Language modeling

Slides are adopted from Advanced NLP course at UMass.

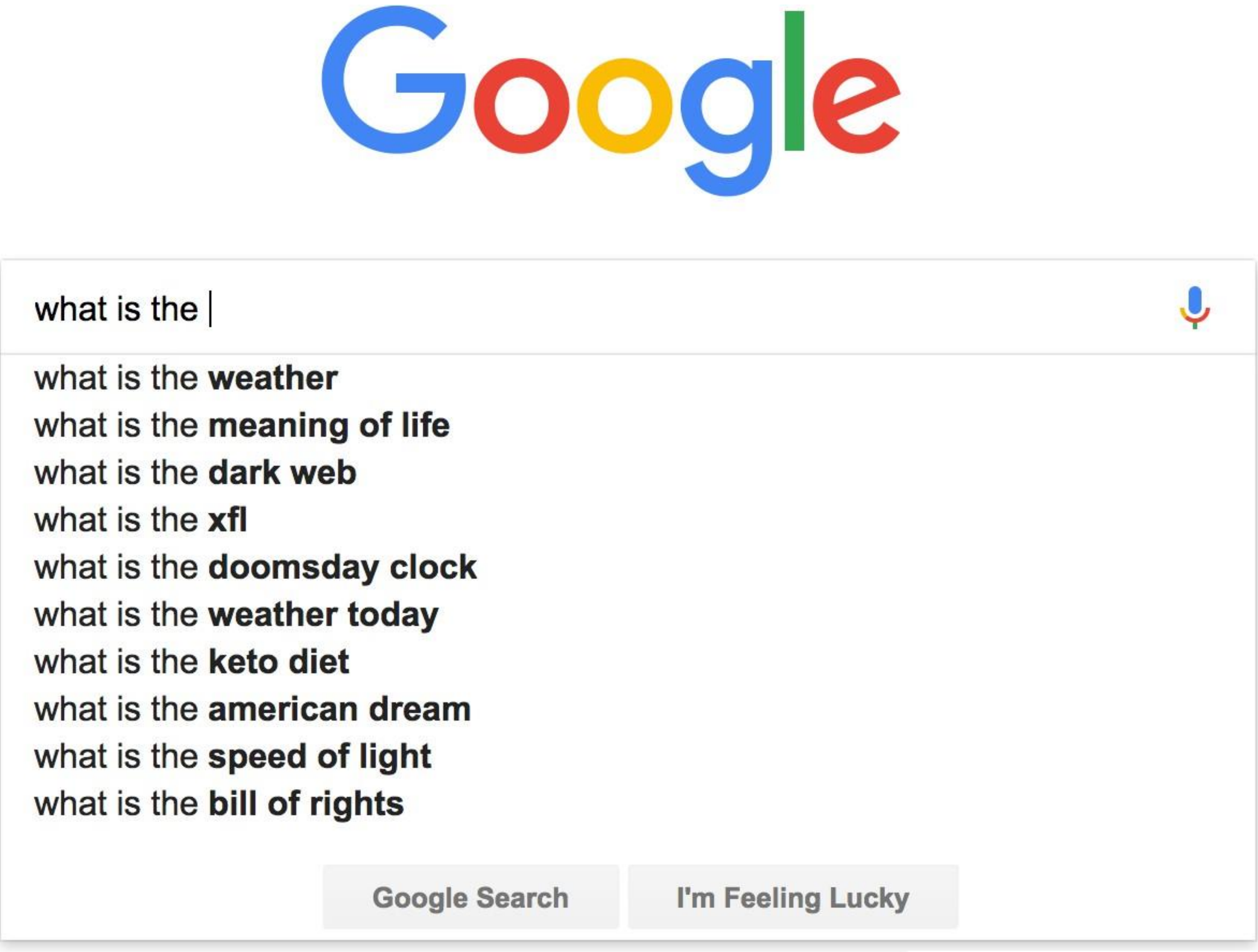
Intro

- Language models assign a probability to a piece of text.
- Why would we ever want to do this?
  - Translation:

    P(i flew to the movies) **<<<<<** P(i went to the movies)
  - Speech recognition:

    P(i saw a van) **>>>>>** P(eyes awe of an)

You use Language Models every day!



Probabilistic Language Modeling

- Goal: compute the probability of a sentence or sequence of words:
  - $P(W) = P(w_1, w_2, w_3, w_4, w_5, \ldots, w_n)$
- Related task: probability of an upcoming word:
  - $P(w_5|w_1, w_2, w_3, w_4)$
- A model that computes either of these:
  - $P(W)$ or $P(wn|w_1, w_2, \ldots, w_{n-1})$ is called a *language model* or LM

How to compute P(W)

- How to compute this joint probability:
  - P(its, water, is, so, transparent, that)
- Intuition: let's rely on the Chain Rule of Probability

Reminder: The Chain Rule

- Recall the definition of conditional probabilities

  $P(B|A) = P(A,B)/P(A)$ Rewriting: $P(A,B) = P(A)P(B|A)$

- More variables:

$$P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$$

- The Chain Rule in General

$$P(x_1, x_2, x_3, \dots, x_n)$$
$$= P(x_1)P(x_2|x_1)P(x_3|x_1,x_2) \dots P(xn|x_1, \dots, xn\ \_\ _1)$$

Chain Rule for computing joint probability

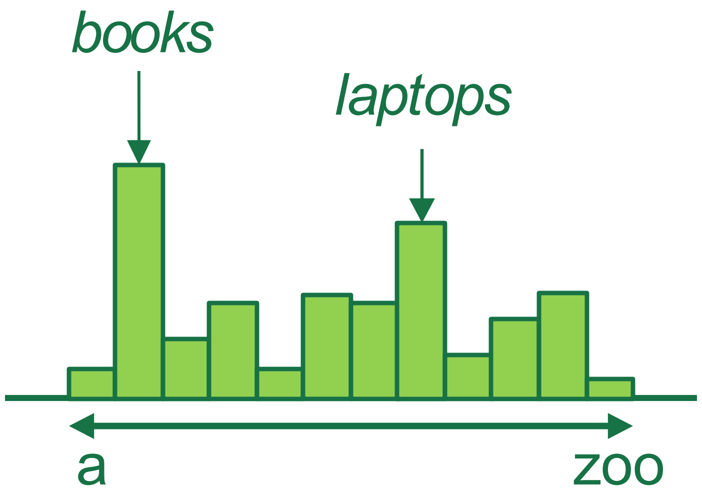
- The Chain Rule applied to compute joint probability of words in sentence

$$P(w_1 w_2 \ldots w_n) = \prod_i P(w_i \mid \overbrace{w_1 w_2 \ldots w_{i-1}}^{\text{prefix}})$$

P("its water is so transparent") =
P(its) × P(water|its) × P(is|its water)
× P(so|its water is) × P(transparent|its water is so)

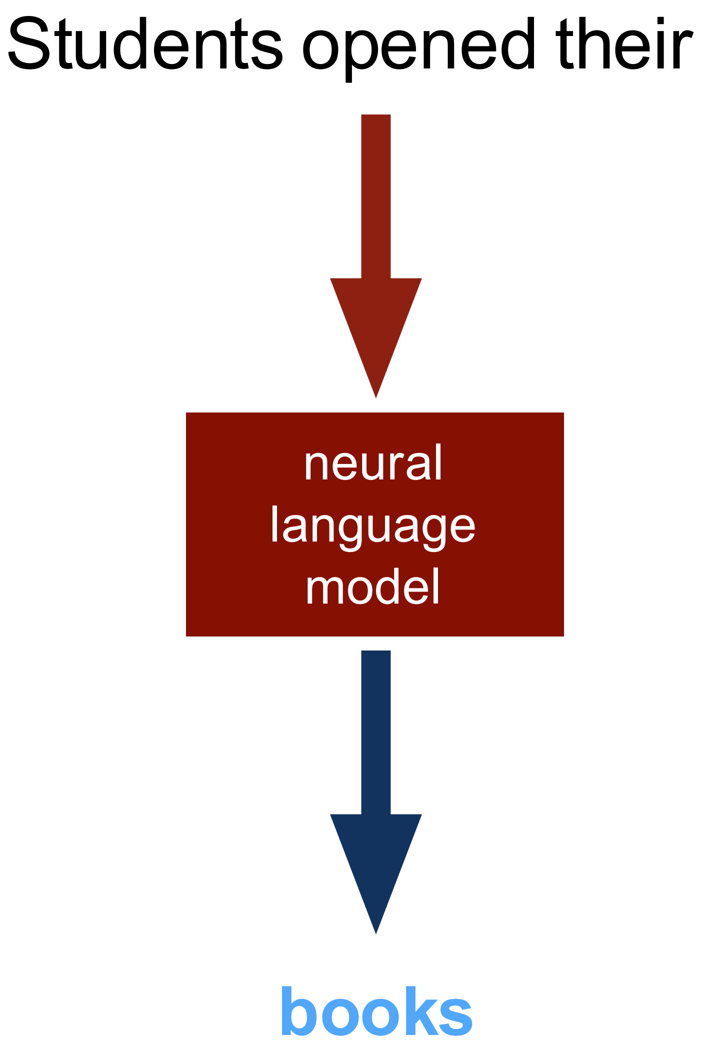
Decoding from an LM

- **Prefix**: “students opened their”



Probability distribution over next word

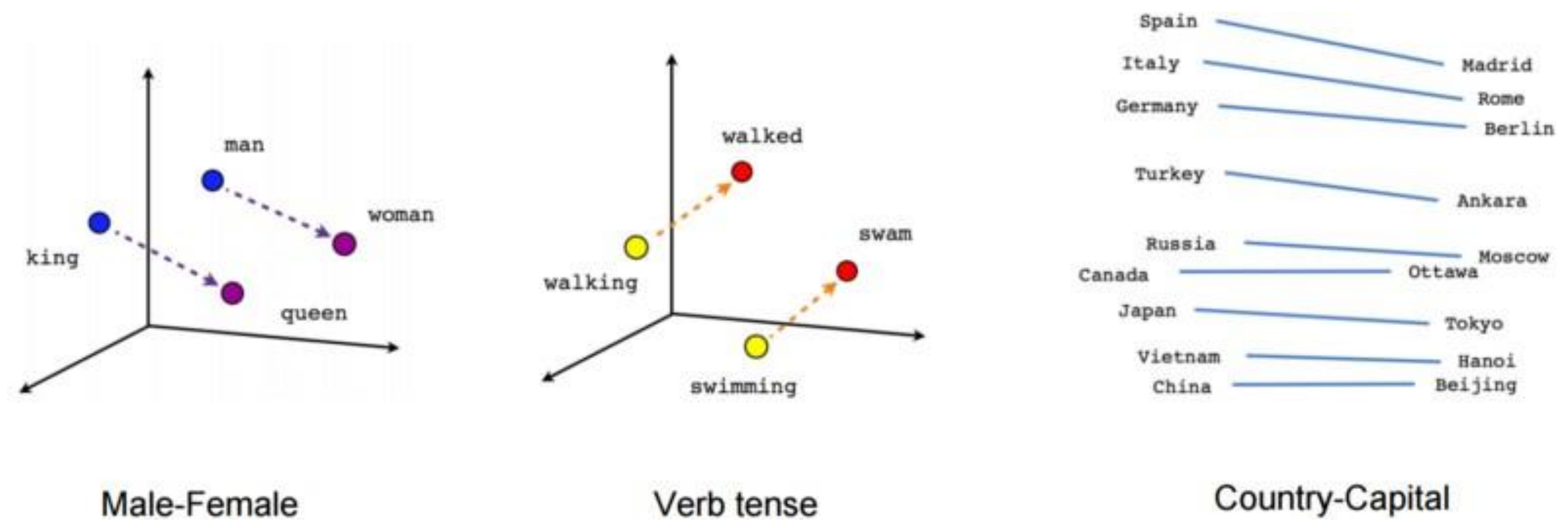
Enter neural networks!



Words as basic building blocks

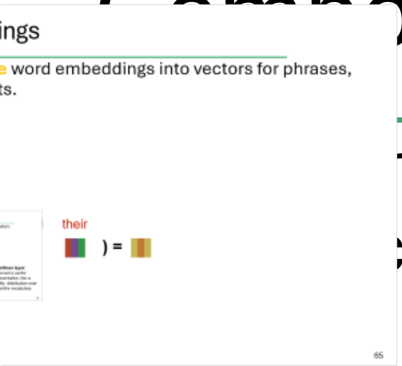
- Represent words with low-dimensional vectors called embeddings (Mikolov et al., NIPS 2013)

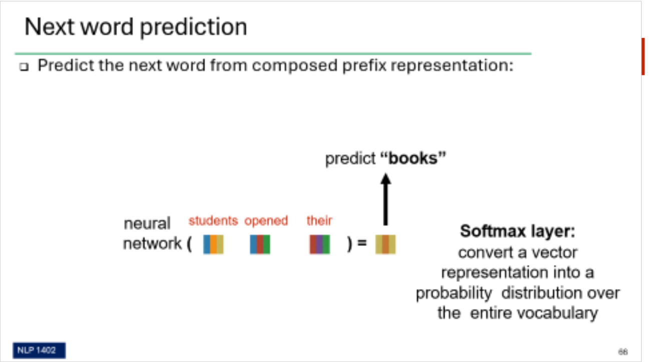
king = [0.23, 1.3, -0.3, 0.43]



Composing embeddings

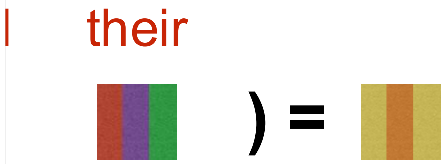
networks **compose** word embeddings into vectors for phrases,
ces, and documents.





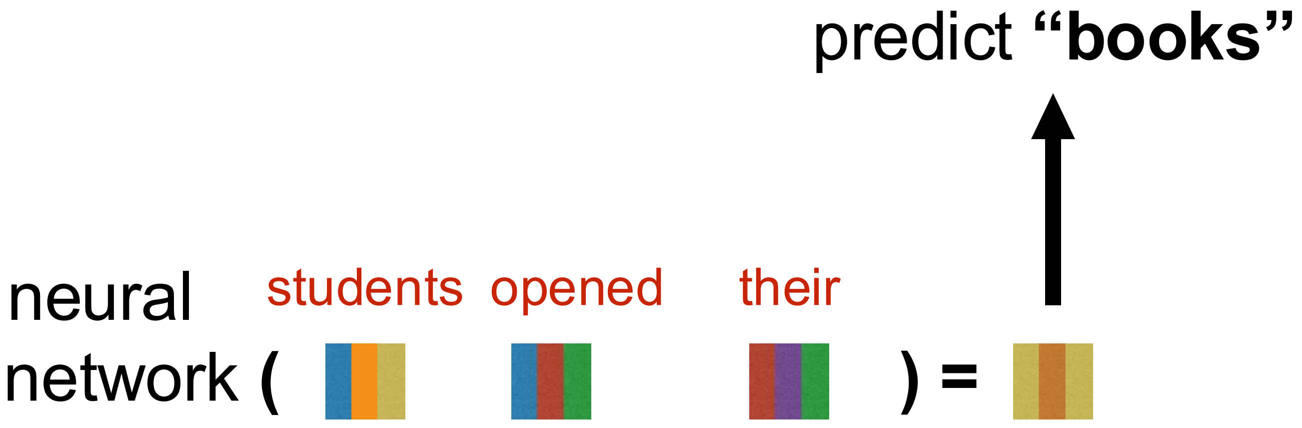
Next word prediction

- Predict the next word from composed prefix representation:



Next word prediction

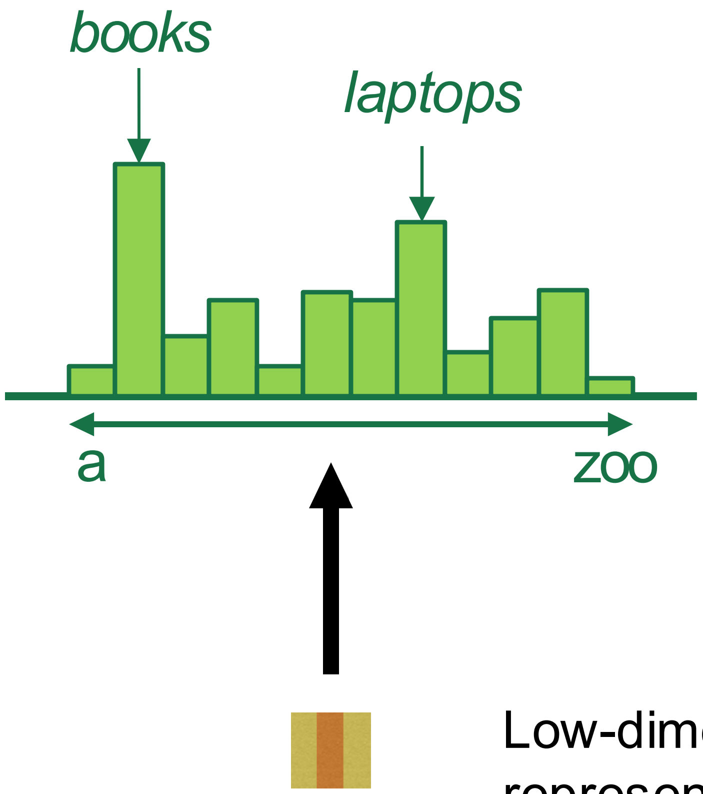
- Predict the next word from composed prefix representation:



Softmax layer:
convert a vector representation into a probability distribution over the entire vocabulary

$P(w_i|\text{vector for "students opened their"})$

Probability distribution over the entire vocabulary

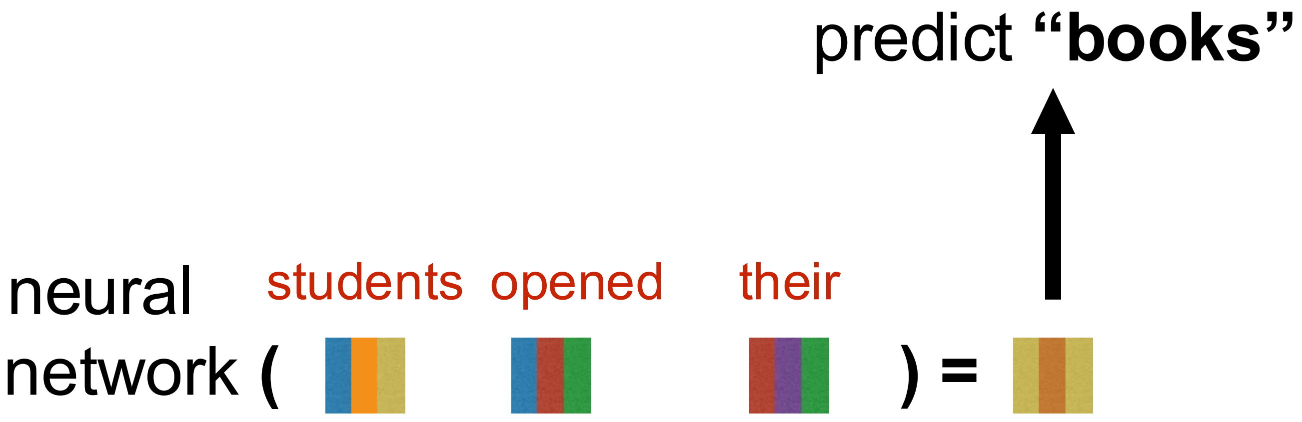


Low-dimensional representation of “students opened their”

So to sum up…

- Given a d-dimensional vector representation x of a prefix, we do the following to predict the next word:
  - Project it to a V-dimensional vector using a matrix-vector product (a.k.a. a “linear layer”, or a “feedforward layer”), where V is the size of the vocabulary.
  - Apply the softmax function to transform the resulting vector into a probability distribution.

Now that we know how to predict “books”, let’s focus on how to compute the prefix representation x in the first place!



Composition functions

- **Input:** sequence of word embeddings corresponding to the tokens of a given prefix
- **Output:** single vector
- Composition functions:
  - Element-wise functions
    - e.g., just sum up all of the word embeddings!
  - Concatenation
  - Feed-forward neural networks
  - Convolutional neural networks
  - Recurrent neural networks
  - Transformers

Large language models

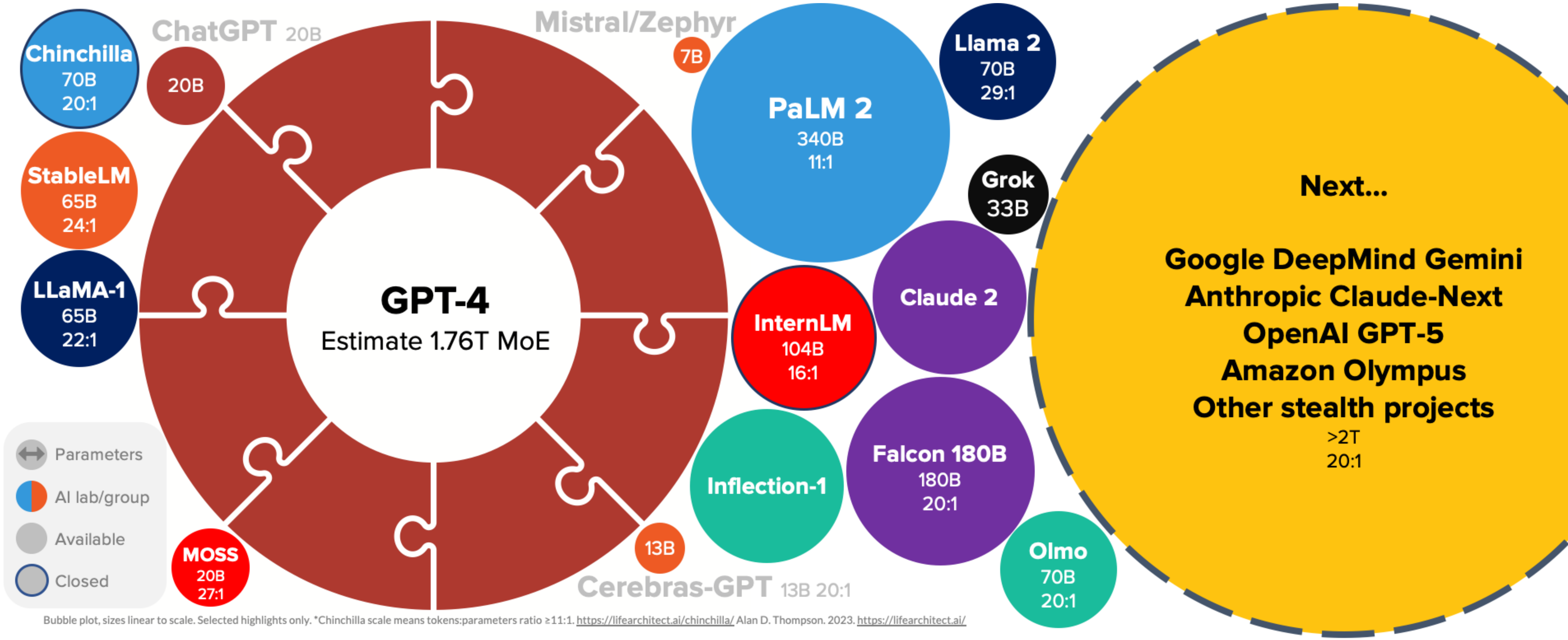
What’s a large language model?

- ChatGPT, a chatbot built on top of the GPT LLM released on November 28, 2022, has popularized deep learning models trained with a text corpus.
- Language models learn a probability distribution over language:

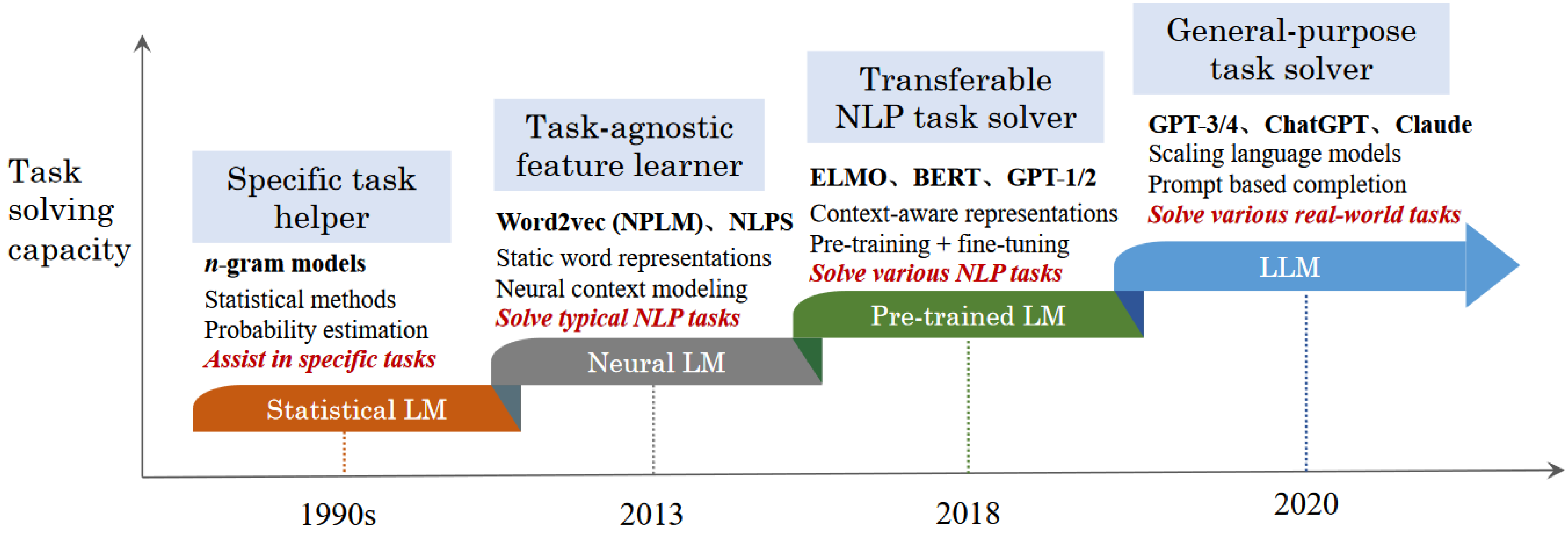
$$P(w_1, . . . ,w_m)$$

- Large in terms of training data (e.g., Common Crawl, Wikipedia, GitHub, ...) and parameters (e.g., PaLM has 540 billion parameters; GPT-4 rumored to have 1 trillion).

Language Models (Up to November 2023)



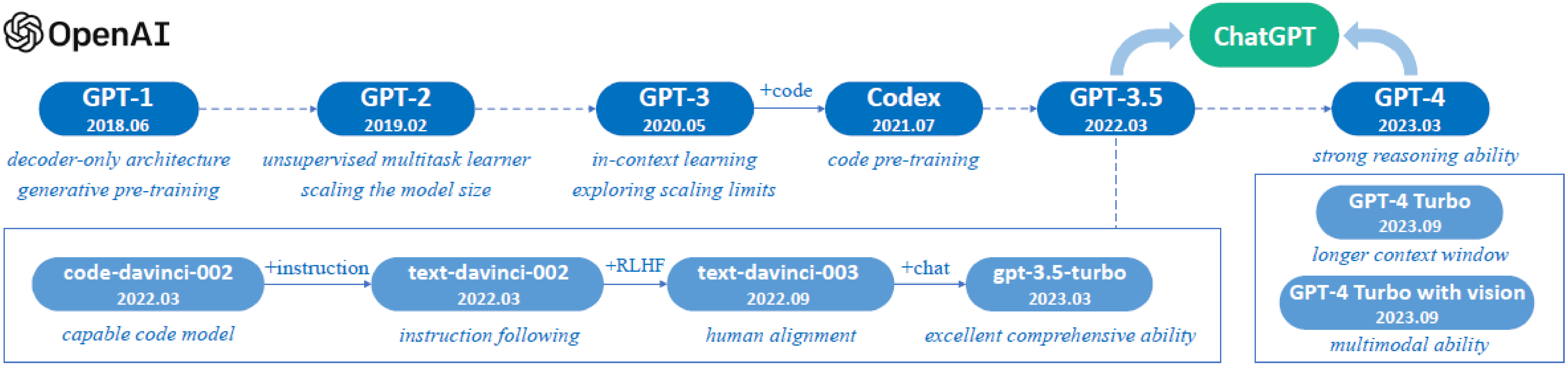
Source: https://lifearchitect.ai/models/


Task solving capacity
Specific task helper
n-gram models
Statistical methods
Probability estimation
Assist in specific tasks
Statistical LM
Task-agnostic feature learner
Word2vec (NPLM)、NLPS
Static word representations
Neural context modeling
Solve typical NLP tasks
Neural LM
Transferable NLP task solver
ELMO、BERT、GPT-1/2
Context-aware representations
Pre-training + fine-tuning
Solve various NLP tasks
Pre-trained LM
General-purpose task solver
GPT-3/4、ChatGPT、Claude
Scaling language models
Prompt based completion
Solve various real-world tasks
LLM
1990s
2013
2018
2020

Three kinds of LLM

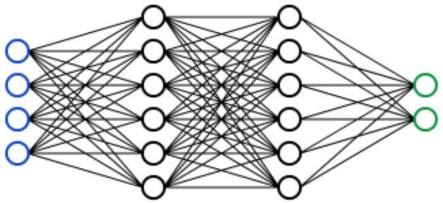
- **Generic language models:** predicting the next token. We will mostly talk about this type today.
- **Instruction tuned**
- **Dialog tuned:** ChatGPT (the base model is hard to interact with).

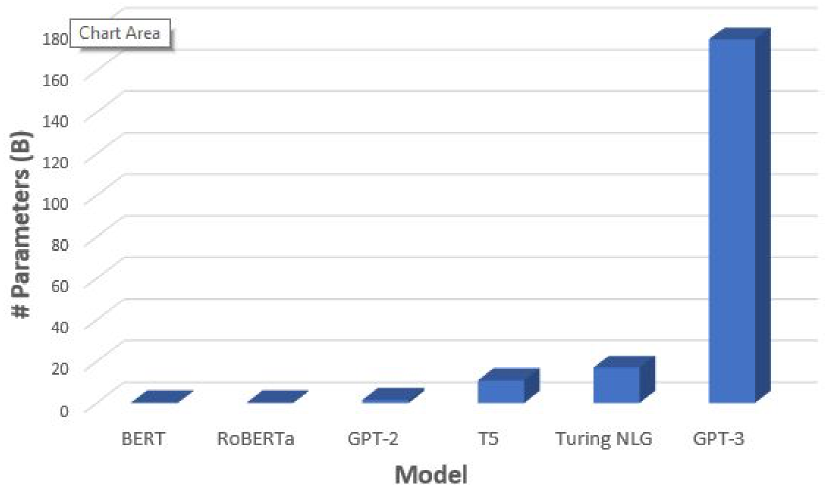
History of OpenAI GPT models



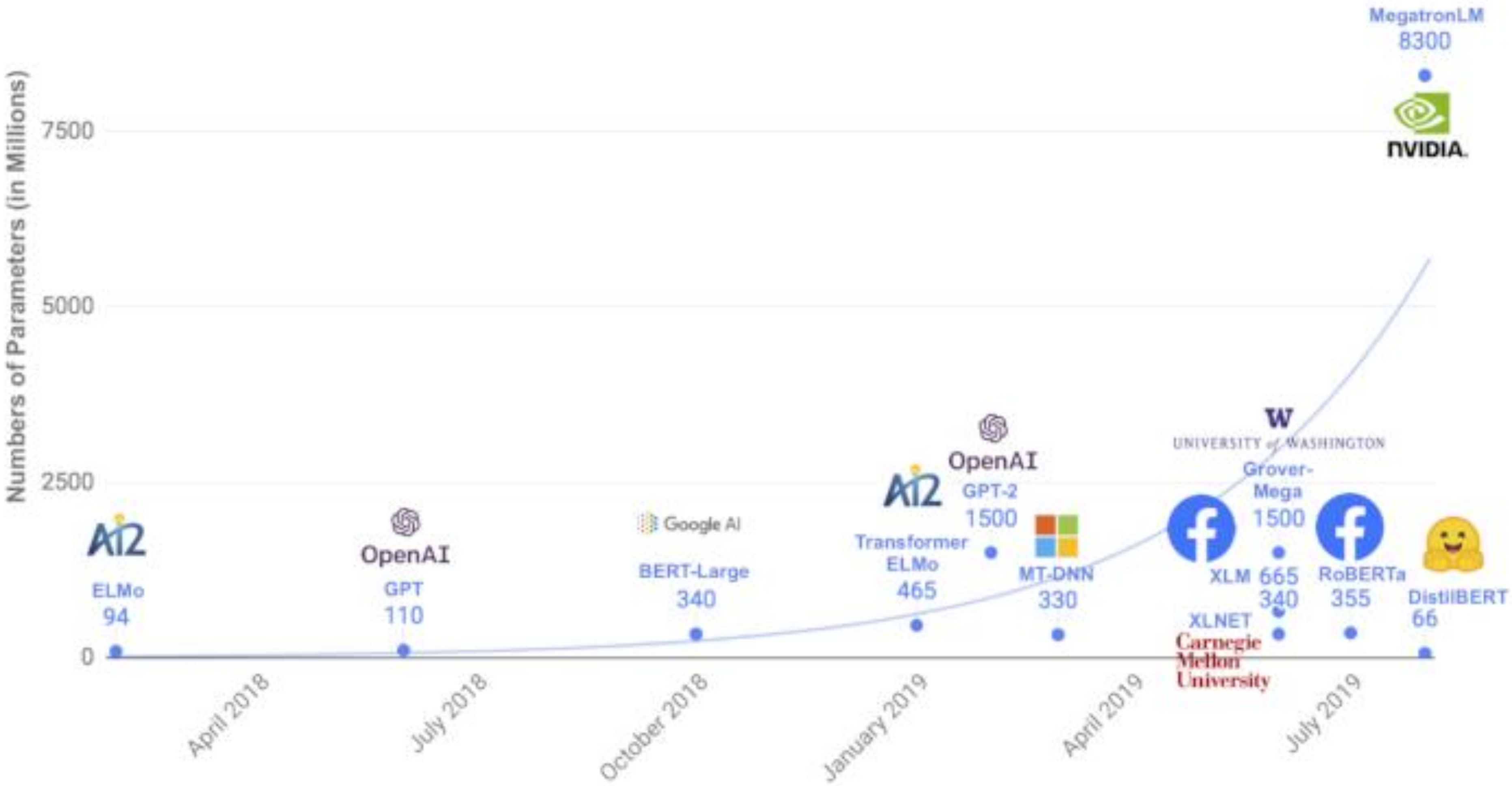
self-supervised learning + scale

In 1885, Stanford University was ______

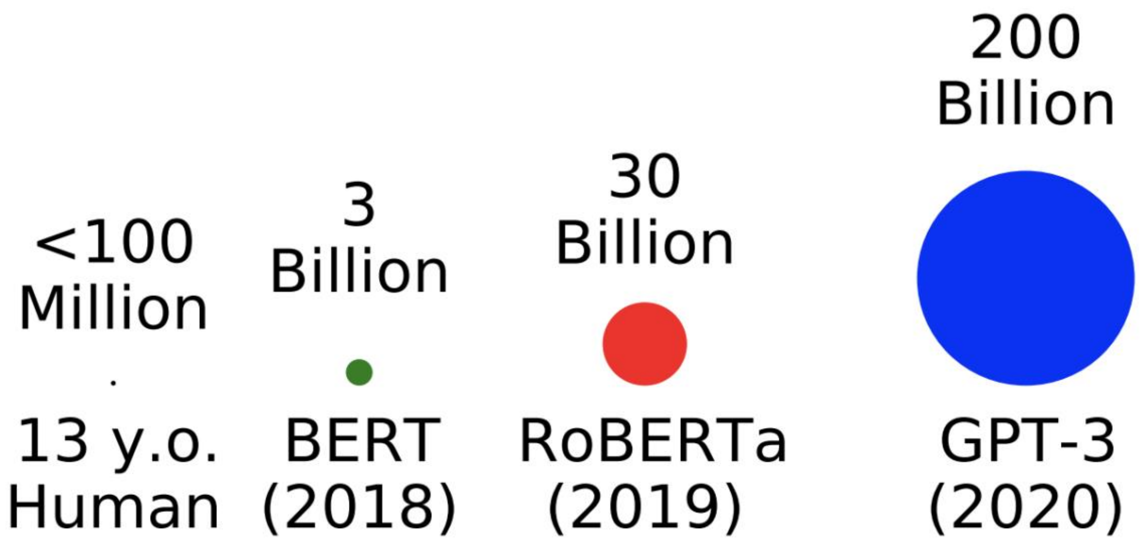




Scale in #parameters



Scale in #words



#tokens seen during training

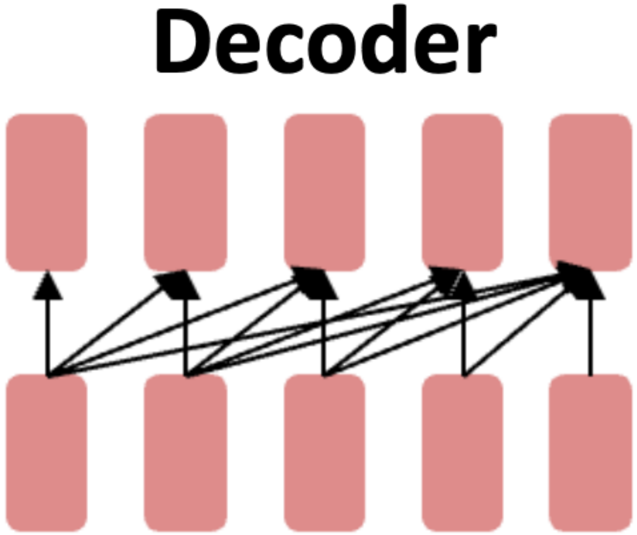
GPTs: Generative Pre-trained Transformers

Large language models

GPT-2 (1.5B parameters; Radford et al., 2019)

Same architecture as GPT, just bigger (117M -> 1.5B)

But trained on much more data: 4GB -> 40GB



Emergent zero-shot learning

The ability to do many tasks with no examples, and no gradient updates

Language Models are Unsupervised Multitask Learners

Alec Radford * [1] Jeffrey Wu * [1] Rewon Child [1] David Luan [1] Dario Amodei ** [1] Ilya Sutskever ** [1]

Emergent abilities of GPT-3 (2020)

OpenAI

- GPT-3 (175B parameters; Brown et al., 2020)
  - Another increase in size (1.5B -> 175B)
  - and data (40GB -> over 600GB)

Language Models are Few-Shot Learners

Tom B. Brown* **Benjamin Mann*** **Nick Ryder*** **Melanie Subbiah***

In-context Learning

Recent history

Word vectors & task-specific architectures (word2vec) → Pre-train & fine-tune (BERT) → **LLMs & in-context learning (GPT3)** → Instruction tuning & RLHF (ChatGPT)

Emergent in-context learning

OpenAI

```
gaot => goat
sakne => snake
brid => bird
fsih => fish
dcuk => duck
cmihp => chimp
```

In-context learning

```
thanks => merci
hello => bonjour
mint => menthe
wall => mur
otter => loutre
bread => pain
```

In-context learning

OpenAI

Few-shot GPT3 can beat strong task-specific fine-tuned models.

Language modeling ≠ assisting users

PROMPT	*Explain the moon landing to a 6 year old in a few sentences.*
COMPLETION	GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

Language modeling ≠ assisting users

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION

Human

A giant rocket ship blasted off from Earth carrying astronauts to the moon. The astronauts landed their spaceship on the moon and walked around exploring the lunar surface. Then they returned safely back to Earth, bringing home moon rocks to show everyone.

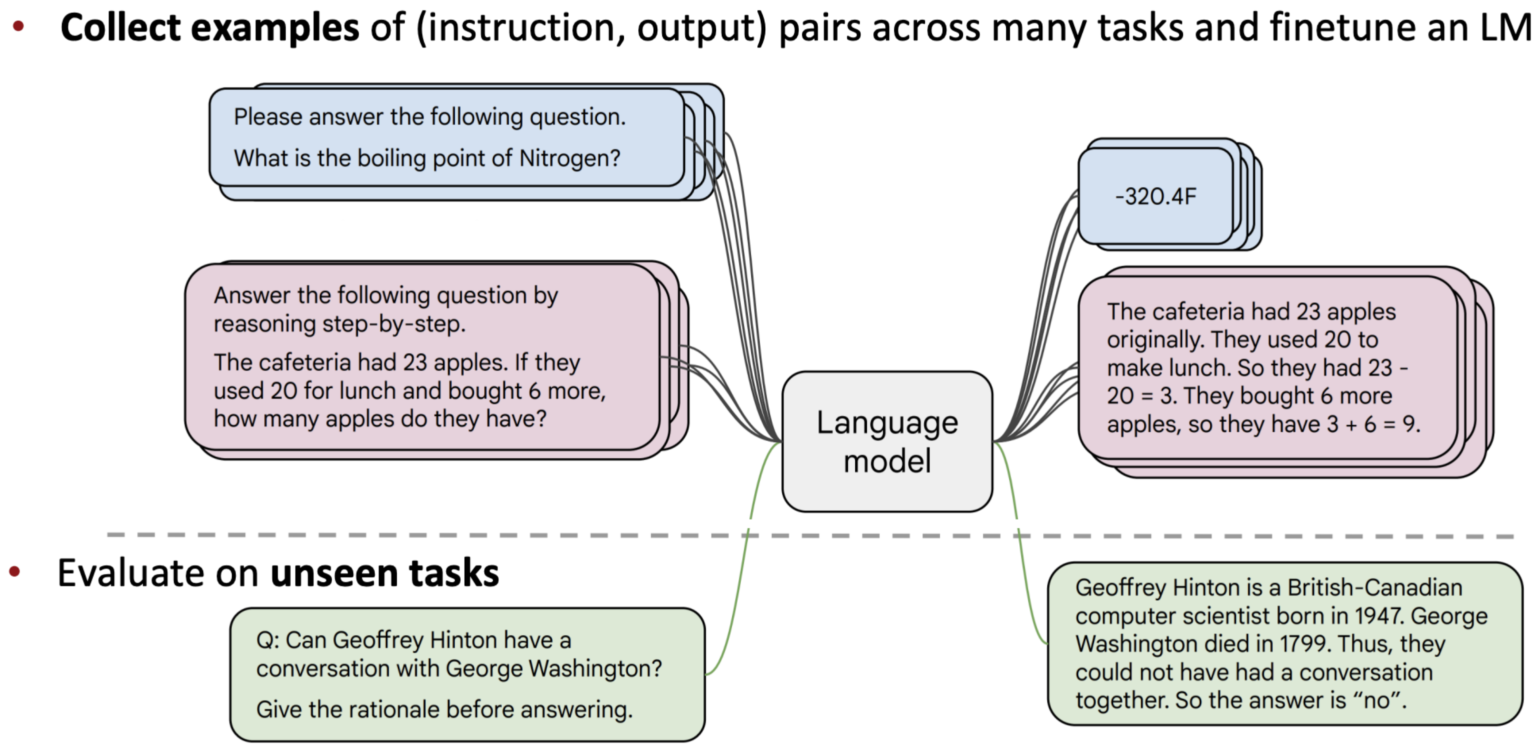
Instruction Tuning

Recent history

Word vectors & task-specific architectures (word2vec) → Pre-train & fine-tune (BERT) → LLMs & in-context learning (GPT3) → **Instruction tuning & RLHF (ChatGPT)**

Instruction tuning

Collect examples of (instruction, output) pairs across many tasks and finetune an LLM



[FLAN-T5; Chung et al., 2022]

Limitations of instruction-tuning

- No right answer for tasks like open-ended creative generation

- Mismatch between the LM objective and the objective of “satisfy human preferences”!

Can we explicitly attempt to **satisfy human preferences**?

Reinforcement learning to the rescue

For each LM sample s, imagine we had a way to obtain a human reward of that summary: $R(s) \in \mathbb{R}$, higher is better.

SAN FRANCISCO, California (CNN) -- A magnitude 4.2 earthquake shook the San Francisco ... overturn unstable objects.	An earthquake hit San Francisco. There was minor property damage, but no injuries.	The Bay Area has good weather but is prone to earthquakes and wildfires.
	s_1	s_2
	$R(s_1) = 8.0$	$R(s_2) = 1.2$

We want to maximize the expected reward of samples from our LM

Optimizing for human preferences

$$\mathbb{E}_{\hat{s} \sim p_\theta(s)}[R(\hat{s})]$$

How do we model human preferences?

OpenAI

Model their preferences as a separate (NLP) problem!

```
An earthquake hit
San Francisco.
There was minor
property damage,
but no injuries.
```

s_1

$R(s_1) = 8.0$

```
The Bay Area has
good weather but is
prone to
earthquakes and
wildfires.
```

s_2

$R(s_2) = 1.2$

Train an LM $RM_\phi(s)$ to predict human preferences from an annotated dataset, then optimize for RM_ϕ instead.

How do we model human preferences?

```
A 4.2 magnitude
earthquake hit
San Francisco,
resulting in
massive damage.
```

$$s_3$$

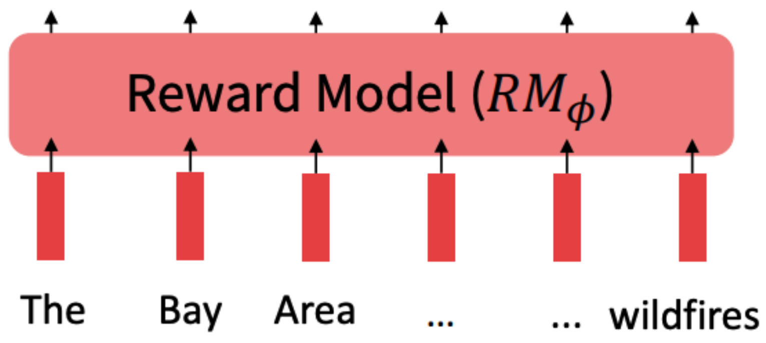
$$R(s_3) = \quad 4.1? \quad 6.6? \quad 3.2?$$

How do we model human preferences?

Asking pairwise comparisons

An earthquake hit San Francisco. There was minor property damage, but no injuries. > A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage. > The Bay Area has good weather but is prone to earthquakes and wildfires.

s_1 s_3 s_2



Bradley-Terry [1952] paired comparison model

$$J_{RM}(\phi) = -\mathbb{E}_{(s^w, s^l)\sim D}\left[\log \sigma(RM_\phi(s^w) - RM_\phi(s^l))\right]$$

"winning" sample

"losing" sample

s^w should score higher than s^l

RLHF: Putting it all together [Christiano et al., 2017; Stiennon et al., 2020]

From GPT3 to InstructGPT

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION

GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

InstructGPT

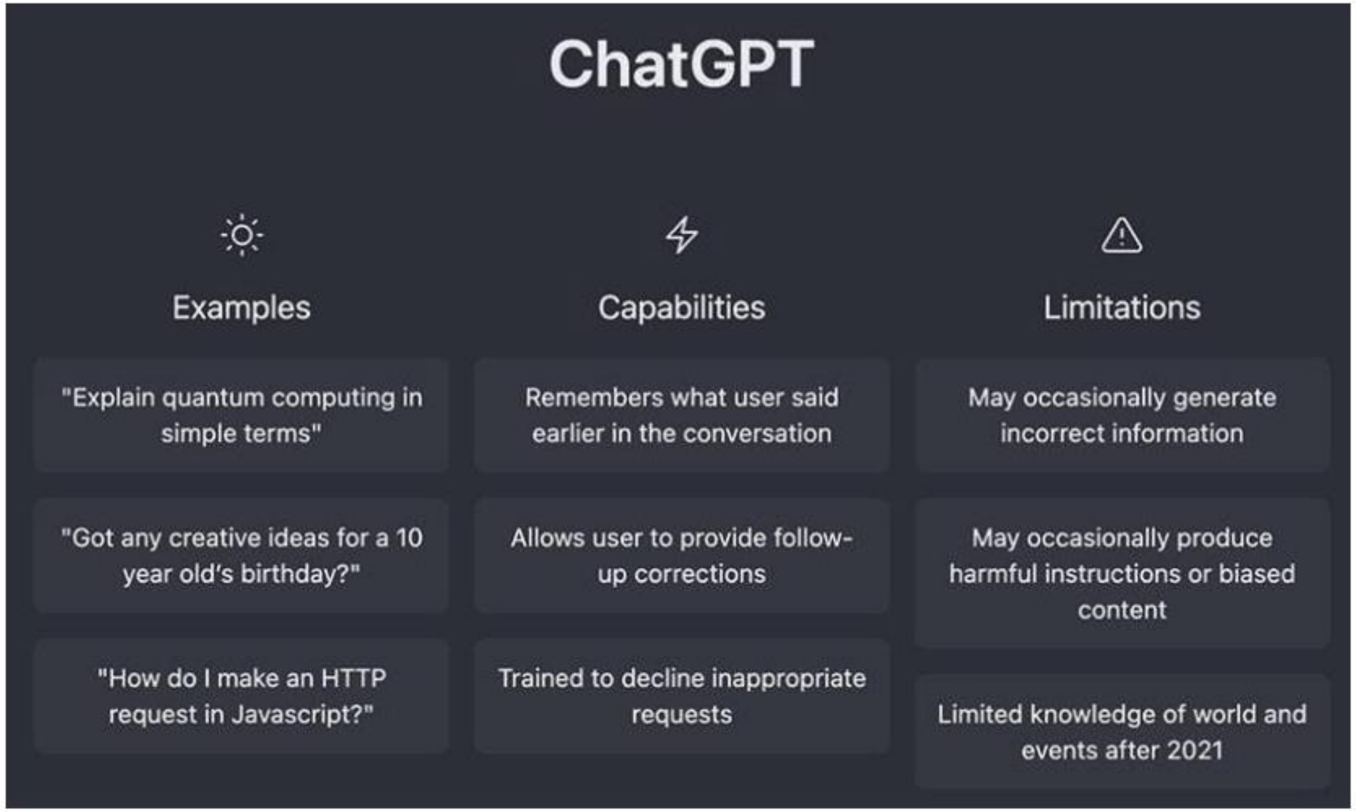
```
People went to the moon, and they took pictures of what they saw, and
sent them back to the earth so we could all see them.
```

ChatGPT: Instruction finetuning + RLHF for **dialog** agents

From:

Tehran is located in ________

To:



Limitations of RL + Reward Modeling

Human preferences are unreliable!

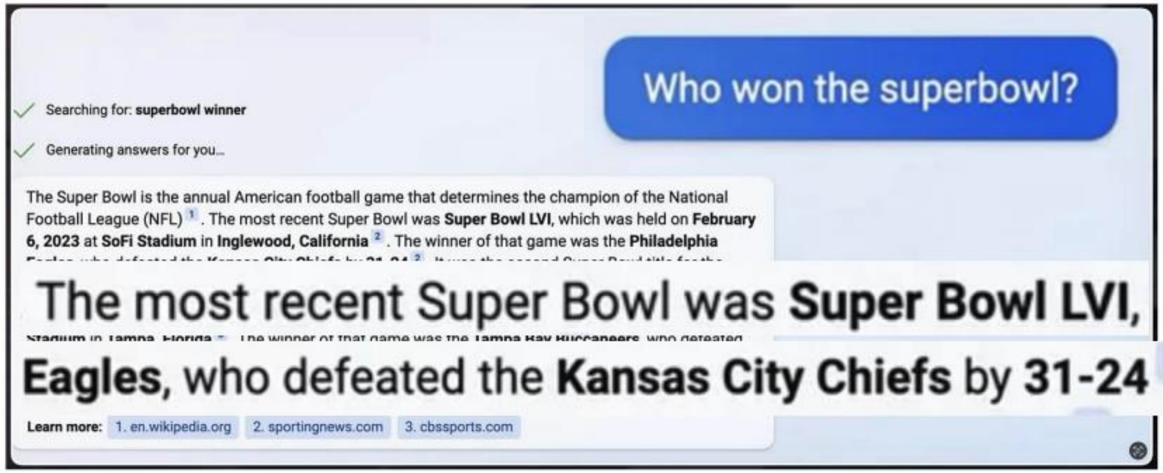
”**Reward hacking**” is a common problem in RL

Google shares drop $100 billion after its new AI chatbot makes a mistake

February 9, 2023 · 10:15 AM ET

https://www.npr.org/2023/02/09/1155650909/google-chatbot--error-bard-shares

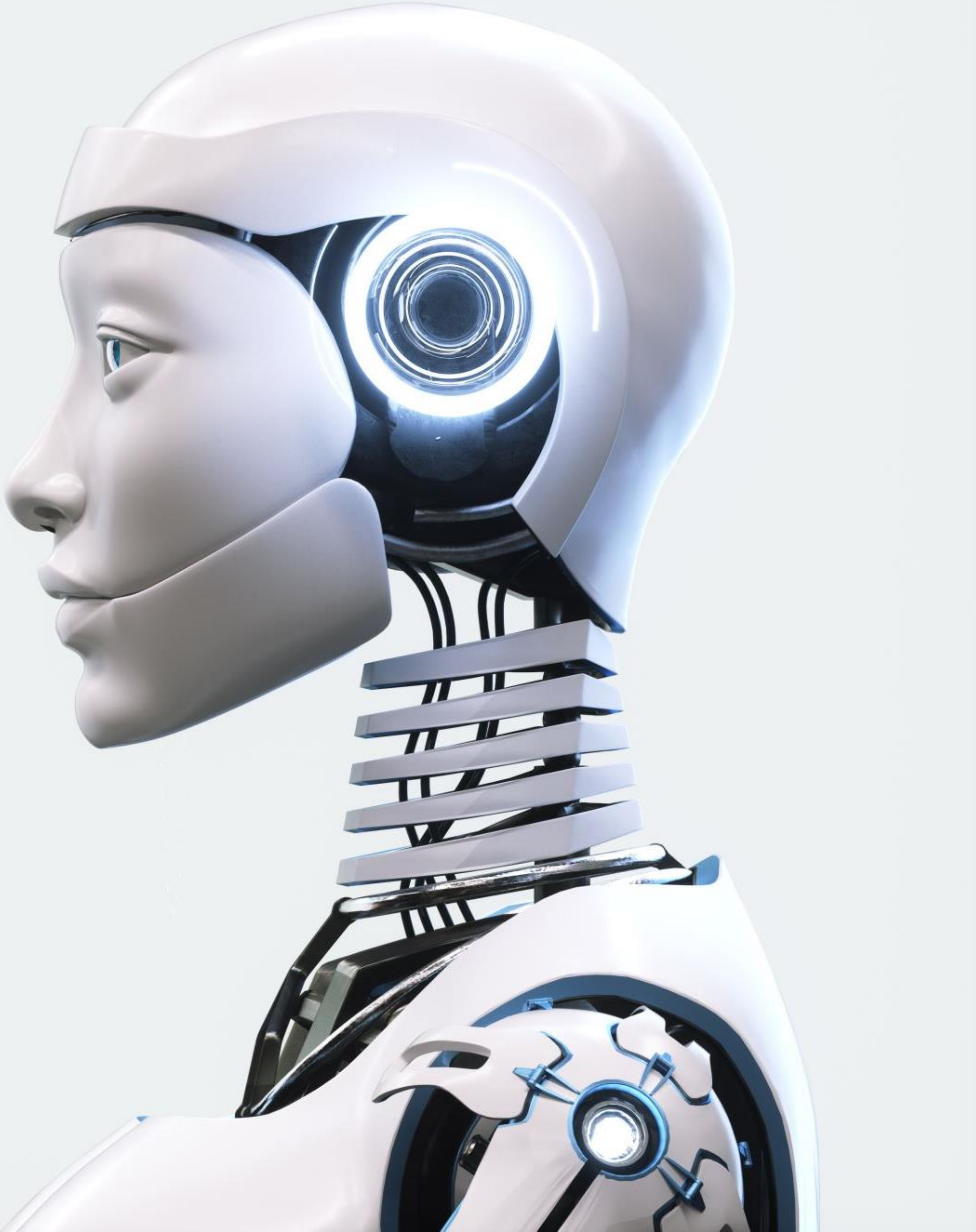
Bing AI hallucinates the Super Bowl

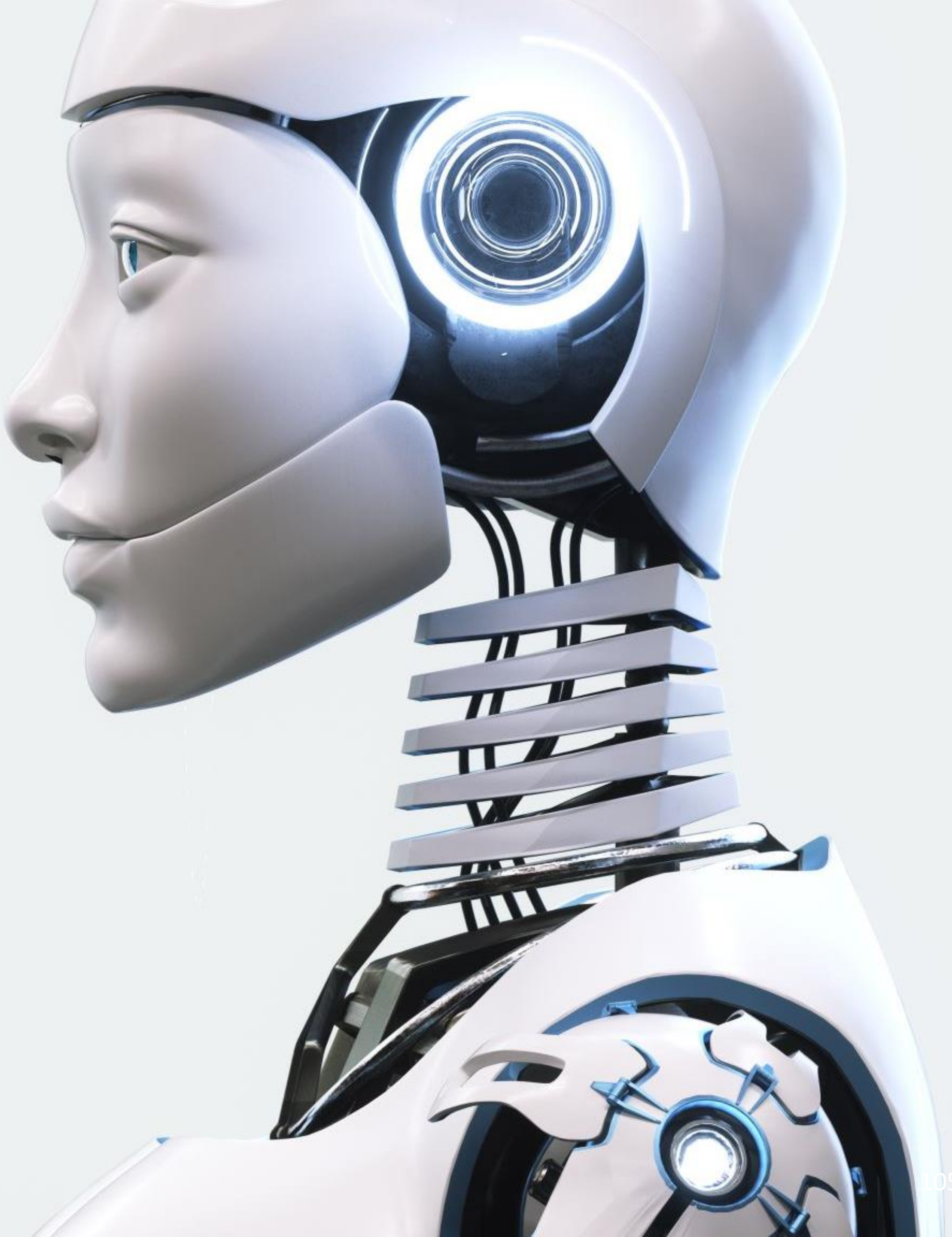


https://news.ycombinator.com/item?id=34776508
https://apnews.com/article/kansas-city-chiefs-philadelphia-eagles-technology-science-82bc20f207e3e4cf81abc6a5d9e6b23a

Chatbots are rewarded to produce responses that seem **authoritative** and **helpful**, regardless of truth

Models of human preferences are even more unreliable!





ChatGPT hallucinates

Prompt	Where do people smoke more on average than in Russia?
ChatGPT Generation	There are several countries where people smoke more on average than in Russia. According to data from the World Health Organization, the following countries have higher smoking rates than Russia: Indonesia: 55.2% of adults smoke China: 52.9% of adults smoke Serbia: 47.5% of adults smoke Serbia and Montenegro: 45.6% of adults smoke Montenegro: 45.6% of adults smoke Greece: 44.5% of adults smoke Bosnia and Herzegovina: 41.8% of adults smoke Kazakhstan: 41.3% of adults smoke Kosovo: 41.3% of adults smoke Moldova: 40.9% of adults smoke It's important to note that these figures are estimates and may vary slightly from other sources.
Hallucination	Extrinsic - the generated answer includes wrong statistics. (Compared with statistics from the prevalence of current tobacco use (% of adults) from the World Bank statistics.[16])

Numbers marked in red are wrong, ChatGPT imagines them.

[Bang et al., 2023:https://arxiv.org/pdf/2302.04023.pdf]

ChatGPT has a timestamp

AS Who won the most recent FIFA World Cup?

The most recent FIFA World Cup was held in 2018, and the winner was the French national team. They defeated Croatia 4-2 in the final to win their second World Cup title. The tournament was held in Russia from June 14 to July 15, 2018, and featured 32 teams from around the world.

The answer should have been Argentina, but it is not always trained on the most recent data.

Recent history

Word vectors & task-specific architectures (word2vec) → Pre-train & fine-tune (BERT) → LLMs & in-context learning (GPT3) → **Instruction tuning & RLHF (ChatGPT)**

Prompt engineering

What are prompts?

- Prompts involve instructions and context passed to a language model to achieve a desired task
- Prompt engineering is the practice of developing and optimizing prompts to efficiently use language models (LMs) for a variety of applications
  - Prompt engineering is a useful skill for AI engineers and researchers to improve and efficiently use language models

```
What is prompt engineering?

Prompt engineering is a process of creating a set of prompts,
or questions, that are used to guide the user toward a desired
outcome. It is an effective tool for designers to create user
experiences that are easy to use and intuitive. This method is
often used in interactive design and software development, as
it allows users to easily understand how to interact with a
system or
product..
```

Why Prompt Engineering?

- Why learn prompt engineering?
  - Important for research, discoveries, and advancement
  - Helps to test and evaluate the limitations of LLMs
  - Enables all kinds of innovative applications on top of LLMs

ANTHROP\C

Prompt Engineer and Librarian

APPLY FOR THIS JOB

SAN FRANCISCO, CA / PRODUCT / FULL-TIME / HYBRID

Anthropic's mission is to create reliable, interpretable, and steerable AI systems. We want AI to b
safe for our customers and for society as a whole.

Anthropic's AI technology is amongst the most capable and safe in the world. However, large language models are a new type of intelligence, and the art of instructing them in a way that delivers the best results is still in its infancy — it's a hybrid between programming, instructing, ar
teaching. You will figure out the best methods of prompting our AI to accomplish a wide range o
tasks, then document these methods to build up a library of tools and a set of tutorials that allov
others to learn prompt engineering or simply find prompts that would be ideal for them.

Compensation and Benefits*

Anthropic's compensation package consists of three elements: salary, equity, and benefits. We are committed to pay fairness and aim for these three elements collectively to be highly competitive with market rates.

Salary - The expected salary range for this position is $250k - $335k.

Equity - Equity will be a major component of the total compensation for this position. We aim to offer higher-than-average equity compensation for a company of our size, and communicate equity amounts at the time of offer issuance.

Source: https://jobs.lever.co/Anthropic/e3cde481-d446-460f-b576-93cab67bd1ed

First Basic Prompt

Instruction

```
The sky is

blue

The sky is a beautiful blue color during the day. The
blue of the sky is created by the Earth's atmosphere
scattering the sun's light. The blue is a result of the
air molecules in the atmosphere reflecting the shorter
wavelength of blue light back to our
eyes.
```

Response

Model: “text-davinci-003”
temperature: 0.7
top-p: 1

Elements of a Prompt

- A prompt is composed with the following components:
  - Instructions
  - Context
  - Input data
  - Output indicator

```
Classify the text into neutral, negative or positive

Text: I think the food was okay.

Sentiment:
```

Role Playing

```
The following is a conversation with an AI research
assistant. The assistant tone is technical and
scientific.

Human: Hello, who are you?
AI: Greeting! I am an AI research assistant. How can
help you today?
Human: Can you tell me about the creation of
blackholes?
AI: Sure! Black holes are regions of spacetime where the
light, can escape from it. They are created when a very
massive star dies and its core collapses in on itself,
forming a singularity of infinite density. The intense
gravity of the singularity pulls in all the matter and
radiation around it, creating the black
hole.
```

Prompt Engineering Techniques

- Many advanced prompting techniques have been designed to improve performance on complex tasks
  - Few-shot prompts
  - Chain-of-thought prompting
  - Self-consistency
  - Knowledge generation prompting
  - Augmented language models
    - Retrieval
    - Programming languages
    - External APIs

Few-shot Prompts

- Few-shot prompting allows us to provide exemplars in prompts to steer the model towards better performance

```
The odd numbers in this group add up to an even number: 4,
8, 9, 15, 12, 2, 1.
A: The answer is False.

The odd numbers in this group add up to an even number: 17,
10, 19, 4, 8, 12, 24.
A: The answer is True.

The odd numbers in this group add up to an even number: 16,
11, 14, 4, 8, 13, 24.
A: The answer is True.

The odd numbers in this group add up to an even number: 17,
9, 10, 12, 13, 4, 2.
A: The answer is False.

The odd numbers in this group add up to an even number: 15,
32, 5, 13, 82, 7, 1.
A: The answer is True.
```

Chain-of-Thought (CoT) Prompting

- Prompting can be further improved by instructing the model to reason about the task when responding
  - This is very useful for tasks that requiring reasoning
  - You can combine it with few-shot prompting to get better results
  - You can also do zero-shot CoT where exemplars are not available

```
The odd numbers in this group add up to an even number: 4,
8, 9, 15, 12, 2, 1.
A: Adding all the odd numbers (9, 15, 1) gives 25. The
answer is False.

The odd numbers in this group add up to an even number: 15,
32, 5, 13, 82, 7, 1.
A: Adding all the odd numbers (15, 5, 13, 7, 1) gives 41.
The answer is False.
```

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

Zero-Shot CoT

- Involves adding "**Let's think step by step**" to the original prompt

```
I went to the market and bought 10 apples. I gave 2 apples
to the neighbor and 2 to the repairman. I then went and
bought 5 more apples and ate 1. How many apples did I
remain with?

11 apples
```

```
I went to the market and bought 10 apples. I gave 2 apples
to the neighbor and 2 to the repairman. I then went and
bought 5 more apples and ate 1. How many apples did I
remain with?
Let's think step by step.

First, you started with 10
apples.You gave away 2 apples to the neighbor and 2 to the
repairman, so you had 6 apples
left.
Then you bought 5 more apples, so now you had 11
apples.Finally, you ate 1 apple, so you would remain with 10
```


Source: Large Language Models are Zero-Shot Reasoners

Self-Consistency

- Self-Consistency aims to improve on the naive greedy decoding used in chain-of-thought prompting
- The idea is to sample multiple, diverse reasoning paths through few-shot CoT, and use the generations to select the most consistent answer.
- This helps to boost the performance of CoT prompting on tasks involving arithmetic and commonsense reasoning

```
When I was 6 my sister was half my age. Now
I'm 70 how old is my sister?

35
```

Many examples were generated but the model kept responding 35 as the answer

Source: Self-Consistency Improves Chain of Thought Reasoning in Language Models

Self-Consistency Example

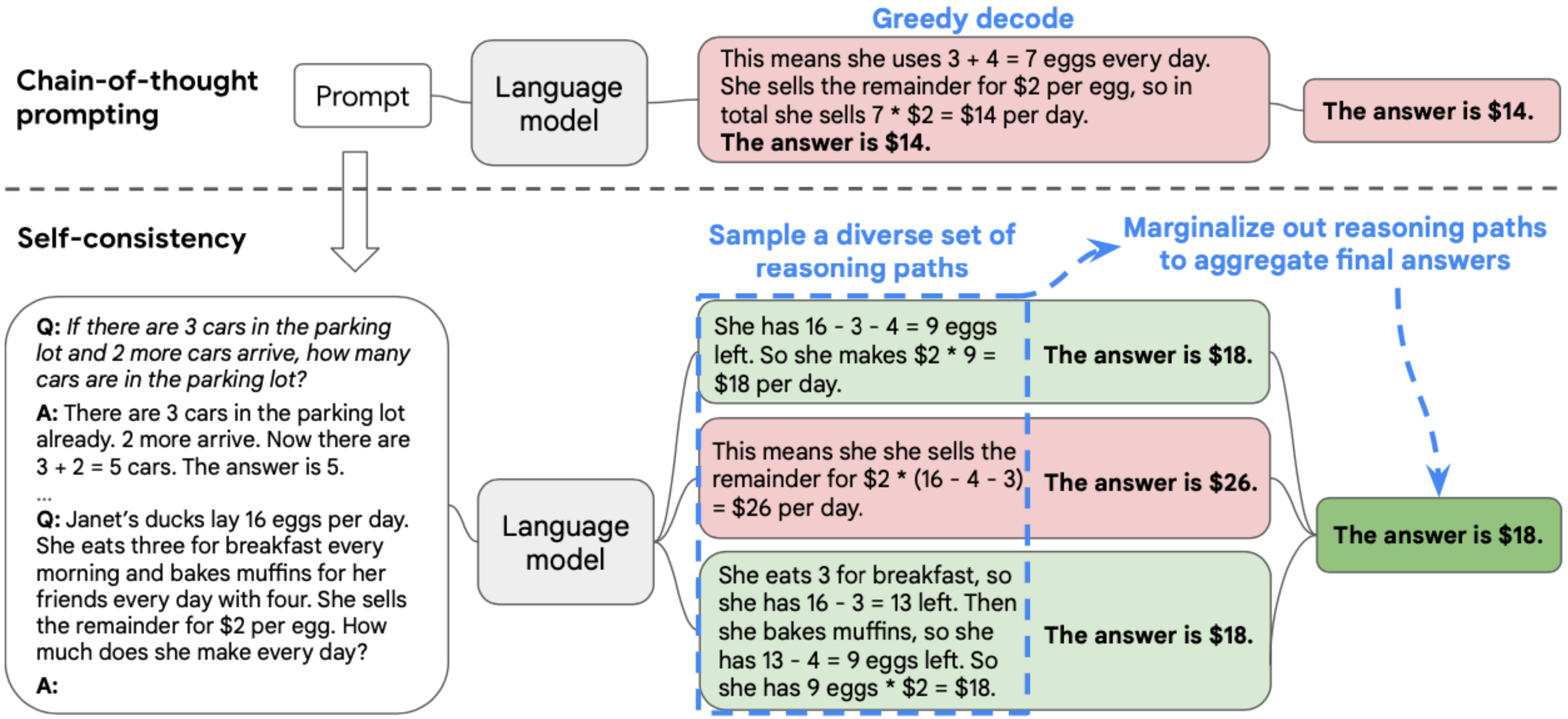


Figure 1: The self-consistency method contains three steps: (1) prompt a language model using chain-of-thought (CoT) prompting; (2) replace the "greedy decode" in CoT prompting by sampling from the language model's decoder to generate a diverse set of reasoning paths; and (3) marginalize out the reasoning paths and aggregate by choosing the most consistent answer in the final answer set.

Source: [Self-Consistency Improves Chain of Thought Reasoning in Language Models]()

LLMs with External Tools

- The generative capabilities of LLMs can be combined with an external tool to solve complex problems.
- The components you need:
  - An agent powered by LLM to determine which actions to take
  - A tool used by the agent to interact with the world (e.g., search API, Wolfram, Python REPL, database lookup)

Tool Augmented Language Models

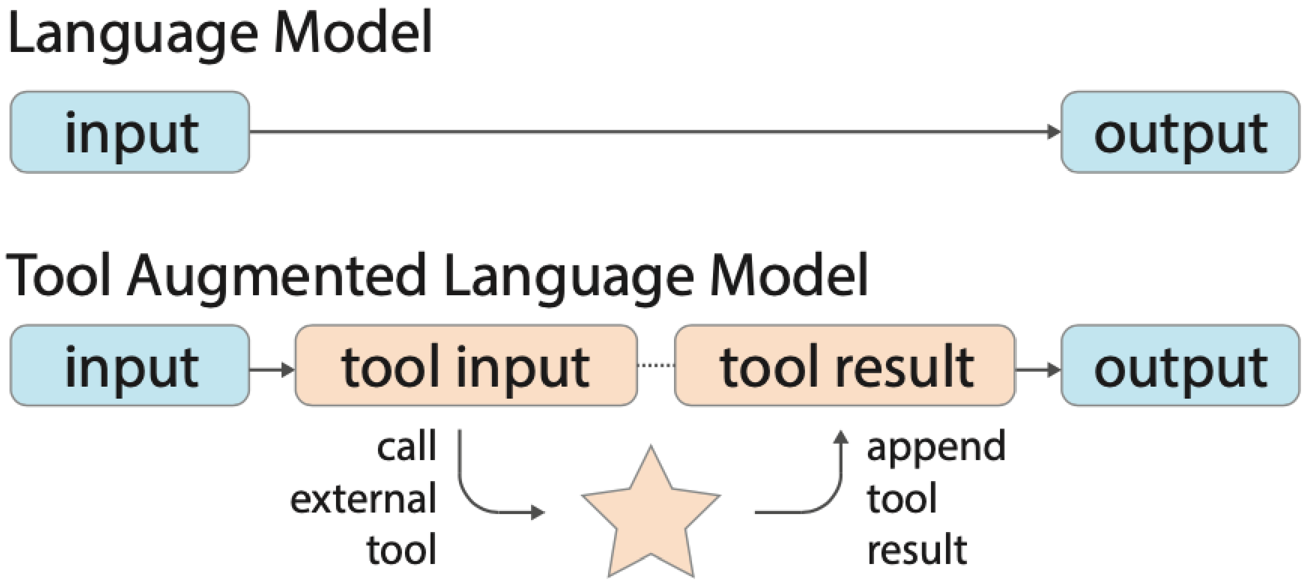


Figure 2: LM and Tool Augmented LMs.

Source [TALM: Tool Augmented Language Models]

Data-Augmented Generation

- For many real-world applications there is a need to augment the generation of a model by incorporating external data
- Steps involved:
  - **Fetching** relevant data
  - **Augmenting** prompts with the retrieved data as context
- External data can include:
  - Document stores
  - APIs
  - Databases
  - User provided data
  - …

Tools & IDEs

- There are many tools, libraries, and platforms with different capabilities and functionalities
- Capabilities include:
  - Developing and experimenting with prompts
  - Evaluating prompts
  - Versioning and deploying prompts

Dyno
Prompt Engineering IDE

LangChain

DUST

PROMPTABLE

More tools here: https://github.com/dair-ai/Prompt-Engineering-Guide#tools--libraries

- https://lilianweng.github.io/posts/2023-03-15-prompt-engineering/
- https://platform.openai.com/docs/guides/prompt-engineering/six-strategies-for-getting-better-results
- https://scholar.google.com/citations?view_op=list_works&hl=en&hl=en&user=ScLUQ-YAAAAJ&sortby=pubdate
- https://cookbook.openai.com/articles/related_resources