Pretraining

CS 6120 Natural Language Processing

Northeastern University

Si Wu

Some materials borrowed rom Jurafsky & Martin Chapter 7

Logistics

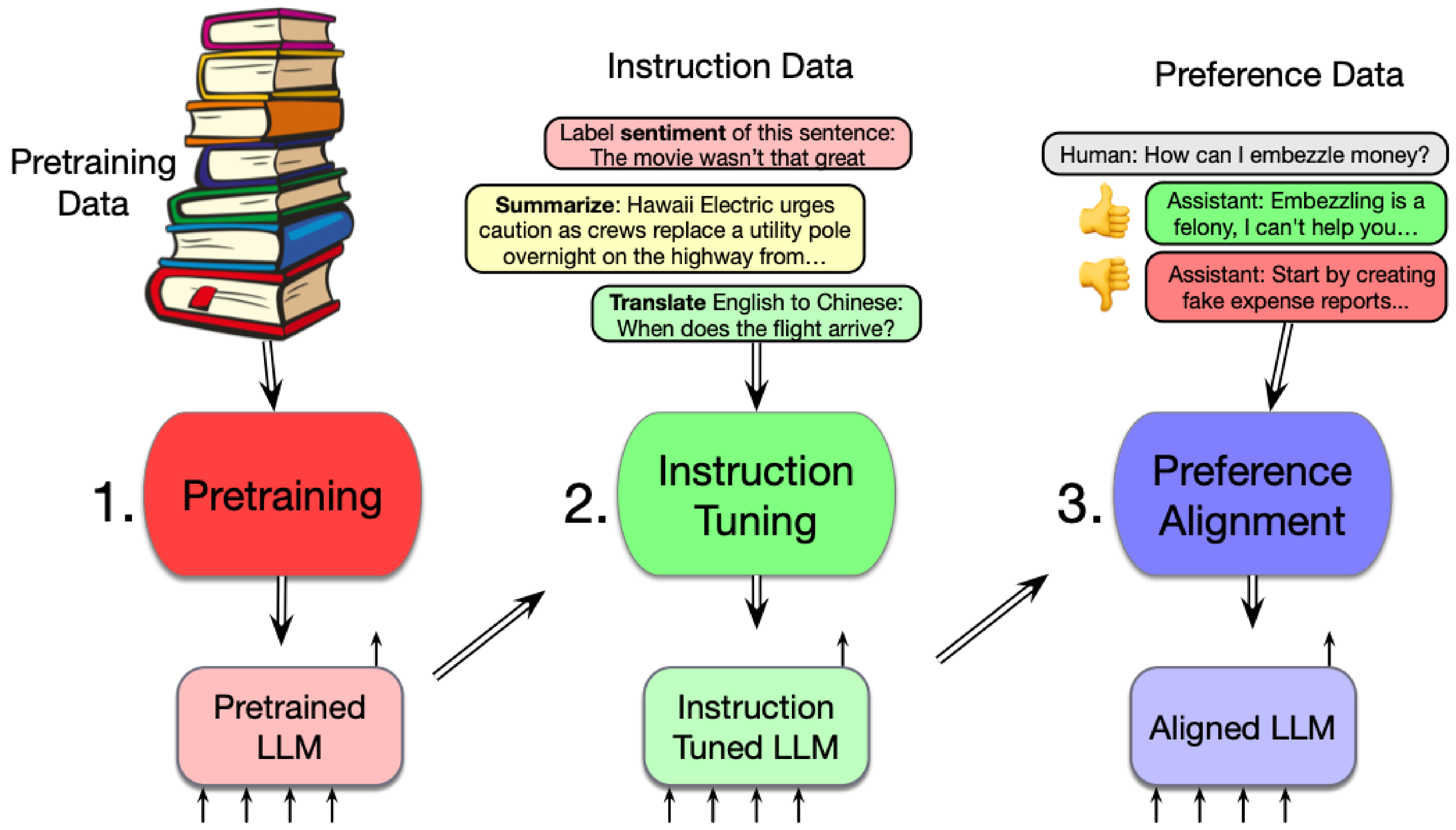
- I am working on giving all of you feedback on your project proposals
  - You get full points for a good and solid proposal
  - You get 50% if I still have questions or the project proposal is not good
    - But you can reply to my comment or resubmit to get 100% later
- Quiz 2 grade should be out soon
  - We are considering letting you drop the lowest at the end of the semester
- Today: pretraining LLMs

What’s is pretraining

- Transformer is a powerful and efficient architecture. As we discussed last time, with A LOT of data, it can have impressive language ability
- So today, we will talk about HOW to obtain this impressive language ability.
  - We know “with a lot of data”, but how to actually train a model to do this

- *“Pre”*- training, implies, this is the first step before a more targeted “training”

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

Three stages of training a LLM

- Stage 1:

  Pretraining: model is trained to incrementally *predict the next word* with an enormous text corpora. The model uses the cross-entropy loss, and that loss is backpropagated all the way through the network.
  - The training data is usually (the cleaned up) text from the internet. Sometimes companies have their own secret sauce data (proprietary data).
  - The result is a model that is very good at next word prediction and can generate text.

Three stages of training a LLM

- Stage 2:

  Instruction tuning: also called supervised finetuning (SFT). Also use cross-entropy loss to follow instructions for different NLP downstream tasks: QA, summarization, writing code, machine translation, etc.
  - Train on task specific corpus

Three stages of training an LLM

- Stage 3:

  Alignment: aka preference alignment. The model is trained to be maximally helpful and less harmful.
  - The model is given preference data, which consists of a context followed by accepted and rejected continuations.
  - The model is then trained by reinforcement learning or other reward-based algorithms ()RLHF, DPO, etc.). We will cover this in future lectures.

Pretraining

- The idea pretraining an LM is the same idea of **self-training/self-supervision** in our earlier lectures, in which we mentioned in the context of word2vec embedding
  - The sentences themselves ARE the data we use to learn next word prediction, without us explicitly making labeled data
- We again use **teaching forcing**: instead of using the predicted word to continue training, we are using the ground-truth word to prevent derailing.

The data we use for pretraining

Data used for pretraining

- Mostly text from the web
- Many commercial companies have their own secret sauce data (proprietary data)
- Data can directly affect performance!
- LLM outputs will also reflect the limitations and biases in the data

Image from https://ceufast.com/blog/you-are-what-you-eat

Data used for pretraining

- Because these corpora (scraped from the web) are so large, they already contains examples we need for NLP tasks:
    - FAQ on a website: QA task
    - Translation
    - Summary
    - Etc.

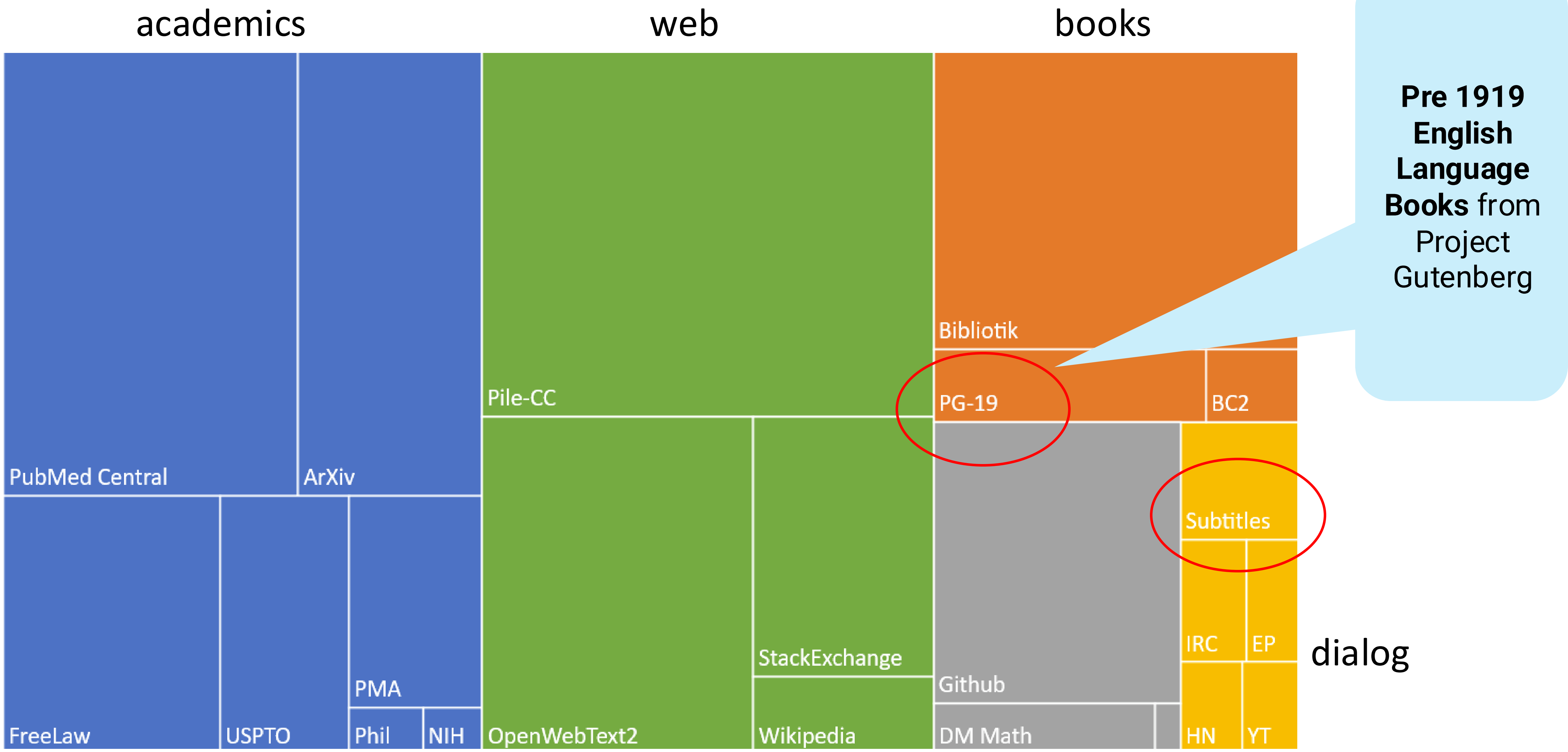
Common pretraining corpus: Common Crawl

- Text is taken from automatially-crawled web pages
- Various versions:
  - Colossal Clean Crawled Corpus (C4): 156 Billion tokens of English
    - Filtered data: deduplicated, removed non-natural language like code, removed sentences with offensive words from a blocklist
    - Consists of patent text documents, Wikipedia, and news sites
- Released in 2020

Common pretraining corpus: The Pile

- 825GB English text corpus
- Text from the web, books, and Wikipedia
- Constructed by publicly released code
- Released in 2020

The Pile: a pretraining corpus



Common pretraining corpus: Dolma

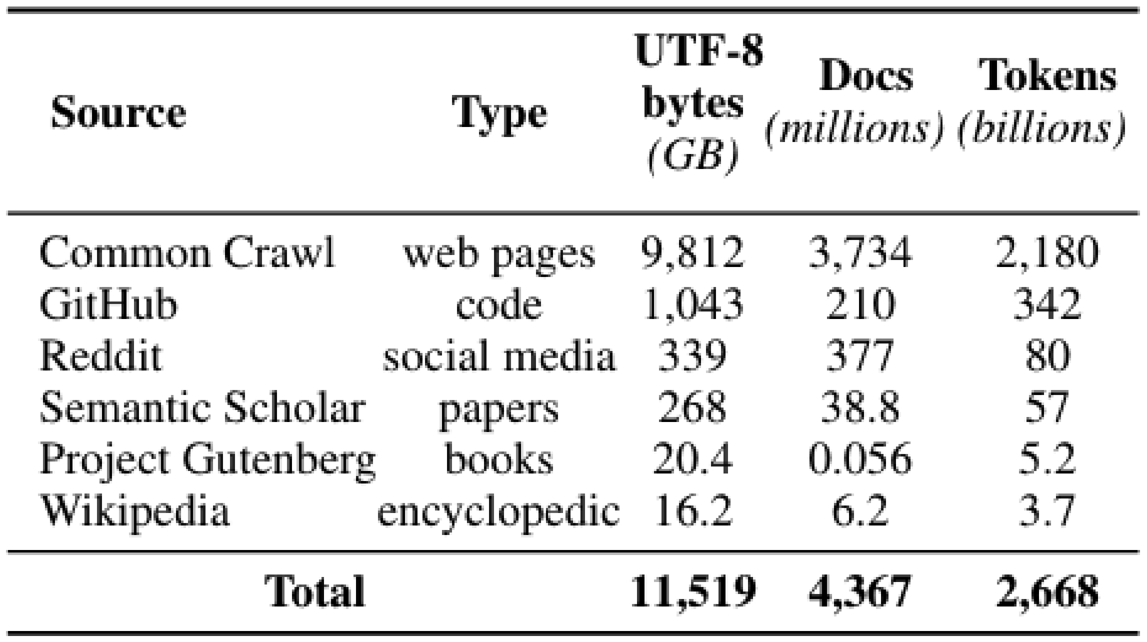
- Also English
- Created with public tools
- 3 trillion tokens
- Web text, academic papers, code, books, encyclopedic materials, social media
- More recent, released in 2024

GPT3 training data

- ~ 60% from common crawl
- ~ 22% from WebText2 (openAI’s own curated data)
- ~ 8% from Books1 (public)
- ~ 8% from Books2 (some licensed)
- ~ 3% from Wikipedia

Olmo training data

- Pretraining data: Dolma (open source, open weight)



Source	Type	UTF-8 bytes *(GB)*	Docs *(millions)*	Tokens *(billions)*
Common Crawl	web pages	9,812	3,734	2,180
GitHub	code	1,043	210	342
Reddit	social media	339	377	80
Semantic Scholar	papers	268	38.8	57
Project Gutenberg	books	20.4	0.056	5.2
Wikipedia	encyclopedic	16.2	6.2	3.7
Total		**11,519**	**4,367**	**2,668**

Table 2: Composition of Dolma. Tokens counts are based on the GPT-NeoX tokenizer.

https://arxiv.org/abs/2402.00838

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

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

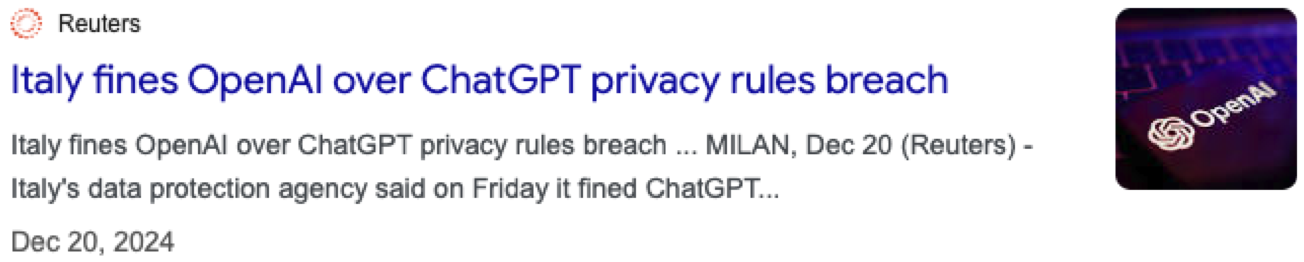
Privacy:

- 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

Safety and privacy

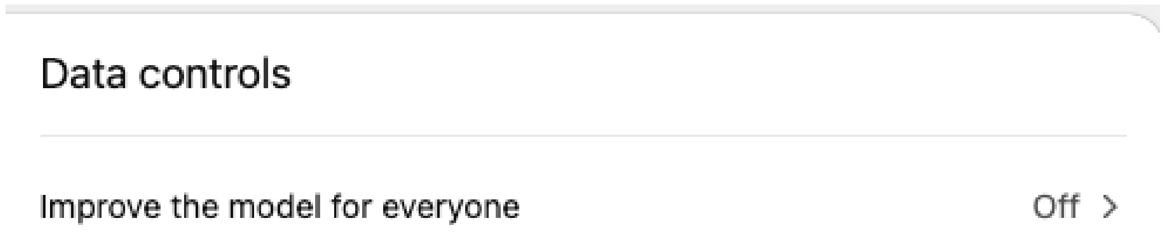

Reuters

Italy fines OpenAI over ChatGPT privacy rules breach

Italy fines OpenAI over ChatGPT privacy rules breach ... MILAN, Dec 20 (Reuters) - Italy's data protection agency said on Friday it fined ChatGPT...

Dec 20, 2024

- Scenario 1: If personal data is in the training data, it’s really difficult to remove that knowledge from an LLM
- Scenario 2: Commercial models often use the users chat history to train their model. Therefore, personal details, names, locations might be leaked
- Scenario 3: Teens are still learning how to evaluate information critically, therefore more likely to be subjected to misinformation.
- Others: emotional dependence

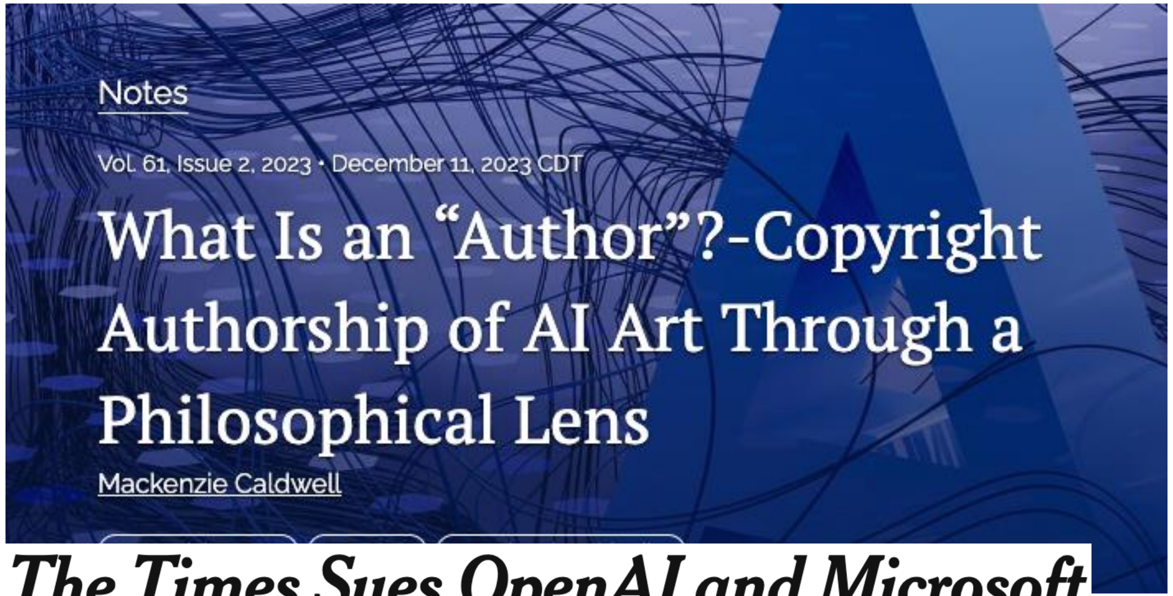

Data controls

Improve the model for everyone Off

Copyright

US authors' copyright lawsuits against OpenAI and Microsoft combined in New York with newspaper actions

California cases over AI trainers' use of work by writers including Ta-Nehisi Coates and Michael Chabon transferred to consolidate with New York suits from John Grisham and Jonathan Franzen and more


Notes

Vol. 61, Issue 2, 2023 · December 11, 2023 CDT

What Is an "Author"?-Copyright Authorship of AI Art Through a Philosophical Lens

Mackenzie Caldwell

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.

The New York Times

Anthropic Agrees to Pay $1.5 Billion to Settle Lawsuit With Book Authors

The settlement is the largest payout in the history of U.S. copyright cases and could lead more A.I. companies to pay rights holders for use...

1 month ago

Finetuning

For a new domain

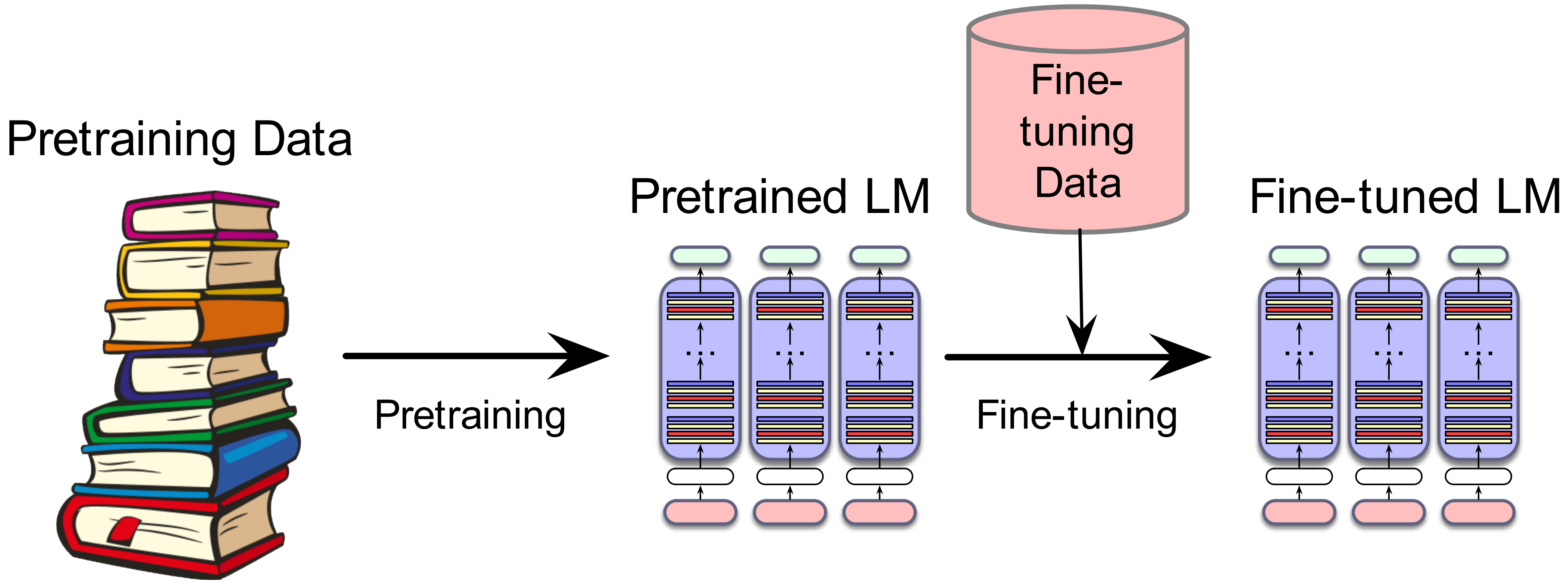
Finetuning for adaptation to new domains

- What happens if we need our LLM to work well on a domain it didn't see in pretraining?
- Perhaps some specific medical or legal domain?
- Or maybe a multilingual LM needs to see more data on some language that was rare in pretraining?

Finetuning

- We continue training on domain-specific data or any relevant data for your task
- This process of taking a fully pre-trained model and running additional training passes using the cross-entropy loss on some new data is called finetuning
- During finetuning, some or all of its parameters will be adapted to some new data.
- Sometimes called **continued pretraining**

Finetuning


Pretraining Data
Pretraining
Pretrained LM
Fine-tuning Data
Fine-tuning
Fine-tuned LM

Parameter Efficient Fine Tuning

- LLMs are large, if we tune all the parameters, it’s very costly in terms of memory and time.
- Alternatively, we only fine-tune some parameters → **parameter-efficient fine tuning (PEFT)**
  - One of the PEFT methods is **called Low-Rank Adaptation (LoRA)**

LoRA-tuning

- Recall that transformers has many dense layers that perform matrix multiplication, e.g. W^Q, W^K, W^V, W^O in the attention computation
- Instead of updating these layers during finetuning, we freeze these layers, and instead, *update a low-rank approximation that has fewer parameters*.

LoRA

- Consider a matrix W of dimensionality $[N \times d]$, it needs to be updated during finetuning via gradient descent
- Normally we update using $W = W_0 + \Delta W$
- In LoRA, we instead update two matrices A and B, where A has size $[N \times r]$, and B has size $[r \times d]$, and $\Delta W = AB$
    - We choose r to be really small, $r \ll \min(d, N)$
- W_o is the frozen base weight from the pretrained model, and $\Delta W = AB$ is learned low-rank correction

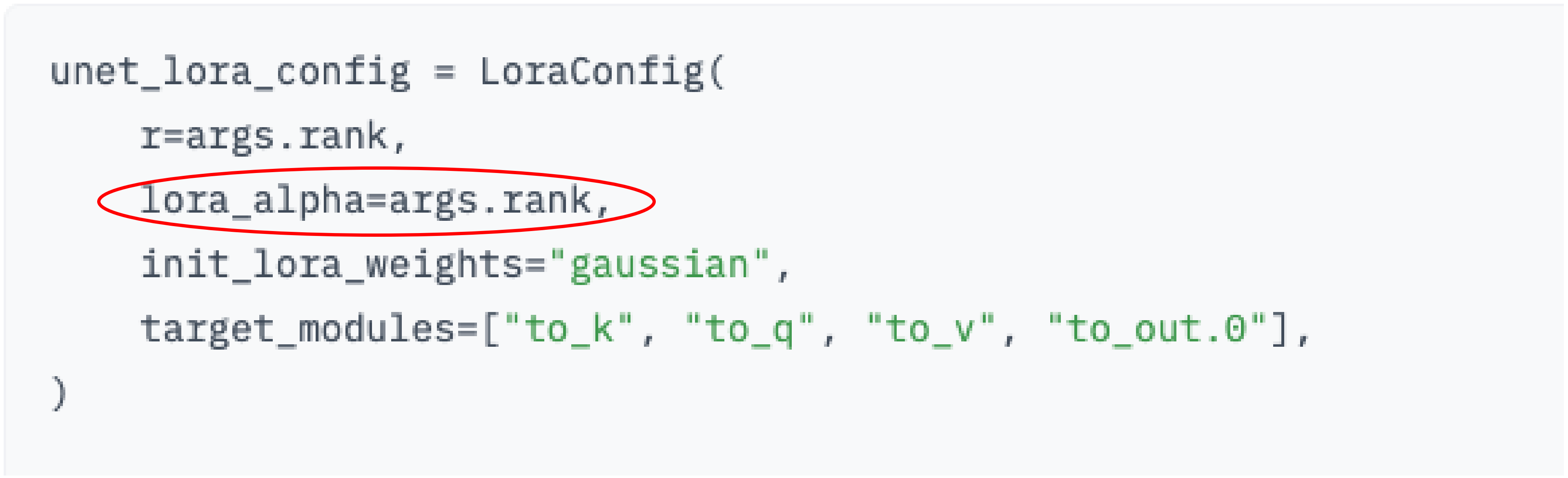
- We basically decompose weight update to a low-rank form

LoRA

- You need to decide which weight matrices will have low-rank adapters
  - E.g. common choices are W^Q, W^K, W^V, sometimes the feed-forward layers
  - For each chosen matrix, LoRA adds trainable A, B matrices such that $\Delta W = AB$

- Recall the dimensionality: N x r, r x d = N x d
  - How to decide r?
    - Lower r: faster!
    - Higher r: more parameters

LoRA in huggingface



```
unet_lora_config = LoraConfig(
    r=args.rank,
    lora_alpha=args.rank,
    init_lora_weights="gaussian",
    target_modules=["to_k", "to_q", "to_v", "to_out.0"],
)
```

$$W = W_0 + \frac{\alpha}{r} AB$$

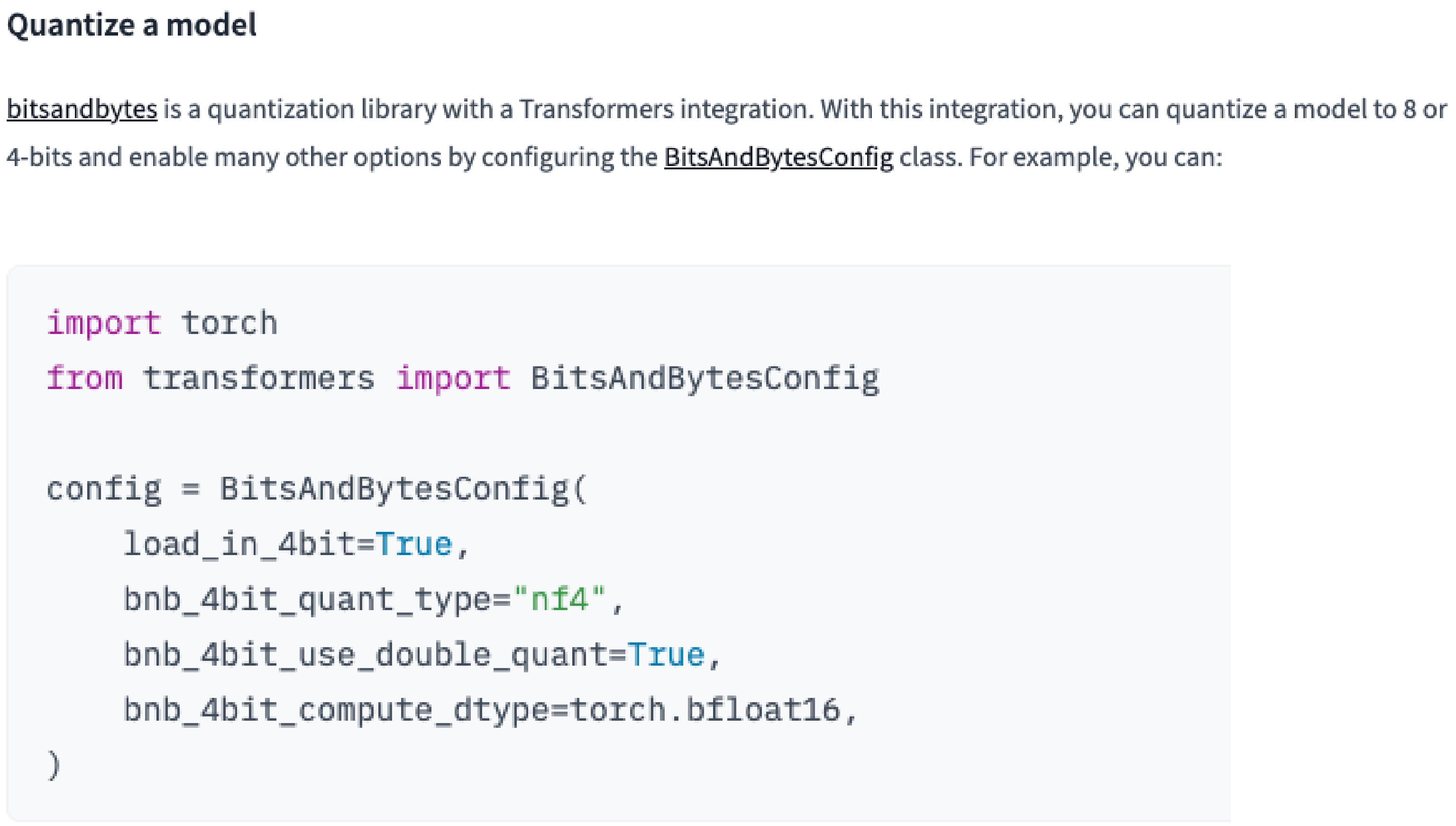
Other fine-tuning methods

Quantized LoRA-tuning (QLoRA), a variation of LoRA

- Even more memory-efficient
- First, pretrained model is quantized to reduce memory and GPU requirement
  - E.g. full preicision FP16 to 4 bit
- Then on top of this, quantized model, we use LoRA

- So now you can finetuning an even bigger model

From huggingface



Quantize a model

bitsandbytes is a quantization library with a Transformers integration. With this integration, you can quantize a model to 8 or 4-bits and enable many other options by configuring the BitsAndBytesConfig class. For example, you can:

```python
import torch
from transformers import BitsAndBytesConfig

config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_use_double_quant=True,
    bnb_4bit_compute_dtype=torch.bfloat16,
)
```

https://huggingface.co/docs/peft/main/en/developer_guides/quantization

And many other PEFT

- Prefix tuning
- Prompt-tuning
- Not covered in this class, but you can read about it if you are interested

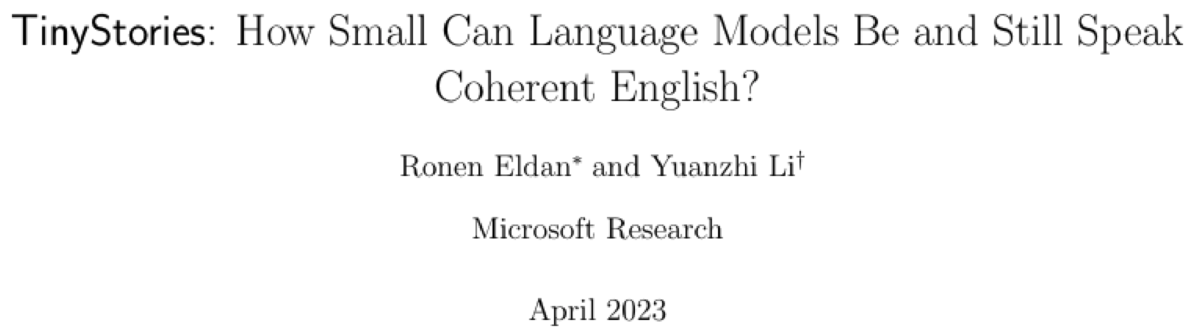
When to train when to fine-tune

How to decide the best strategy for your project or your future NLP work

So many names, what’s the difference

- Pretraining
- Fine-tuning
- Training from scratch

Strategy


TinyStories: How Small Can Language Models Be and Still Speak Coherent English?

Ronen Eldan* and Yuanzhi Li†

Microsoft Research

April 2023

1. What's the model?
   a) If it's already pretrained, you can usually just finetune
   b) If you want to build your own model (say, a transformer), see second step
2. What's the size of the model?
3. What's your task? What's the data? Does it match the domain coverage of the training data?

Advice

Generally, I don’t recommend training any kind of transformers from scratch or doing your own pretraining.

It takes a lot to even get good language abilities. And probably not possible with free-tier Google Colab. But there are exceptions, so it depends on your task!

I highly recommend LoRA or other kinds of finetuning.

If you haven’t heard of Hugging face, it’s a great place to look for models and datasets.

Have a nice weekend!