

Introduction to Large Language Models

Nikita Saxena
Research Engineer

Intros

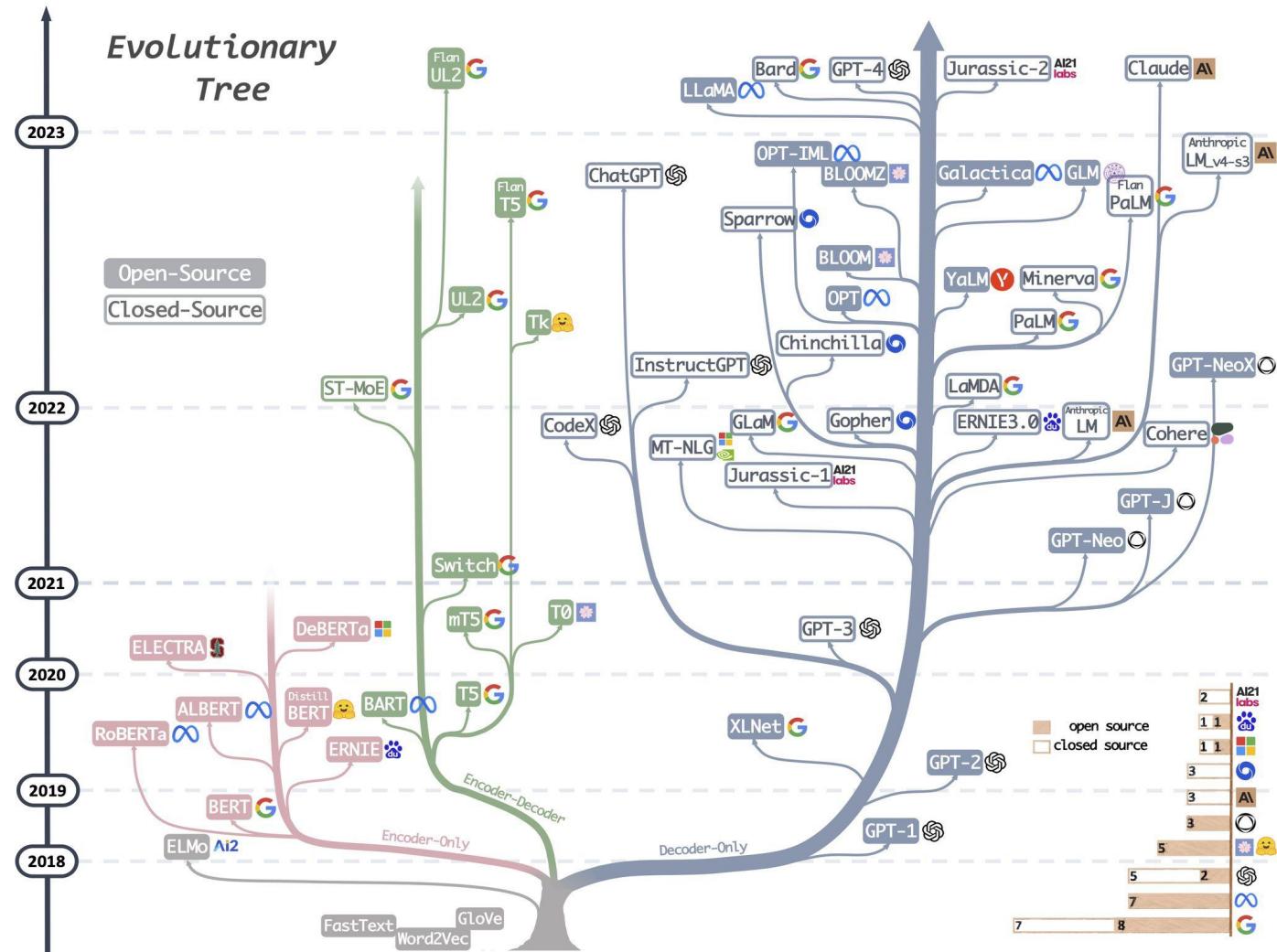
1. Name
2. Educational Background
3. Dream Job/Where do you see yourself in 2-5 years?
4. [Optionally] Specific Thing you want to learn this week

Evolution of Large Language Models:

A handful of organizations like Google, OpenAI, and Meta are driving most of the major advancements.

Open vs. Closed Source:

There's a dynamic split between proprietary, closed-source models (like GPT-4, PaLM) and powerful open-source models (like LLaMA, BLOOM) that fuel community research.

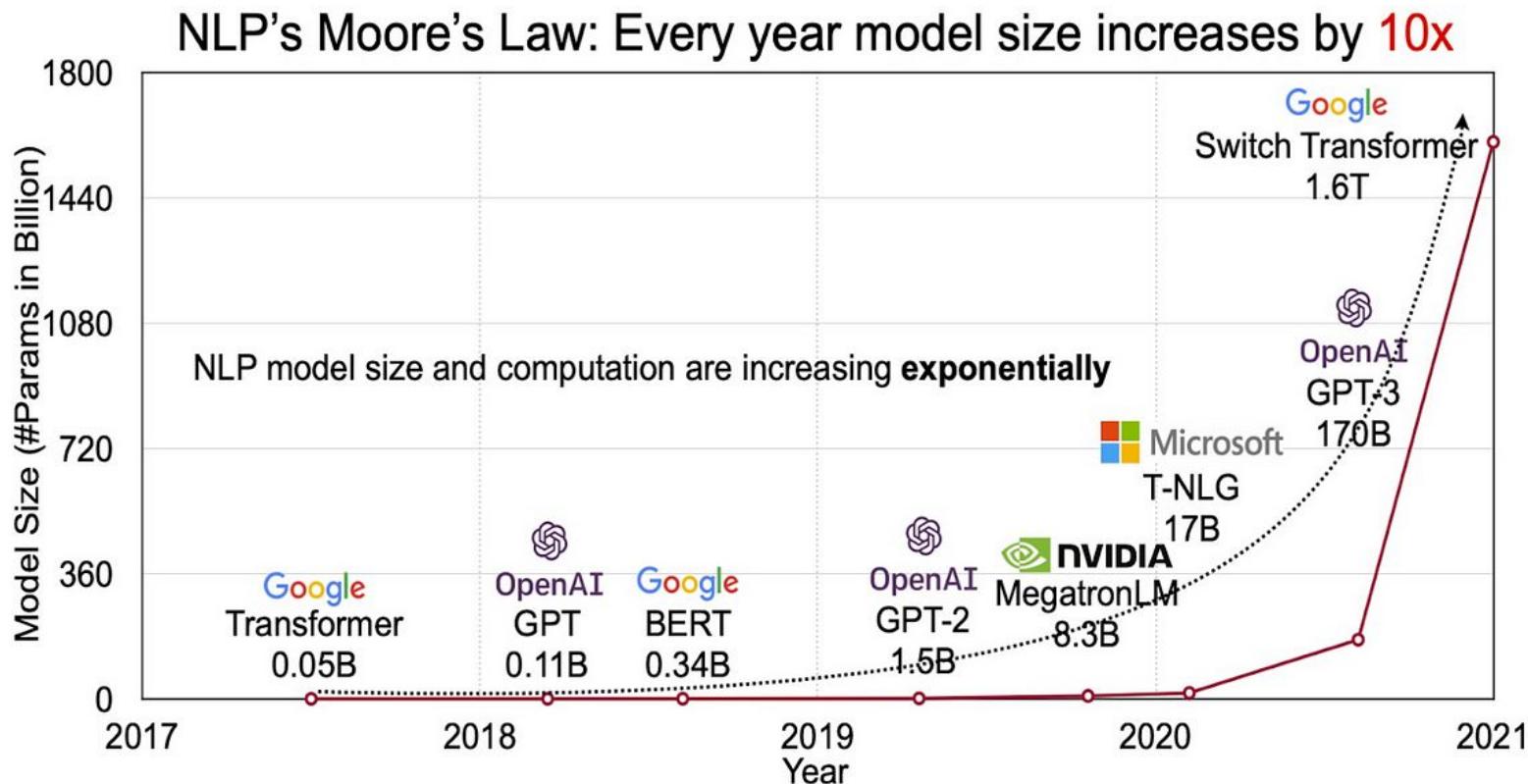


Let's us look at the number of parameters in a CNN:

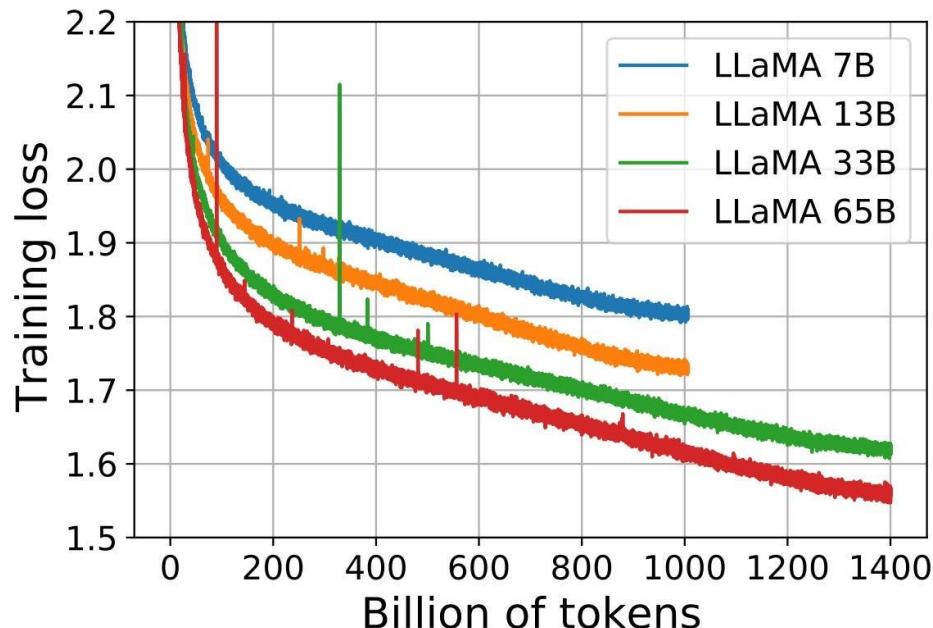
<u>SL.No</u>		<u>Activation Shape</u>	<u>Activation Size</u>	<u># Parameters</u>
1.	Input Layer:	(32, 32, 3)	3072	0
2.	CONV1 (f=5, s=1)	(28, 28, 8)	6272	608
3.	POOL1	(14, 14, 8)	1568	0
4.	CONV2 (f=5, s=1)	(10, 10, 16)	1600	3216
5.	POOL2	(5, 5, 16)	400	0
6.	FC3	(120, 1)	120	48120
7.	FC4	(84, 1)	84	10164
8.	Softmax	(10, 1)	10	850

70,000

And now let's compare it with some LLMs



- **Learning = Reducing "Loss":** During training, a model's performance is measured by "loss" (an error score). The goal is to get this score as low as possible.
- **More Data is Better:** As models process more data (moving right on the chart), their loss consistently goes down. They get better at predicting the next word.
- **Bigger Models Learn More:** For the same amount of training data, larger models (like the 65B in red) achieve a lower loss, meaning they are more effective learners than smaller models (like the 7B in blue).



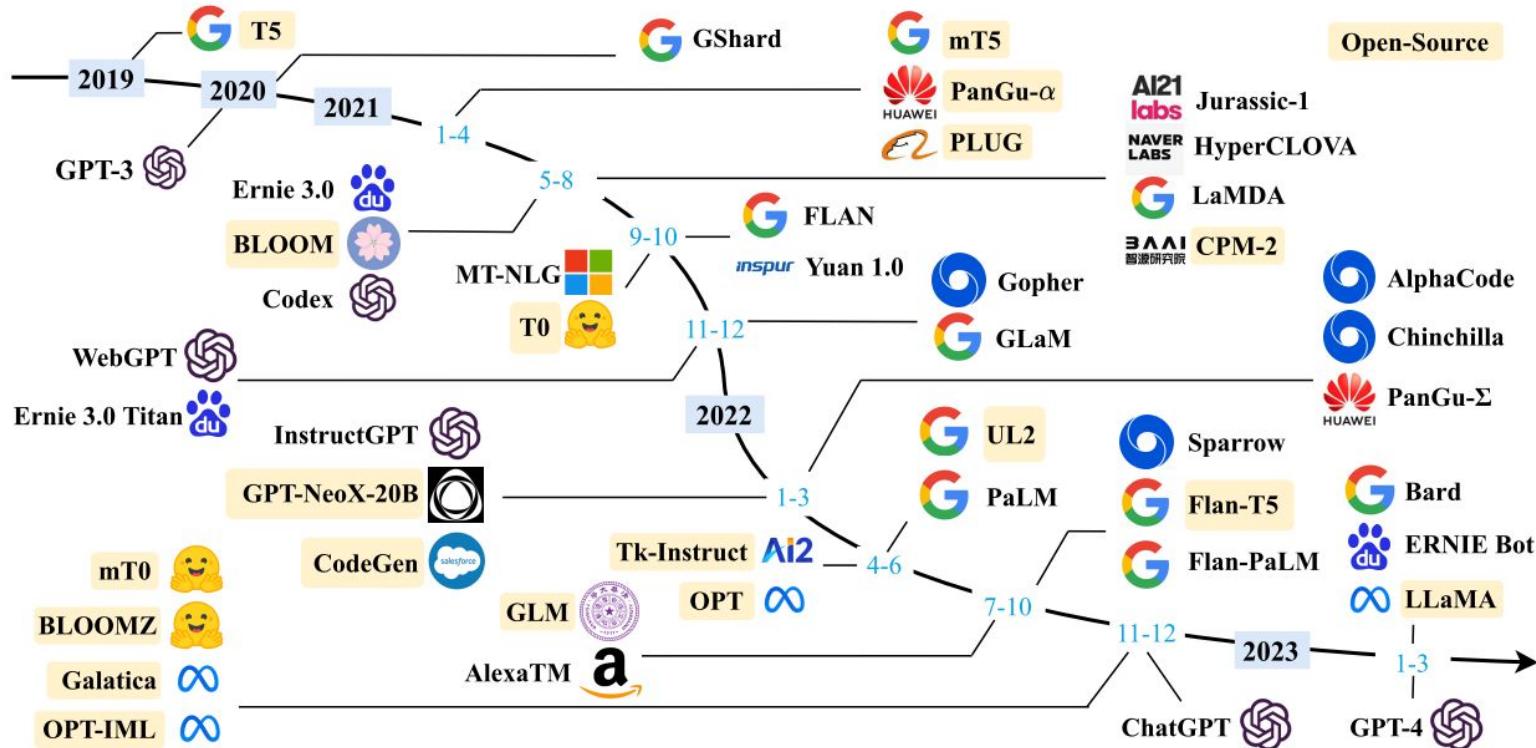


Fig. 1. A timeline of existing large language models (having a size larger than 10B) in recent years. We mark the open-source LLMs in yellow color.

Training Cost of LLMs

The investment in larger models has been substantial since 2020.

- For instance, training the GPT-2, a 1.5-billion-parameter model, in 2019 cost \$50,000;
- In contrast, training PaLM, a massive 540-billion-parameter model in 2022, came with a staggering price tag of \$8 million.

When training a 65B-parameter model, our code processes around 380 tokens/sec/GPU on 2048 A100 GPU with 80GB of RAM. This means that training over our dataset containing 1.4T tokens takes approximately 21 days.

Amount of computing resources for training the LLaMA model. Image from LLaMA paper

Inference Cost of LLMs

- Training consumes about 6 FLOPs (Floating-Point Operations) per parameter per token;
- Inference typically ranges from 1 to 2 FLOPs per parameter per token.

Model	GPT-3.5 4K Context	GPT-3.5 16K Context	GPT-4 8K Context	GPT-4 32K Context
Yearly cost – 1K requests per day	\$949	\$1,898	\$28,470	\$56,940
Yearly cost – 1M requests per day	\$949,000	\$1,898,000	\$28,470,000	\$56,940,000

Cost of LLMs

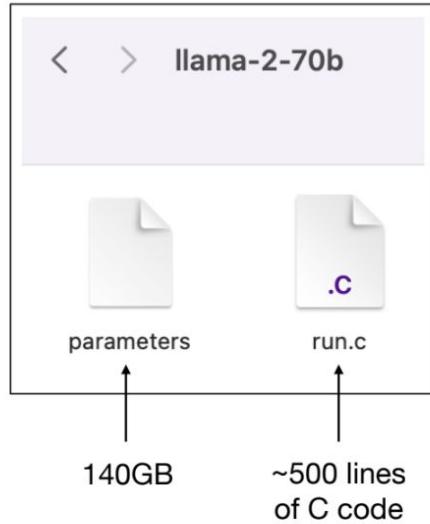
- **GPT-3.5/4 API Costs:** ChatGPT API is priced at \$0.002 for 1,000 tokens. For lower usage, the cost is reasonable, but processing a million documents a day can cost around \$1 million a year. More powerful models like GPT-4 have varying costs based on usage, ranging from \$1,000 to \$56 million annually.
- **Open-Source Model Hosting Costs:** Hosting LLMs involves GPU RAM and hourly charges. For a 7 billion parameter model at 16-bit precision, it would cost around \$2-3 per hour. Larger open-source models require more GPU memory and can cost around \$20 per hour.
- **On-Premises Hosting Costs:** Running LLMs on private servers requires high-quality GPUs, which can be expensive. Scaling up on-premises involves purchasing additional infrastructure, setting it up, and handling maintenance. It's important to consider other cloud-related infrastructure, such as load balancers, for maintaining network traffic and ensuring low latency.

Model	Release Time	Size (B)	Base Model	Adaptation IT	Adaptation RLHF	Pre-train Data Scale	Latest Data Timestamp	Hardware (GPUs / TPUs)	Training Time	Evaluation ICL	Evaluation CoT
Closed Source	GPT-3 [55]	May-2020	175	-	-	300B tokens	-	-	-	✓	-
	GShard [91]	Jun-2020	600	-	-	1T tokens	-	2048 TPU v3	4 d	-	-
	Codex [92]	Jul-2021	12	GPT-3	-	100B tokens	May-2020	-	-	✓	-
	ERNIE 3.0 [93]	Jul-2021	10	-	-	375B tokens	-	384 V100	-	✓	-
	Jurassic-1 [94]	Aug-2021	178	-	-	300B tokens	-	800 GPU	-	✓	-
	HyperCLOVA [95]	Sep-2021	82	-	-	300B tokens	-	1024 A100	13.4 d	✓	-
	FLAN [62]	Sep-2021	137	LaMDA-PT	✓	-	-	128 TPU v3	60 h	✓	-
	Yuan 1.0 [96]	Oct-2021	245	-	-	180B tokens	-	2128 GPU	-	✓	-
	Anthropic [97]	Dec-2021	52	-	-	400B tokens	-	-	-	✓	-
	WebGPT [72]	Dec-2021	175	GPT-3	-	✓	-	-	-	✓	-
	Gopher [59]	Dec-2021	280	-	-	300B tokens	-	4096 TPU v3	920 h	✓	-
	ERNIE 3.0 Titan [98]	Dec-2021	260	-	-	-	-	-	-	✓	-
	GLaM [99]	Dec-2021	1200	-	-	280B tokens	-	1024 TPU v4	574 h	✓	-
	LaMDA [63]	Jan-2022	137	-	-	768B tokens	-	1024 TPU v3	57.7 d	-	-
	MT-NLG [100]	Jan-2022	530	-	-	270B tokens	-	4480 80G A100	-	✓	-
	AlphaCode [101]	Feb-2022	41	-	-	967B tokens	Jul-2021	-	-	-	-
	InstructGPT [61]	Mar-2022	175	GPT-3	✓	✓	-	-	-	✓	-
	Chinchilla [34]	Mar-2022	70	-	-	1.4T tokens	-	-	-	✓	-
	PaLM [56]	Apr-2022	540	-	-	780B tokens	-	6144 TPU v4	-	✓	✓
	AlexaTM [102]	Aug-2022	20	-	-	1.3T tokens	-	128 A100	120 d	✓	✓
	Sparrow [103]	Sep-2022	70	-	-	✓	-	64 TPU v3	-	✓	-
	WeLM [104]	Sep-2022	10	-	-	300B tokens	-	128 A100 40G	24 d	✓	-
	U-PaLM [105]	Oct-2022	540	PaLM	-	-	-	512 TPU v4	5 d	✓	✓
	Flan-PaLM [64]	Oct-2022	540	PaLM	✓	-	-	512 TPU v4	37 h	✓	✓
	Flan-U-PaLM [64]	Oct-2022	540	U-PaLM	✓	-	-	-	-	✓	✓
	GPT-4 [46]	Mar-2023	-	-	✓	✓	-	-	-	✓	✓
	PanGu-Σ [106]	Mar-2023	1085	PanGu- α	-	329B tokens	-	512 Ascend 910	100 d	✓	-
	PaLM2 [107]	May-2023	16	-	✓	100B tokens	-	-	-	✓	✓

01

Overview

Open Source LLMs: Under the Hood



LLM \approx Weights (Binary File) + Code (run.c)

- **Weights**: model's "brain" - billions of numbers).
- **Code (run.c)**: Load weights and perform inference.

*This core combination allows anyone to run the model locally.
Llama (by Meta) is an example of an open source model.*

In contrast, closed source models (e.g., ChatGPT, Gemini):

- Accessed via APIs, **not direct file downloads**.
- Internal workings and full weights are **proprietary**.
- More like a "**service**" than a downloadable artifact.

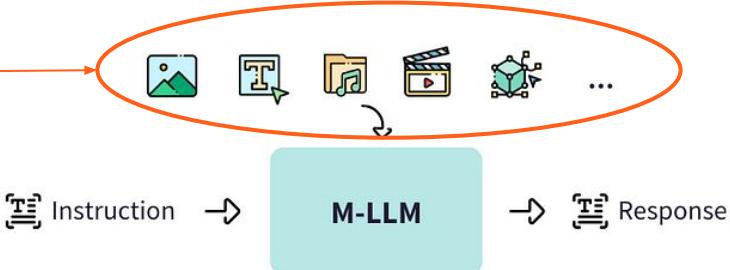
LLM Training: The TLDR

Core Concept: LLM training is essentially a lossy compression of (a large part of) the internet.

01

Vast Internet Data
Textbooks, Websites, Code

Multimodal Datasets:



LLM Training: The TLDR

Core Concept: LLM training is essentially a lossy compression of (a large part of) the internet.

01

Vast Internet Data
Textbooks, Websites, Code

02

Training Process
Massive GPU Clusters

LLM Training: The TLDR

Core Concept: LLM training is essentially a lossy compression of (a large part of) the internet.

01

Vast Internet Data
Textbooks, Websites, Code

02

Training Process
Massive GPU Clusters

03

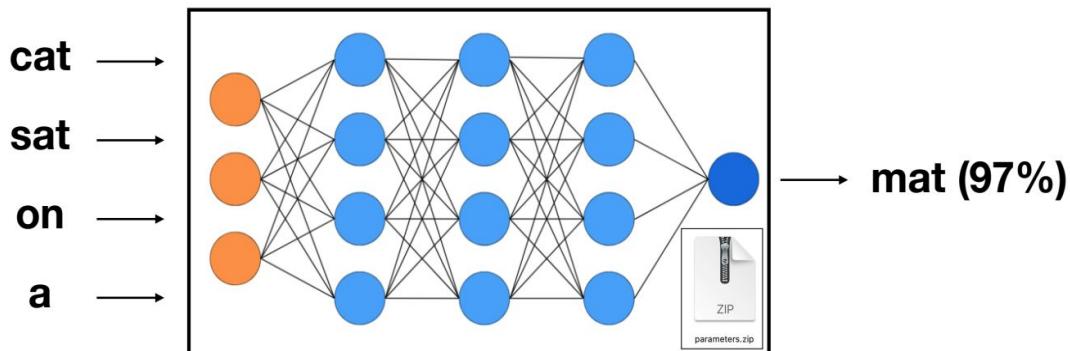
Model Weights: "The LLM"
Compact Binary File



Core Task: Next Word Prediction

Fundamental Mechanism: Next-token predictor.

Emergent Capabilities: Translation, summarization, coding emerge from repeating next token prediction.



e.g. context of 4 words

predict next word

The LLM calculates the probability of every possible next word:

- "...mat" (**high**)
- "...chair" (**medium**)
- "...sky" (**low**)

It then usually picks one of the high-probability words.

Let's visualize what an LLM looks like

<https://bbycroft.net/llm>

time: 5 mins



02

Key Stages in LLM Training

Pre-training

Building foundational knowledge.

Supervised Fine-Tuning (SFT):

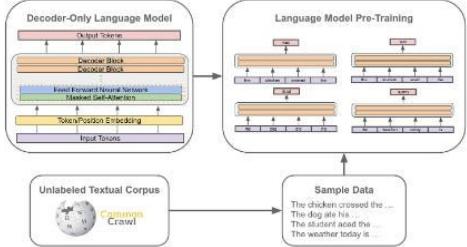
Teaching instruction following and specific skills.

Reinforcement Learning from Human Feedback (RLHF):

Aligning with human preferences.

Alignment

Pre-Training



SFT

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

RLHF

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

RLHF

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.

Building World Knowledge

Goal

Create a *base model* with **broad understanding**.

Input

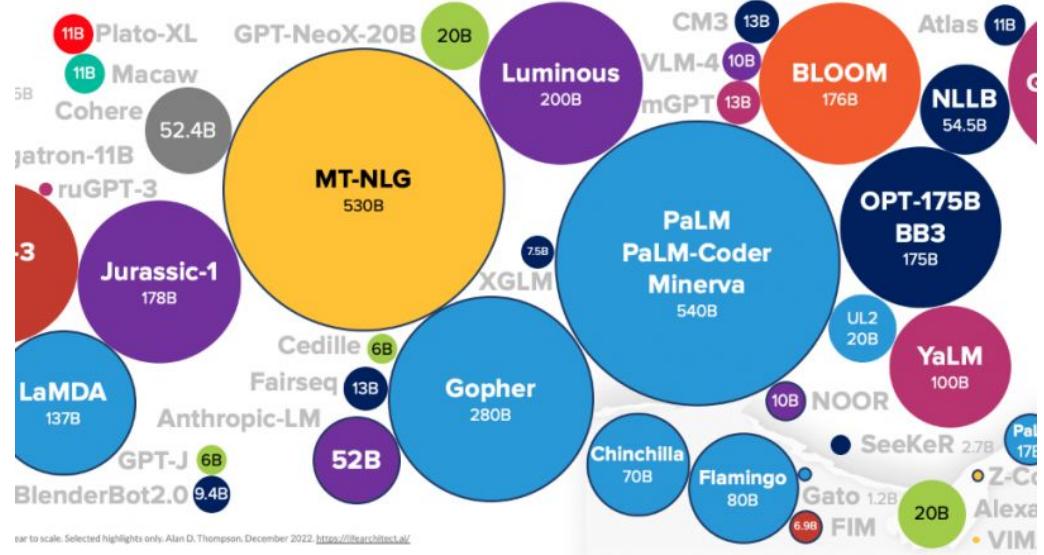
Massive, diverse corpus of text and code.
(e.g., *Common Crawl*, *Wikipedia*, *Books*, *GitHub*).

Process: *self-supervised learning*

Typically predicting masked words or the next sentence.

Outcome: *base model*

- Possesses extensive world knowledge: grammar, facts, common sense, coding patterns..
- Can complete text, but not yet optimized for conversation or instruction following.



Shaping the Assistant

Goal

Teach the base model to **follow instructions** and act like an assistant.

Input

- The pre-trained base model.
- A smaller, high-quality dataset of instruction-response pairs.

Instruction: "Explain gravity like I'm five."

Response: "Imagine..."

Process: alignment

Train the model to generate the desired 'response' given an 'instruction'.

Output: *instruction-tuned model*.

Better at answering questions, following commands, and engaging in dialogue.

SFT teaches the model behave in a helpful, instructive way.

Gemini

2.5 Flash ▾

❖ Invite a friend

User

Hi, can you help me with an itinerary for Kazakhstan?

Assistant



Kazakhstan is a vast and diverse country, offering everything from futuristic cities to stunning natural landscapes. The ideal itinerary depends heavily on how much time you have and what your interests are. Here are a few options, ranging from a shorter trip focusing on a specific region to a more comprehensive exploration.

General Tips for Traveling in Kazakhstan:

- **Best Time to Visit:** May to September generally offers the best weather for outdoor activities, though spring and autumn can be beautiful for less extreme temperatures.
- **Transportation:** Distances are vast. For long distances, consider domestic flights or overnight trains. Within cities, taxis (Yandex.Taxi is popular) are affordable. For national parks around Almaty, renting a car or hiring a driver can offer more flexibility.

Preference Alignment

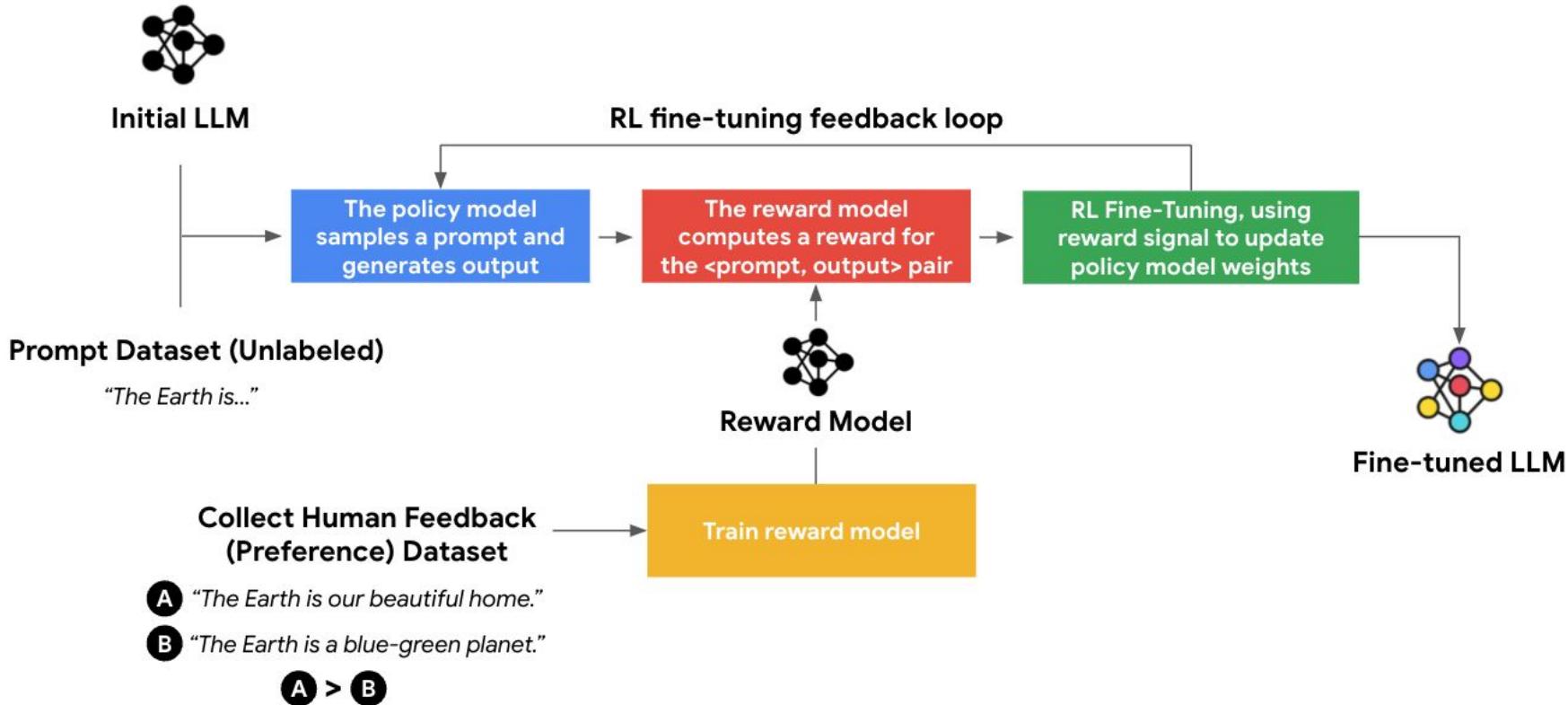
Goal

Improve alignment by teaching the model what **humans prefer**.

Process: Refining with Preferences

Comparative analysis – "Which response is better?"

- Take the instruction-tuned model.
- Generate multiple responses to various prompts.
 - Humans rank these responses.
 - This preference data is used to train a **Reward Model**.
- **Reward Model** is used to fine-tune the LLM using reinforcement learning techniques to **maximize "favourable" responses**.



Pre-training Size
 $O(100\text{TB})$

Forcing a student to read every single book in the world's largest library. They won't become an expert in any one thing, but they'll gain an immense, broad-based knowledge about everything.

SFT Size
 $O(100\text{K})$

Given a set of high-quality flashcards with specific questions and the perfect answers, learn style and format of how to answer questions.

RLHF Size
 $O(100\text{K})$

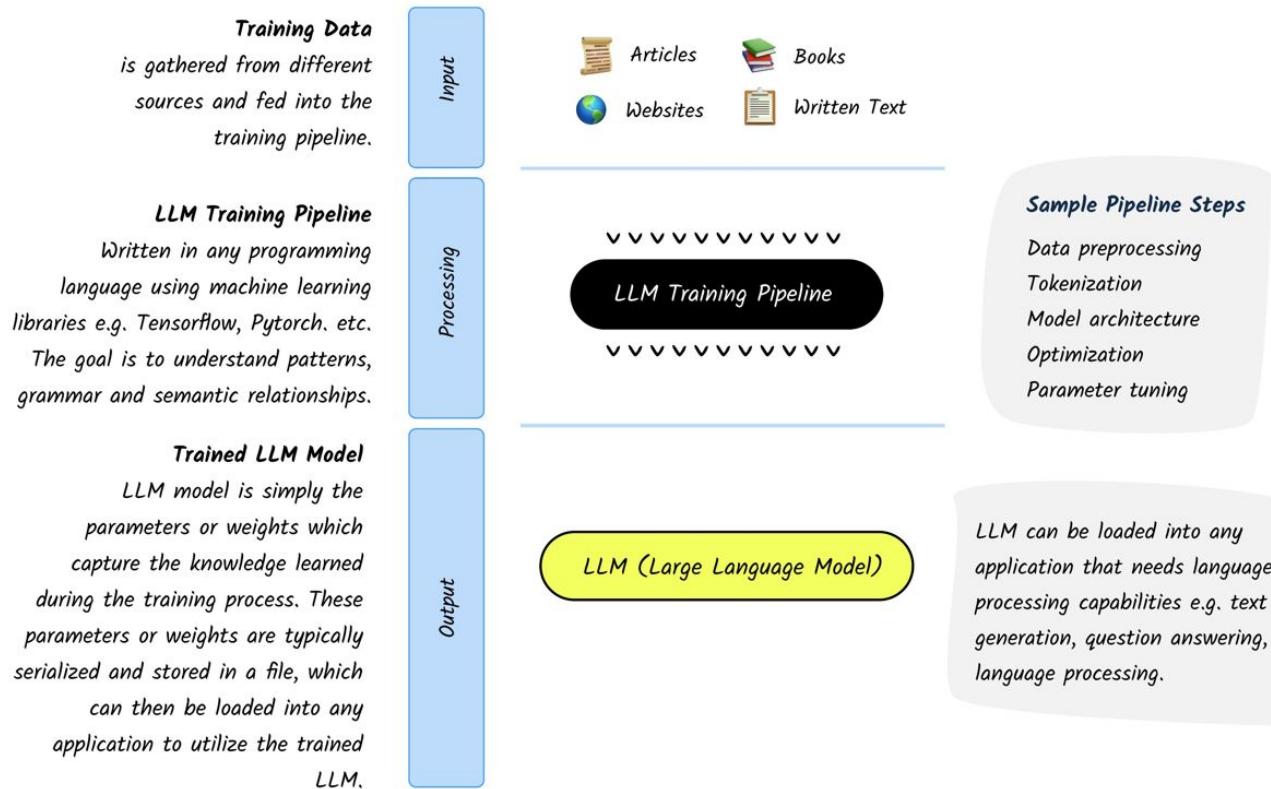
Provide several possible answers to a question. Refine their tone, style, and judgment by getting feedback from experts on each answer.

03

Summing Up

LLM - Large Language Models

AI systems designed to understand and generate human like text.



03

LMSYS

LMSYS (Large Model Systems Organization)

Who are they?

Open research organization (collaborators from UC Berkeley, UC San Diego, CMU, etc.)

Key Contributions & Projects

- **Chatbot Arena:** A crowd-sourced platform for anonymous, side-by-side LLM evaluation. Users chat with two models and vote for the better response.
Provided a dynamic leaderboard.
- **Public Datasets:** Release valuable human preference data (e.g., from Chatbot Arena interactions), crucial for RLHF and alignment research.

Why are they important?

Provides an independent, dynamic way to compare LLM capabilities.

Let's compare different models in LMSYS

<https://lmarena.ai/>

time: 5 mins





Thank you.



Nikita Saxena
Research Engineer