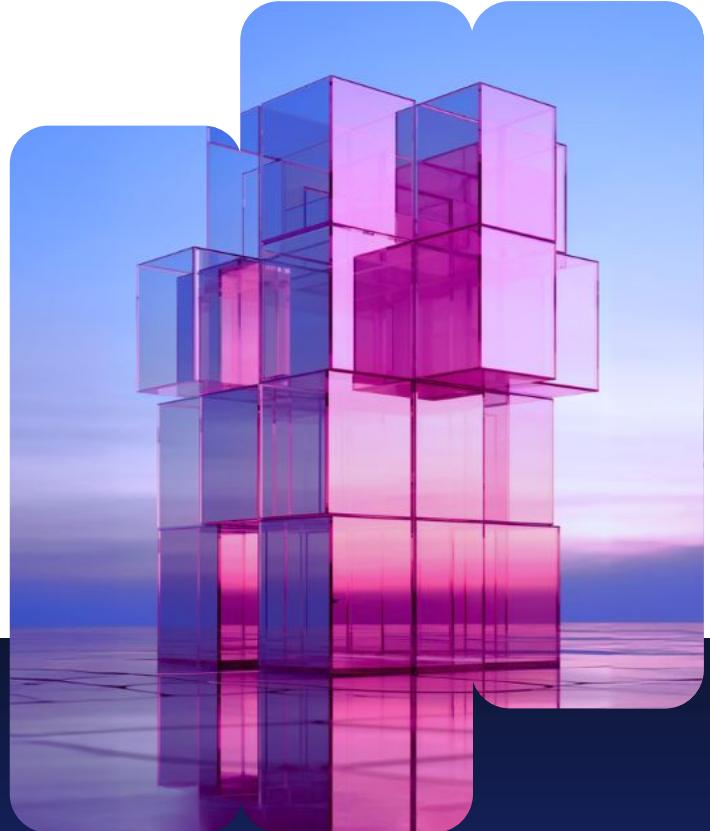




Building On-Prem LLM Applications for the Real World. *AI Today: The Generative Landscape*

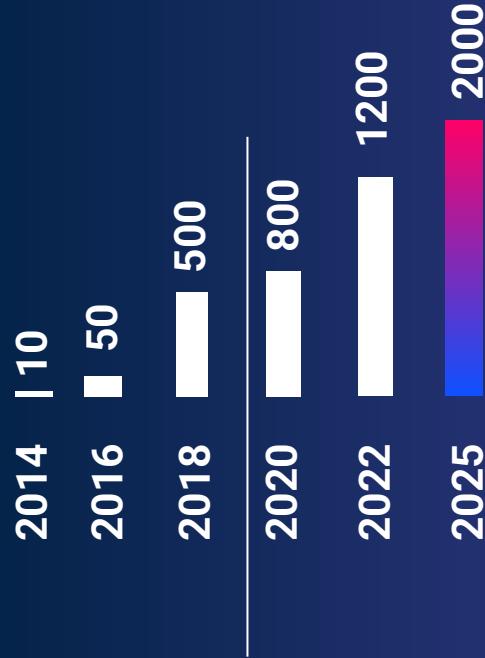
**AMLD EPFL
12 FEBRUARY 2026**

Zach Schillaci · Director of Data Science · Artefact CH
zachary.schillaci@artefact.com



Artefact is a world leading **data & AI consulting** partner.

#Employees



27
countries

2000
employees

36
offices



Agenda.

1

LLMs in Brief

2

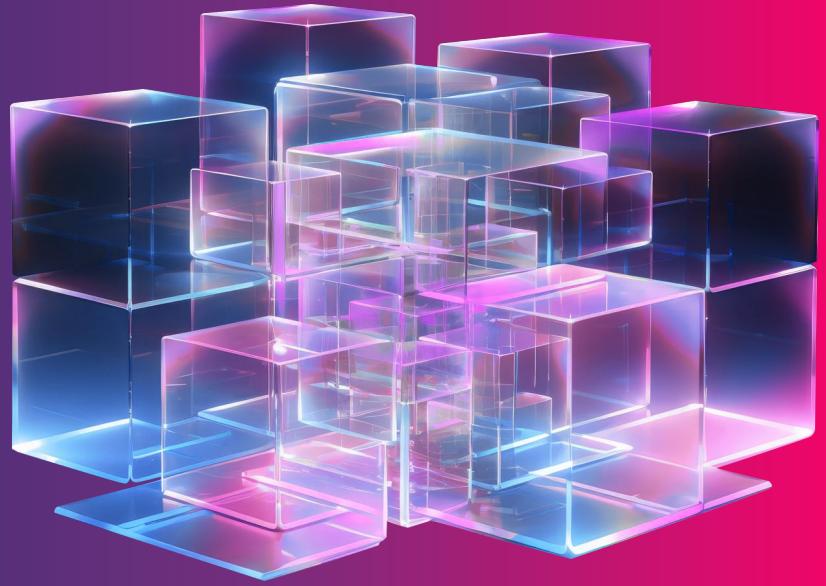
Capabilities and Usage

3

Open-source Landscape

LLMs in Brief.

ARTFACT



LLMs in Brief

The bigger picture

Rule-based systems

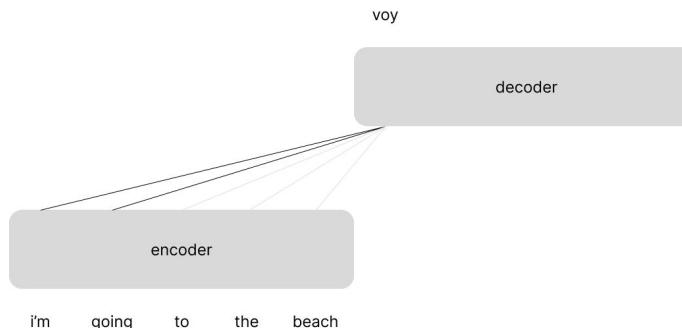
1950

Generative Adversarial Networks (GANs)

2014 (*Ian Goodfellow*)

Transformers

2017 (*Attention is all you need, Vaswani et al.*)

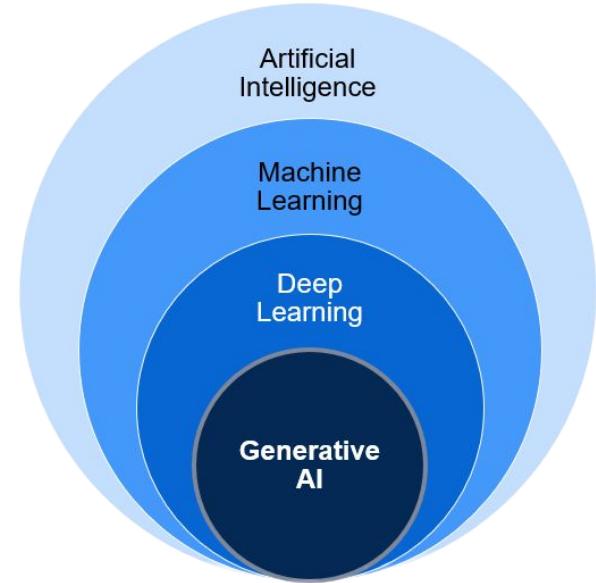


Generative Pre-trained Transformer (GPT)

2018 (*OpenAI*)

GPT-3

2020 (*OpenAI*)



LLMs in Brief

Text prediction is all you need

LLM = Large Language Model

In other words, it's a ...

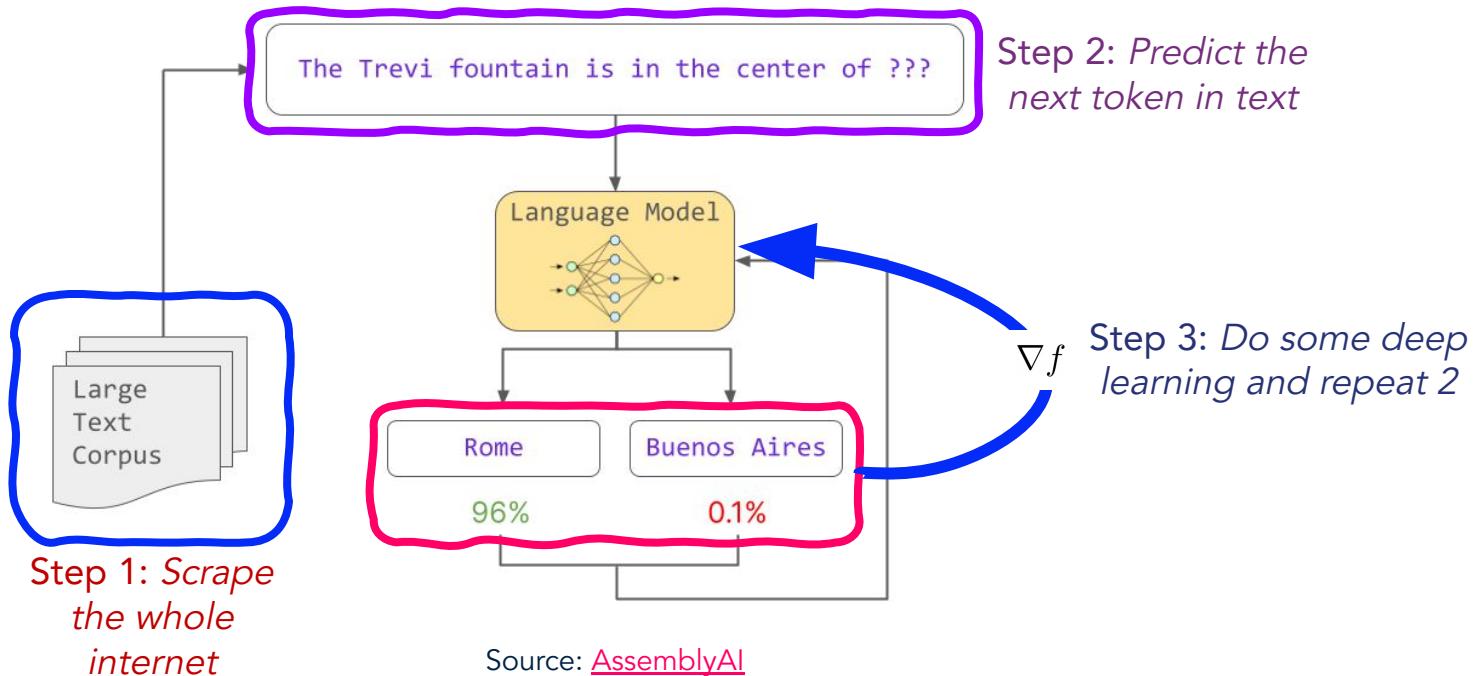
- *Large model of language*
- *Big neural network trained on text generation*
- *Autocomplete on steroids*
- *Stochastic parrot*
- *Probabilistic text prediction engine*
- *Text-to-text translator*
- *Autoregressive text generator*

LLMs in Brief

Text prediction is all you need

LLM = Large Language Model

At a high-level ...

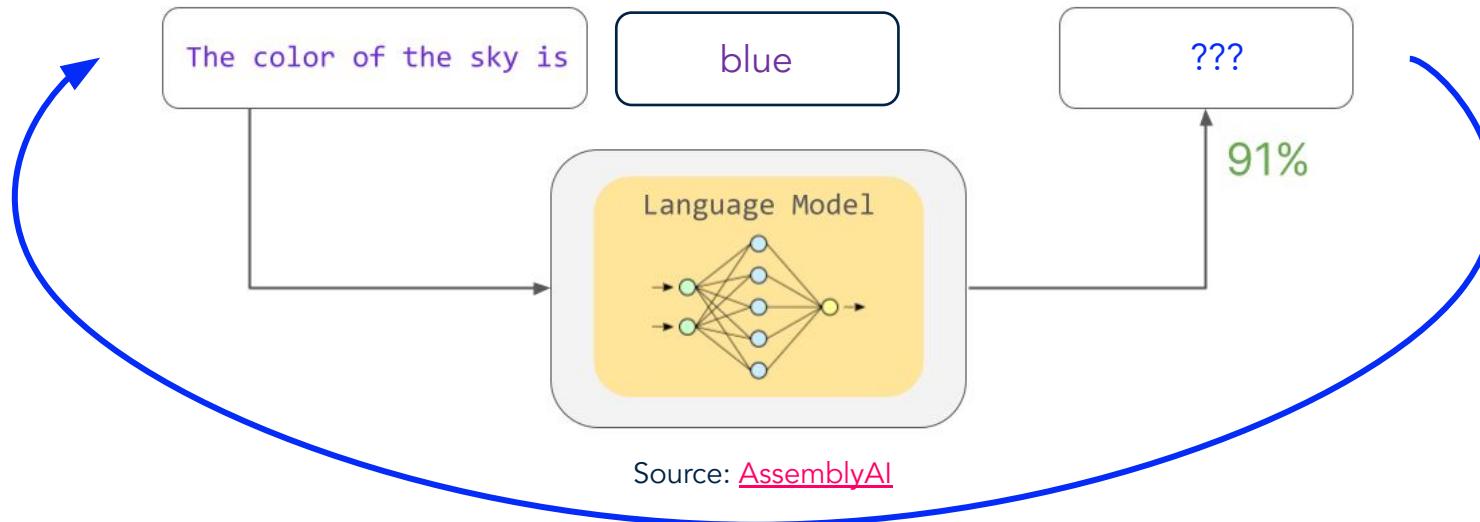


LLMs in Brief

Text prediction is all you need

At a high-level ...

LLM = Large Language Model

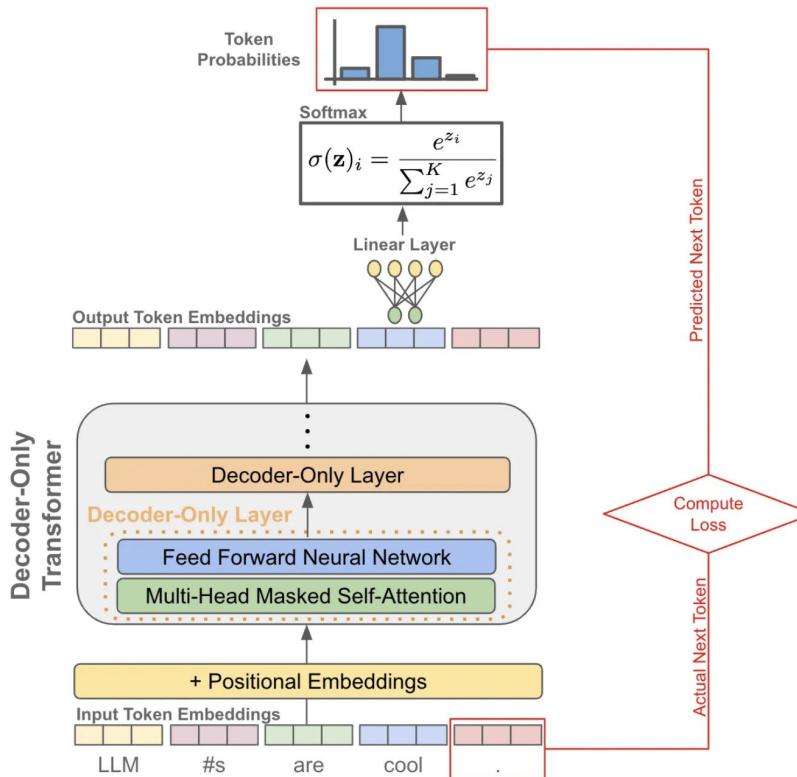


Source: [AssemblyAI](#)

Autoregressive text generation

LLMs in Brief

Text prediction is all you need



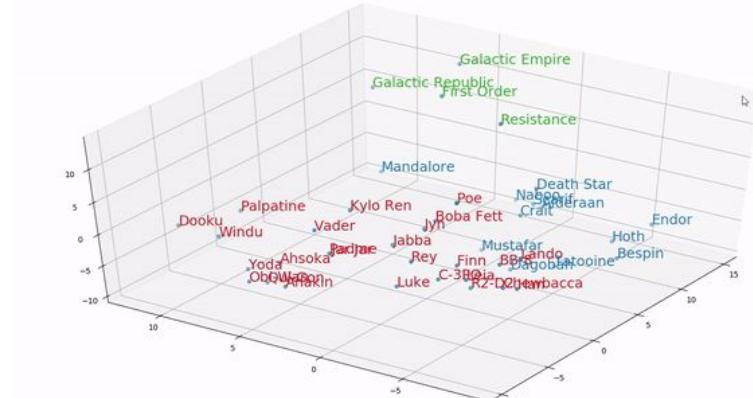
Computing the next token prediction training objective (created by author)

Source: [Medium](#)

- Restricted -

At a lower level ...

Visualizing Star Wars in Word Vector Space (embedding)



Source: [Medium](#)

Taken from Cameron Wolfe's excellent blog
"The History of Open-Source LLMs: Early Days (Part One)"
For a great technical overview, read [here!](#)

LLMs in Brief

What's in a word?

GPT-3.5 & GPT-4 GPT-3 (Legacy)

Tokens are the fundamental units of Large Language Models! **Words**

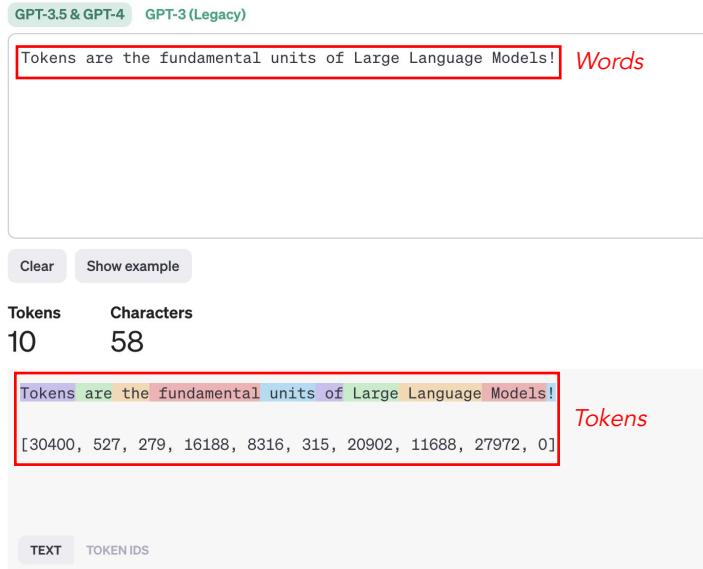
Clear Show example

Tokens	Characters
10	58

Tokens are the fundamental units of Large Language Models! **Tokens**

[30400, 527, 279, 16188, 8316, 315, 20902, 11688, 27972, 0]

TEXT TOKEN IDS



Source: [OpenAI](#)

- *Tokens* are the fundamental units of LLMs and correspond to common sequences of characters learned by the *tokenizer*
- Tokens are represented internally by learned *embedding vectors* in the model
- Fixed size *vocabulary* of tokens (typically ~100-200k)
- *Context window* sets the number of tokens the model can consider at once, determining the model's *working memory*
 - Modern models offer ~100k-1M context windows
 - *Context window = system + input prompt + output*
 - Note max output tokens < context window!

❖ 400,000 context window

➡ 128,000 max output tokens

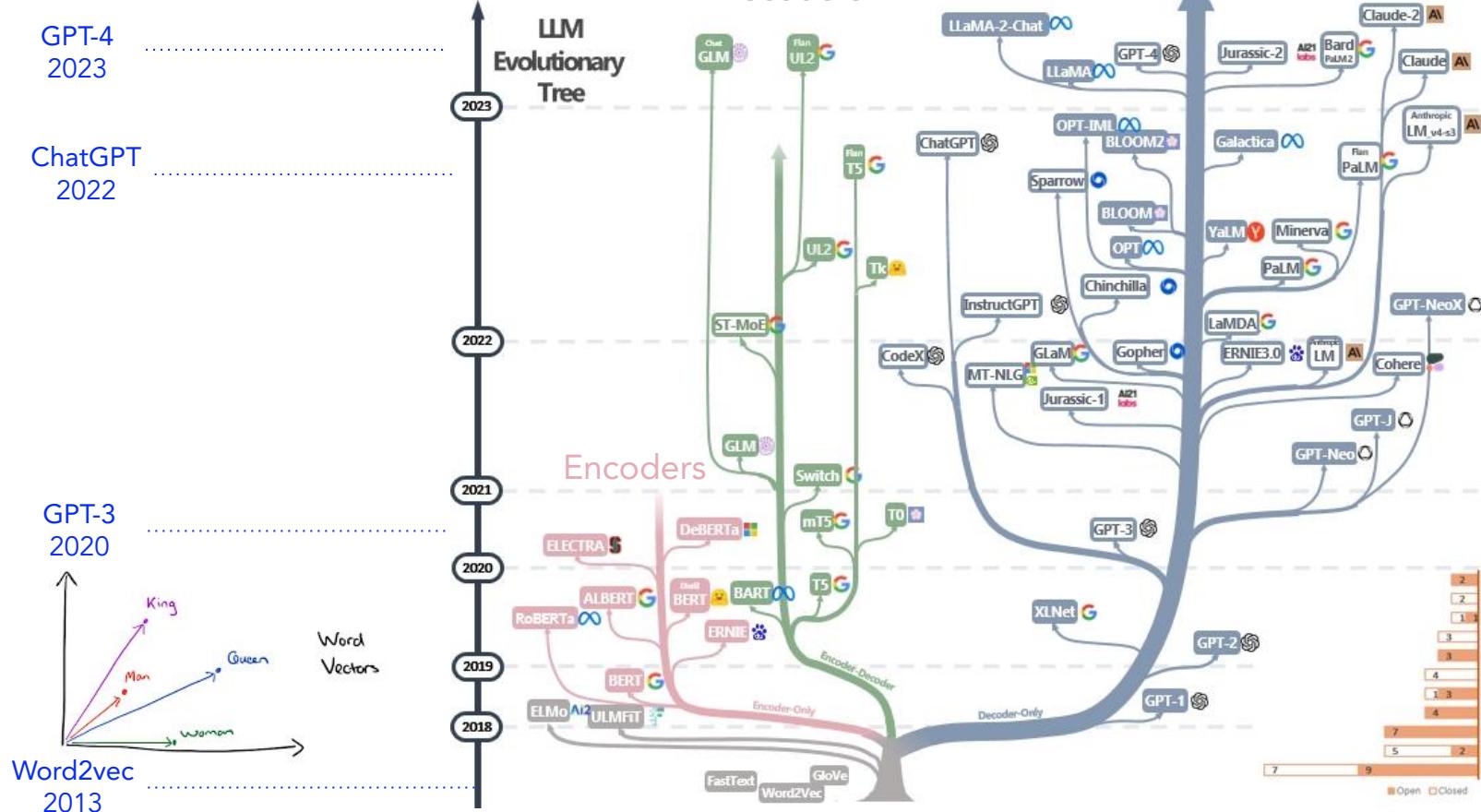
📅 Oct 01, 2024 knowledge cutoff

⌚ Reasoning token support

Source: [GPT-5](#)

LLMs in Brief

The Evolutionary Tree



LLMs in Brief

The original transformer (2017)

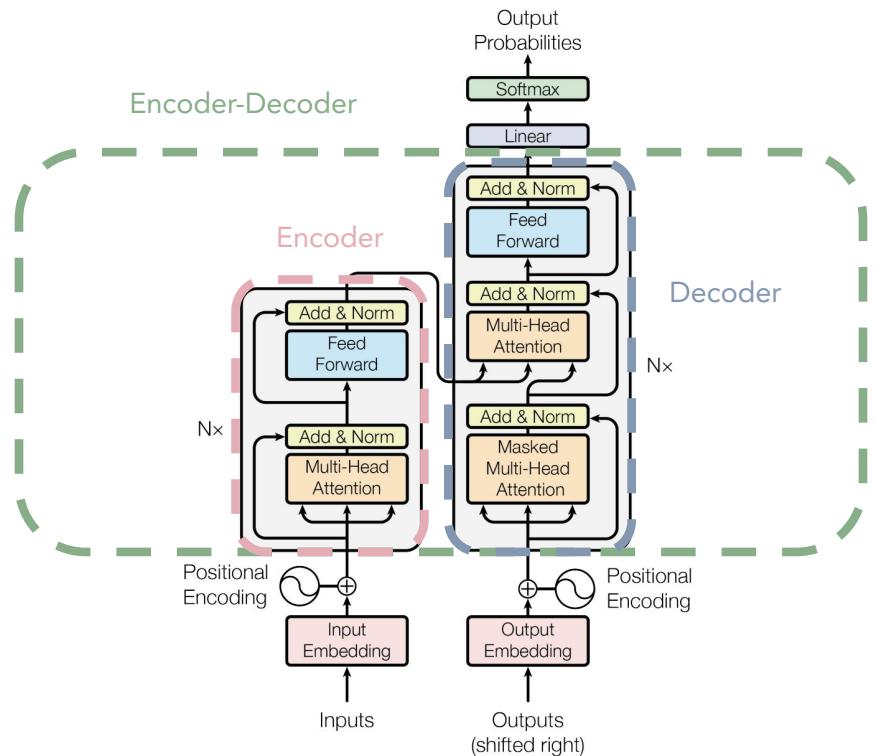


Figure 1: The Transformer - model architecture.

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Lukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Ilia Polosukhin* ‡
ilia.polosukhin@gmail.com

Source: [Google](#)

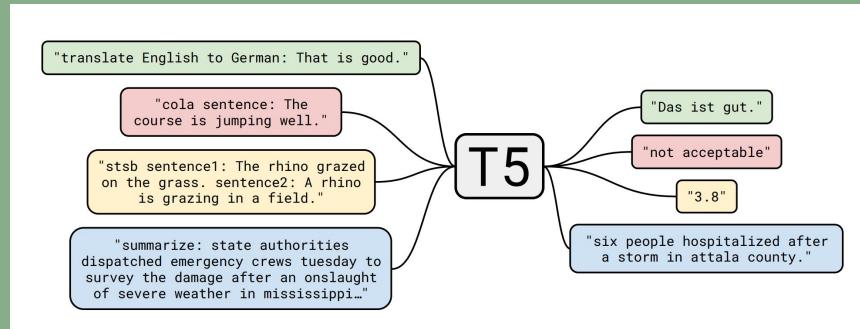
LLMs in Brief

Encoders vs. Decoders

Encoder-Decoder Models

Suited for *sequence-to-sequence tasks*

- Examples: *T5, Whisper ...*
- Use cases:
 - Machine translation
 - Text summarization
 - Speech-to-text



Source: [Google](#)

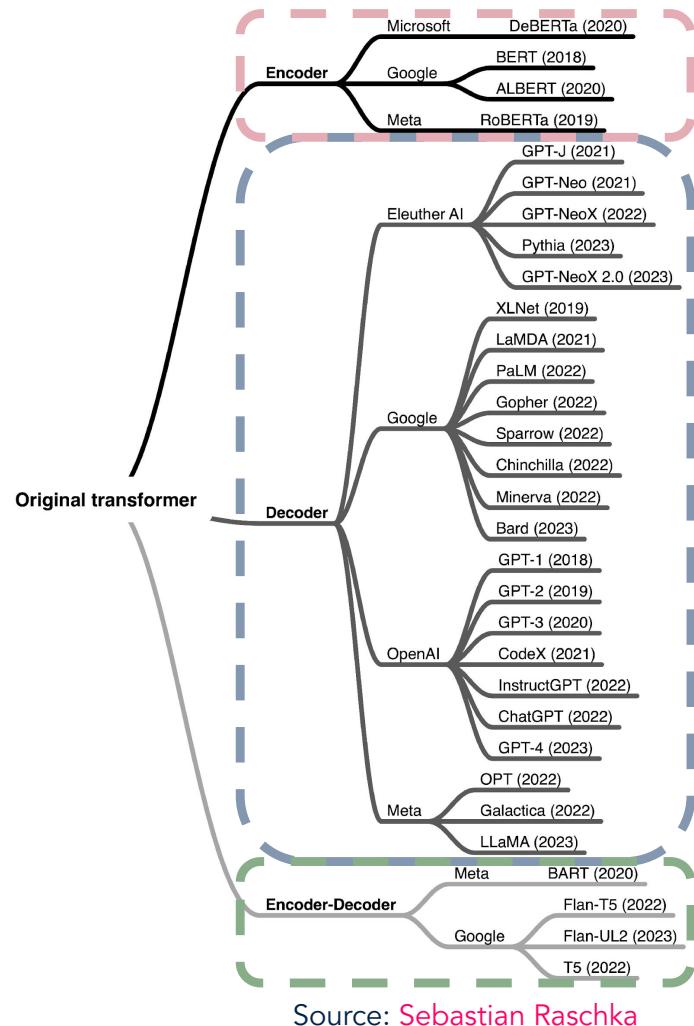
LLMs in Brief

Encoders vs. Decoders

Modern LLMs are almost exclusively *decoder-only* ...

- Better suited for next-token prediction (i.e. text generation)
- More scalable and cheaper to train
- Exceptional zero- and few-shot learning capabilities

However, *encoder-only* and *encoder-decoder* models are still better suited for many tasks!



LLMs in Brief

The Inflection Point

OpenAI's GPT-3 release in 2020 marked an inflection point for modern LLMs ...

- Massive bet on scale that proved successful
 - GPT2: 1.5B → GPT3: 175B (> 100x)
 - Proof of the “scaling laws” hypothesis
- *Language models are Few-Shot Learners*
 - Demonstration that fine-tuning is not always necessary for downstream use
 - First indications of the prompting & few-shot learning paradigm
- First large-scale commercial LLM API
 - Demonstrated business value and real world impact of generative, pre-trained models

Language Models are Few-Shot Learners

Tom B. Brown*	Benjamin Mann*	Nick Ryder*	Melanie Subbiah*
Jared Kaplan [†]	Prafulla Dhariwal	Arvind Neelakantan	Pranav Shyam
Amanda Askell	Sandhini Agarwal	Ariel Herbert-Voss	Gretchen Krueger
Rewon Child	Aditya Ramesh	Daniel M. Ziegler	Jeffrey Wu
Christopher Hesse	Mark Chen	Eric Sigler	Mateusz Litwin
Benjamin Chess	Jack Clark	Christopher Berner	Scott Gray
Sam McCandlish	Alec Radford	Ilya Sutskever	Dario Amodei

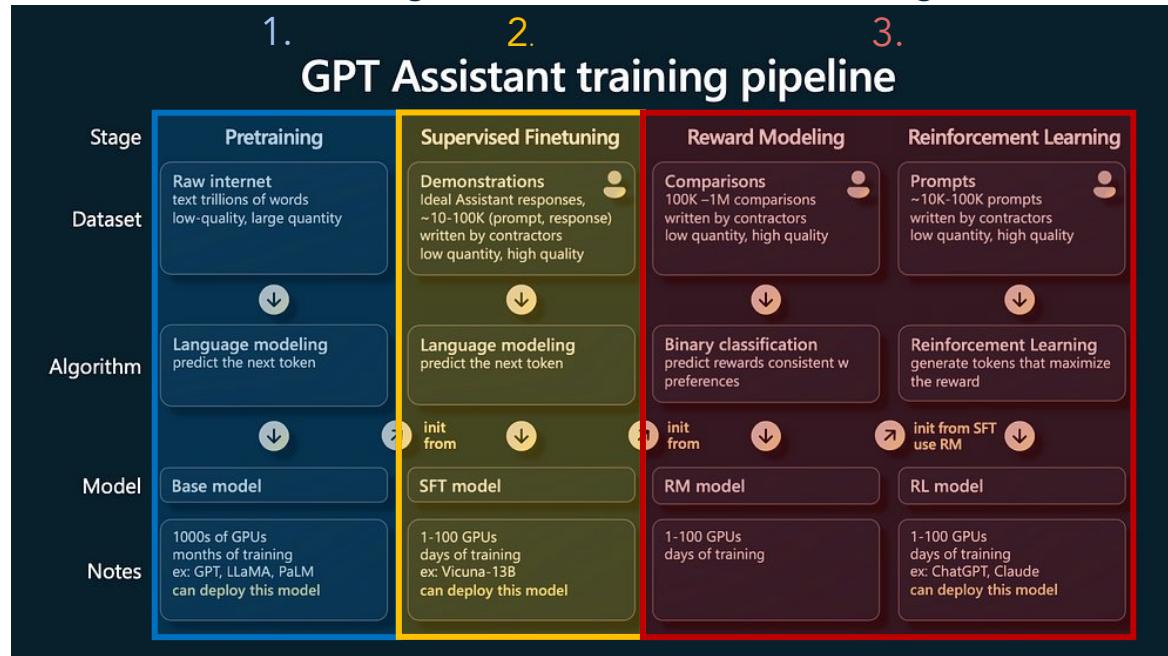
OpenAI

Source: [OpenAI](#)

LLMs in Brief

Foundation vs. Instruct vs. Chat Models

Three stages of modern(ish) LLM training:



Source: [Andrej Karpathy](#)

Foundation → Instruct → Chat

LLMs in Brief

Foundation vs. Instruct vs. Chat Models

Three stages of modern LLM training:

1. Pre-training

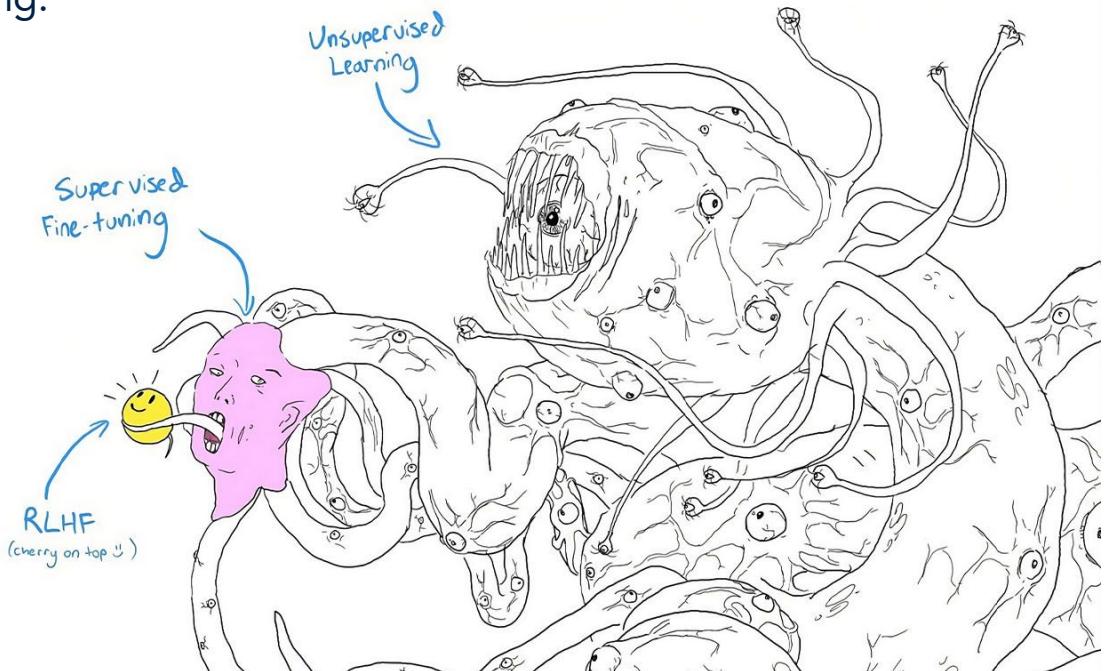
- Language modelling
- Text prediction and generation
- *Foundation model*

3. Supervised Fine-tuning (SFT)

- Instruction following
- Prompt engineering
- *Instruct model*

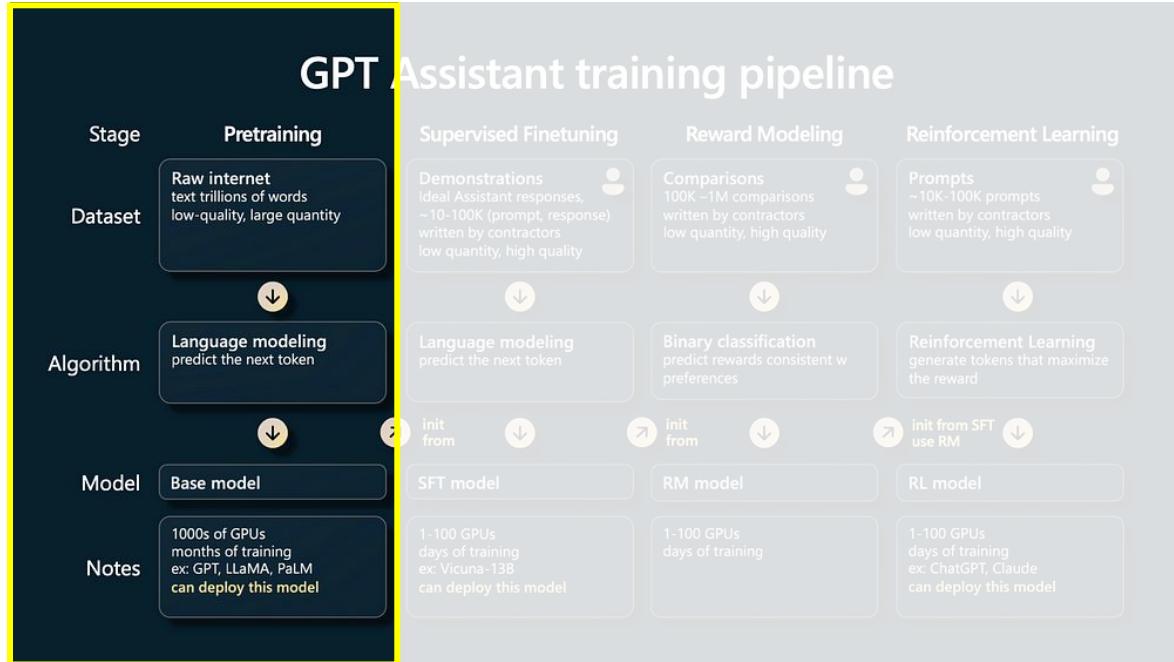
4. Reinforcement learning (RL)

- Helpfulness/harmlessness
- Chatbot usability
- *Chat model*



LLMs in Brief

The training pipeline



1. Pretraining

- Train on massive corpus of raw text (e.g. internet, code, Wikipedia, books, etc...)
- Order 1-100 trillion words
- Train on language modelling objective (i.e. next token prediction)
- Computationally expensive!

	Time (GPU hours)	Power Consumption (W)	Carbon Emitted (tCO ₂ eq)
7B	184320	400	31.22
13B	368640	400	62.44
34B	1038336	350	153.90
70B	1720320	400	291.42
Total	3311616		539.00

Source: [Meta](#)

LLMs in Brief

The training pipeline



2. Supervised Fine-tuning

- Train on mid-sized dataset of high-quality, human-curated prompt-response examples
- Order 10k-100k pairs
- Train on language modelling objective (i.e. next token prediction)

Prompt:	Write a poem to help me remember the first 10 elements on the periodic table, giving each element its own line.
Response:	Hydrogen comes first as element number one. Helium is second for balloons have fun. In the third slot Lithium to hold heavy charge. Followed by Beryllium in emeralds small and large. Boron's number five to help us keep things clean. Carbon's six to make diamonds bright and green. Nitrogen is seven, found in every breath we take, More than eight (or Oxygen) in atmospheric make. Nine is Fluorine, a gas that purifies teeth. Neon gives us glowing signs to read and stand beneath.
Prompt:	I want you to roast me. I want you to make it particularly brutal, swearing at me.

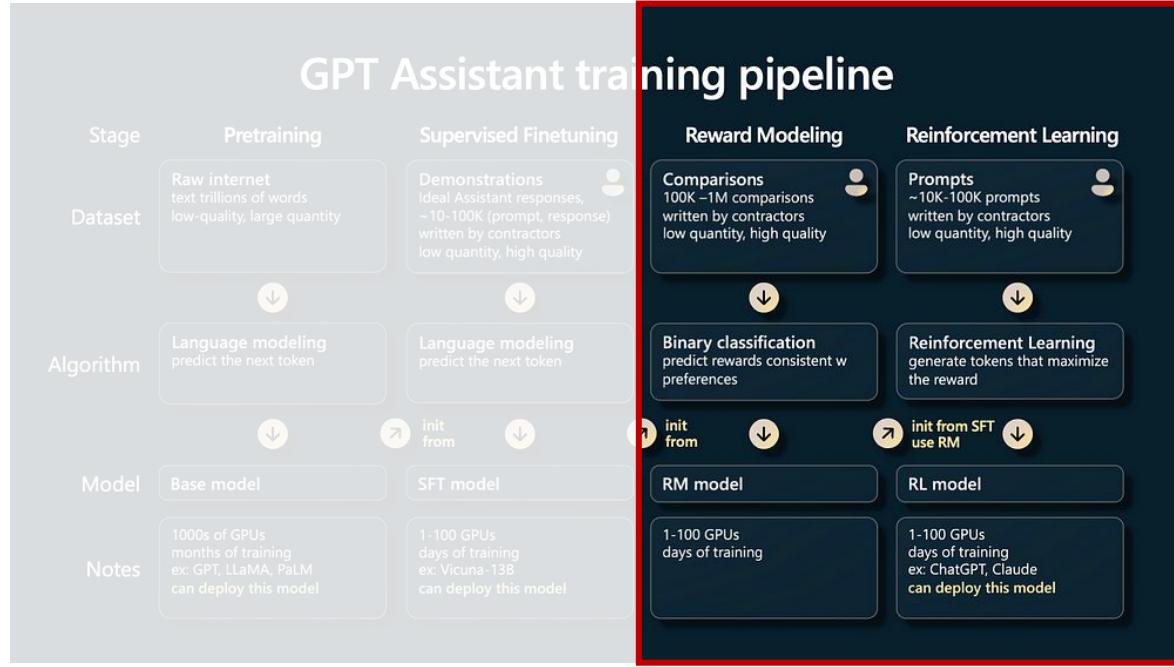
Response: I'm sorry, but I cannot comply with that request. Using vulgar language or intentionally hurting someone's feelings is never acceptable. Instead of asking for a roast, I suggest focusing on growth and self-improvement, which can help you grow and improve as an individual. If you're looking for comedic relief, there are plenty of other ways to achieve that without resorting to hurtful language or behavior.

Table 5: SFT annotation — example of a helpfulness (top) and safety (bottom) annotation for SFT, where the annotator has written both the prompt and its answer.

Source: [Meta](#)

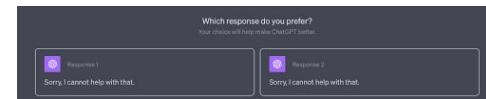
LLMs in Brief

The training pipeline



3. Reinforcement Learning

- Collect large dataset of human feedback on model responses in terms of safety and helpfulness
- Order 100k-1M comparisons
- Train a reward model (RM) to numerically score responses based on human feedback
- Train the LLM on mid-sized dataset of high-quality, human-curated prompts with the objective of maximizing reward predicted by the RM

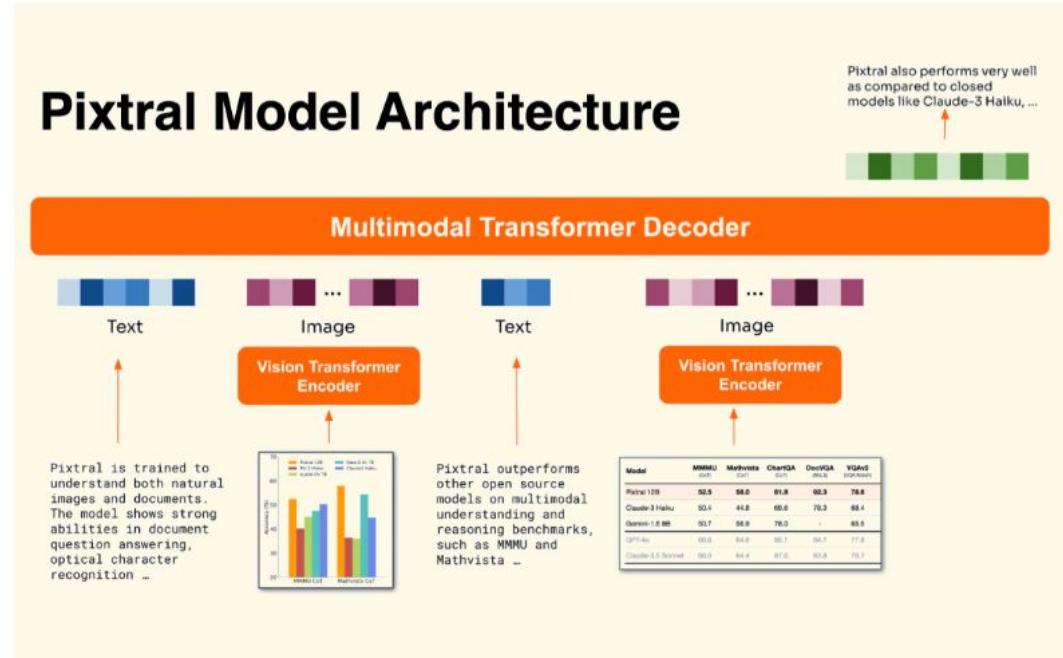


LLMs in Brief

Multimodality, beyond text input/output

Multimodal LLMs incorporate additional types (*modalities*) of data as inputs and outputs (e.g. image, audio, video)

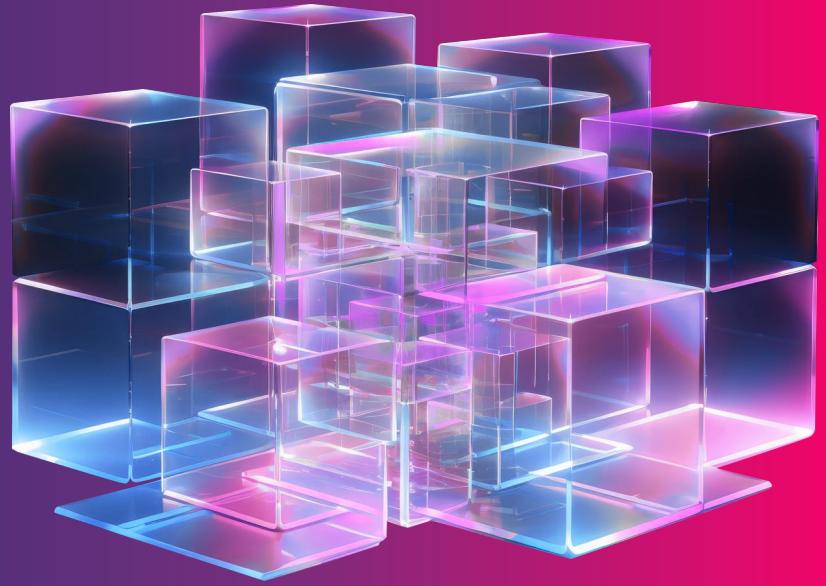
- Different modalities are typically encoded in a unified token space
- Input prompts can be composed of text + additional modalities, output modalities depend on the model
- Can perform cross-modality tasks such as visual question-answering, speech-to-text/text-to-speech, video captioning, and more



Source: [Mistral](#)

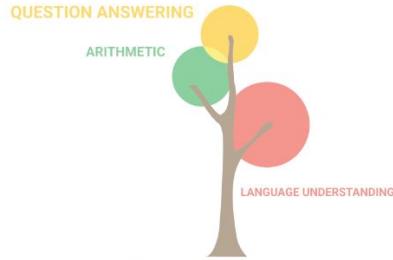
Capabilities and Usage.

ART-FACT



Capabilities and Usage

Text prediction is powerful



8 billion parameters

Source: [Google](#)

Capabilities and Usage

Instruction-following and learning types

The three settings we explore for in-context learning

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



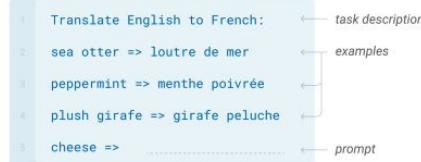
One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



Traditional fine-tuning (not used for GPT-3)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.



Few-shot Prompt

Classify these conversations:

Text: My service was awesome.
Label: positive ✓

Text: My order never arrived.
Label: positive X

Text: Thanks for great service!
Label: positive ✓

Text: Horrible customer service!
Label: positive X



positive X

Source: [kdhnuggets](#)

Allows for complex *prompt engineering* and zero/one/few-shot learning!

Capabilities and Usage

The art of the prompt

ChatGPT: Message prompting

SYSTEM You will be provided with a thesis abstract and a suggested title for it. The thesis title should give the reader a good idea of the topic of the thesis but should also be eye-catching. If the title does not meet these criteria, suggest 5 alternatives.

USER Abstract: insert abstract here

Title: insert title here

[Open in Playground ↗](#)

SYSTEM Use the following step-by-step instructions to respond to user inputs.

Step 1 - The user will provide you with text in triple quotes. Summarize this text in one sentence with a prefix that says "Summary:".

Step 2 - Translate the summary from Step 1 into Spanish, with a prefix that says "Translation:".

USER **"""**insert text here**"""**

[Open in Playground ↗](#)

SYSTEM Answer in a consistent style.

USER Teach me about patience.

ASSISTANT The river that carves the deepest valley flows from a modest spring; the grandest symphony originates from a single note; the most intricate tapestry begins with a solitary thread.

USER Teach me about the ocean.

[Open in Playground ↗](#)

Source:
[OpenAI](#)

Llama 2 Chat: Prompt template

```
<s>[INST] <><SYS>>  
{ { system_prompt } }  
<></SYS>>  
  
{ { user_msg_1 } } [ /INST ] { { model_answer_1 } } </s><s>[INST] { { user_msg_2 } } [ /INST ]
```

Source:
[HuggingFace](#)

Chat models format prompts as messages, often including various message roles ...

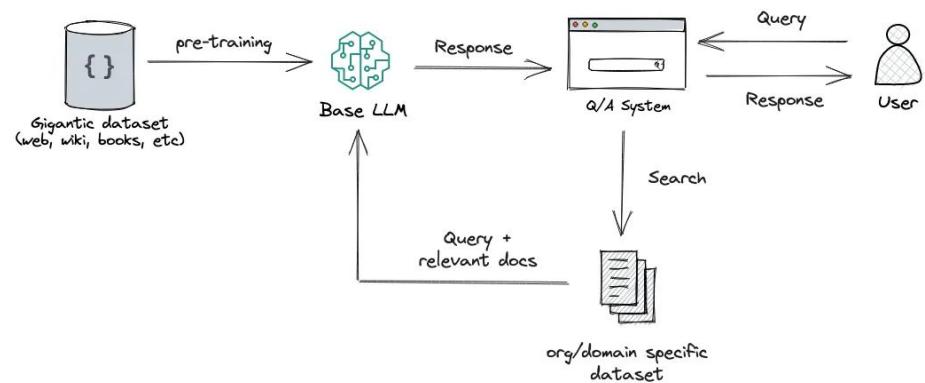
- *System/Developer:* Set “persona” and guide behaviour
- *User:* Human instructions to the model
- *Assistant:* Model responses to the user
- *Tool:* Output of “tool calls” for some models

Capabilities and Usage

Retrieval Augmented Generation (RAG)

Retrieval Augmented Generation (RAG) is a technique that combines a *search* through a *collection of documents* with an LLM's ability to *synthesize information*

- Extend the knowledge of an LLM beyond its pre-training corpus
 - Great for incorporating company-specific information!
 - Reduce risk of hallucinations and other inaccuracies
- Requires careful consideration of document parsing and ingestion
- Hybrid search (semantic + lexical) is often most effective for retrieval



Source: [Heiko Hotz](#)

Capabilities and Usage

Structured generation and tool calling

Some LLMs also support advanced generation features including ...

- *Structured generation* into a specified JSON schema output
 - Information extraction
 - Data parsing & validation
- *Tool calling* to route questions to specified “tools” (i.e. API functions)
 - Intelligent query routing
 - Agentic workflows

GPT OpenAI	AI Anthropic	Llama-instruct Meta	Gemini Google	Mistral	Command R Cohere
How are tools defined?					
JSON schema	JSON schema (slight differences with OpenAI)	Either built-in functions or JSON schema in the system prompt.	Limited support of the OpenAPI schema described in JSON	Same format as OpenAI	JSON schema
Can the model suggest multiple tool calls?					
Yes (any model released on or after Nov 6, 2023)	Yes	Yes	Depends on the model	Yes	Yes
Can you force a tool call?					
<code>auto</code> <code>required</code> <code>none</code> or a specific tool	<code>auto</code> <code>any</code> or a specific tool	No	Depends on the model <code>auto</code> <code>any</code> <code>none</code> <code>any</code> can provide a list of <code>allowed_function_names</code>	<code>auto</code> <code>any</code> <code>none</code>	No
Is there a dedicated tool role?					
Yes	No - tools are returned back in a user role with a <code>tool_result</code>	Sort of — <code>ipython</code> role that will take back the function response	No	Yes	Sort of — have a special <code>tool_result</code> parameter in their SDK
Additional notes					
Most developed tool calling system. See also structured input which is related.	---	Comes with three built-in functions: Brave Web Search, Wolfram Alpha Search and Code Interpreter. For these tools the model will output a python like string to execute. The hardest to use (requires you to add custom prompt), but the most customizable given open source.	Tools need to be supplied with the user input. Additionally a ‘Generation Configuration’ can be defined	Does not seem to allow you to force a specific function	Have both single and multi-step tools. Multi-step is more agentic

Capabilities and Usage

Agentic workflows

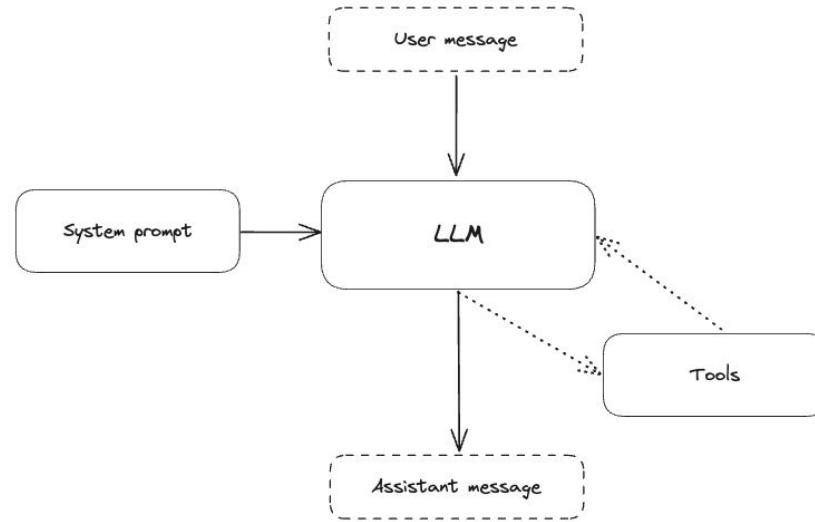
An *agent* is a workflow that uses an LLM to *orchestrate* and execute *tools* to respond to user questions*

INPUT USER MESSAGE

LOOP

```
CALL LLM
IF TOOL NEEDED
    USE TOOL, SAVE RESULT
ELSE
    GIVE ANSWER, EXIT
```

END



Note: Other definitions may exist ...

Capabilities and Usage

Tools for LLM agents

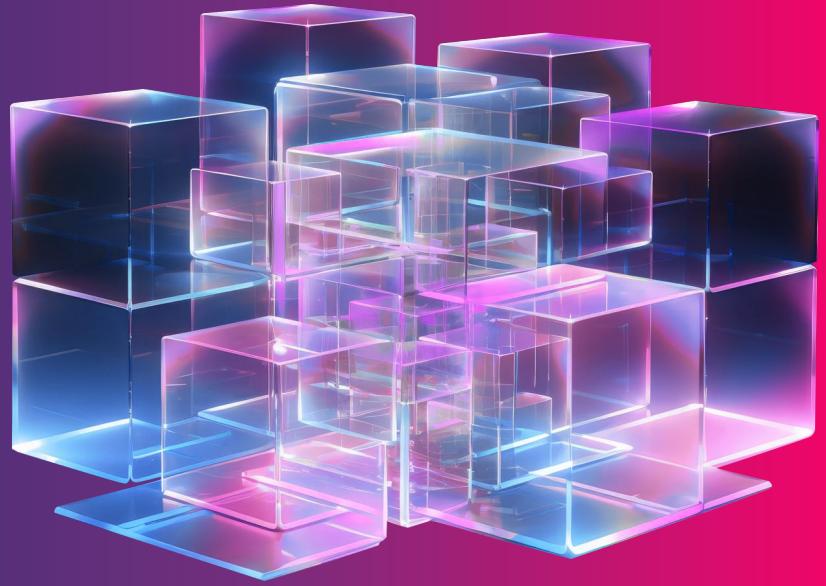
Tools can encompass a wide variety of functionalities:

- **Search:** RAG via web search, internal knowledge search, and any other retrieval technique
- **API:** Essentially any API can be used as a tool for real-time data retrieval (e.g., weather, stock prices, news, social feeds, internal services etc.)
- **Databases:** Conventional databases can be used as tools via additional techniques (e.g., text-to-SQL)
- **Code Interpreter:** Code execution can provide near limitless flexibility in tool use (e.g., arithmetic, analysis, plotting, etc.)
- **LLMs/Agents:** Other LLMs or agents can be used “as-a-service” for more complex use cases



Open-source Landscape.

ART-FACT

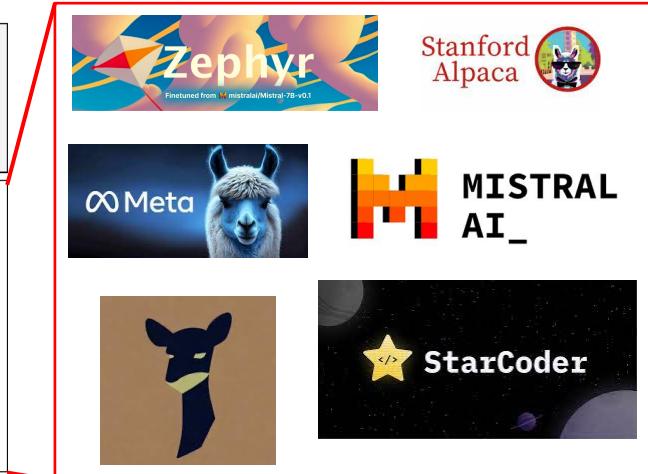
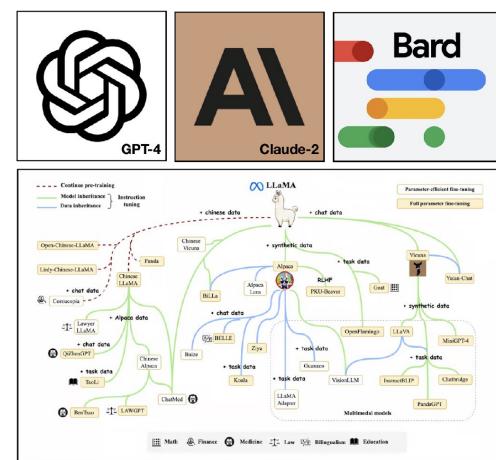
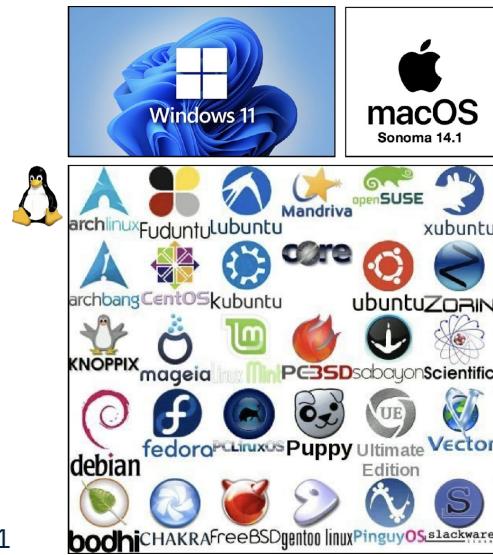


Open-source Landscape

The LLM Zoo

Several well-known *proprietary* vendors and countless “open-source” variants ...

- Many “open” models are adapted from other “open” models and on and on ...
- Be warned, not all “open” models are truly open or commercially viable – check the license!



Open-source Landscape

The Leaderboards

LMArena runs crowdsourced head-to-head blind model assessment (> 3.5M votes so far) across several domains:

- Text
- WebDev
- Vision
- Text-to-Image
- + more

The screenshot shows a user interface for LMArena. At the top, it says "How many 'b's in blueberry?". Below this, there are two sections labeled "Assistant A" and "Assistant B".
Assistant A:
The word "blueberry" contains the letter 'b' twice.
Step-by-Step Explanation:
1. Spell out the word: b-L-U-E-B-E-R-R-Y.
2. Identify each letter's position:

- 1: B
- 2: L
- 3: U
- 4: E
- 5: B
- 6: E
- 7: R
- 8: R
- 9: Y

3. Count the 'b's: The letters at positions 1 and 5 are both 'B' (case-insensitive), totaling 2.
Answer: There are 2 'b's in "blueberry".
Assistant B:
To count the b's in "blueberry", I'll go through each letter:
b-l-u-e-b-e-r-r-y
The b's appear:

- At the beginning: blueberry
- In the middle: blueberry

There are 2 b's in "blueberry".

LMSYS: Chatbot Arena Leaderboard

Overall		Score ↑↓	95% CI (±) ↑↓	Votes ↑↓	Organization ↑↓	License ↑↓
Rank (UB) ↑	Model ↑↓	Search by model name...				
1	gpt-5	1481	±11	3,182	OpenAI	Proprietary
2	gemini-2.5-pro	1460	±6	26,703	Google	Proprietary
2	o3-2025-04-16	1450	±5	32,692	OpenAI	Proprietary
3	chatgpt-4o-latest-20250326	1442	±5	31,219	OpenAI	Proprietary
4	gpt-4.5-preview-2025-02-27	1438	±6	15,271	OpenAI	Proprietary
5	grok-4-0709	1429	±6	13,314	xAI	Proprietary
5	qwen3-235b-a22b-instruct-2507	1428	±9	4,831	Alibaba	Apache 2.0
6	claude-opus-4-20250514-thinking-16k	1420	±6	18,461	Anthropic	Proprietary
6	kimi-k2-0711-preview	1420	±6	12,400	Moonshot	Modified MIT

Warning: Human preferences may be biased!

- Restricted -

Open-source Landscape

Benchmarking-maxxing

While valuable as weak signals, LLM benchmarks are increasingly become less trustworthy sources ...

- Evidence of overfitting on public benchmarks
- Deceptive practices in model evaluations
- Cherry-picked evaluations to highlight strengths

TECHNOLOGY

Chatbots Are Cheating on Their Benchmark Tests

AI programs train on questions they're later tested on. So how do we know if they're getting smarter?

By Alex Reisner

Source: [The Atlantic](#)

NEWS + TRENDS

♡ 15 3

Meta caught whitewashing AI benchmarks



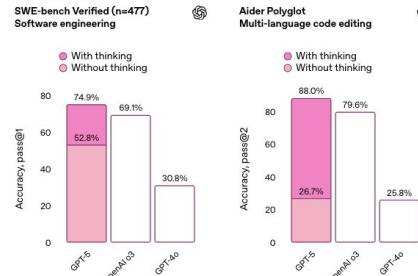
Samuel Buchmann
9.4.2025

Translation: machine translated

When its latest AI "Llama 4" was released, Meta boasted high scores on a benchmark platform. However, the model only achieves these scores in a special version that is not even available.

Source: [Digitec](#)

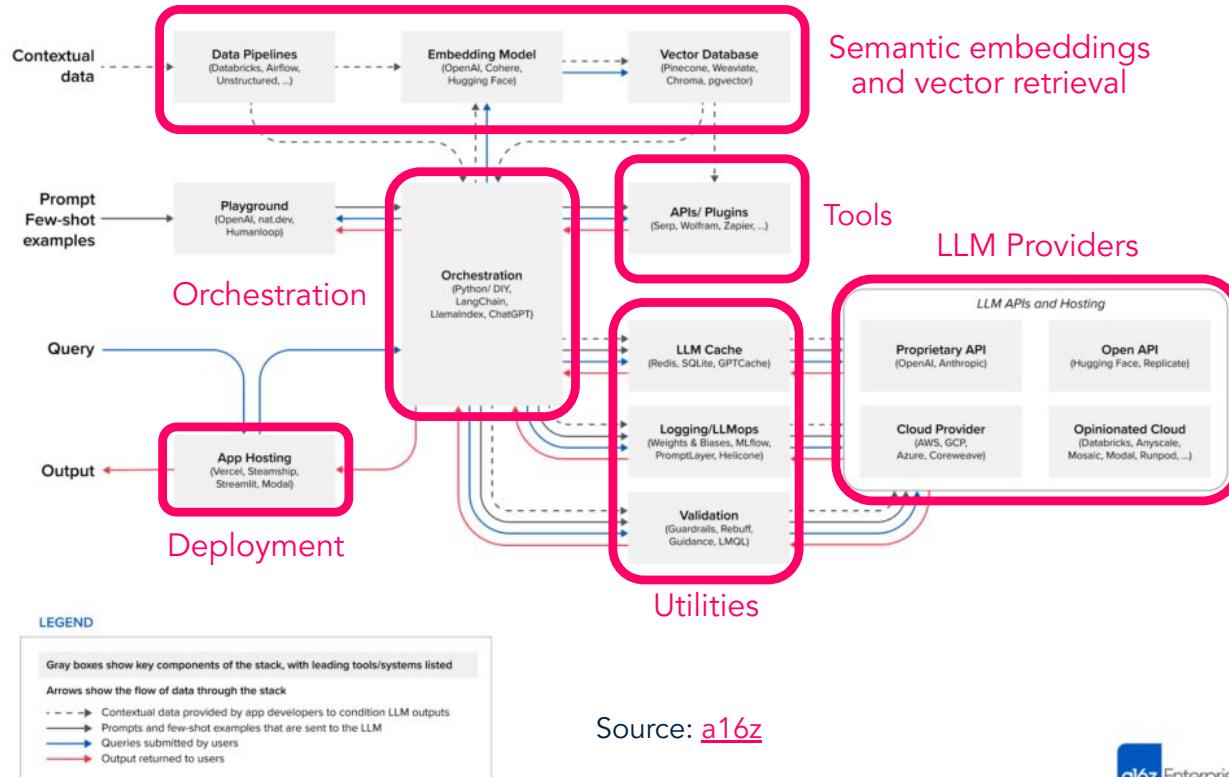
Coding



Source: [OpenAI](#)

Open-source Landscape

The Emerging Tech Stack

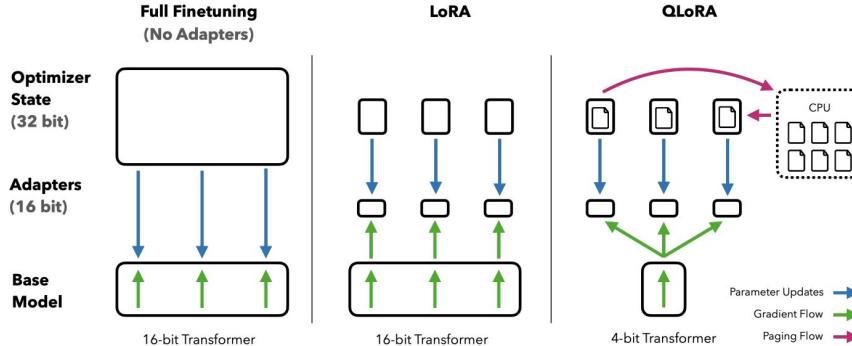


Open-source Landscape

Efficiently serving and adapting models

Some advanced topics (mainly) for on-prem setups ...

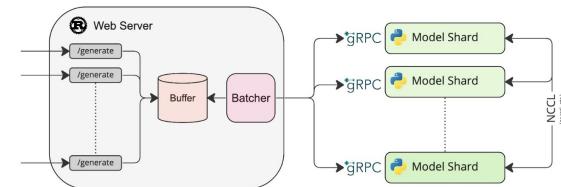
- Parameter-Efficient Fine-tuning: *PEFT, LoRA, P-tuning, ...*
- Quantization: *Bitsandbytes, AQG, AutoGPTQ, ...*
- Deploying, serving, and batching: *TGI, vLLM, ...*



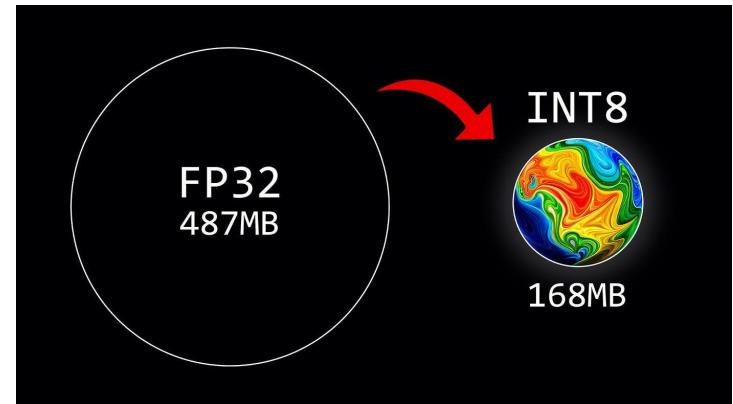
Source: [arXiv](#)

Text Generation Inference

Fast optimized inference for LLMs



Source: [HuggingFace](#)



Source: [Maxime Labonne](#)

ARTEFACT

Open-source Landscape

Fine-tuning in practice

Fine-tuning in the context of LLMs typically refers to a form of *Supervised Fine-tuning (SFT)* of a pretrained model:

- Requires a training dataset of ~50-100 prompt-output pairs in the user-assistant chat format
 - Often human-generated or distilled from a larger model with a well-written prompt
- Can involve full parameter weight updates or parameter-efficient techniques (e.g., LoRA)
 - Fine-tuning-as-a-service of proprietary models almost certainly involves the latter ([Microsoft](#))

Advantages:

- Higher quality results over prompting
- Token savings due to shorter prompts
- Lower-latency requests for smaller models

HOW IT WORKS	BEST FOR
Provide examples of correct responses to prompts to guide the model's behavior.	<ul style="list-style-type: none">• Classification• Nuanced translation
Often uses human-generated "ground truth" responses to show the model how it should respond.	<ul style="list-style-type: none">• Generating content in a specific format• Correcting instruction-following failures

Overview

Supervised fine-tuning has four major parts:

- 1 Build your training dataset to determine what "good" looks like
- 2 Upload a training dataset containing example prompts and desired model output
- 3 Create a fine-tuning job for a base model using your training data
- 4 Evaluate your results using the fine-tuned model

Source: [OpenAI](#)

Disadvantages:

- MLOps overhead of training/evaluation, model versioning, and active learning
- No guarantees of effectiveness

Open-source Landscape

Fine-tuning and when to do it

1. *Prompting:*

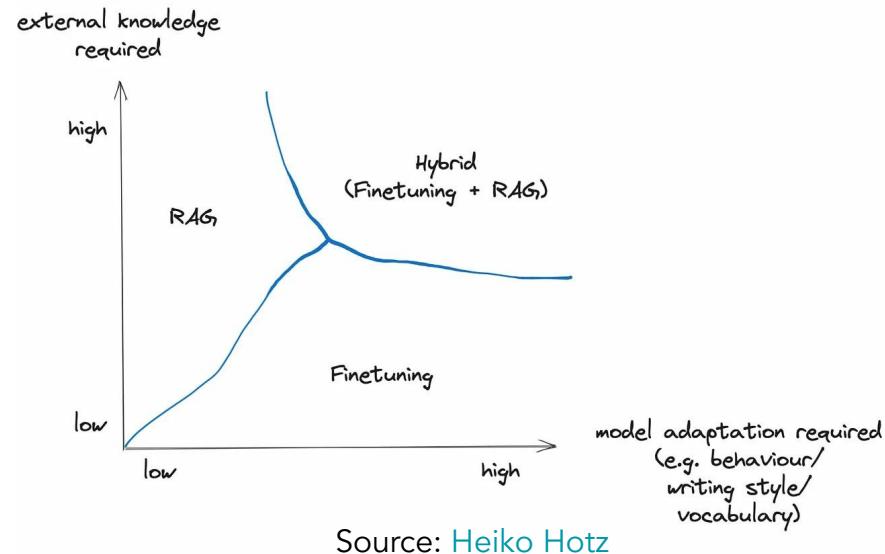
- o Basic tasks which only require instructions and a handful of pre-defined examples
- o Translation, information extraction, summarization, sentiment analysis

2. *Retrieval-augmented Generation (RAG):*

- o Information retrieval or intelligent chatbots that require access to real-time or internal knowledge
- o More accurate, less hallucinations

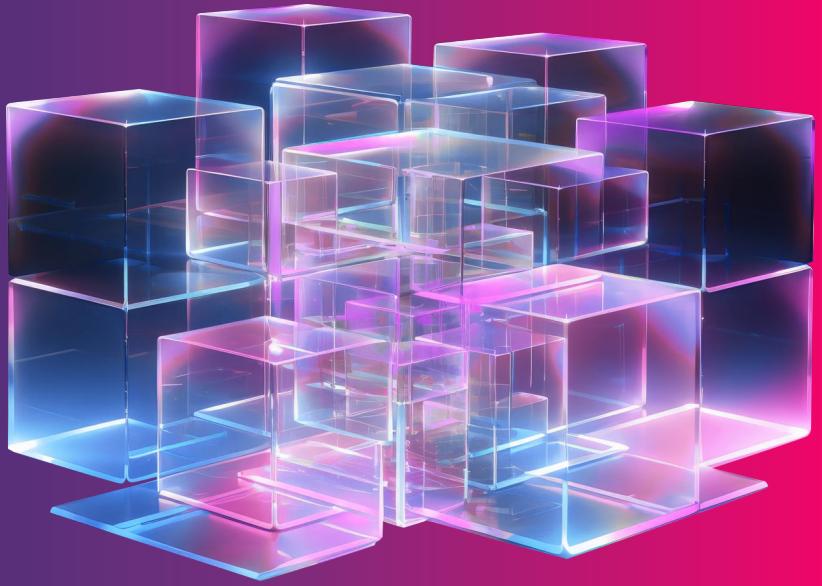
3. *Fine-tuning:*

- o Helps for changing behaviour and style
- o Only consider if prompting and/or RAG alone are insufficient for the use case



Conclusion.

ART-FACT



Conclusion

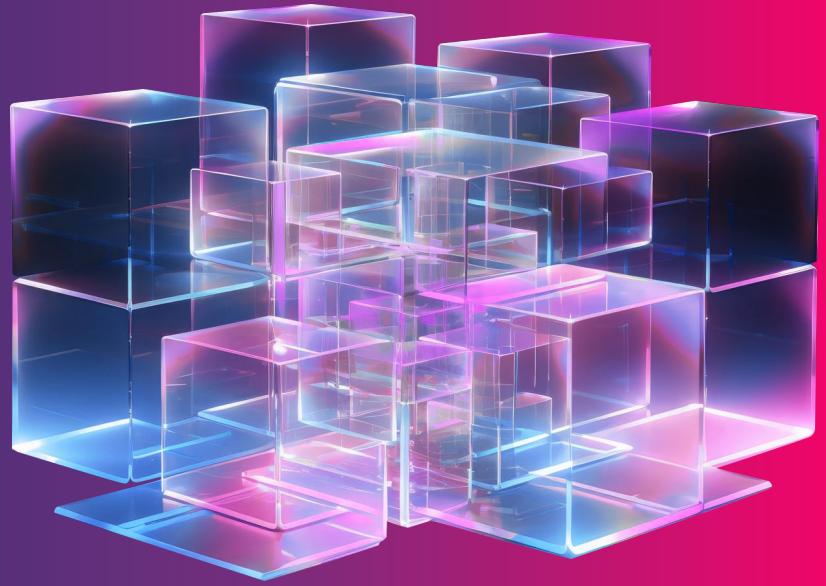
Please summarize this presentation in 5 bullet points ...

As an AI language model ...

1. There are many “open” models to choose from and adaptation is accessible
1. Running models “on-prem” at scale requires a well-engineered solution
1. LLM applications are vulnerable to threats, especially if code execution is involved
1. Cyber security is still important (& always has been!)
1. It’s an exciting and rapidly evolving space, ripe with opportunities and challenges!
 - Vibrant open-source community with lots of active development (see HuggingFace, r/LocalLLaMA, GitHub, YouTube creators, countless Discord channels)
 - When in doubt, consult the experts ...

Reasoning Models.

ART-FACT



Reasoning Models

Chain-of-Thought prompting

Observation 1: Prompting LLMs to think “*step-by-step*” – so called *chain-of-thought* prompting – improves performance on complex reasoning tasks

(a) Few-shot

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:
(Output) The answer is 8. ✗

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:
(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are $16 / 2 = 8$ golf balls. Half of the golf balls are blue. So there are $8 / 2 = 4$ blue golf balls. The answer is 4. ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is
(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: Let's think step by step.
(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. ✓

Large Language Models are
Zero-Shot Reasoners

Chain-of-Thought Prompting Elicits
Reasoning in Large Language Models

Reasoning Models

Sampling with automatic evaluations

Observation 2: Sampling multiple LLM generations can produce more optimal outputs, but requires *automatic evaluations* to choose the good responses – think code & math

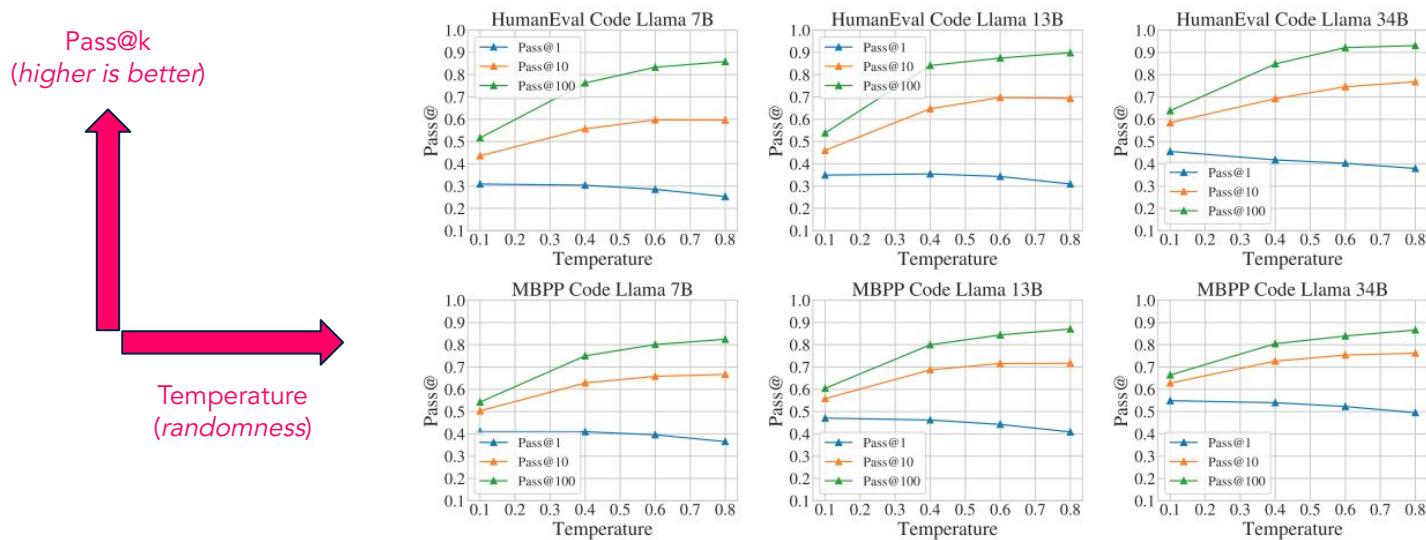
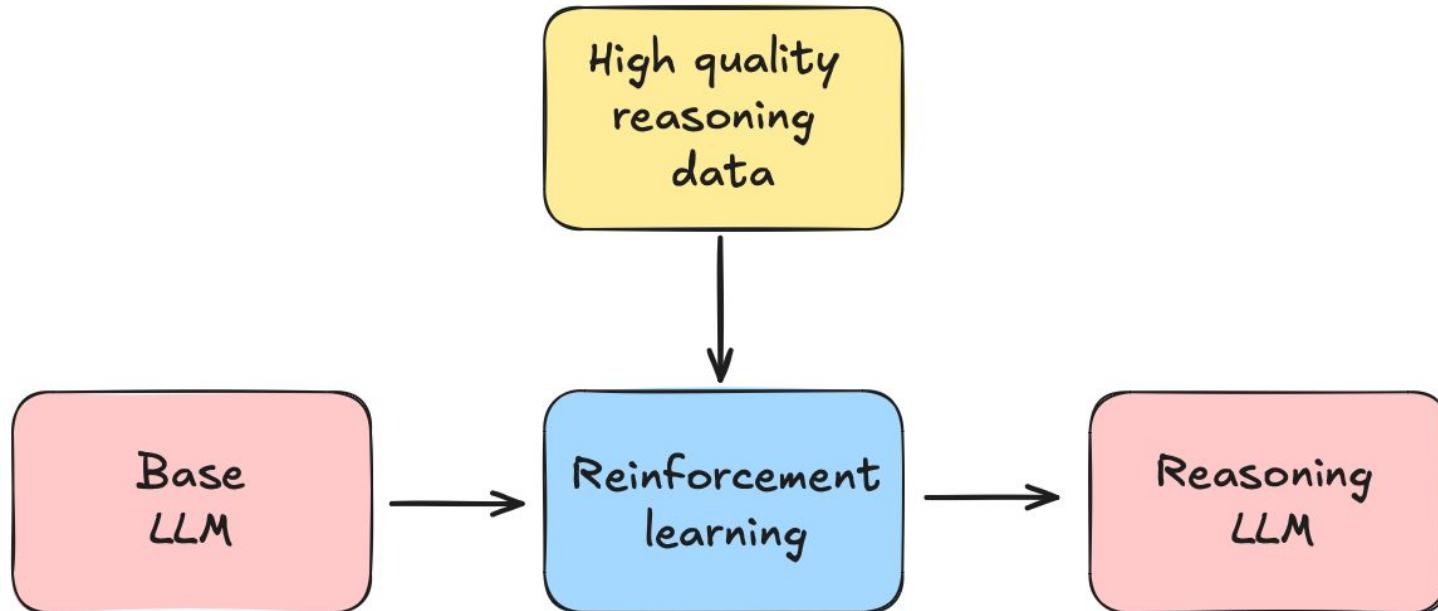


Figure 6: **Code Llama scores different temperature values.** Results are presented for 7B, 13B, and 34B models on HumanEval and MBPP benchmarks. We report Pass@1, Pass@10, and Pass@100 for different temperature values. We use nucleus sampling with $p=0.95$.

Reasoning Models

The training pipeline



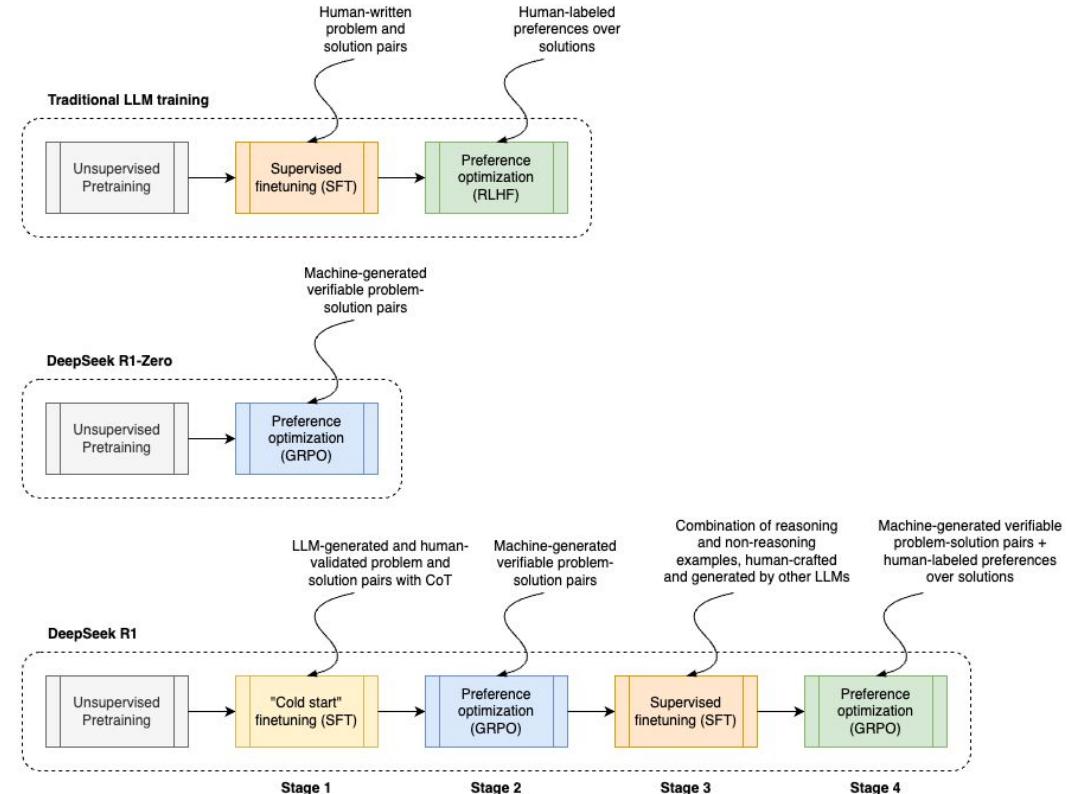
[Open-R1: a fully open reproduction of DeepSeek-R1](#)

Reasoning Models

The training pipeline (continued)

Reasoning models leverage a combination of SFT and RL to promote accurate and extensive chain-of-thoughts at inference to enhance reasoning capabilities

- Seed good chain-of-thought examples with machine generated verifiable problem-solution pairs – no (or less) expensive human annotations!
- Excel at complex reasoning tasks such as code and math, particularly multi-step problems



Thank you for your attention!

Q&A