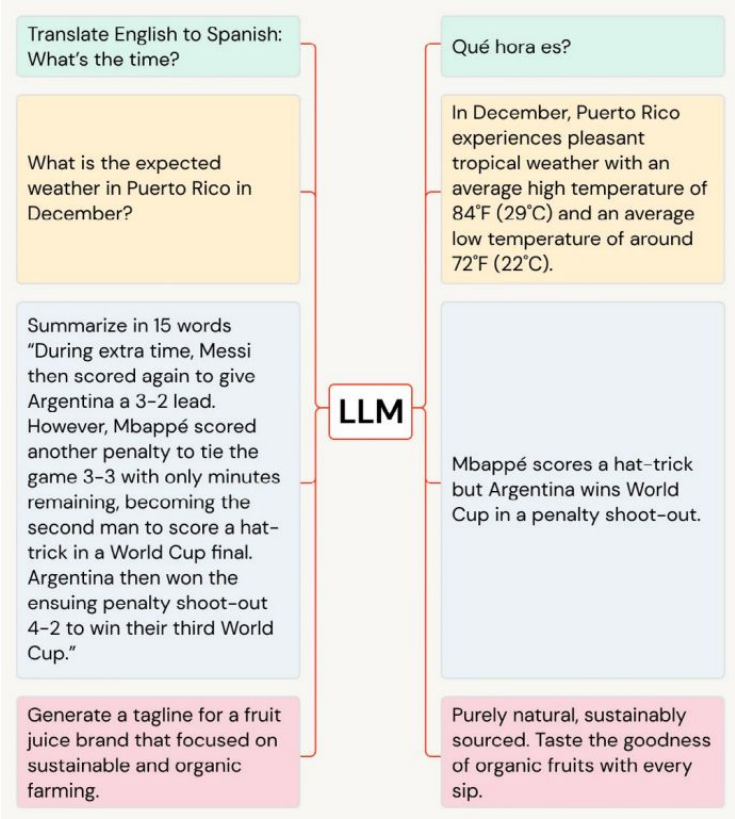
## Large Language Models

Eduardo Fernández Carrión, PhD

# LLM

- Los modelos de lenguaje grande son sistemas de IA que analizan **grandes cantidades de texto** para generar **respuestas naturales** a variadas **tareas** escritas, empleando para ello conjuntos de datos extensos y algoritmos de aprendizaje automático avanzados.

- Son fundamentales en aplicaciones como el procesamiento de lenguaje natural y la traducción automática, formando parte de un campo más amplio de IA generativa que incluye proyectos como la generación de arte, audio y video.



# Disponibilización de LLM

## Servicios propietarios

- Los servicios propietarios como ChatGPT de OpenAI **son altamente eficientes,** ofreciendo respuestas rápidas y manejan tareas complejas, pero su operación y entrenamiento tienen un costo elevado.
- Estos servicios requieren enviar datos a sus servidores, lo que puede generar **preocupaciones de privacidad** y control limitado sobre los modelos y su entrenamiento.
- Los servicios propietarios **no son gratuitos** más allá de un uso básico, siendo el coste un factor clave para su aplicación a gran escala.

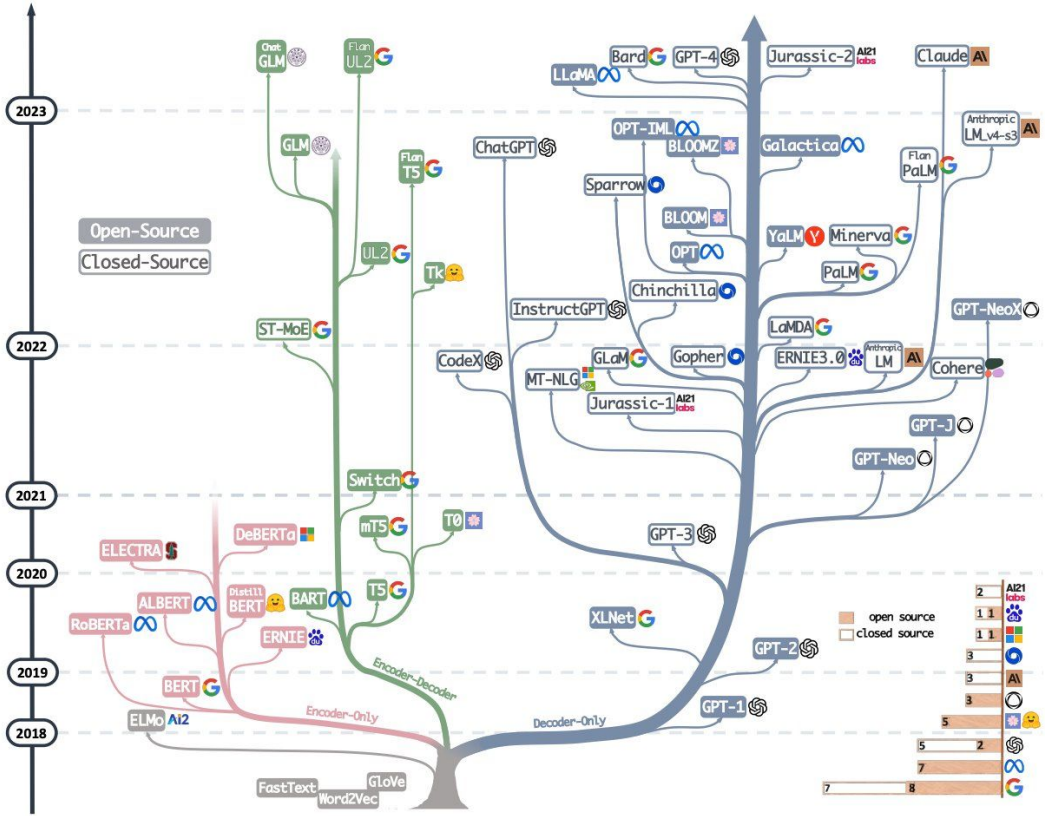
# Disponibilización de LLM

## Modelos de código abierto

- Los modelos de código abierto proporcionan una alternativa, con un crecimiento explosivo y comunidades como **Hugging Face** que ofrecen cientos de miles de modelos.
- Aunque los modelos de **código abierto están mejorando en rendimiento**, todavía no alcanzan la eficacia de modelos propietarios avanzados.
- En muchos casos de uso, es **suficiente con un LLM entrenado para una tarea específica**.
- Actualmente, implementar modelos de lenguaje abiertos **requiere más esfuerzo**, pero se está avanzando rápidamente para facilitar su acceso y uso.
- Utilizar modelos de código abierto permite un **control más directo sobre la privacidad y los costes**, y también la posibilidad de afinar los modelos con datos específicos.

# Transformers timeline

(hasta 2023)




https://www.kaggle.com/code/ashusma/progress-in-transformer-based-language-model

# Releases

| LATEST MODEL | DESCRIPTION | MAX TOKENS |
|---|---|---|
| `gpt-3.5-turbo` | Most capable GPT-3.5 model and optimized for chat at 1/10th the cost of `text-davinci-003`. Will be updated with our latest model iteration. | 4,096 tokens |
| `gpt-3.5-turbo-0301` | Snapshot of `gpt-3.5-turbo` from March 1st 2023. Unlike `gpt-3.5-turbo`, this model will not receive updates, and will be deprecated 3 months after a new version is released. | 4,096 tokens |
| `text-davinci-003` | Can do any language task with better quality, longer output, and consistent instruction-following than the curie, babbage, or ada models. Also supports some additional features such as | 4,097 tokens |

## Size

Generalmente hay varios tamaños que dependen de dos características principalmente,

- El **número de tokens** que admite de entrada
- El **número de parámetros** con el que ha sido entrenado (que depende de las capas)

| Model | Parameters (Millions) |
|---|---|
| BERT-Tiny | 4.4 |
| BERT-Mini | 11.3 |
| BERT-Small | 29.1 |
| BERT-Medium | 41.7 |
| BERT-Base | 110.1 |
| BERT-Large | 340 |

Cuanto más grande, más potente y preciso, pero más costoso computacionalmente.

## Flavors

- **Instruct**: Optimizadas para seguir instrucciones de manera más eficaz. El modelo ha sido entrenado y afinado específicamente para comprender y ejecutar mejor las instrucciones dadas por los usuarios, buscando producir respuestas que sean más útiles.
- **Chat**: Optimizadas para interactuar en formatos de diálogo. Estos modelos están ajustados para mantener conversaciones coherentes, contextuales y engaging con los usuarios, imitando el estilo de una conversación humana

# Ventajas y desventajas LLM fine-tuning

## Pros

- Adaptación a una Tarea específica

Crear un modelo específico para tu caso de uso.

- Costo de Inferencia

Modelos más adaptados suelen ser más pequeños, haciéndolos más rápidos en el tiempo de inferencia.

- Control

Toda la información de los datos y del modelo permanece completamente bajo tu control.

## Cons

- **Coste Computacional**

Este es el uso más costoso de un LLM ya que requerirá tanto tiempo de entrenamiento como costo computacional.

- **Requisitos de Datos**

Modelos más grandes requieren conjuntos de datos más grandes.

# Alternativas al Fine-Tuning

- **Few-shot learning** es un enfoque en el campo de la IA que busca crear modelos capaces de aprender una nueva tarea con **muy pocos ejemplos de entrenamiento.**
  Este tipo de aprendizaje es especialmente relevante en situaciones donde hay una escasez de datos etiquetados

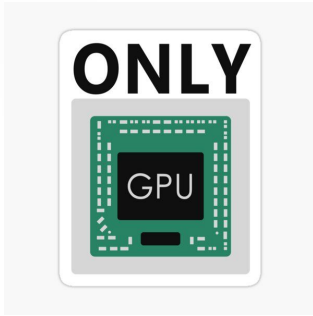
huggingface/**setfit**

Efficient few-shot learning with Sentence Transformers

```
Clasifica el siguiente texto según si pertenece al ámbito de hogar, deportes o electrónica.

Ejemplo 1:
Texto: "Los mejores ejercicios para aumentar la resistencia cardiovascular."
Clasificación: Deportes

Ejemplo 2:
Texto: "Comparativa de los últimos modelos de smartphones del año."
Clasificación: Electrónica

Ejemplo 3:
Texto: "Consejos para decorar tu sala de estar con un presupuesto limitado."
Clasificación: Hogar

Ejemplo 4:
Texto: "Cómo instalar una red Wi-Fi en casa para mejorar la cobertura."
Clasificación: Electrónica

Ejemplo 5:
Texto: "Técnicas de meditación para después de hacer yoga."
Clasificación: Deportes

Texto a clasificar:
"Guía de mantenimiento de electrodomésticos para prolongar su vida útil."
Clasificación:
```

- Los modelos como GPT-3 de OpenAI también muestran capacidades de few-shot learning. Se les puede presentar unos pocos ejemplos en su entrada (**como parte de un *prompt***), tras lo cual pueden generar predicciones o continuar patrones en contextos que no habían visto durante su entrenamiento inicial.

# Recursos para Fine-Tuning




## Model Memory Calculator

The minimum recommended vRAM needed for a model is denoted as the size of the "largest layer", and training of a model is roughly 4x its size (for Adam).

Memory usage for 'mistral-community/Mistral-7B-v0.2'

| dtype | Largest Layer or Residual Group | Total Size | Training using |
|---|---|---|---|
| float16/bfloat16 | 432.02 MB | 13.74 GB | 54.98 GB |
| float32 | 864.03 MB | 27.49 GB | 109.96 GB |

Training using Adam explained:

When training on a batch size of 1, each stage of the training process is expected to have near the following memory results for each precision you selected:

| dtype | Model | Gradient calculation | Backward pass | Optimizer s |
|---|---|---|---|---|
| float16/bfloat16 | 27.49 GB | 41.23 GB | 54.98 GB | 54.98 GB |
| float32 | 27.49 GB | 27.49 GB | 54.98 GB | 109.96 GB |

Memory usage for 'NousResearch/Llama-2-7b-chat-hf'

| dtype | Largest Layer or Residual Group | Total Size | Training usin |
|---|---|---|---|
| float16/bfloat16 | 388.02 MB | 12.37 GB | 49.48 GB |
| float32 | 776.03 MB | 24.74 GB | 98.96 GB |

Training using Adam explained:

When training on a batch size of 1, each stage of the training process is expected to have near the following memory results for each precision you selected:

| dtype | Model | Gradient calculation | Backward pass | Optimizer s |
|---|---|---|---|---|
| float16/bfloat16 | 24.74 GB | 37.11 GB | 49.48 GB | 49.48 GB |
| float32 | 24.74 GB | 24.74 GB | 49.48 GB | 98.96 GB |

https://huggingface.co/docs/accelerate/main/en/usage_guides/model_size_estimator

# Quantization

Es una técnica para **reducir la precisión de 32 bits en coma flotante** de los números que representan los parámetros del modelo como los **pesos, sesgos** o **activaciones**.

La quantization reduce significativamente el **coste computacional**, **los tiempos** de entrenamiento e inferencia, pero **reduce la calidad** del modelo.

Los tipos de datos de menor precisión más comunes son:

- **float16**, tipo de dato de acumulación float16
- **bfloat16**, tipo de dato de acumulación float32
- **int16**, tipo de dato de acumulación int32
- **int8**, tipo de dato de acumulación int32

# QLoRA

**QLoRA (*Quantized Low-Rank Adaptation*)** es una **técnica de optimización de modelos** de inteligencia artificial, específicamente para la **quantization** y la **reducción de la complejidad computacional** en los LLMs

## QLoRA: Efficient Finetuning of Quantized LLMs

| Paper | Adapter Weights | Demo |

This repo supports the paper "QLoRA: Efficient Finetuning of Quantized LLMs", an effort to democratize access to LLM research.

QLoRA uses bitsandbytes for quantization and is integrated with Hugging Face's PEFT and transformers libraries.
QLoRA was developed by members of the University of Washington's UW NLP group.

https://github.com/artidoro/qlora

**bitsandbytes** y **PEFT** también forman parte de HF

# PEFT

***Parameter-Efficient Fine-Tuning*** es una técnica que permite adaptar de manera eficiente LLMs  a diversas aplicaciones específicas **sin necesidad de ajustar todos los parámetros del modelo**

Los ajustes PEFT crean **checkpoints** de modelo de tamaño reducido, lo cual es más manejable y económico en términos de almacenamiento.

PEFT

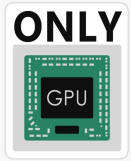
State-of-the-art Parameter-Efficient Fine-Tuning (PEFT) methods

https://github.com/huggingface/peft

# LLM Fine-Tuning

## Libraries

```
from transformers import (
    AutoModelForCausalLM,
    AutoTokenizer,
    BitsAndBytesConfig,
    TrainingArguments,
)
from peft import LoraConfig, PeftModel
from trl import SFTTrainer
```




PEFT

```
import torch

torch.cuda.is_available()
```

```
True
```

## Instruction based Dataset

**text**
string · *lengths*

96 155 16.3%

| |
|---|
| <s>[INST] What does OPL stand for in the OPL programming language? [/INST] OPL is short for Open Programming Language </s> |
| <s>[INST] Which company developed the OPL programmung language? [/INST] Psion Ltd created OPL for the Psion Organiser </s> |
| <s>[INST] Which was the original name for the OPL programming language? [/INST] The OPL language was originally named Organiser Programming Language </s> |
| <s>[INST] Who created the OPL programming language? [/INST] The OPL language was designed by Colly Myers and implemented by Richard Harrison and Martin Stamp of Psion Ltd. </s> |
| <s>[INST] What was the first implementation of OPL? [/INST] The first implementation of the OPL programming language was in 1984 for the Psion Organiser. </s> |
| <s>[INST] What does the REM keyword stand for in OPL? [/INST] REM is short for REMark. it is used to represent a comment in OPL. For example: REM i am a comment </s> |

```
!pip install -q  transformers==4.45.0 datasets==3.6.0
accelerate==0.34.0 peft==0.13.2 bitsandbytes==0.45.5
trl==0.11.2 sentencepiece==0.2.0 einops==0.8.1
```

ntic master
revolucionamos la comunicación

UNIVERSIDAD COMPLUTENSE MADRID