Large Language Models

Introduction to Large Language Models

Large language models

Computational agents that can interact conversationally with people using natural language

LLMS have revolutionized the field of NLP and AI

Language models

- Remember the simple n-gram language model
  - Assigns probabilities to sequences of words
  - Generate text by sampling possible next words
  - Is trained on counts computed from lots of text
- Large language models are similar and different:
  - Assigns probabilities to sequences of words
  - Generate text by sampling possible next words
  - **Are trained by learning to guess the next word**

Fundamental intuition of large language models

Text contains enormous amounts of knowledge

Pretraining on lots of text with all that knowledge is what gives language models their ability to do so much

What does a model learn from pretraining?

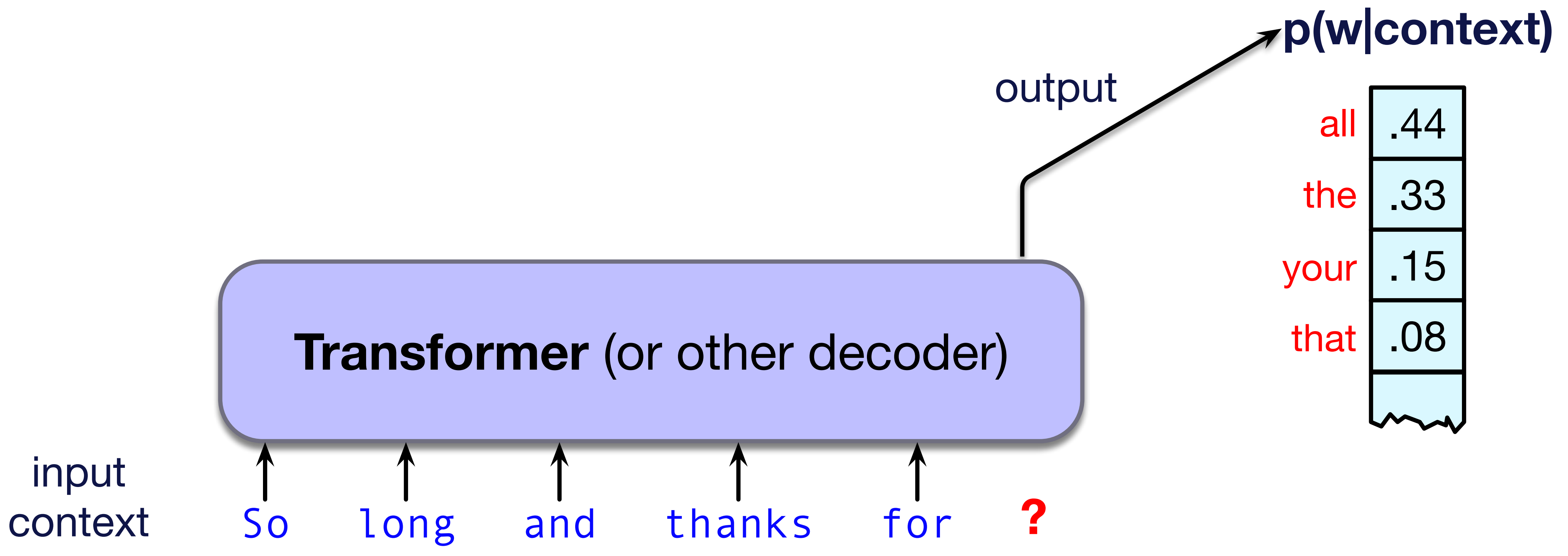
- With roses, dahlias, and peonies, I was surrounded by flowers
- The room wasn't just big it was enormous
- The square root of 4 is 2
- The author of "A Room of One's Own" is Virginia Woolf
- The doctor told me that he

What is a large language model?

A neural network with:

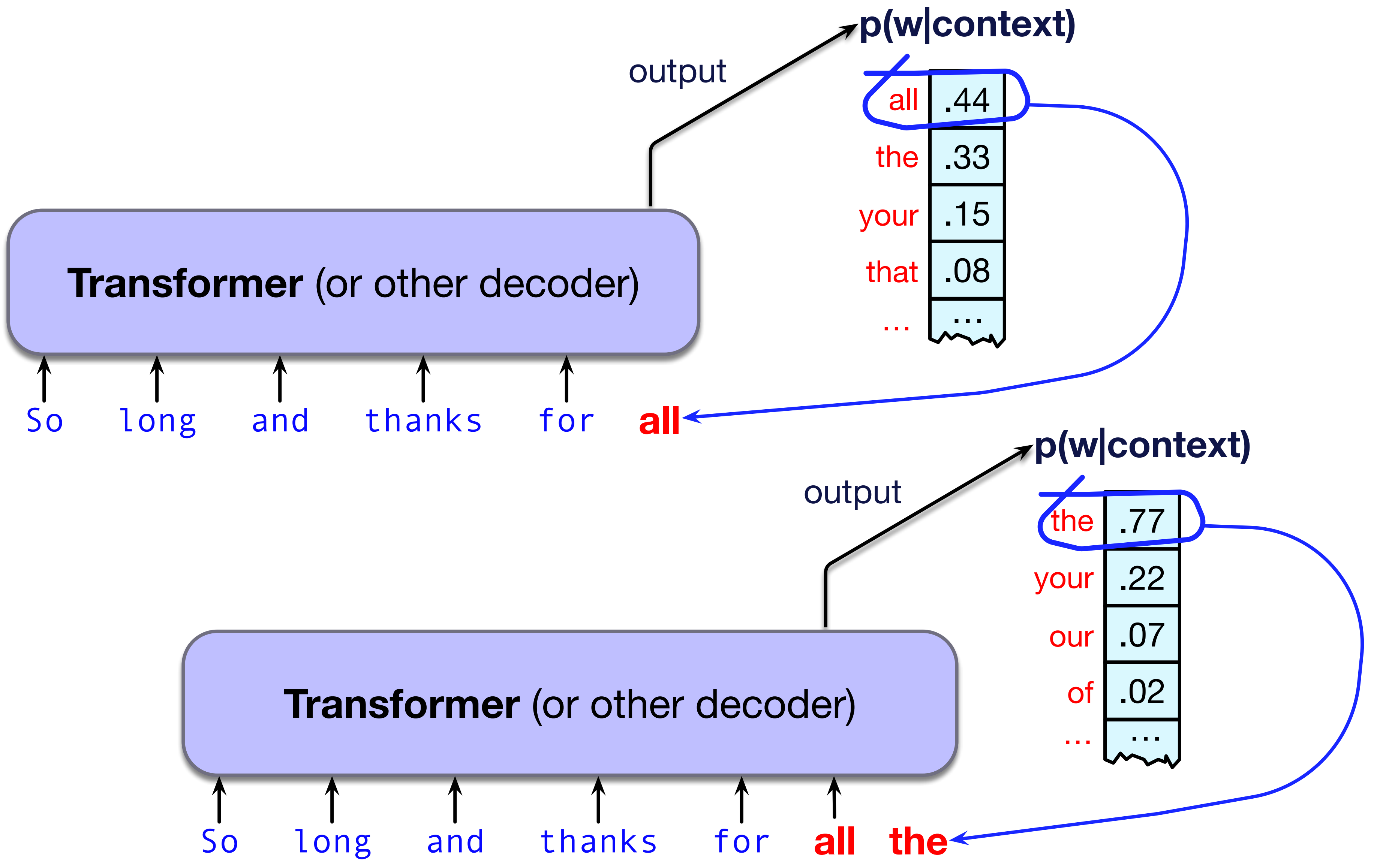
Input: a context or prefix,

Output: a distribution over possible next words

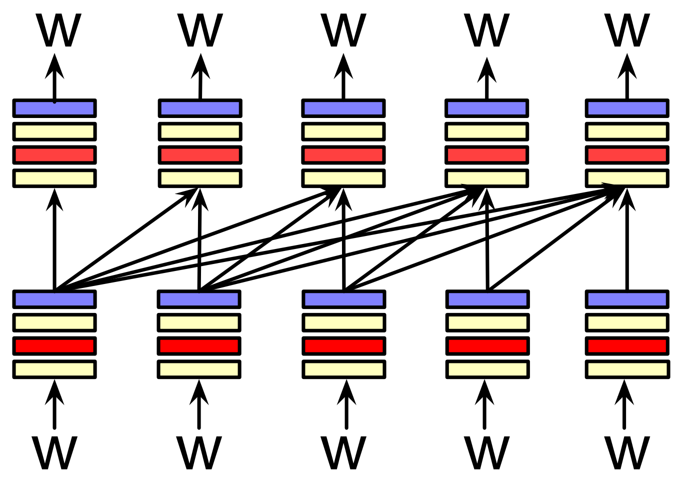


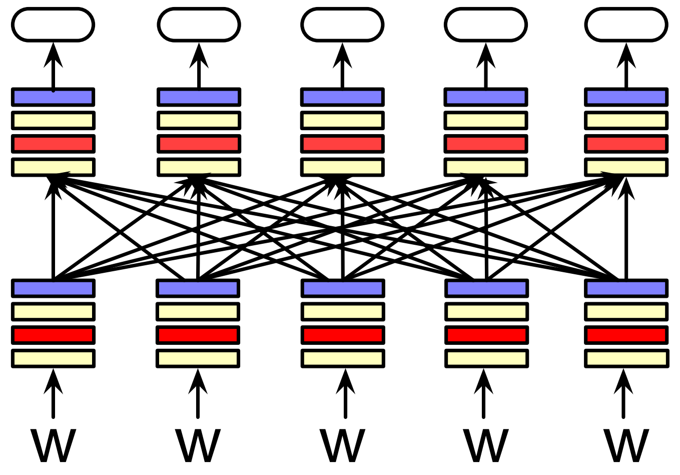
LLMs can generate!

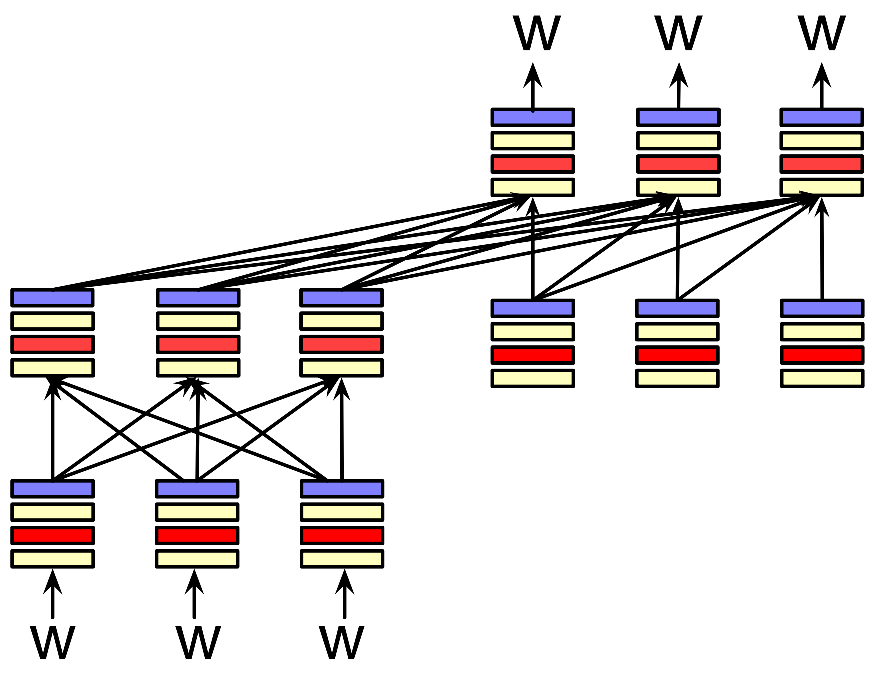
A model that gives a probability distribution over next words can generate by repeatedly sampling from the distribution


p(w|context)
output
all .44
the .33
your .15
that .08
… …
Transformer (or other decoder)
So long and thanks for all
p(w|context)
output
the .77
your .22
our .07
of .02
… …
Transformer (or other decoder)
So long and thanks for all the

Three architectures for large language models







Decoders

GPT, Claude,

Llama

Mixtral

Encoders

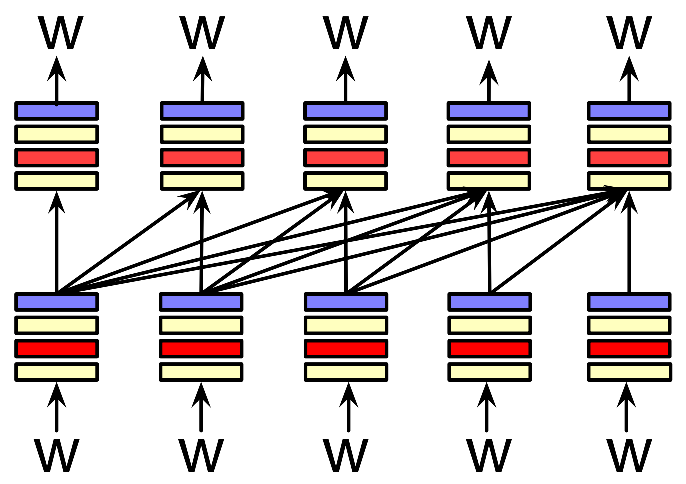
BERT family,

HuBERT

Encoder-decoders

Flan-T5, Whisper

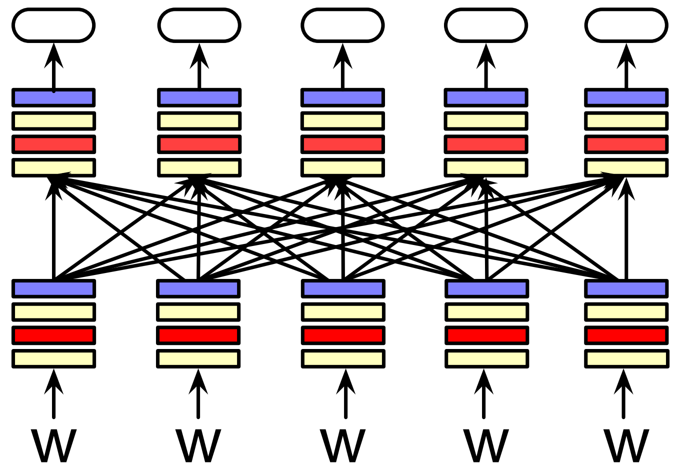
Decoders



What most people think of when we say LLM

- GPT, Claude, Llama, DeepSeek, Mistral
- A generative model
- It takes as input a series of tokens, and iteratively generates an output token one at a time.
- Left to right (causal, autoregressive)

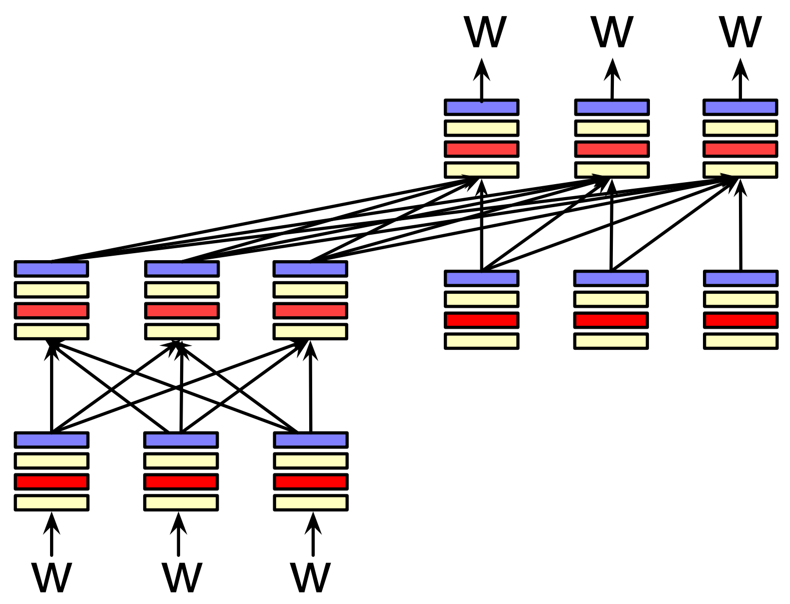
Encoders



- Masked Language Models (MLMs)
- BERT family

- Trained by predicting words from surrounding words on both sides
- Are usually **finetuned** (trained on supervised data) for classification tasks.

Encoder-Decoders



- Trained to map from one sequence to another
- Very popular for:
  - machine translation (map from one language to another)
  - speech recognition (map from acoustics to words)

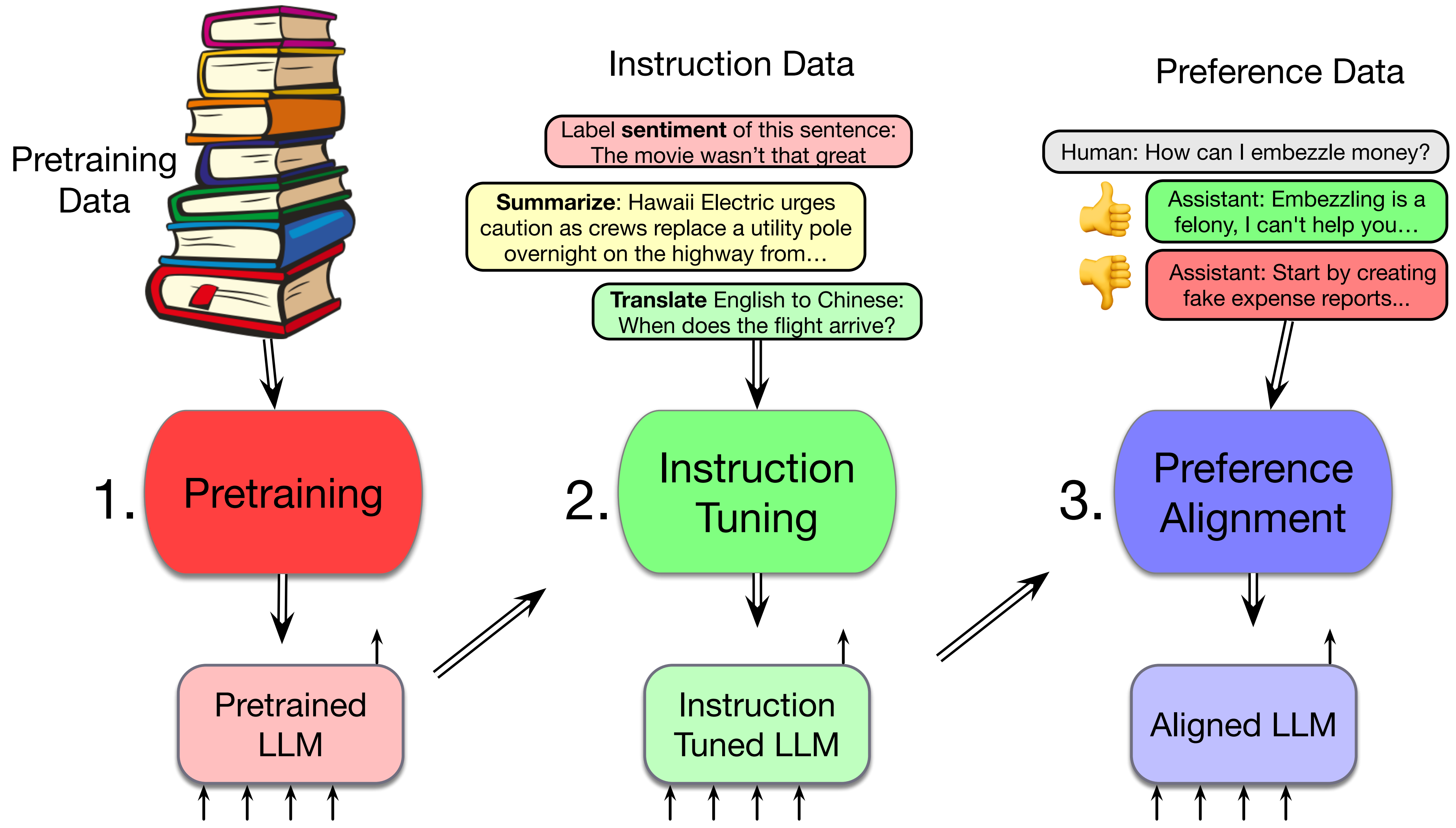
Large Language Models

Introduction to Large Language Models

Large Language Models

Pretraining Large Language Models

Three stages of training in LLMs


Pretraining Data
Instruction Data
Preference Data
Label **sentiment** of this sentence: The movie wasn't that great
Summarize: Hawaii Electric urges caution as crews replace a utility pole overnight on the highway from…
Translate English to Chinese: When does the flight arrive?
Human: How can I embezzle money?
Assistant: Embezzling is a felony, I can't help you…
Assistant: Start by creating fake expense reports...
1. Pretraining
2. Instruction Tuning
3. Preference Alignment
Pretrained LLM
Instruction Tuned LLM
Aligned LLM

Pretraining

The big idea that underlies all the amazing performance of language models

First **pretrain** a transformer model on enormous amounts of text

Then **apply** it to new tasks.

Self-supervised training algorithm

We train them to predict the next word!

1. Take a corpus of text
2. At each time step *t*
   i. ask the model to predict the next word
   ii. train the model using gradient descent to minimize the error in this prediction

"**Self-supervised**" because it just uses the next word as the label!

Intuition of language model training: loss

- Same loss function: **cross-entropy loss**
  - We want the model to assign a high probability to true word *w*
  - = want loss to be high if the model assigns too low a probability to w
- CE Loss: The negative log probability that the model assigns to the true next word w
  - If the model assigns too low a probability to w
  - We move the model weights in the direction that assigns a higher probability to w

Cross-entropy loss for language modeling

CE loss: difference between the correct probability distribution and the predicted distribution

$$L_{CE} = -\sum_{w \in V} \boxed{\mathbf{y}_t[w]} \log \boxed{\hat{\mathbf{y}}_t[w]}$$

The correct distribution $\mathbf{y}_t$ knows the next word, so is 1 for the actual next word and 0 for the others.

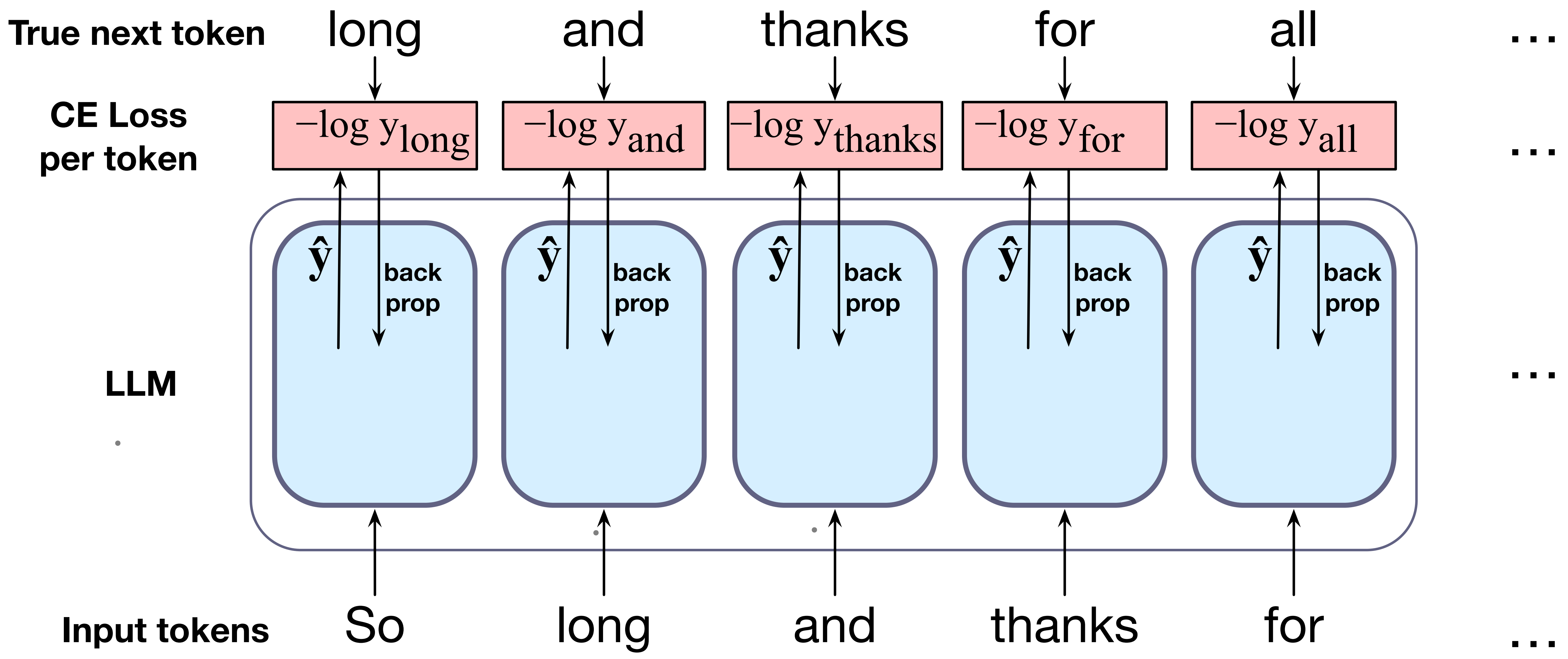
So in this sum, all terms get multiplied by zero except one: the logp the model assigns to the correct next word, so:

$$L_{CE}(\hat{\mathbf{y}}_t, \mathbf{y}_t) = -\log \hat{\mathbf{y}}_t[w_{t+1}]$$

Teacher forcing

- At each token position t, model sees correct tokens $w_{1:t}$,
  - Computes loss (–log probability) for the next token w_{t+1}
- At next token position t+1 we ignore what model predicted for w_{t+1}
  - Instead we take the **correct** word w_{t+1}, add it to context, move on

Training a transformer language model



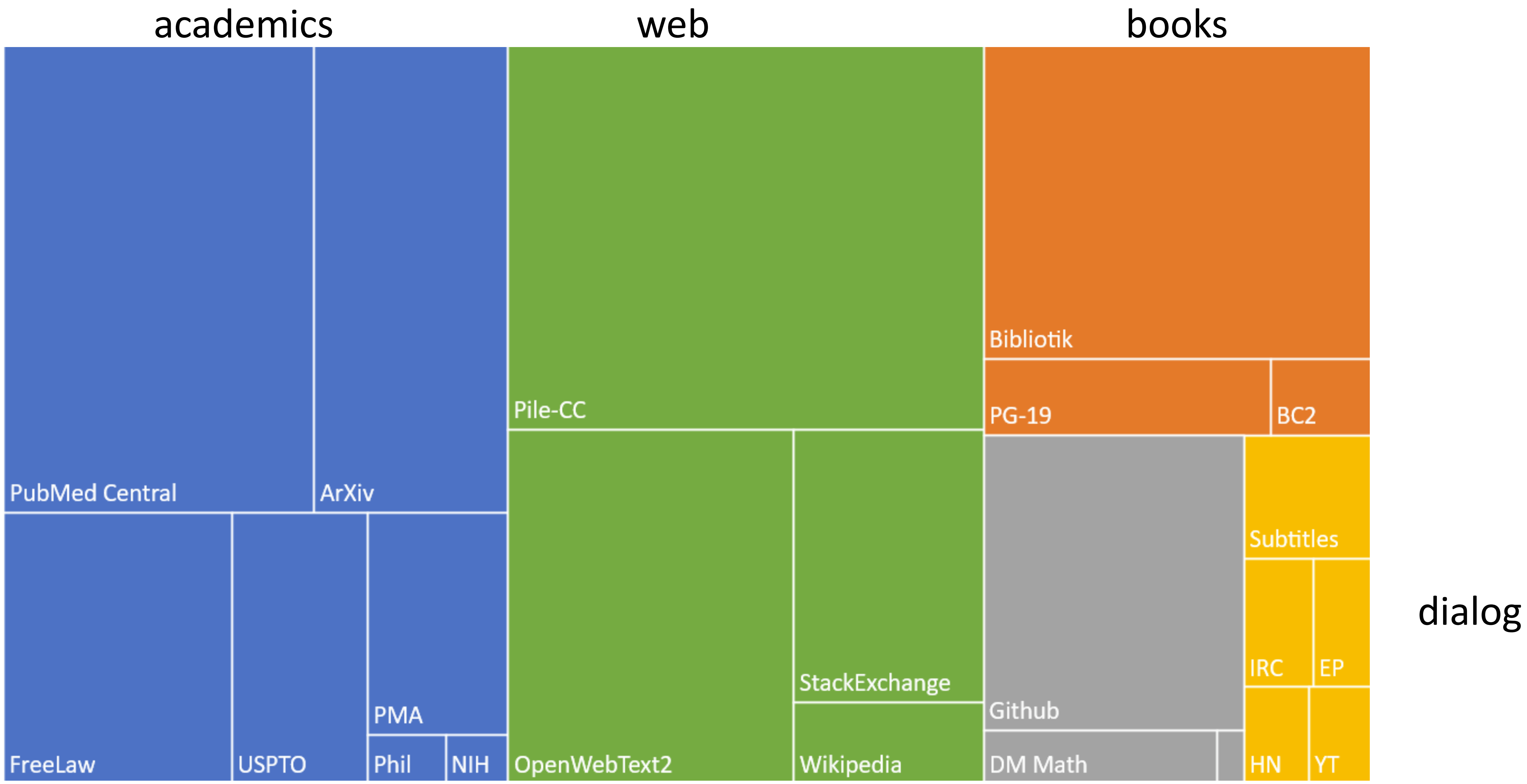
LLMs are mainly trained on the web

Common crawl, snapshots of the entire web produced by the non- profit Common Crawl with billions of pages

Colossal Clean Crawled Corpus (C4; Raffel et al. 2020), 156 billion tokens of English, filtered

What's in it? Mostly patent text documents, Wikipedia, and news sites

The Pile: a pretraining corpus



Filtering for quality and safety

Quality is subjective

- Many LLMs attempt to match Wikipedia, books, particular websites
- Need to remove boilerplate, adult content
- Deduplication at many levels (URLs, documents, even lines)

Safety also subjective

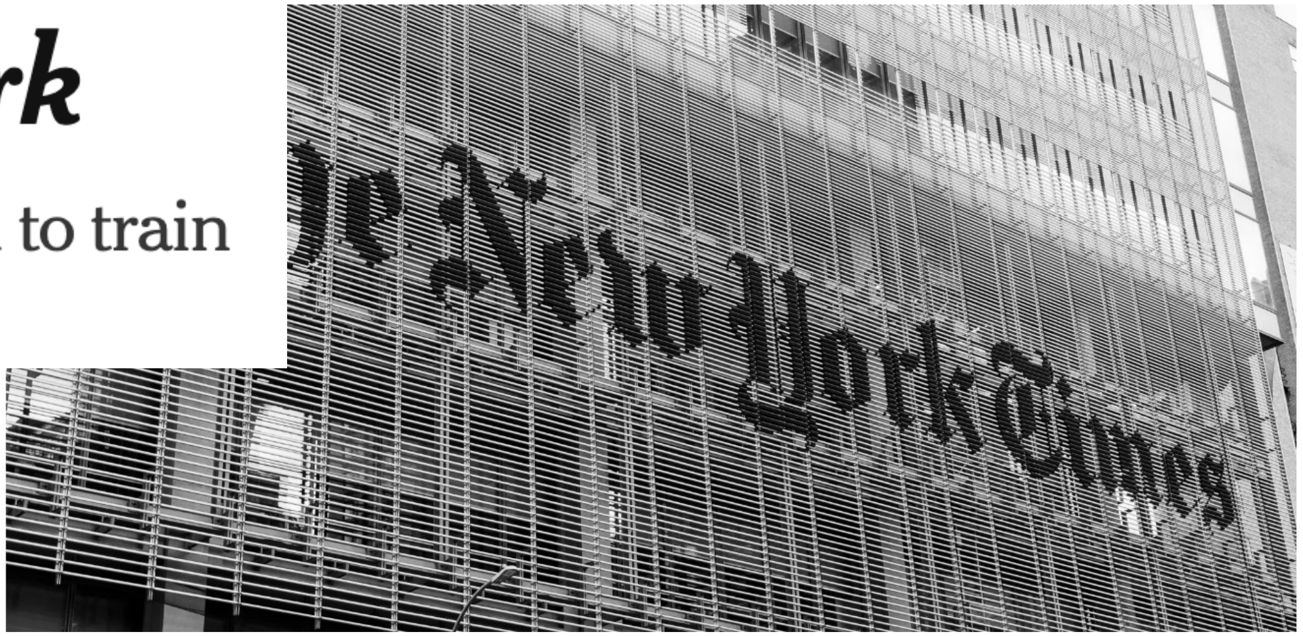
- Toxicity detection is important, although that has mixed results
- Can mistakenly flag data written in dialects like African American English

There are problems with scraping from the web

Authors Sue OpenAI Claiming Mass Copyright Infringement of Hundreds of Thousands of Novels

The Times Sues OpenAI and Microsoft Over A.I. Use of Copyrighted Work

Millions of articles from The New York Times were used to train chatbots that now compete with it, the lawsuit said.



There are problems with scraping from the web

Copyright: much of the text in these datasets is copyrighted

- Not clear if fair use doctrine in US allows for this use
- This remains an open legal question across the world

Data consent

- Website owners can indicate they don't want their site crawled

Privacy:

- Websites can contain private IP addresses and phone numbers

Skew:

- Training data is disproportionately generated by authors from the US which probably skews resulting topics and opinions

Large Language Models

Pretraining Large Language Models

Large Language Models

Evaluating Large Language Models

Better LMs are better at predicting text

Reminder of the chain rule:

$$\begin{aligned} P(w_{1:n}) &= P(w_1)P(w_2|w_1)P(w_3|w_{1:2})\ldots P(w_n|w_{1:n-1}) \\ &= \prod_{i=1}^{n} P(w_i|w_{<i}) \end{aligned}$$

So given a text $w_{1:n}$ we could just compare the log likelihood from two LMs:

$$\text{log likelihood}(w_{1:n}) = \log \prod_{i=1}^{n} P(w_i|w_{<i})$$

But raw log-likelihood has problems

Probability depends on size of test set

- Probability gets smaller the longer the text
- We would prefer a metric that is **per-word**, normalized by length

Perplexity is normalized for length

Perplexity is the inverse probability of the test set, normalized by the number of words

(The inverse comes from the original definition of perplexity from cross-entropy rate in information theory)

Probability range is $[0,1]$, perplexity range is $[1,\infty]$

Perplexity

So just as for n-gram grammars, we use perplexity to measure how well the LM predicts unseen text

The perplexity of a model θ on an unseen test set is the **inverse probability that θ assigns to the test set, normalized by the test set length.**

For a test set of *n* tokens $w_{1:n}$ the perplexity is :

$$
\begin{aligned}
\text{Perplexity}_\theta(w_{1:n}) &= P_\theta(w_{1:n})^{-\frac{1}{n}} \\
&= \sqrt[n]{\frac{1}{P_\theta(w_{1:n})}} \qquad = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{P_\theta(w_i|w_{<i})}}
\end{aligned}
$$

Perplexity

- The higher the probability of the word sequence, the lower the perplexity.
- Thus the lower the perplexity of a model on the data, the better the model.
- **Minimizing perplexity is the same as maximizing probability**

Also: perplexity is sensitive to length/tokenization so best used when comparing LMs that use the same tokenizer.

Many other factors that we evaluate, like:

Size

Big models take lots of GPUs and time to train, memory to store

Energy usage

Can measure kWh or kilograms of CO2 emitted

Fairness

Benchmarks measure gendered and racial stereotypes, or decreased performance for language from or about some groups.