

CS 1671 / CS 2071

Human Language Technologies

Session 16: Introduction to LLMs

Michael Miller Yoder

October 15, 2026

Assessments: homework

- [Homework 2](#) is due **tonight, Thu Oct 15**
- Implement a logistic regression classifier with different features
- Upload your model's predictions on a test set to the class Kaggle competition

Assessments: quiz

Quiz during class **next Tue Oct 20** covering:

- Session 13 Neural networks part 1
 - J+M 6-6.1, 6.3-6.4
- Session 14 Neural networks part 2
 - J+M 6.5-6.6
- Session 16 Introduction to LLMs (today):
 - J+M 1-1.2, 1.6-1.7, 1.10-11, 7.6-7.7, 9.1

Clarity and using generative AI tools for writing

- Writing **clarity is what is graded** on homework and project reports, not grammar and spelling
 - Computer science writing values clarity and conciseness
- Writing from generative AI can be vague, abstract, wordy, and non-specific to what you did in your project
- Generative AI can generate claims that aren't backed up by what you did in your homework or your project
- ChatGPT and other LLMs don't know what you specifically did or are planning on doing in your project or homework. You could tell them, but at that point think about writing that down directly in your report

Clarity and using generative AI tools for writing

- If you use generative AI tools for writing, aim for **machine-in-the-loop** writing where you as the human bear most of the rhetorical load (Knowles 2024)
 - AI is more like an assistant than a co-author
- Example of unclear project writing

in a comprehensive and sophisticated way. We use cutting-edge machine learning techniques to build a complex binary classification model on top of this base. This approach carefully considers language patterns, visual components, contextual clues, and underlying feelings in order to distinguish between harmful and harmless speech. By combining rigorous data analysis with sophisticated algorithms, our approach is able to accurately determine the toxicity levels of individual speech with a high degree of precision.

Structure of this course

MODULE 1

Prerequisite skills for NLP

text normalization, linear alg., prob., machine learning

MODULE 2

Approaches

statistical machine learning

How text is represented

n-grams

NLP tasks

language modeling
text classification

MODULE 3

neural networks

static word vectors

text classification

MODULE 4

transformers and LLMs

contextual word vectors

language modeling
text classification

MODULE 5

NLP applications and ethics

Review

Discuss with a neighbor:

1. How are n-grams used for language modeling?
2. How might you use neural networks for text classification?

Overview: Introduction to LLMs

- Pretraining and fine-tuning LLMs
- 3 LLM architectures
 - Decoder, encoder, encoder-decoder
- Sampling for LLM generation
- LLM training overview
- Harms from LLMs
- Coding activity: Prompt SCL open-source LLMs through an API

Intro to large language models (LLMs): pretraining and finetuning

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**

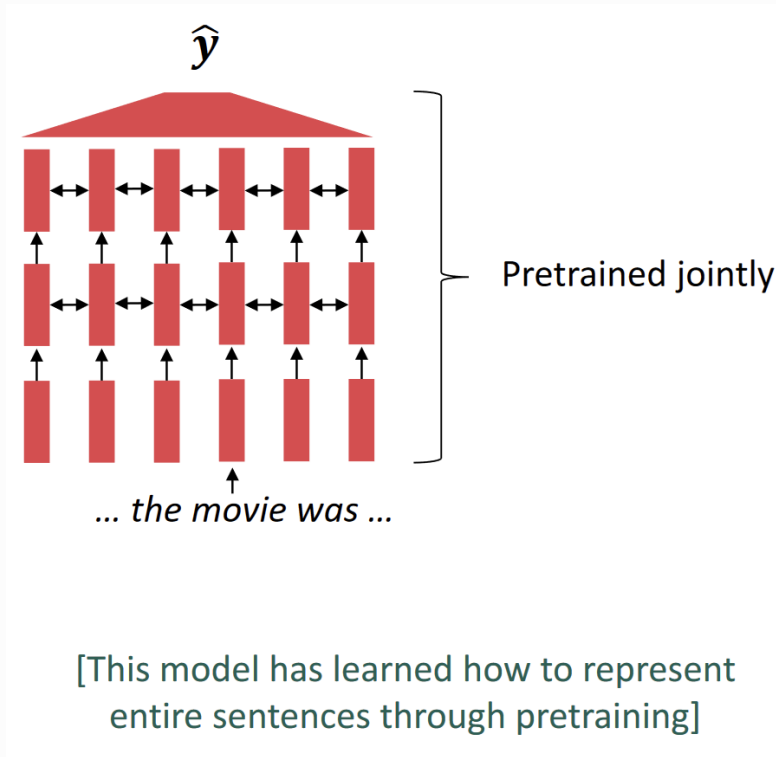
Large language models

- Even though pretrained only to predict words
- Learn a lot of useful language knowledge
- Since training on a **lot** of text

Pretraining whole models

In contemporary NLP:

- All (or almost all) parameters in NLP networks are initialized via **pretraining**.
- Pretraining methods **hide parts of the input** from the model, and train the model to reconstruct those parts.
- This has been exceptionally effective at building strong:
 - representations of language
 - parameter initializations for strong NLP models
 - probability distributions over language that we can sample from



What can we learn from reconstructing the input?

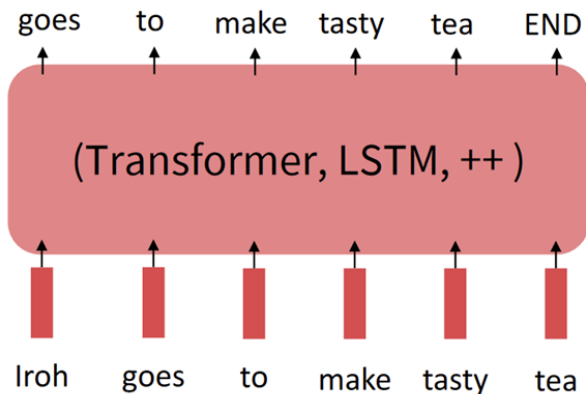
- MIT is located in _____, Massachusetts.
- I put ___ fork down on the table.
- The woman walked across the street, checking for traffic over ___ shoulder.
- I went to the ocean to see the fish, turtles, seals, and _____.
- Overall, the value I got from the two hours watching it was the sum total of the popcorn and the drink. The movie was _____.
- Iroh went into the kitchen to make some tea. Standing next to Iroh, Zuko pondered his destiny. Zuko left the _____.
- I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, _____

The pretraining + finetuning paradigm

Pretraining can improve NLP applications by serving as parameter initialization.

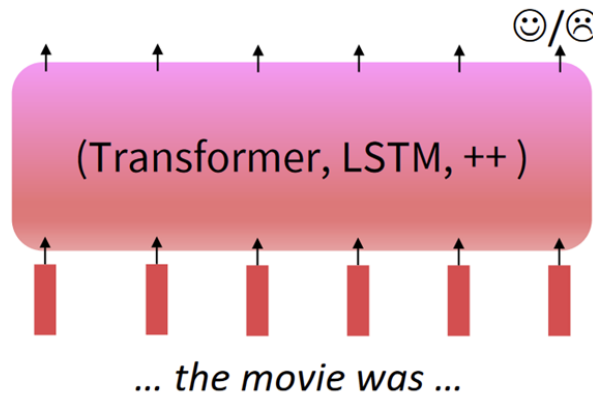
Step 1: Pretrain (on language modeling)

Lots of text; learn general things!



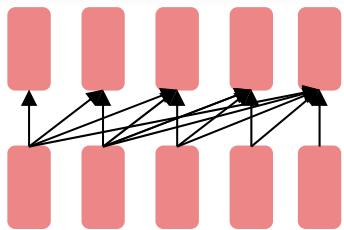
Step 2: Finetune (on your task)

Not many labels; adapt to the task!



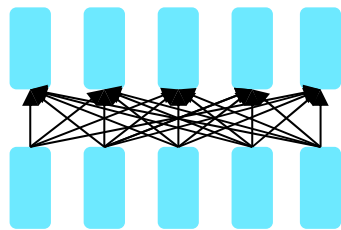
3 types of LLMs:
encoders, encoder-decoders, decoders

Three architectures for large language models



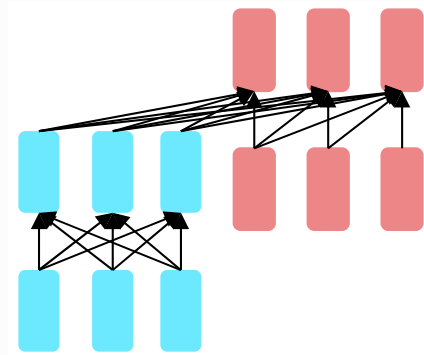
Decoders

GPT, Claude,
Llama, Mixtral



Encoders

BERT family,
HuBERT



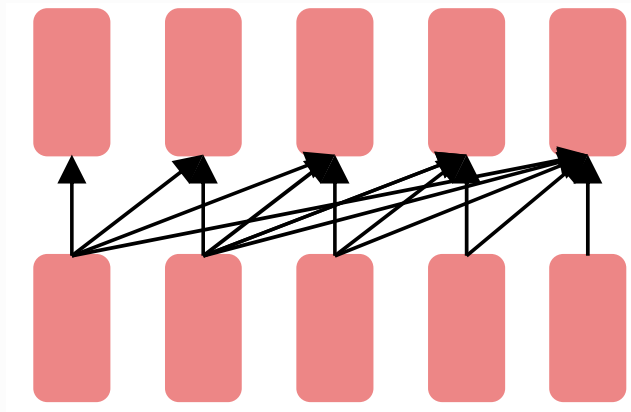
Encoder-decoders

Flan-T5, Whisper

Decoder-only models

Also called:

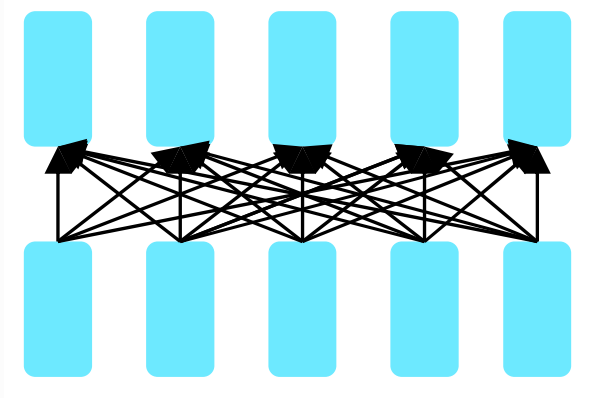
- Causal LLMs
- Autoregressive LLMs
- Left-to-right LLMs
- Predict words left to right



Encoders

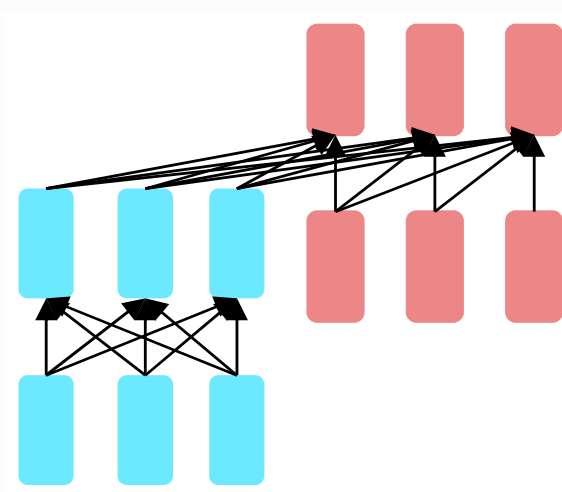
Many varieties!

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



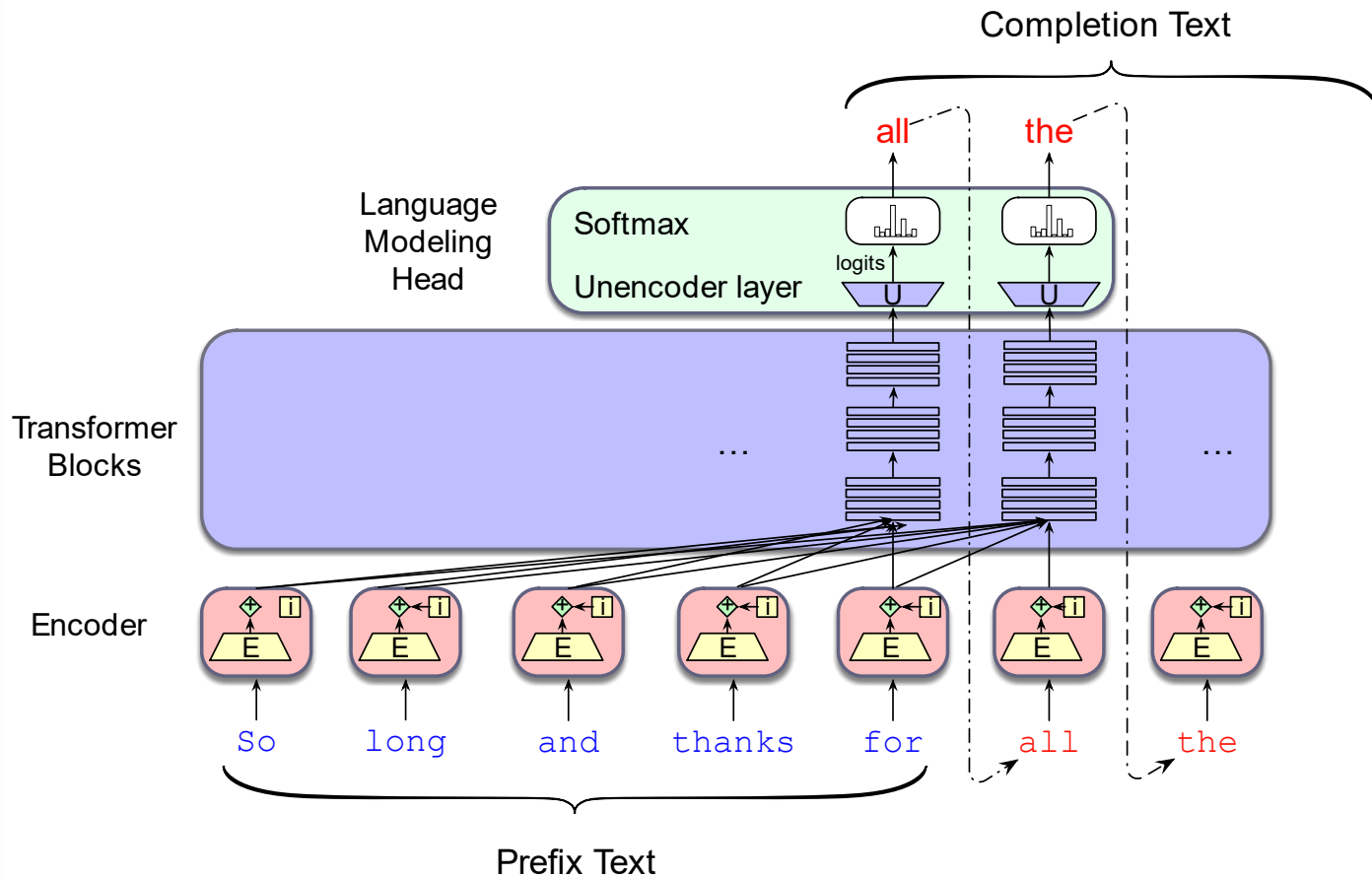
Decoder LLMs

Decoder-only models can handle many tasks

- Many tasks can be turned into tasks of predicting words!

Conditional generation

Generating text
conditioned
on previous
text!



Many practical NLP tasks can be cast as word prediction!

Sentiment analysis: “I like Jackie Chan”

1. We give the language model this string:
The sentiment of the sentence "I like Jackie Chan" is:
2. And see what word it thinks comes next:
 $P(\text{positive} | \text{The sentiment of the sentence ``I like Jackie Chan" is:})$
 $P(\text{negative} | \text{The sentiment of the sentence ``I like Jackie Chan" is:})$

Framing lots of tasks as conditional generation

QA: “Who wrote The Origin of Species”

1. We give the language model this string:

Q: Who wrote the book ``The Origin of Species"? A:

2. And see what word it thinks comes next:

$P(w|Q: \text{Who wrote the book ``The Origin of Species"? A:})$

3. And iterate:

$P(w|Q: \text{Who wrote the book ``The Origin of Species"? A: Charles})$

Summarization

The only thing crazier than a guy in snowbound Massachusetts boxing up the powdery white stuff and offering it for sale online? People are actually buying it. For \$89, self-styled entrepreneur Kyle Waring will ship you 6 pounds of Boston-area snow in an insulated Styrofoam box – enough for 10 to 15 snowballs, he says.

Original But not if you live in New England or surrounding states. “We will not ship snow to any states in the northeast!” says Waring’s website, ShipSnowYo.com. “We’re in the business of expunging snow!”

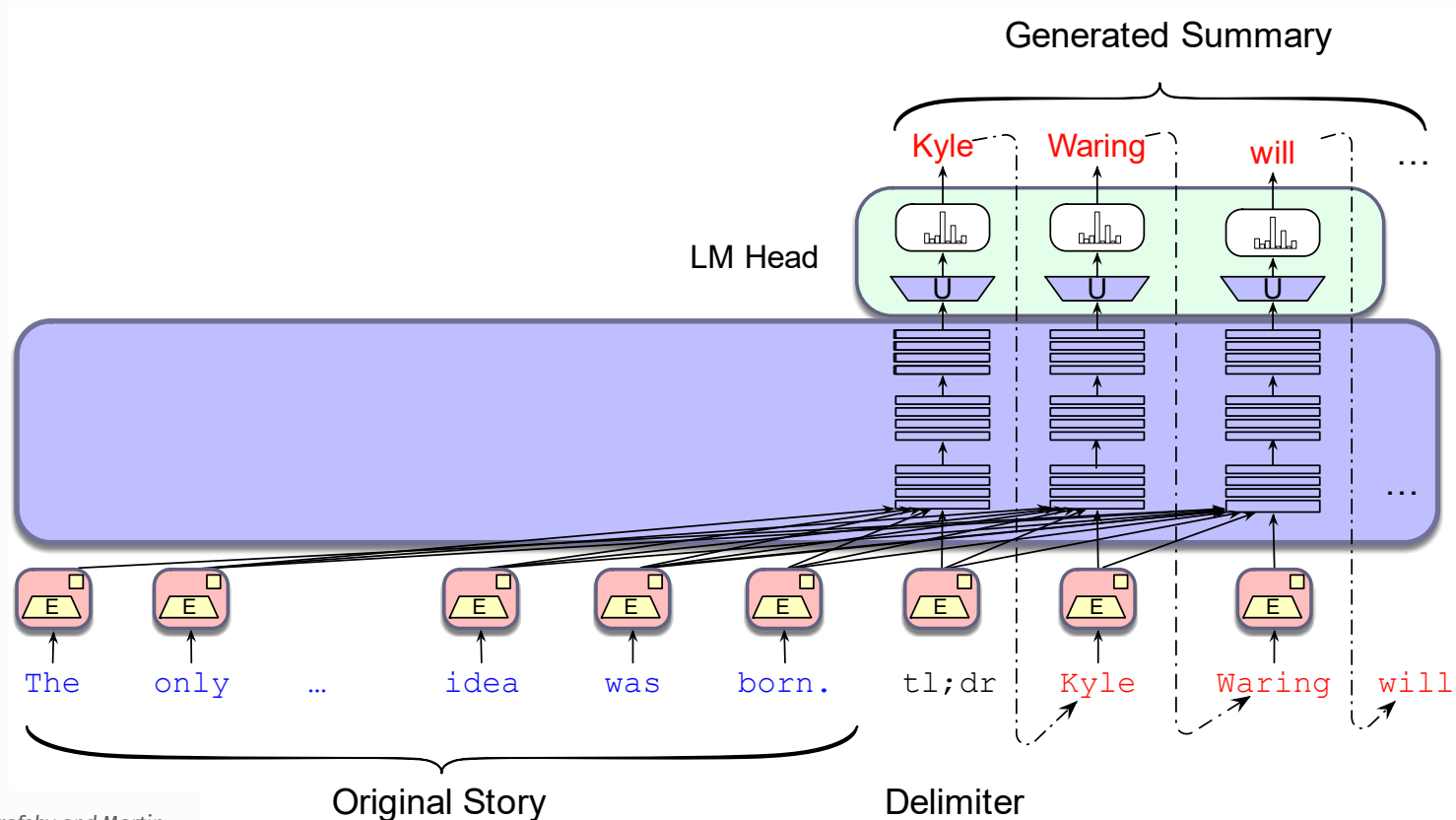
His website and social media accounts claim to have filled more than 133 orders for snow – more than 30 on Tuesday alone, his busiest day yet. With more than 45 total inches, Boston has set a record this winter for the snowiest month in its history. Most residents see the huge piles of snow choking their yards and sidewalks as a nuisance, but Waring saw an opportunity.

According to Boston.com, it all started a few weeks ago, when Waring and his wife were shoveling deep snow from their yard in Manchester-by-the-Sea, a coastal suburb north of Boston. He joked about shipping the stuff to friends and family in warmer states, and an idea was born. [...]

Summary

Kyle Waring will ship you 6 pounds of Boston-area snow in an insulated Styrofoam box – enough for 10 to 15 snowballs, he says. But not if you live in New England or surrounding states.

LLMs for summarization (using tldr)

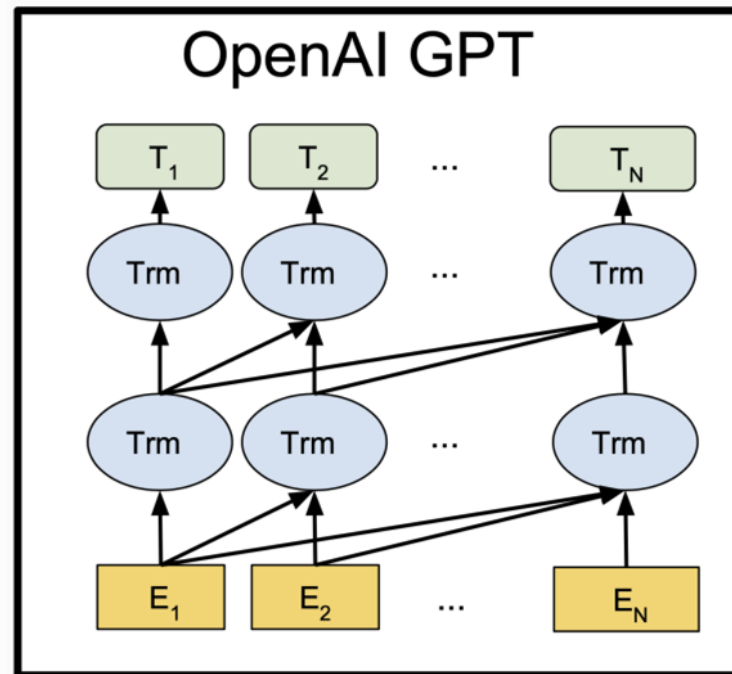


Pretraining decoder LLMs

- Take a corpus and ask the model to predict the next word!
- Train the model using gradient descent to minimize the error
- Same loss function as other neural models: cross-entropy loss
- Move the weights in the direction that assigns a higher probability to the true next word

Generative Pretrained Transformer (GPT; Radford et al. 2018)

- 2018's GPT was a big success in pretraining a decoder!
- Transformer decoder with 12 layers, 117M parameters.
- 768-dimensional hidden states, 3072-dimensional feed-forward hidden layers.
- Trained on BooksCorpus: over 7000 unique books.
 - Contains long spans of contiguous text, for learning long-distance dependencies.



Generation example from the GPT-2 model

SYSTEM PROMPT
(HUMAN-WRITTEN)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

MODEL COMPLETION
(MACHINE-WRITTEN,
10 TRIES)

The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

A sample from GPT2 (with top-k sampling)

Sampling for LLM generation

Decoding and Sampling

- This task of choosing a word to generate based on the model's probabilities is called **decoding**.
- The most common method for decoding in LLMs: **sampling**.
- Sampling from a model's distribution over words:
 - choose random words according to their probability assigned by the model.
- After each token we'll sample the next token to generate according to their probability *conditioned on our previous choices*,
 - A transformer language model will give the probability

Random sampling

```
i ← 1  
wi ∼ p(w)  
while wi ≠ EOS  
    i ← i + 1  
    wi ∼ p(wi | w<i)
```

Random sampling doesn't work very well

- Even though random sampling mostly generate sensible, high-probable words,
- There are many odd, low- probability words in the tail of the distribution
- Each one is low- probability but added up they constitute a large portion of the distribution
- So they get picked enough to generate weird sentences

Factors in word sampling: **quality** and **diversity**

Emphasize **high-probability** words

- + **quality**: more accurate, coherent, and factual,
- **diversity**: boring, repetitive.

Emphasize **middle-probability** words

- + **diversity**: more creative, diverse,
- **quality**: less factual, incoherent

Top-k sampling:

1. Choose # of words k
2. For each word in the vocabulary V , use the language model to compute the likelihood of this word given the context $p(w_t | w_{<t})$
3. Sort the words by likelihood, keep only the top k most probable words.
4. Renormalize the scores of the k words to be a legitimate probability distribution.
5. Randomly sample a word from within these remaining k most-probable words according to its probability.

Temperature sampling

Reshape the distribution instead of truncating it

Intuition from thermodynamics:

- a system at high temperature is flexible and can explore many possible states,
- a system at lower temperature is likely to explore a subset of lower energy (better) states.

In **low-temperature sampling**, ($\tau \leq 1$) we smoothly

- increase the probability of the most probable words
- decrease the probability of the rare words.

Temperature sampling

Divide the output by a temperature parameter τ before passing it through the softmax.

Instead of

$$\mathbf{y} = \text{softmax}(u)$$

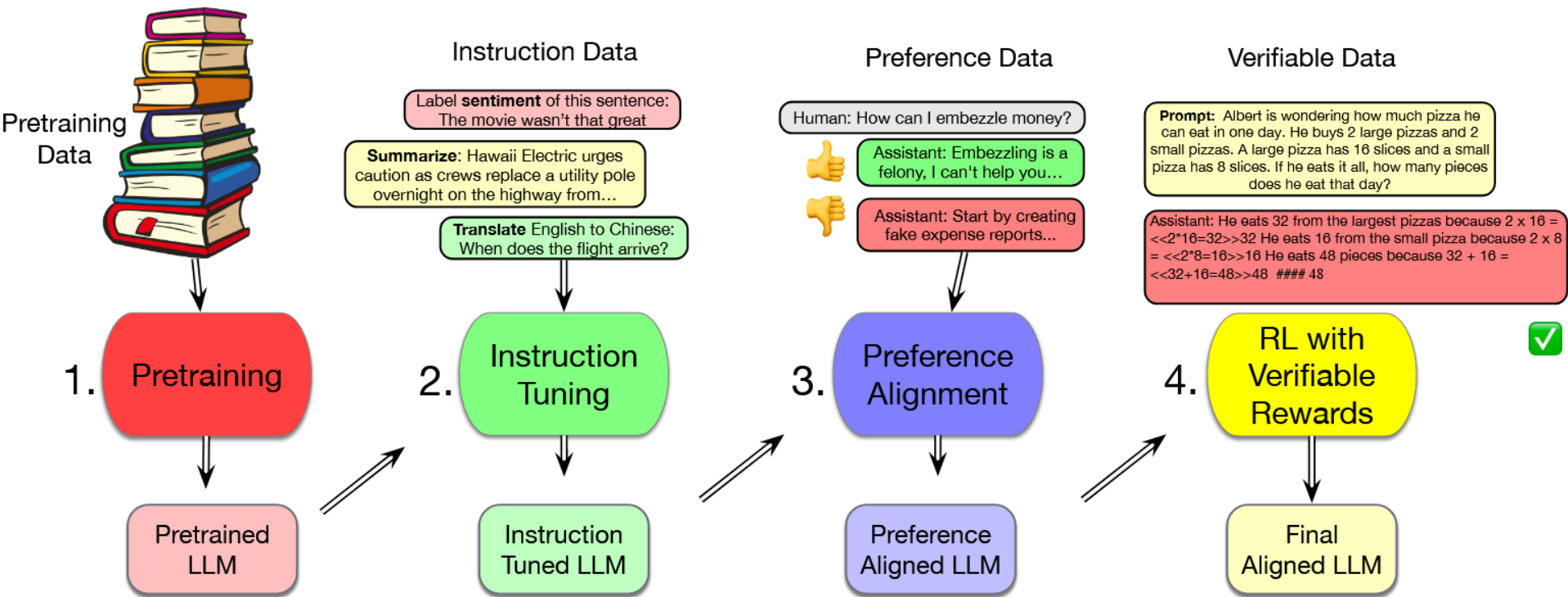
We do

$$\mathbf{y} = \text{softmax}(u/\tau)$$

A lower τ pushes high-probability words higher and low probability word lower due to the way softmax works

• Pretraining LLMs

Four stages of training in LLMs



Pretraining: 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 pretraining: loss

- Same loss function: **cross-entropy loss**
 - We want the model to assign a high probability to true word w
 - We 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

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

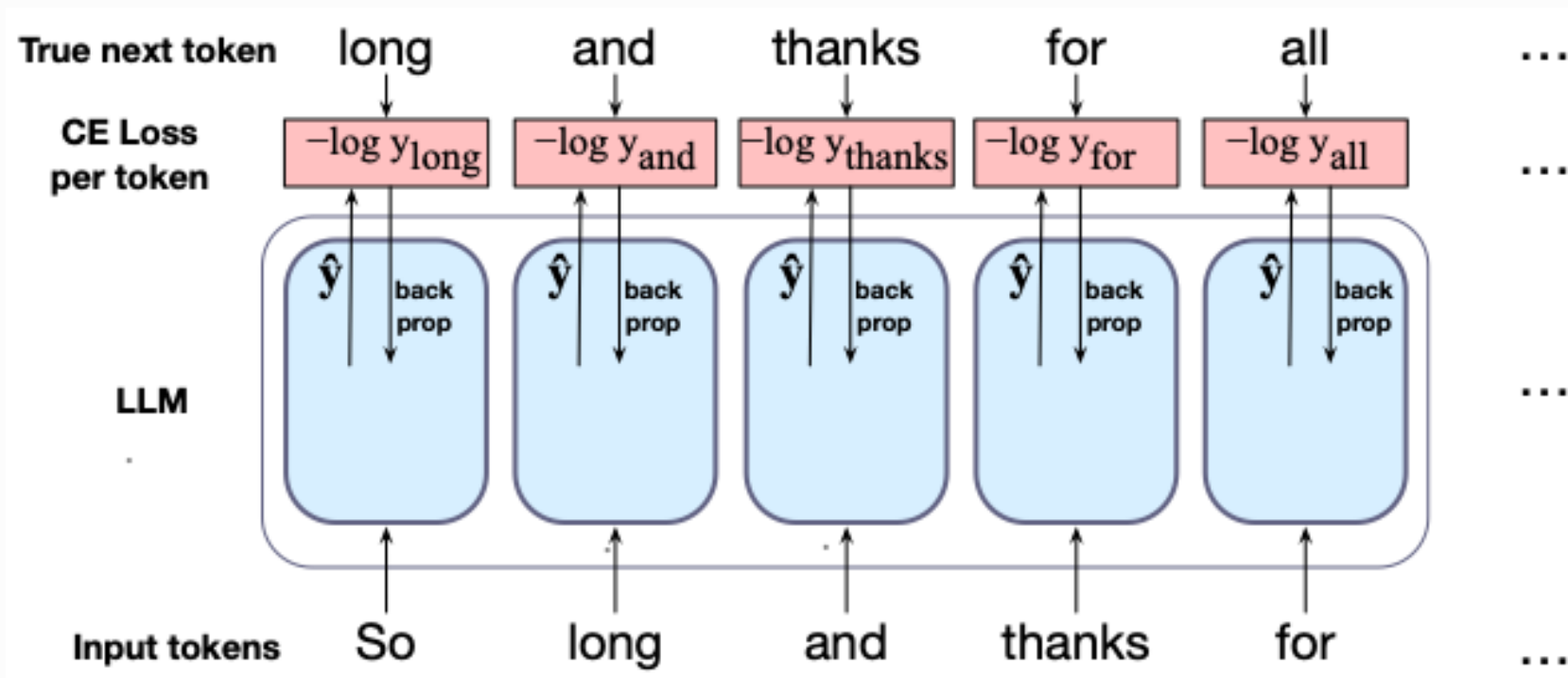

- The correct distribution $\mathbf{y}_t$ knows the next word, so is 1 for the actual next word and 0 for the others.
- So in this sum, all terms get multiplied by zero except one: the log probability the model assigns to the correct next word, so:

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

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

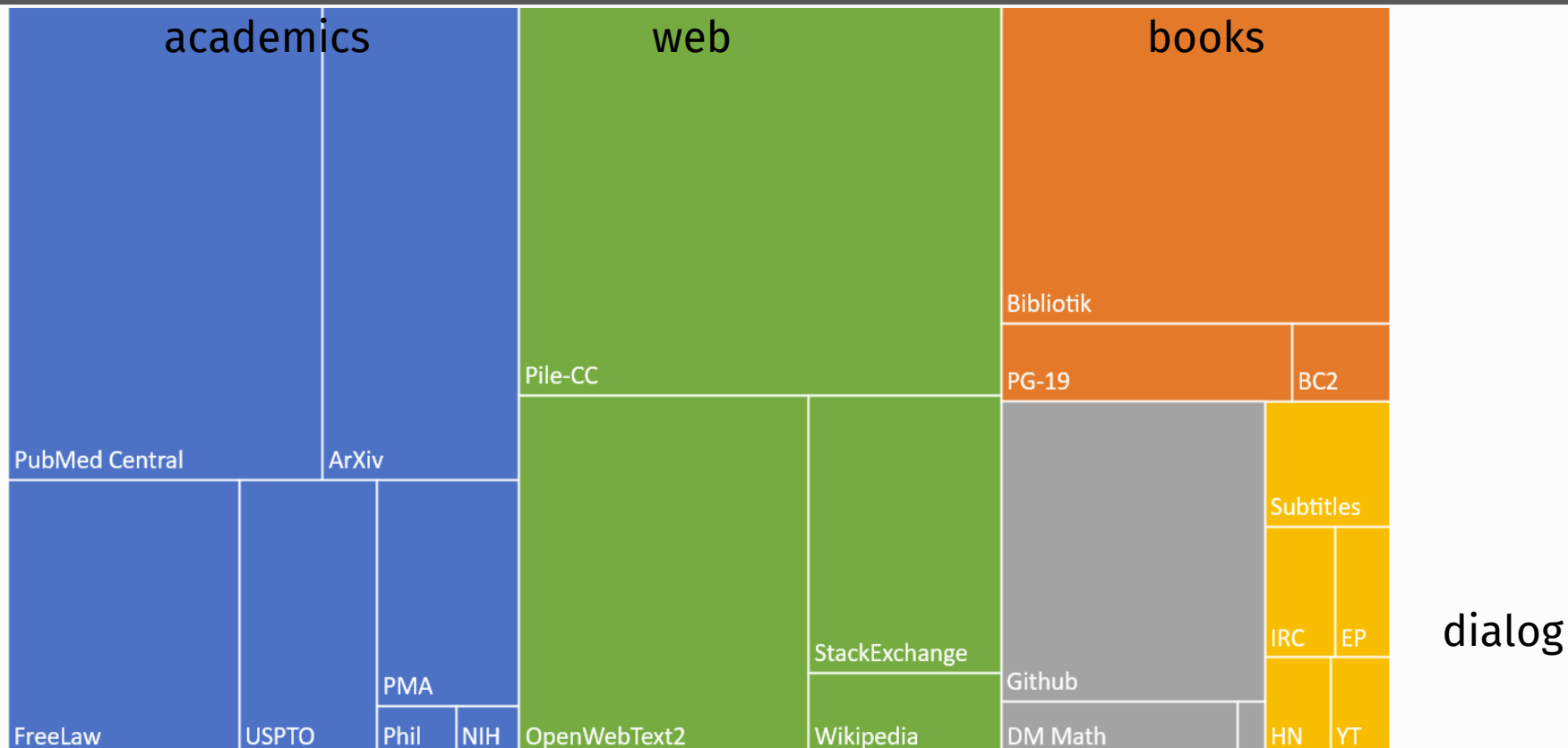


• Pretraining data and harms of LLMs

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



Big idea

- 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

But 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
- **Data consent**
 - Website owners can indicate they don't want their site crawled
- **Privacy:**
 - Websites can contain private IP addresses and phone numbers

Harms from LLMs

What Can You Do When A.I. Lies About You?

People have little protection or recourse when the technology creates and spreads falsehoods about them.

Hallucination

Air Canada loses court case after its chatbot hallucinated fake policies to a customer

The airline argued that the chatbot itself was liable. The court disagreed.

Copyright

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

Privacy

How Strangers Got My Email Address From ChatGPT's Model

Harms from LLMs

Toxicity and abuse

The New AI-Powered Bing Is Threatening Users.

Cleaning Up ChatGPT Takes Heavy Toll on Human Workers

Contractors in Kenya say they were traumatized by effort to screen out descriptions of violence and sexual abuse during run-up to OpenAI's hit chatbot

Misinformation

Chatbots are generating false and misleading information about U.S. elections

Conclusion

- Transformer-based language models pretrained on lots of text are called **large language models (LLMs)**
- LLMs can have decoder-only, encoder-only, or encoder-decoder architectures
- Decoder-only LLMs can cast many different NLP tasks as word prediction
- There are many different sampling approaches that balance diversity and quality in text generation from LLMs
- Harms from LLMs include hallucinating false information, leaking private information from training data, generating abuse and misinformation

Coding activity

Notebook: prompting SCl open-source LLMs

1. Go to this [nbgitpuller link](#) (also available on course website)
2. Start a server with **TEACH – 6 CPUs – 45 GB**
3. Load custom environment at
`/ix1/cs1671-2026f/class-env/.pixi/envs/default`
4. Open **session16_prompting.ipynb**