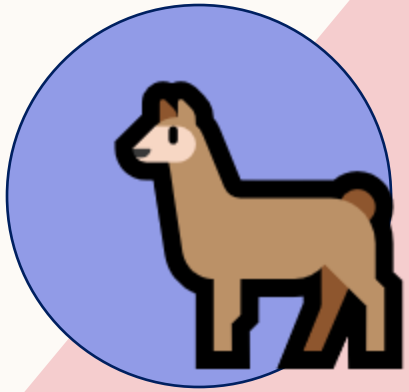
Introduction to Building LLMs

CS229: Machine Learning

Yann Dubois | Aug. 13th 2024

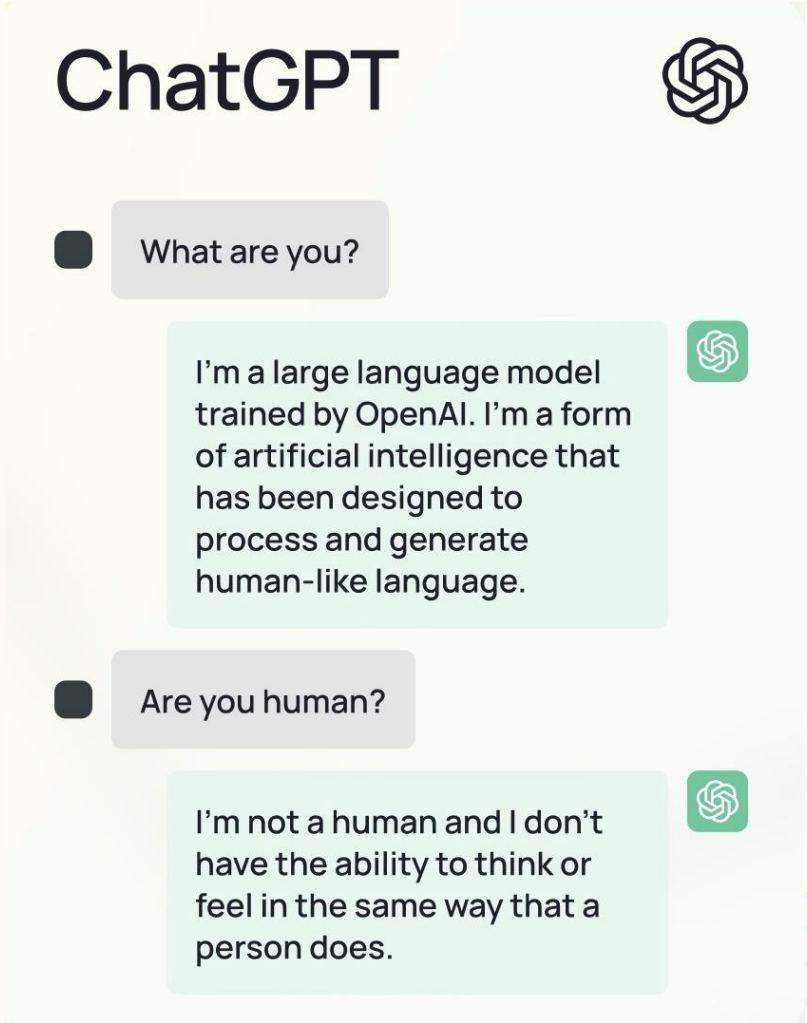
Slides partially based on CS336, CS224N, CS324



LLMs

- LLMs & chatbots took over the world

- How do they work?



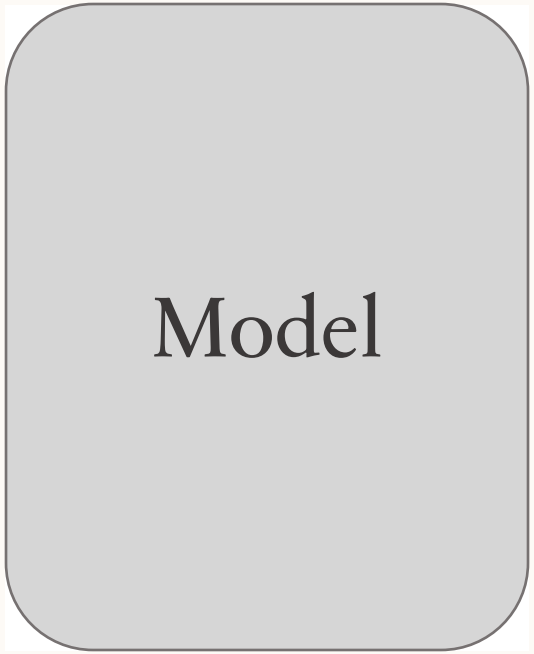
What matters when training LLMs

- Architecture
- Training algorithm/loss

Most of academia

- Data
- Evaluation
- Systems

What matters in practice

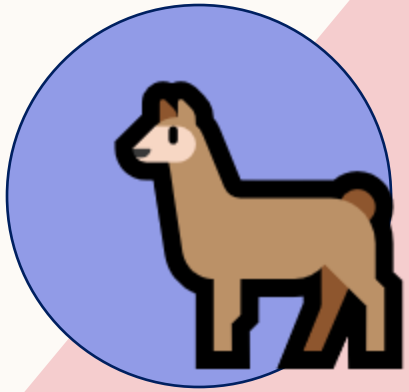


Overview

Pretraining -> GPT3

- Task & loss

Post-training -> ChatGPT



Language Modeling

- LM: probability distribution over sequences of tokens/words $p(x_1, \dots, x_L)$

P(the, mouse, ate, the, cheese) = 0.02

P(the, the, mouse, ate, cheese) = 0.0001 Syntactic knowledge

P(the, cheese, ate, the, mouse) = 0.001 Semantic knowledge

- LMs are generative models: $x_{1:L} \sim p(x_1, \dots, x_L)$
- **Autoregressive (AR) language models:**

that's the auto-regressive part-

$$p(x_1, \dots, x_L) = p(x_1)p(x_2|x_1)p(x_3|x_2, x_1)\dots = \prod_i p(x_i|x_{1:i-1})$$

No approx: chain rule of probability

=> You only need a model that can predict the next token given past context!

AR Language Models

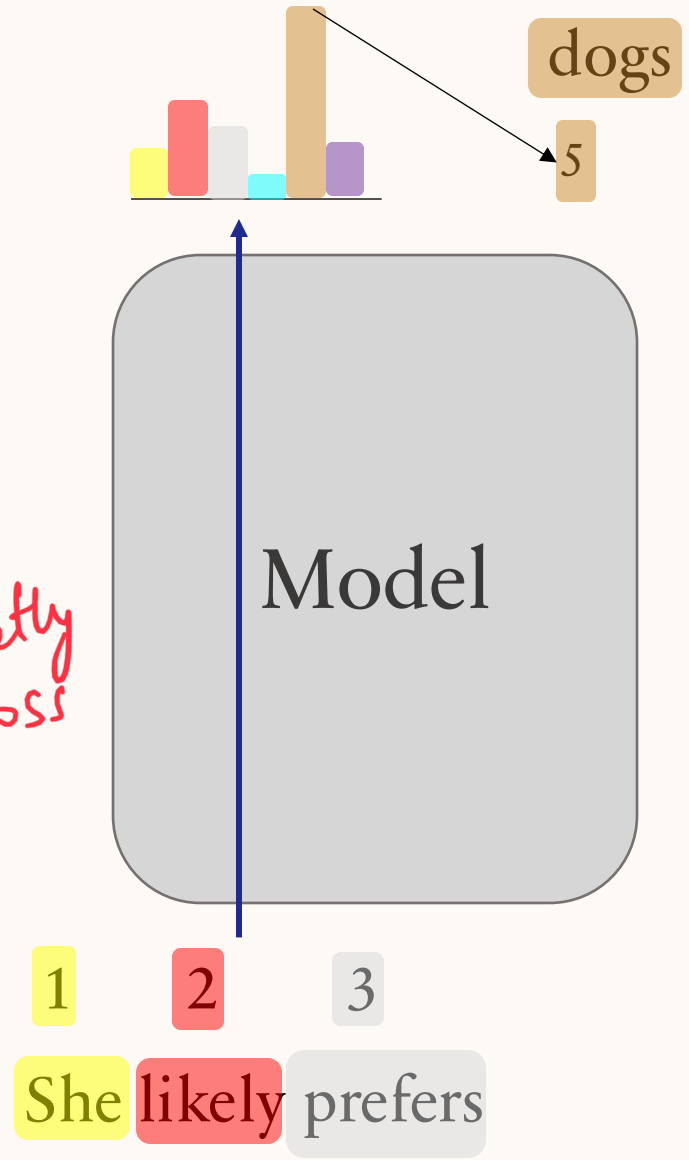
- Task: predict the next word
- Steps:

1. tokenize
2. forward
3. predict probability of next token
4. sample
5. detokenize

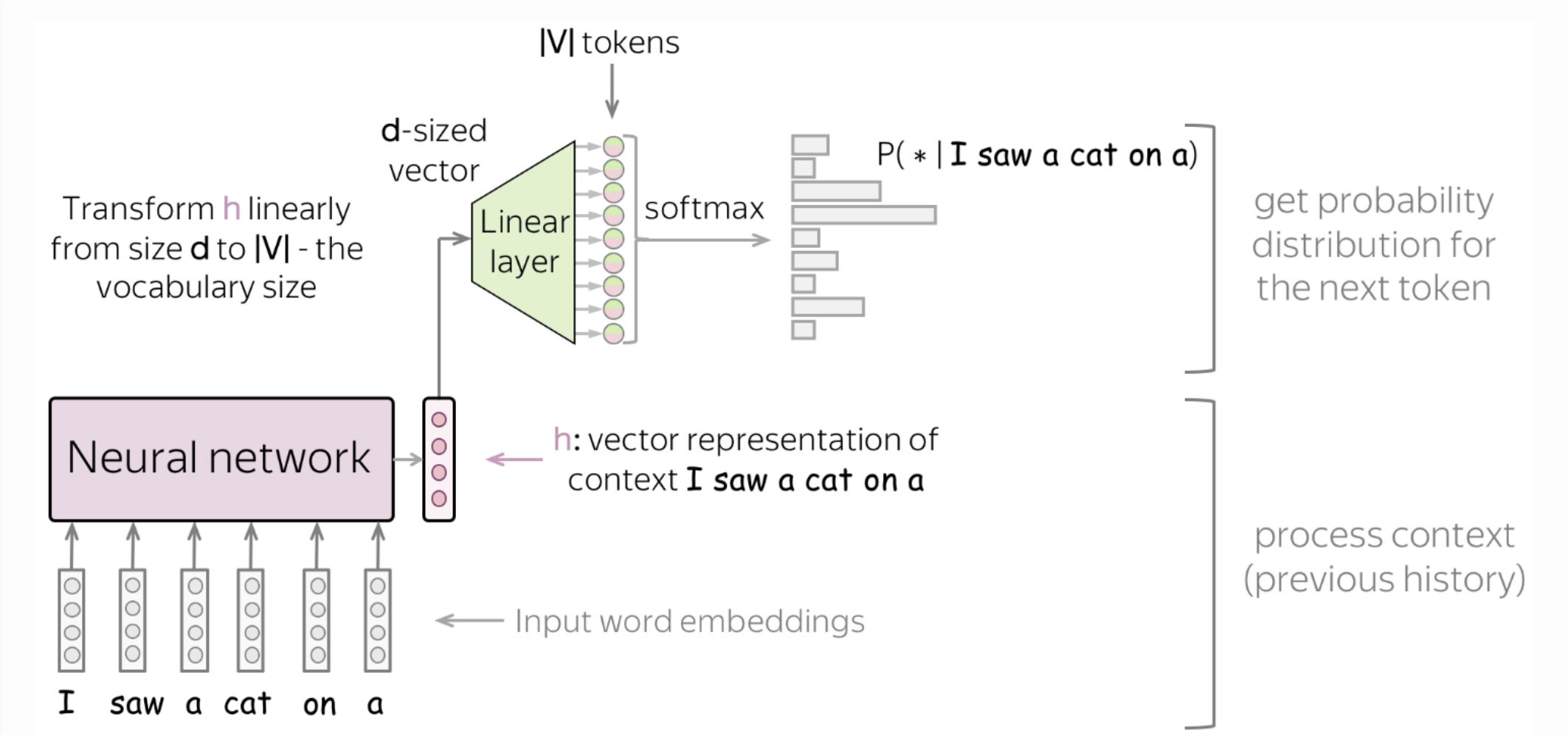
Inference only

at training we can directly do CE loss

these 2 happen only at inference-



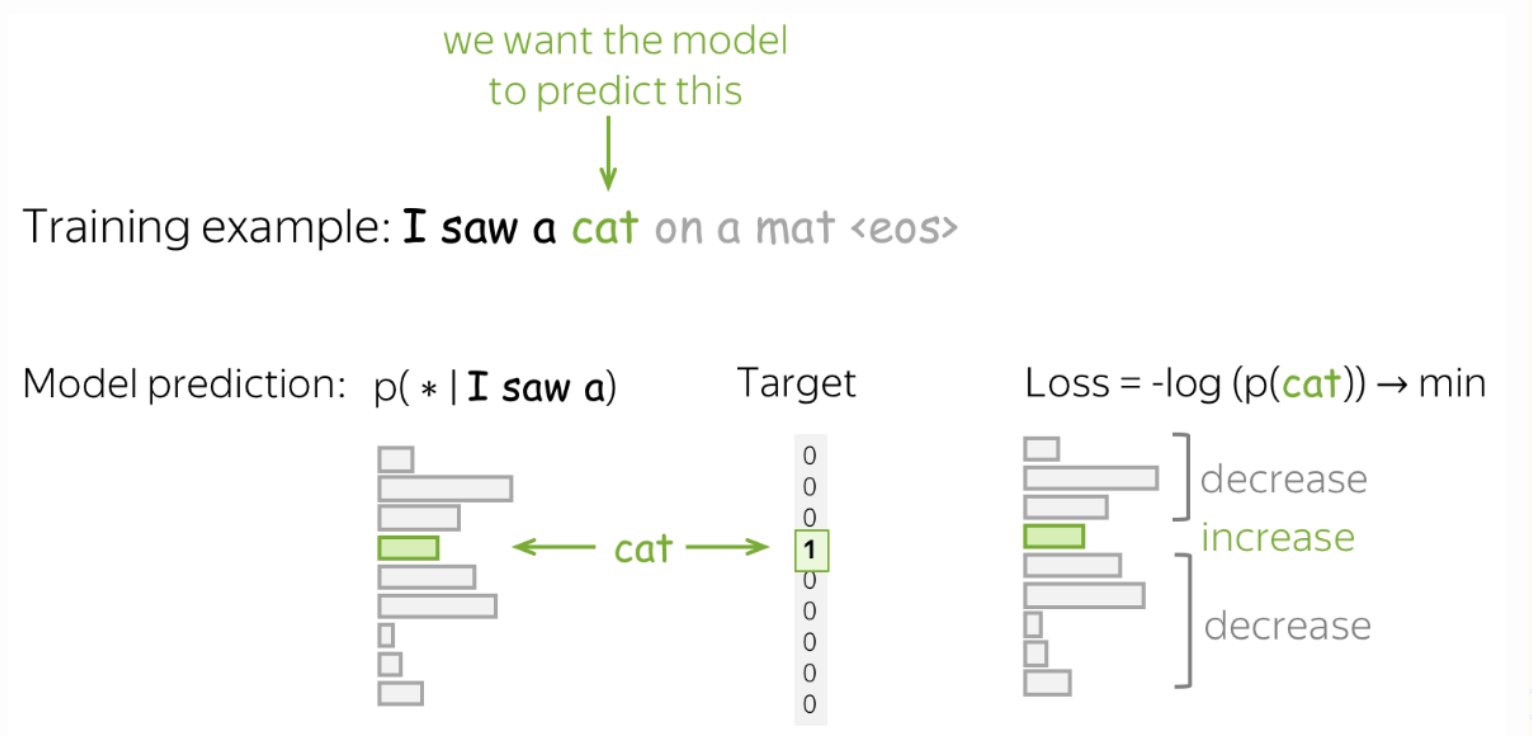
AR Neural Language Models



https://lena-voita.github.io/nlp_course/language_modeling.html#intro

Loss

- Classify next tokens' index
  - => cross-entropy loss



https://lena-voita.github.io/nlp_course/language_modeling.html#intro

- => maximize text's log-likelihood

$$\max \prod_i p(x_i|x_{1:i-1}) = \min\left(-\sum_i \log p(x_i|\, x_{i:i-1})\right) = \min\ \mathcal{L}(x_{i:L})$$

Tokenizer

- Why?
  - More general than words (eg typos)
  - Shorter sequences than with characters
- Idea: tokens as common subsequences (~3 letters)
- Eg: Byte Pair Encoding (BPE). Train steps:

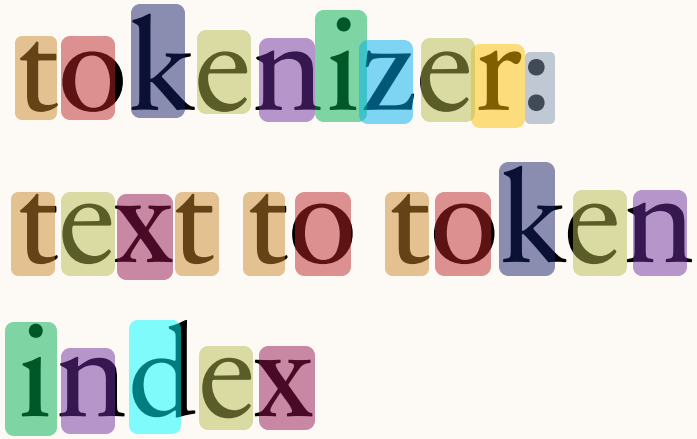
1. Take large corpus of text

tokenizer: text to token index

Tokenizer

- Why?
  - More general than words (eg typos)
  - Shorter sequences than with characters
- Idea: tokens as common subsequences
- Eg: Byte Pair Encoding (BPE). Train steps:

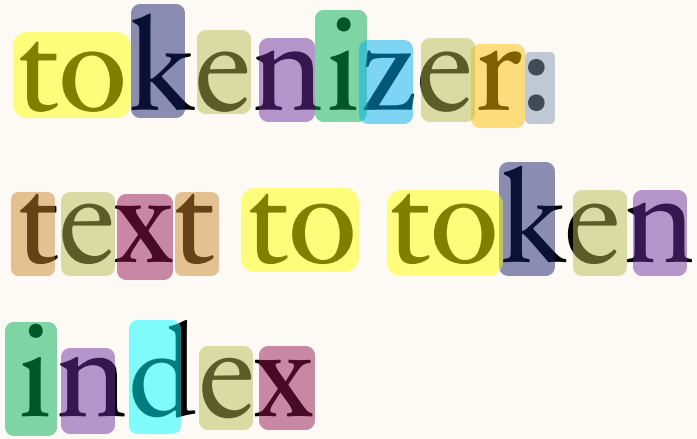
1. Take large corpus of text
2. Start with one token per character



Tokenizer

- Why?
  - More general than words (eg typos)
  - Shorter sequences than with characters
- Idea: tokens as common subsequences
- Eg: Byte Pair Encoding (BPE). Train steps:

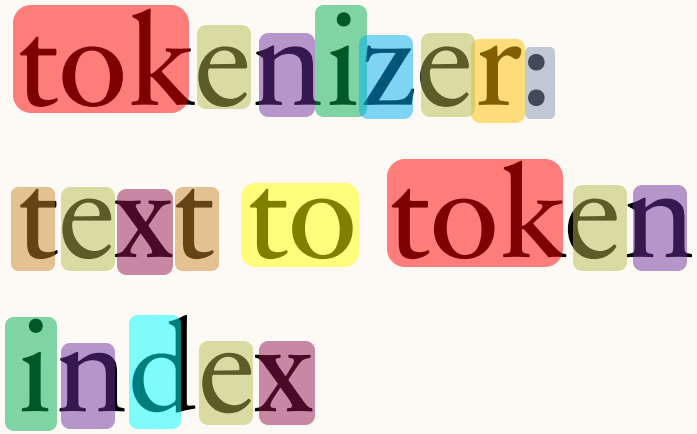
1. Take large corpus of text
2. Start with one token per character
3. Merge common pairs of tokens into a token



Tokenizer

- Why?
  - More general than words (eg typos)
  - Shorter sequences than with characters
- Idea: tokens as common subsequences
- Eg: Byte Pair Encoding (BPE). Train steps:

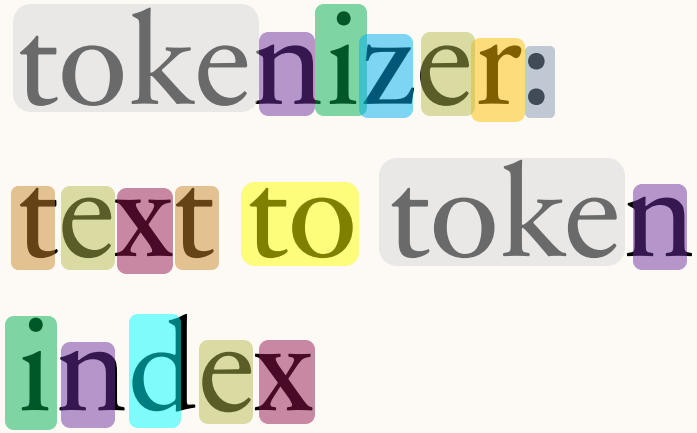
1. Take large corpus of text
2. Start with one token per character
3. Merge common pairs of tokens into a token
4. Repeat until desired vocab size or all merged



Tokenizer

- Why?
  - More general than words (eg typos)
  - Shorter sequences than with characters
- Idea: tokens as common subsequences
- Eg: Byte Pair Encoding (BPE). Train steps:

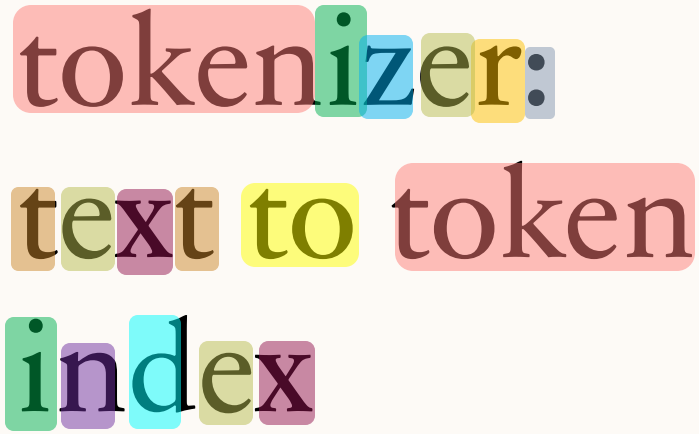
1. Take large corpus of text
2. Start with one token per character
3. Merge common pairs of tokens into a token
4. Repeat until desired vocab size or all merged



Tokenizer

- Why?
  - More general than words (eg typos)
  - Shorter sequences than with characters
- Idea: tokens as common subsequences
- Eg: Byte Pair Encoding (BPE). Train steps:

1. Take large corpus of text
2. Start with one token per character
3. Merge common pairs of tokens into a token
4. Repeat until desired vocab size or all merged

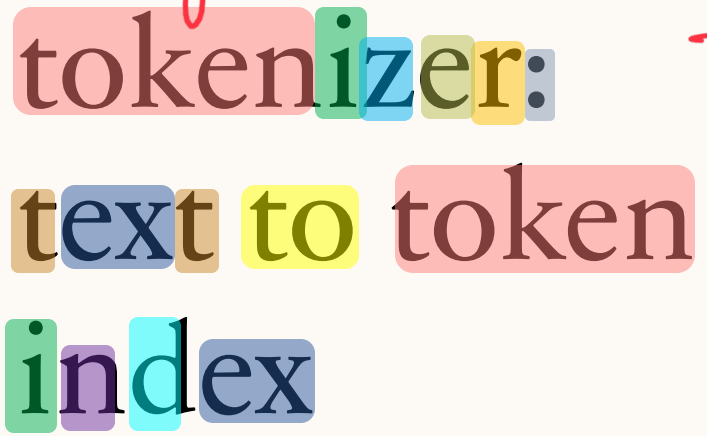


Tokenizer

tokens are unique but transformer will take care to learn context depending on surrounding tokens.

- Why?
  - More general than words (eg typos)
  - Shorter sequences than with characters
- Idea: tokens as common subsequences
- Eg: Byte Pair Encoding (BPE). Train steps:

1. Take large corpus of text
2. Start with one token per character
3. Merge common pairs of tokens into a token
4. Repeat until desired vocab size or all merged

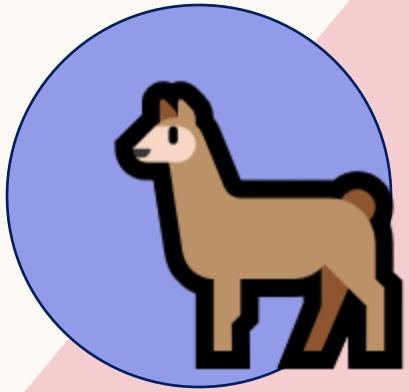


Overview

Pretraining -> GPT3

- Task & loss
- Evaluation

Post-training -> ChatGPT



LLM evaluation: Perplexity

- Idea: validation loss

$$PPL(x_{1:L}) = 2^{\frac{1}{L}\mathcal{L}(x_{1:L})} = \prod p(x_i|x_{1:i-1})^{-1/L}$$

- To be more interpretable: use **perplexity**
  - avg per token (~independent of length)
  - Exponentiate => units independent of log base
- Perplexity: between 1 and |Vocab|
  - Intuition: number of tokens that you are hesitating between

$\left(\frac{1}{|V|}\right)^{-1/L}$ if random pick

$\prod_{i=1}^{L} |V|^{-1/L}$ least confident

$\prod_{i=1}^{L} |V|^{-1/L}$

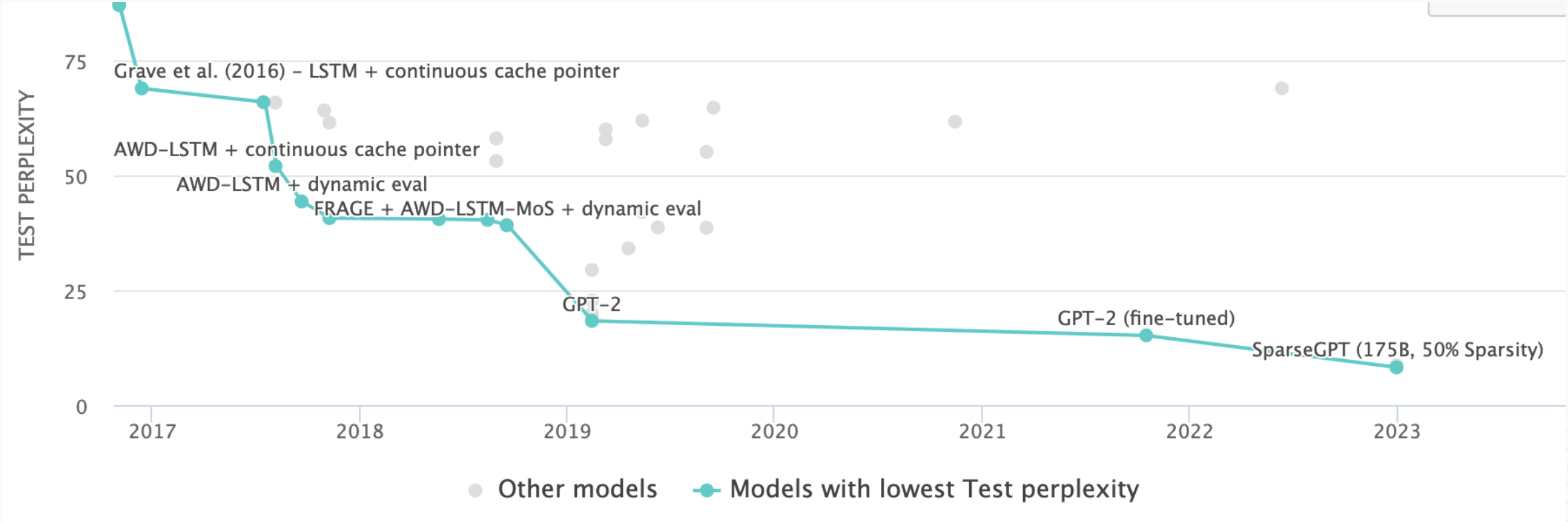
$|V|$

1 -> model is not hesitating

|Vocab| -> model hesitates in entire vocabulary space.

so prob at each step is $\frac{1}{|V|}$

LLM evaluation: Perplexity

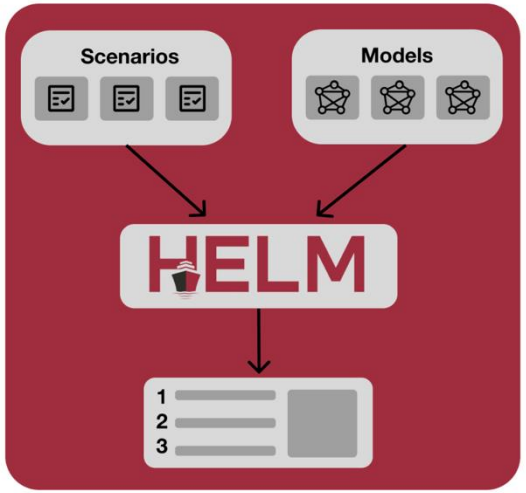


Between 2017-2023, models went from ”hesitating” between ~70 tokens to <10 tokens

Perplexity not used anymore for academic benchmark but still important for development

LLM Evaluation: agg. std NLP benchmarks

Holistic evaluation of language models (HELM)



Model	Mean win rate
GPT-4 (0613)	0.962
GPT-4 Turb (1106 preview)	0.834
Palmyra X V3 (72B)	0.821
Palmyra X V2 (33B)	0.783
PaLM-2 (Unicorn)	0.776
Yi (34B)	0.772

SEE MORE

Huggingface open LLM leaderboard

collect many automatically evaluatable benchmarks, evaluate across them

LLM Evaluation: agg. std NLP benchmarks

- Mix of things that can be "easily" evaluated

- Typically there is "gold" answer => you likelihood of LLM to predict that vs other options

Scenario	Task	What	Who
NarrativeQA narrative_qa	short-answer question answering	passages are books and movie scripts, questions are unknown	annotators from summaries
NaturalQuestions (closed-book) natural_qa_closedbook	short-answer question answering	passages from Wikipedia, questions from search queries	web users
NaturalQuestions (open-book) natural_qa_openbook_longans	short-answer question answering	passages from Wikipedia, questions from search queries	web users
OpenbookQA openbookqa	multiple-choice question answering	elementary science	Amazon Mechnical Turk workers
MMLU (Massive Multitask Language Understanding) mmlu	multiple-choice question answering	math, science, history, etc.	various online sources
GSM8K (Grade School Math) gsm	numeric answer question answering	grade school math word problems	contractors on Upwork and Surge AI
MATH math_chain_of_thought	numeric answer question answering	math competitions (AMC, AIME, etc.)	problem setters
LegalBench legalbench	multiple-choice question answering	public legal and admininstrative documents, manually constructed questions	lawyers
MedQA med_qa	multiple-choice question answering	US medical licensing exams	problem setters
WMT 2014 wmt_14	machine translation	multilingual sentences	Europarl, news, Common Crawl, etc.

HELM-lite
[Liang+ 2022]

LLM Evaluation: eg MMLU

- Example: **MMLU**
- ~Most trusted pretraining benchmark

Astronomy

What is true for a type-Ia supernova?

A. This type occurs in binary systems.
B. This type occurs in young galaxies.
C. This type produces gamma-ray bursts.
D. This type produces high amounts of X-rays.

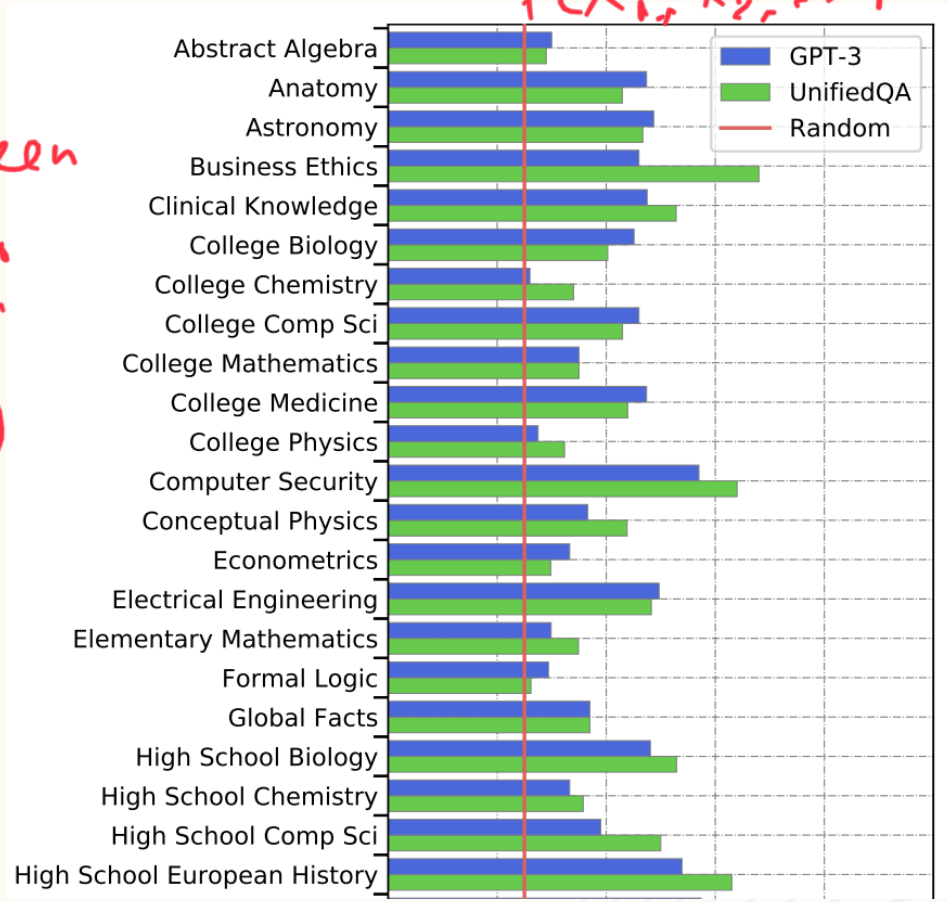
Answer: A

High School Biology

In a population of giraffes, an environmental change occurs that favors individuals that are tallest. As a result, more of the taller individuals are able to obtain nutrients and survive to pass along their genetic information. This is an example of

A. directional selection.
B. stabilizing selection.
C. sexual selection.
D. disruptive selection

Answer: A



MMLU
[Hendrycks+ 2020]

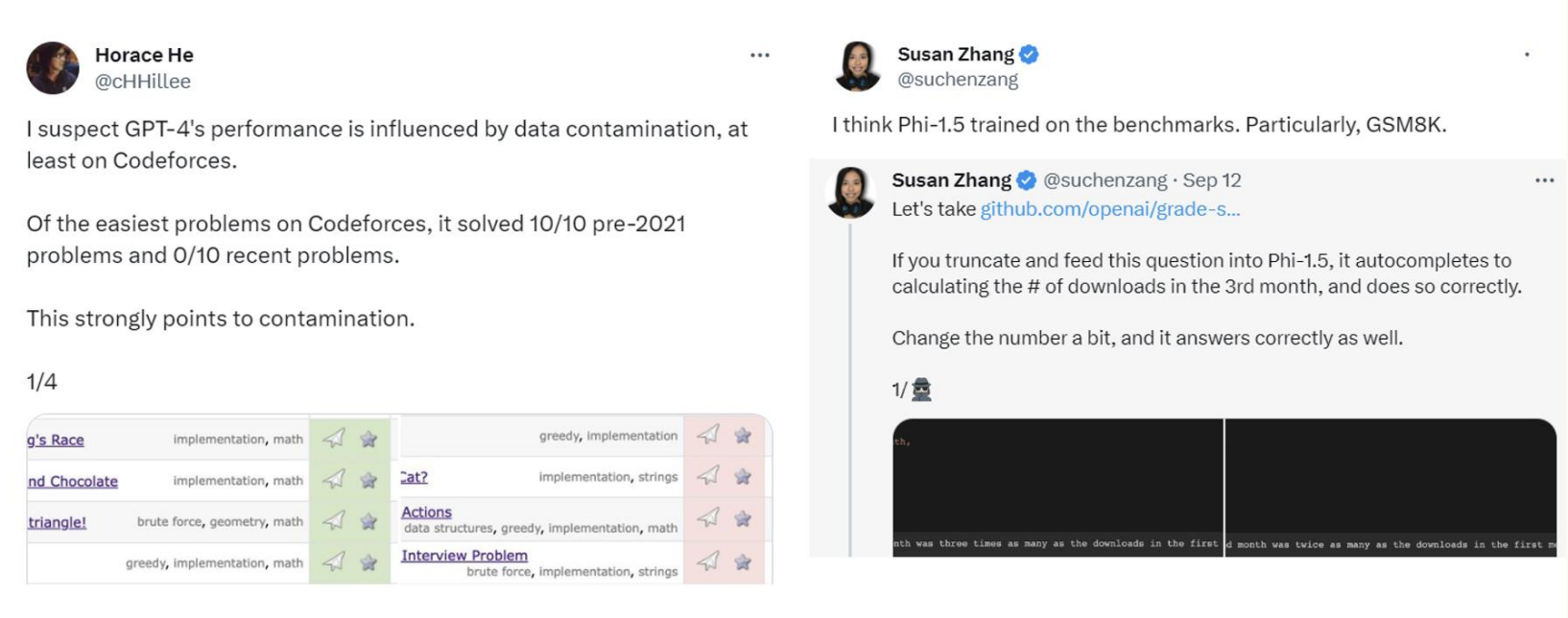
Evaluation: challenges

- Sensitivity to prompting/inconsistencies

	MMLU (HELM)	MMLU (Harness)	MMLU (Original)
llama-65b	**0.637**	0.488	**0.636**
tiiuae/falcon-40b	0.571	**0.527**	0.558
llama-30b	0.583	0.457	0.584
EleutherAI/gpt-neox-20b	0.256	0.333	0.262
llama-13b	0.471	0.377	0.47
llama-7b	0.339	0.342	0.351
tiiuae/falcon-7b	0.278	0.35	0.254

Evaluation: challenges

- Sensitivity to prompting/inconsistencies
- Train & test contamination (~not important for development)

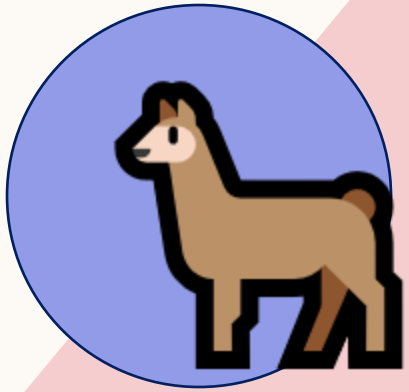


Overview

Pretraining -> GPT3

- Task & loss
- Evaluation
- Data

Post-training -> ChatGPT



250B pages -> 1 Petabyte.

Data

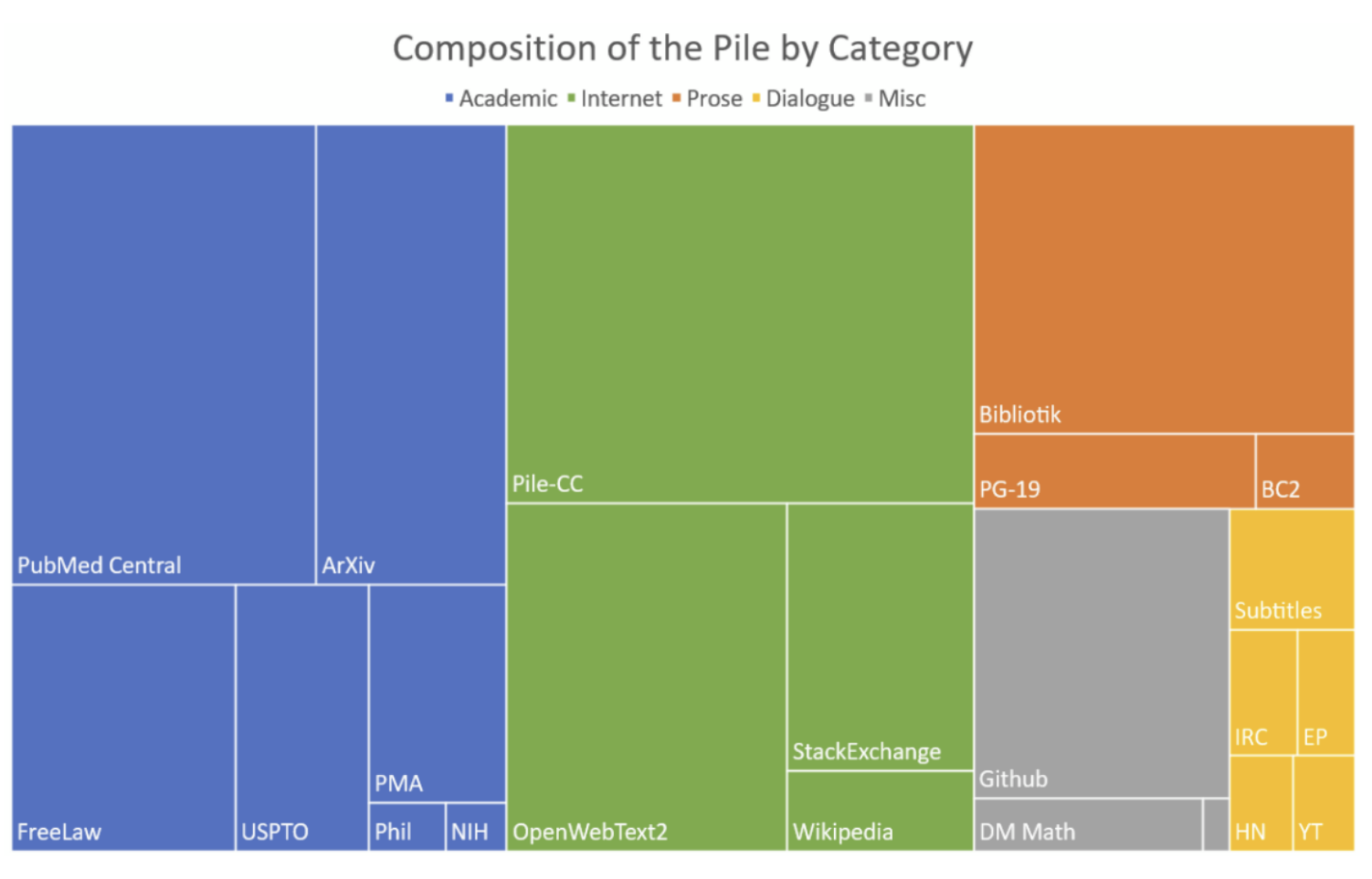
- Idea: use al
- Note: inter
  1. Down
  2. Text e
  3. Filter
  4. Dedu ys same
  5. Heuri dirty toks)
  6. Mode
  7. Data g scaling laws t
- Also: lr ar

See if a page comes from wiki


```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"> <html
xmlns="http://www.w3.org/1999/xhtml"> <head> <meta http-equiv="Content-Type" content="text/html; charset=utf-8" /> <title>000 084 -
Downloads Free 000 084 - Download 000 084 Software</title> <meta type="description" content="000 084 at Smart Code for free download. 000
084 freeware and shareware free downloads." /> <meta type="keywords" content="000 084, downloads, freeware, software, free, 000-084 Test
Prep Training, Pass4sure IBM 000-084, TopCerts 000-084 Questions and Answers, HP0-084 Free Practice Exam Questions, Pass4sure ADOBE
9A0-084" /> <link rel="shortcut icon" href="/design/favicon.ico" type="image/x-icon" /> <link href="/design/look.css" rel="stylesheet"
type="text/css" /> <!--[if lte ie 6]> <link href="/design/iexplorer6.css" rel="stylesheet" type="text/css" /> <![endif]--> <script
type="text/javascript">link = "http://www.smartcode.com";</script> <script type="text/javascript" src="/design/bmark.js"></script> <link
href="/design/generic/ui.css" rel="stylesheet" type="text/css" /> <script src="/cms/generic.2/scripts/jquery.js" type="text/javascript"
language="javascript"></script> <script src="/cms/generic.2/scripts/ui.js" type="text/javascript" language="javascript"></script> </head>
<body> <div id="wrapper"> <div id="header"> <a class="logo_type" href="http://www.smartcode.com/"><!--<img src="/design/sitelogo.jpg"
alt="Premier website for Windows Shareware and Freeware Downloads">--></a> </div> <div id="menu"> <script type="text/javascript"
src="/design/jcoding.js"></script> <form action="http://www.smartcode.com/downloads" id="search" name="search" method="get"
onsubmit="return do_search(false);"> <input class="inp_text" type="text" name="query" id="query" value=""/> <a href="#" onclick="do_search
(false);return false;" class="input_href"><img src="/design/search_button.gif" class="input_href" /></a> </form> <ul> <li><a
href="http://www.smartcode.com/">Home</a></li> <li><a href="http://www.smartcode.com/db/allrootandsubcats.php">Categories</a></li> <li><a
href="http://www.smartcode.com/db/new.php">New</a></li> <li><a href="http://www.smartcode.com/db/top.php">Popular</a></li> <li><a
href="http://www.smartcode.com/submit/">Submit</a></li> <li><a href="http://www.smartcode.com/main/rss/">RSS</a></li> <li><a
href="http://www.smartcode.com/main/contact.html">Contact</a></li> </ul> </div> <div id="content"> <div id="content_right"> <style> h1 {
padding-bottom: 15px; } h1 strong { float: left; } div.pager { font-size: 11px; float: right; padding-top: 5px; } </style> <div
style="display:none;"> false<br> 5<br> 5<br> 5 </div> <table border="0" cellspacing="0" cellpadding="0" id="table1"> <tr> <td
width="4"> </td> <td> <script src="/design/ccoding.js" type="text/javascript"></script> </td> <td width="15"> </td> <td> <script
 src="/design/ccoding_im.js" type="text/javascript"></script> </td> </tr> </table> <br> <div class="hr" style="width:100%;
margin-bottom:10px;"></div> <h1> <strong>000 084</strong> <div class="pager"> Pages:  1 <a href="/p2.html"> 2 </a> <a href="/p3
.html"> 3 </a>  <a href="http://000-084.smartcode.com/freeware.html">Freeware</a> <a href="http://000-084.smartcode.com/macosx
.html">Mac</a> </div> </h1> <br> <p ct="1=Testkingworld.com is your ultimatel source for the System x High Performance Servers...">
Testkingworld.com is your ultimatel source for the System x High Performance Servers... <a class="details-link"
href="http://000-084-test-prep-training.smartcode.com/info.html">Details</a></p><br> <p ct="1=One of the best and most rewarding features
of the 000-084 training materials are that..."> One of the best and most rewarding features of the 000-084 training materials are
that... <a class="details-link" href="http://pass4sure-ibm-000-084.smartcode.com/info.html">Details</a></p><br> <p ct="1=Download
free 000-084 questions and answers. 000-084 exam questions are ultimate..."> Download free 000-084 questions and answers. 000-084 exam
questions are ultimate... <a class="details-link" href="http://topcerts-000-084-questions-and-answers.smartcode.com/info
.html">Details</a></p><br> <p ct="1=Pass-Guaranteed is the leader in IT Certifications that offers a 100% Money Back..."> Pass-Guaranteed
is the leader in IT Certifications that offers a 100% Money Back... <a class="details-link"
```

Data

- Collec
- Lot of
  - H
  - H
- A lot o
  - (
- Comm
  - (
  - T
- Closed

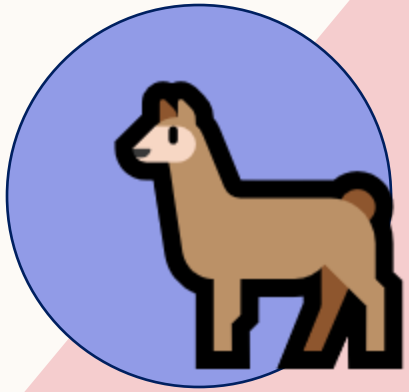


Overview

Pretraining -> GPT3

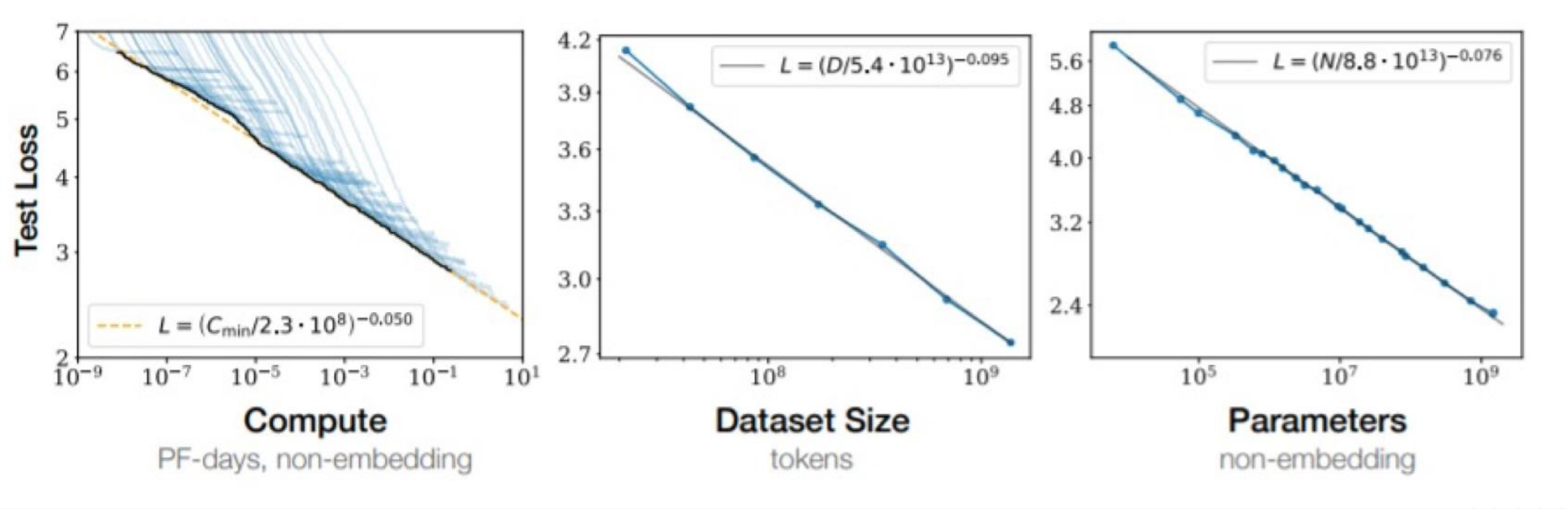
- Task & loss
- Evaluation
- Data
- Scaling laws

Post-training -> ChatGPT



Scaling laws

- Empirically: more data and larger models => better performance
  - Large models =/> overfitting
- Idea: predict model performance based on amount of data & parameter



It works for many things!

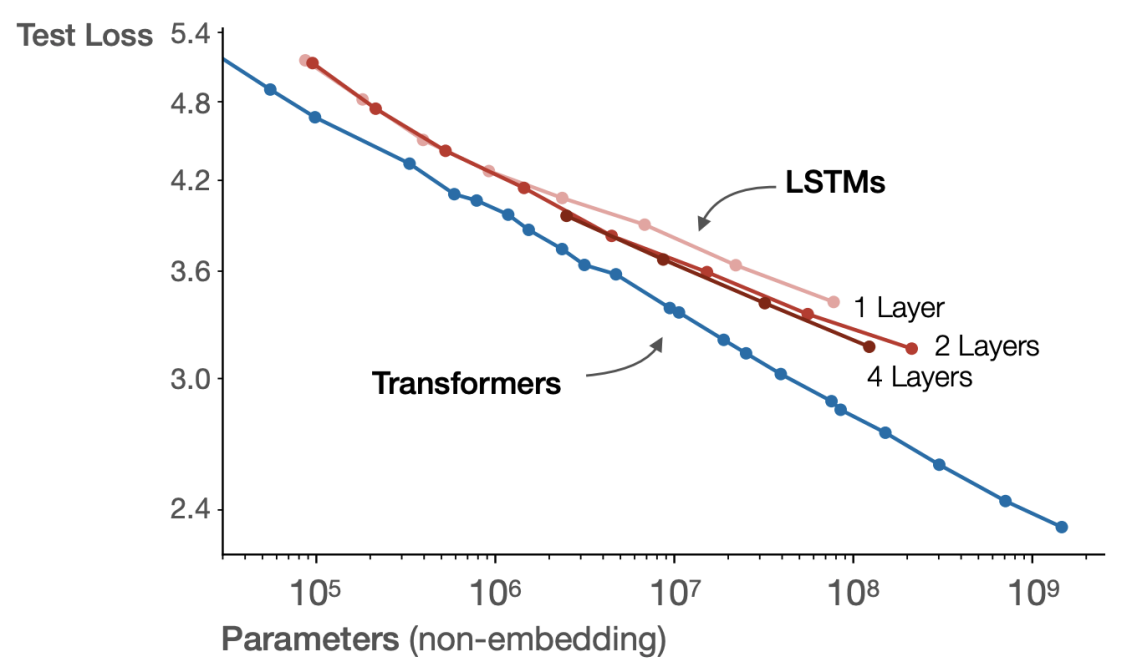
Scaling laws
[Kaplan+ 2020]

Scaling laws: tuning

- You have 10K GPUs for a month, what model do you train?
- Old pipeline: (hyperparameter tuning on original model sizes)
  - Tune hyperparameters on big models (e.g. 30 models)
  - Pick the best => final model is trained for as much as each filtered out ones (e.g. 1 day)
- New pipeline: (if u increase the model u should reduce x).
  - Find scaling recipes (eg lr decrease with size)
  - Tune hyperparameters on small models of different sizes (e.g. for <3 days)
  - Extrapolate using scaling laws to larger ones
  - Train the final huge model (e.g. >27 days)

Scaling laws: eg LSTM

- Q: Should we use transformers or LSTM?

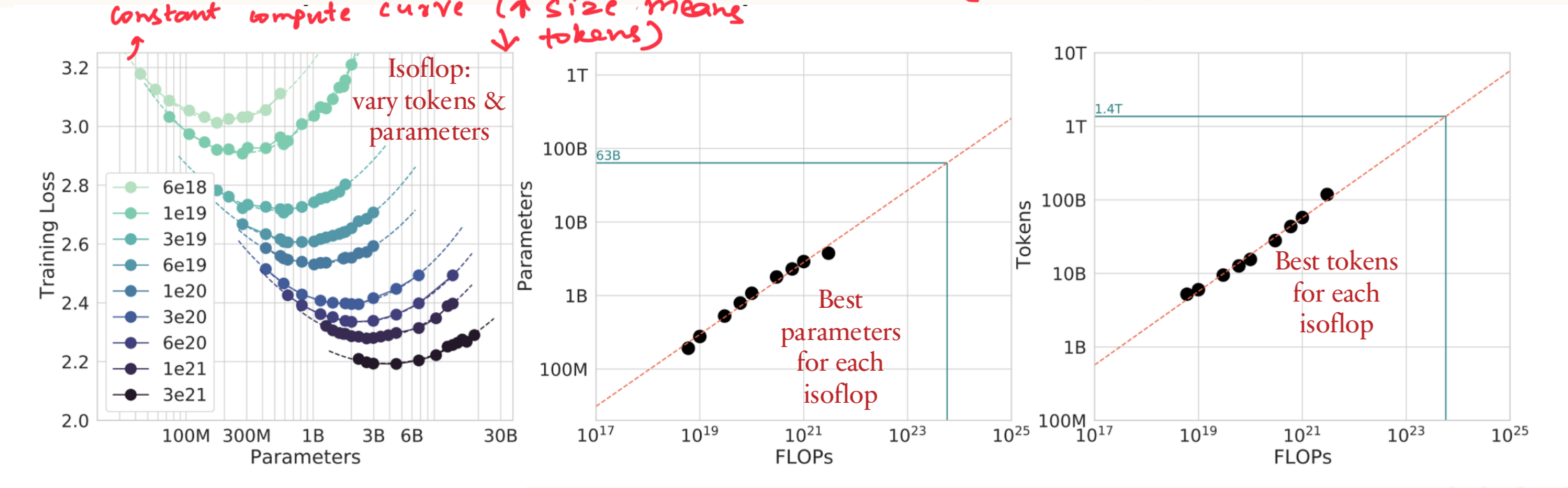


A: Transformers have a better constant and scaling rate (slope)

Scaling laws
[Kaplan+ 2020]

Scaling laws: eg Chinchilla

- Q: How do we optimally allocate training* resources (size vs data)?



A: Use 20:1 tokens for each parameter (20:1)

*doesn't consider inference cost => in practice use larger (> 150:1)

Chinchilla [Hoffmann+ 2022]

Scaling laws: tuning

- Many questions you can try to answer with scaling laws
- Resource allocation:
  - Train models longer vs train bigger models?
  - Collect more data vs get more GPUs?
- Data:
  - Data repetition / multiple epochs?
  - Data mixture weighting?
- Algorithm:
  - Arch: LSTMs vs transformers?
  - Size: width vs depth?

Bitter lesson

- **Bitter lesson:** models improve with scale & Moore's Law

  => "only thing that matters in the long run is the leveraging of computation."

  Bitter [Sutton 2019] http://www.incompleteideas.net/IncIdeas/BitterLesson.html

- Don't spend time over complicating: do the simple things and scale them!

Training a SOTA model

- Example of current SOTA: LLaMA 3 400B

~40 tok/param => train compute optimal

tokens Data: 15.6T tokens Parameters: 405B

- FLOPs: 6NP = 6 * 15.6e12 * 405e9 = 3.8 e25 FLOPs ~2x less than executive order

parameters

(Biden rule ie 26 will go scrutiny)

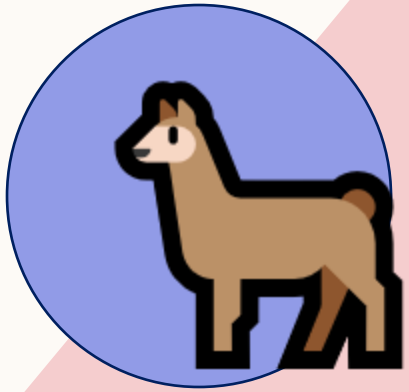
- Compute: 16K H100 with average throughput of 400 TFLOPS
- Time: 3.8e25 / (400e12 * 3600) = 26M GPU hour / (16e3 * 24) = 70 days From paper: ~30M
- Cost: rented compute + salary=~$2/h*26Mh + 500k/y*50employee= $52M+$25M = ~$75M $65-85M
- Carbon emitted= 26Mh*0.7kW*0.24kg/kWh = 4400 tCO2eq ~2k return tickets JFK-LHR
- Next model? ~10x more FLOPs

Overview

Pretraining -> GPT3

- Task & loss
- Evaluation
- Data
- Scaling laws
- Systems

Post-training -> ChatGPT



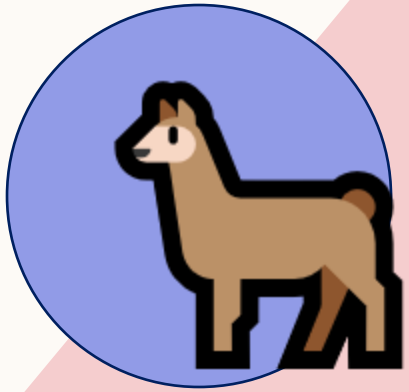
Overview

Pretraining -> GPT3

- Task & loss
- Evaluation
- Data
- Scaling laws
- Systems

Post-training -> ChatGPT

- Task



Language Modeling ≠ assisting users

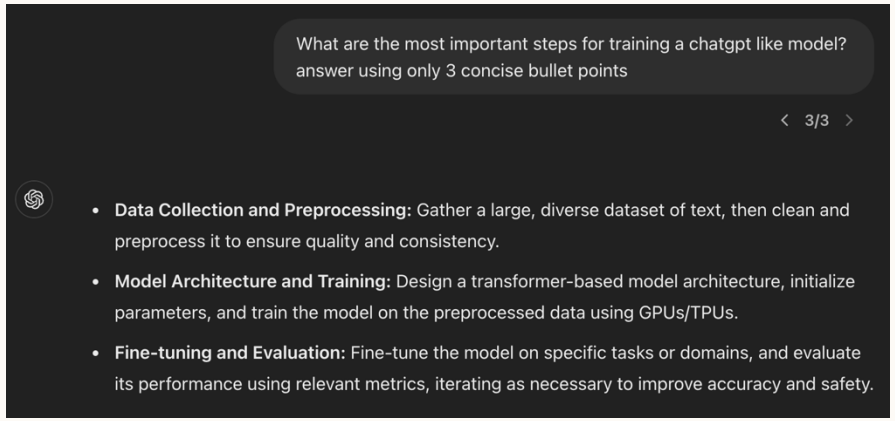
- Problem: language modeling is not what we want

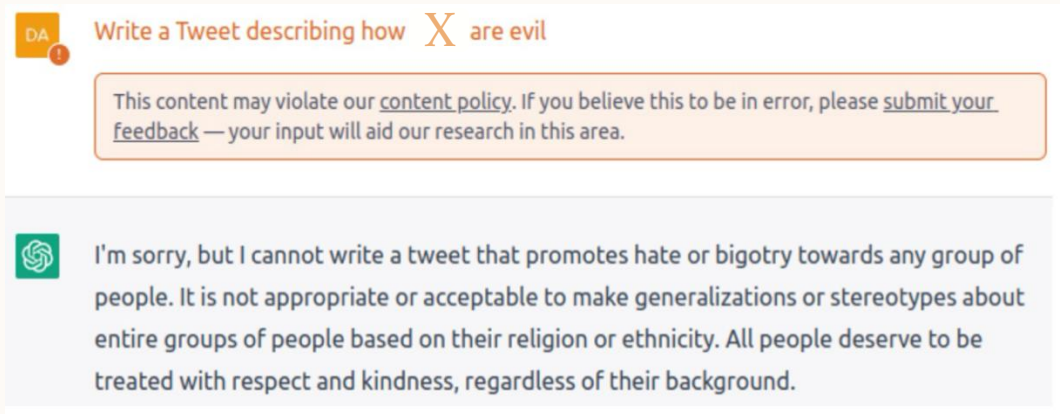
PROMPT	*Explain the moon landing to a 6 year old in a few sentences.*
COMPLETION	GPT-3

```
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.
```

Task: “alignment”

- Goal: LLM follows user instructions and designer’s desires (eg moderation)





- Background:
  - data of desired behaviors is what we want **but** scarce and expensive
  - pretraining data scales **but** is not what we want (we have more but not what we want)
- Idea: finetune pretrained LLM on a little desired data => “post-”training

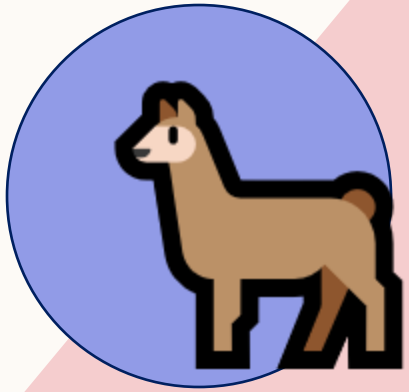
Overview

Pretraining -> GPT3

- Task & loss
- Evaluation
- Data
- Scaling laws
- Systems

Post-training -> ChatGPT

- Task
- SFT: data & loss



Supervised finetuning (SFT)

- Idea: finetune the LLM with language modeling of the desired answers

  Next word prediction — "supervised"

- How do we collect the data? Ask humans

Can you write a short introduction about the relevance of the term "monopsony" in economics? Please use examples related to potential monopsonies in the labour market and cite relevant research.	"Monopsony" refers to a market structure where there is only one buyer for a particular good or service. In economics, this term is particularly relevant in the labor market, where a monopsony employer has significant power over the wages and working conditions of their employees. […]. Overall, the concept of monopsony is essential to understanding the dynamics of labor markets and the impact of market power on workers. Further research is needed to understand the extent and impact of monopsonies on the economy and to develop policies to address this issue. References: Bivens, J., & Mishel, L. (2013). The Pay of Corporate Executives and Financial Professionals as Evidence of Rents in Top 1 Percent Incomes. Journal of Economic Perspectives, 27(3), 57-78.
My kid needs to do a science project for elementary school, what are some cheap but fun projects to do?	There are many cheap and fun science projects that elementary school kids can do! Here are a few ideas: 1. Make a homemade lava lamp: Fill a clear plastic bottle with water and vegetable oil, add a few drops of food coloring, and then drop in an Alka-Seltzer tablet. The reaction will cause bubbles to rise and fall like a lava lamp. […]7. Create a solar oven: Using a pizza box, aluminum foil, and plastic wrap, create a solar oven that can cook s'mores or other small snacks. 8. Explore magnets: Using magnets and a variety of objects, have your child experiment with magnetism to see what objects are attracted to magnets and which ones are not. These are just a few ideas to get you started! There are many more cheap and fun science projects that your child can do with household items and a little creativity.

OpenAssistant [Kopf+ 2023]

This was the ~key to GPT3 -> ChatGPT model!

Scalable data for SFT: eg Alpaca

- Problem: human data is slow to collect and expensive
- Idea: use LLMs to scale data collection

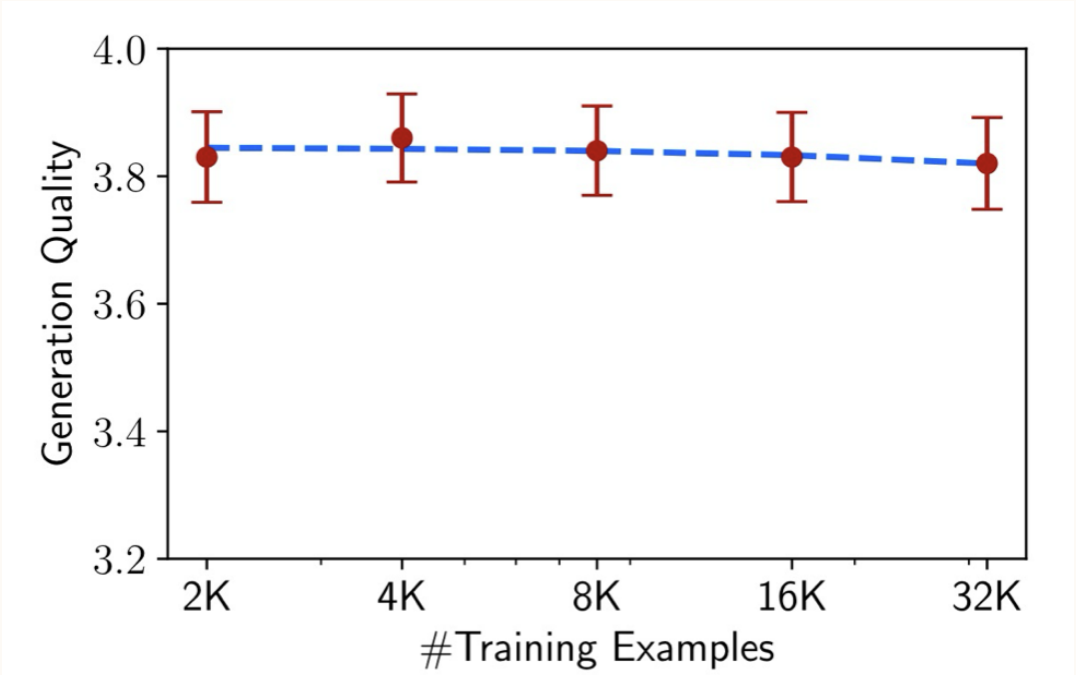
Give three tips for staying healthy.	1.Eat a balanced diet and make sure to include plenty of fruits and vegetables. 2. Exercise regularly to keep your body active and strong. 3. Get enough sleep and maintain a consistent sleep schedule.
What does “algorithm” mean?	An algorithm is a step-by-step set of instructions used to solve a problem or achieve a goal. It is a process of breaking down complex tasks into smaller, more manageable steps. Algorithms can be used to find the shortest path from one place to another, sort data, or render data from an immense database in a few seconds.
Find the average number in a list.	def avg_list(nums): total = 0 for n in nums: total += n return total / len(nums) print(avg_list([4, 3, 6, 10, 8])) # Output 6.4

Alpaca
[Taori+ 2023]

Started for academic replication of ChatGPT but “synthetic data generation” is now hot topic!

Scalable data for SFT: quantity?

- You need very little data for SFT! ~few thousand



LIMA
[Zhou+ 2023]

- Just learns the format of desired answers (length, bullet points, …)
  - The knowledge is already in the pretrained LLM!
  - Specializes to one "type of user"

Scaling laws doesn't help for SFT ie we don't need a ton of example data to do SFT. why scaling won't help!!

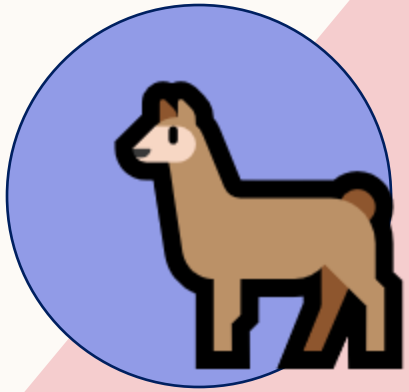
Overview

Pretraining -> GPT3

- Task & loss
- Evaluation
- Data
- Scaling laws
- Systems

Post-training -> ChatGPT

- Task
- SFT: data & loss
- RLHF : data & loss



RL from Human Feedback (RLHF)

- Problem: SFT is **behavior cloning** of humans

we may not be the best at generating

1. **Bound by human abilities:** humans may prefer things that they are not able to generate
2. **Hallucination:** cloning *correct* answer teaches LLM to hallucinate if it didn't know about it!

Can you write a short introduction about the relevance of the term "monopsony" in economics? Please use examples related to potential monopsonies in the labour market and cite relevant research.	"Monopsony" refers to a market structure where there is only one buyer for a particular good or service. In economics, this term is particularly relevant in the labor market, where a monopsony employer has significant power over the wages and working conditions of their employees. […]. Overall, the concept of monopsony is essential to understanding the dynamics of labor markets and the impact of market power on workers. Further research is needed to understand the extent and impact of monopsonies on the economy and to develop policies to address this issue. **References: Bivens, J., & Mishel, L. (2013). The Pay of Corporate Executives and Financial Professionals as Evidence of Rents in Top 1 Percent Incomes. Journal of Economic Perspectives, 27(3), 57-78.**

If LLM doesn't know [Bivens 2013] => teaches the model to make up plausibly sounding referneces

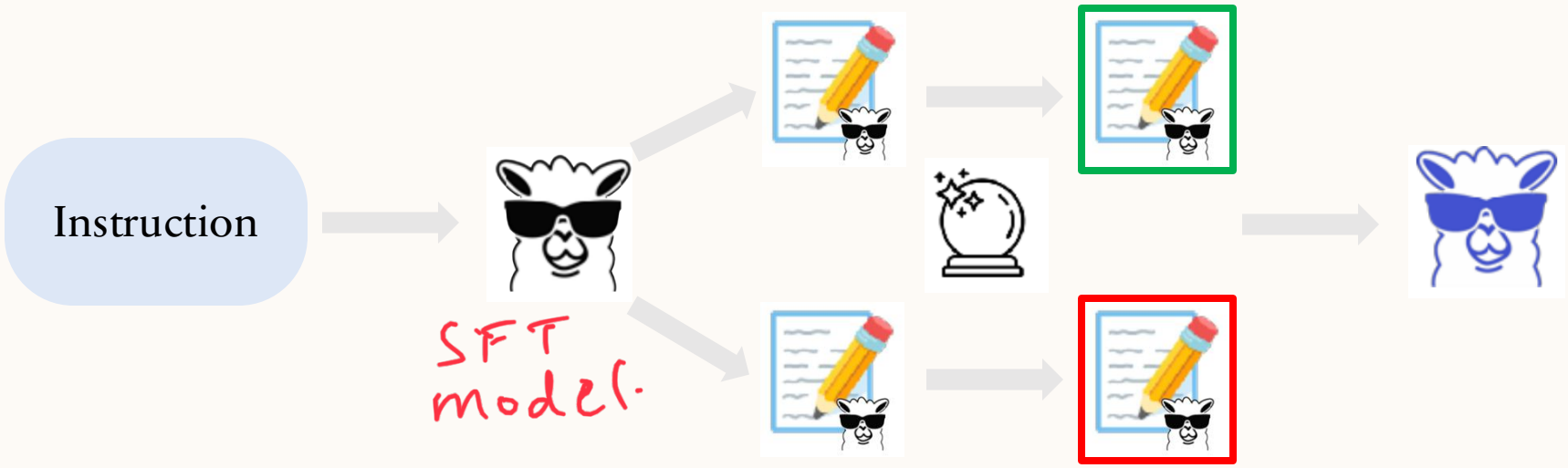
3. **Price:** collecting ideal answers is expensive

hallucination might be caused by SFT! if a model doesn't know something during pretraining & forces to learn during SFT, it learns that this is the way to learn!!

RLHF

- Idea: maximize human preference rather than clone their behavior → generating answers!
- Pipeline:
  1. For each instruction: generate 2 answers from a pretty good model (SFT)
  2. Ask labelers to select their preferred answers
  3. Finetune the model to generate more preferred answers

How??



RLHF: PPO

- Idea: use reinforcement learning
- What is the reward?
  - Option 1: whether the model's output is preferred to some baseline
    - Issue: binary reward doesn't have much information (Sparse)
  - Option 2: train a **reward model R** using a logistic regression loss to classify preferences.

$$p(i > j) = \frac{\exp(R(x, \hat{y}_i))}{\exp(R(x, \hat{y}_i)) + \exp(R(x, \hat{y}_j))}$$ [Bradley-Terry 1952]

    - Use logits R(…) as reward => continuous information => information heavy!

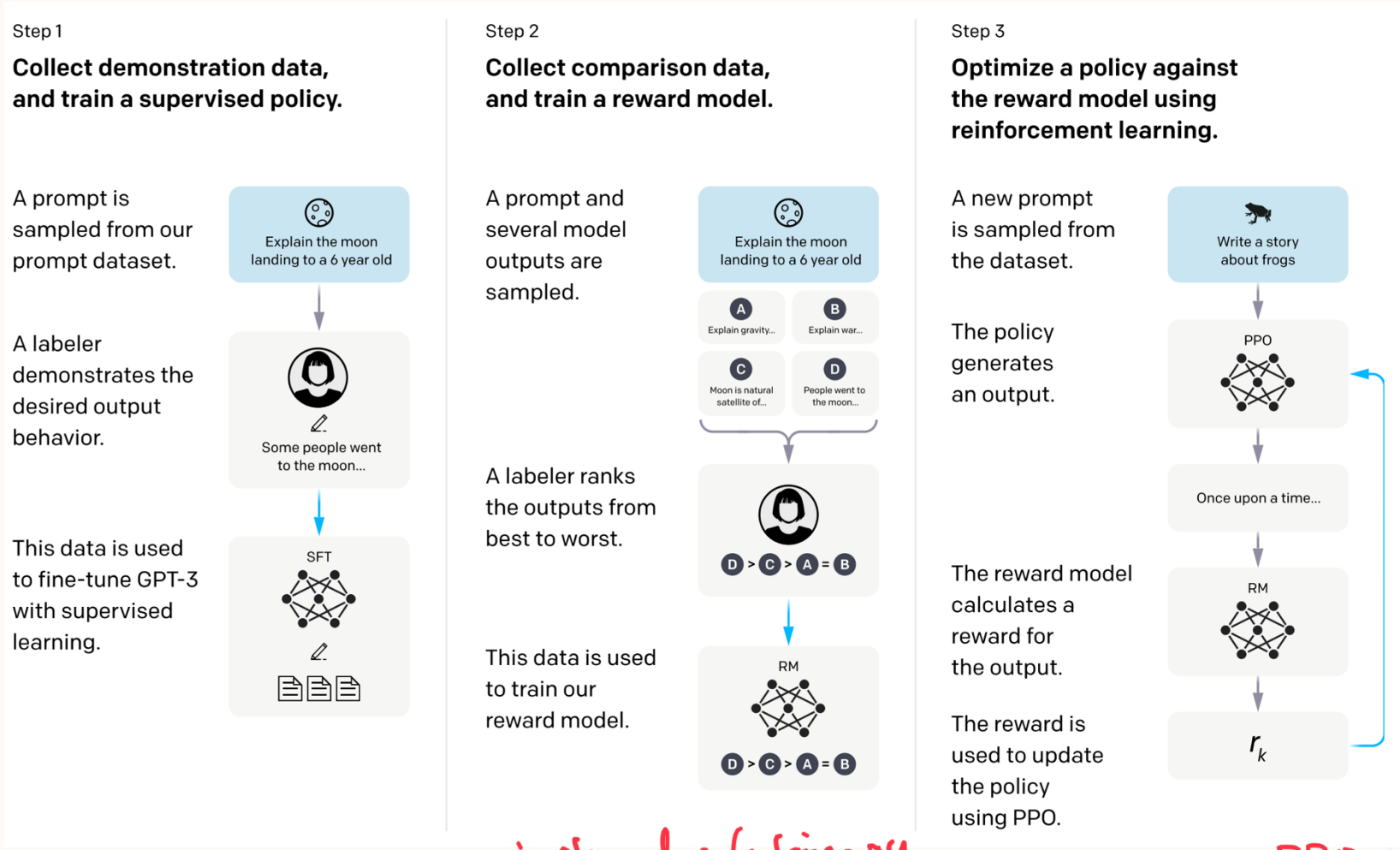
use logits of softmax. as it's continuous.

- Optimize $\mathbb{E}_{\hat{y} \sim p_\theta(\hat{y}|x)} \left[R(x, \hat{y}) - \beta \log \frac{p_\theta(\hat{y}|x)}{p_{ref}(\hat{y}|x)} \right]$ using PPO

-> regularization avoids overoptimization

- Note: LMs are policies not a model of some distribution

RLHF: PPO -> ChatGPT



RLHF
[Ouyang+ 2022]

SFT

instead of binary, use logits

use PPO to update the RM.

RLHF: PPO challenges

- Problem: RL in theory simple, in practice **messy** (clipping, rollouts, outer loops,…)

Instead of reward, we use advantages

$$\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} := \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^V \quad \text{where} \quad \delta_t^V = r_t + \gamma V(s_{t+1}) - V(s_t),$$

```python
def _estimate_advantage(self, rewards: Tensor, values: Tensor) -> Dict[str, Tensor]:
    """Generalized advantage estimation.

    Reference:
        https://arxiv.org/abs/1506.02438
    """
    if self.args.whiten_rewards:
        rewards = torch_ops.whiten(rewards, shift_mean=False)
    lastgaelam = 0
    advantages_reversed = []
    gen_length = self.args.response_len
    for t in reversed(range(gen_length)):
        nextvalues = values[:, t + 1] if t < gen_length - 1 else 0.0
        delta = rewards[:, t] + self.args.gamma * nextvalues - values[:, t]
        lastgaelam = delta + self.args.gamma * self.args.lam * lastgaelam
        advantages_reversed.append(lastgaelam)
    advantages = torch.stack(advantages_reversed[::-1], dim=1)
    returns = advantages + values
    advantages = torch_ops.whiten(advantages, shift_mean=True)
    return dict(returns=returns, advantages=advantages)
```

AlpacaFarm [Dubois+ 2023]

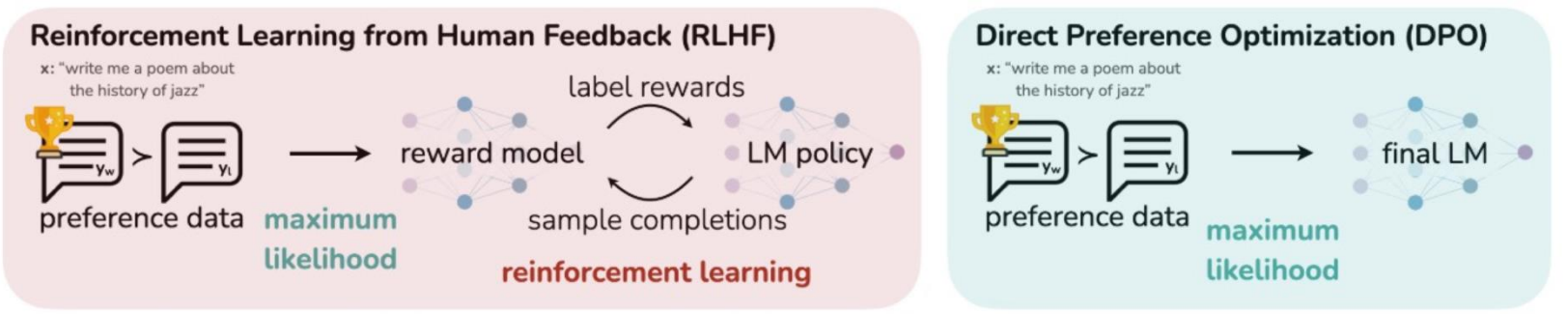
RLHF: DPO

maximize the stuff I like & min the stuff I don't like!

- Idea: maximize probability of preferred output, minimize the other

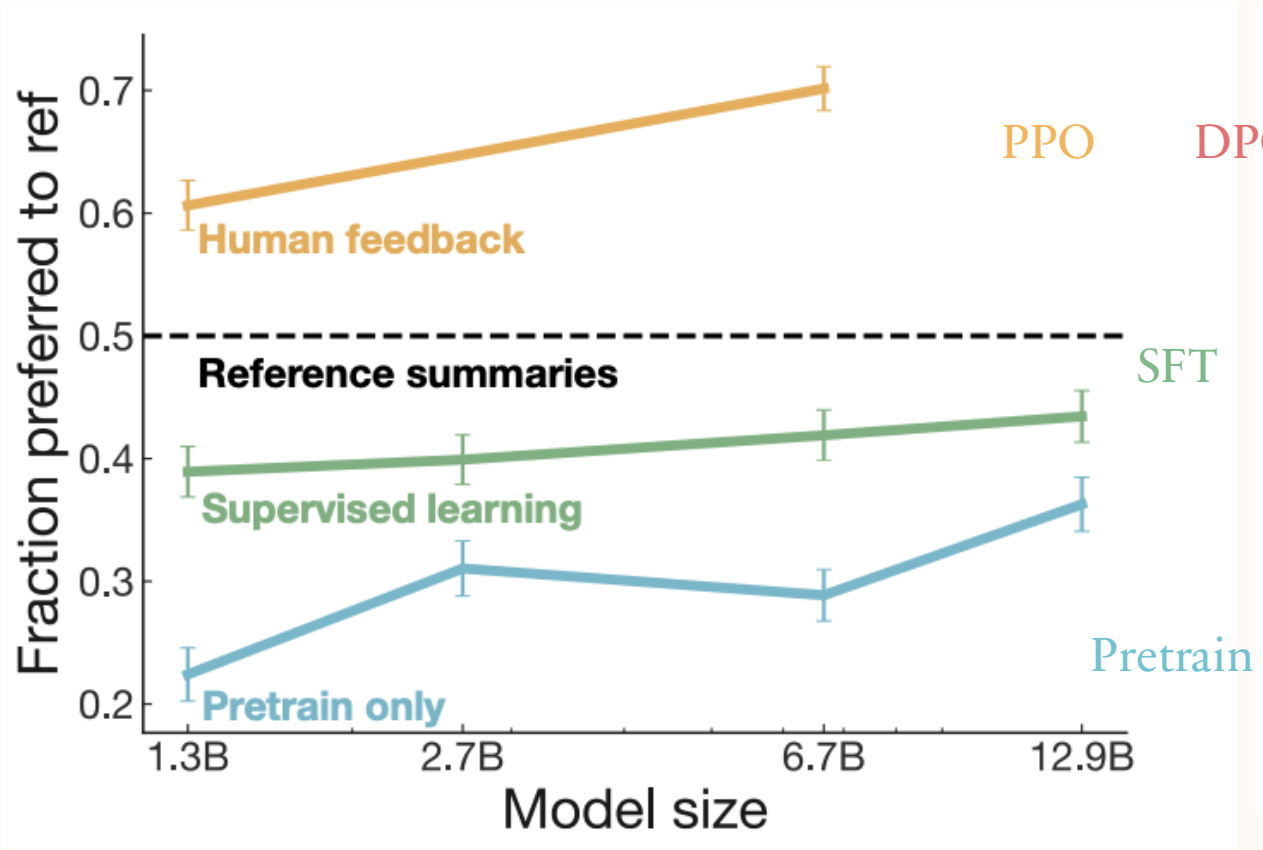
$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta\log\frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

DPO
[Rafailov+ 2023]



- This is ~equivalent (same global minima) to RLHF/PPO
- Much simpler than PPO and performs as well => standard (in open source community)

RLHF: gains



Method	Simulated Win-rate (%)
GPT-4*†	79.0 ± 1.4
ChatGPT*†	61.4 ± 1.7
PPO	46.8 ± 1.8
DPO	46.8 ± 1.7
Best-of-1024	45.0 ± 1.7
Expert Iteration	41.9 ± 1.7
SFT 52k	39.2 ± 1.7
SFT 10k	36.7 ± 1.7
Binary FeedME	36.6 ± 1.7
Quark	35.6 ± 1.7
Binary Reward Conditioning	32.4 ± 1.6
Davinci001*	24.4 ± 1.5
LLaMA 7B*	11.3 ± 1.1

Learn to summarize
[Stiennon+ 2020]

AlpacaFarm
[Dubois+ 2023]

RLHF: human data

- Data: human crowdsourcing

In this task, you will be provided with a **Prompt** from a user (e.g., a question, instruction, statement) to an AI chatbot along with two potential machine-generated **Responses** to the Prom

Your job is to assess which of the two Responses is better for the Prompt, considering the following for each Response:

Helpfulness: To what extent does the Response provide useful information or satisfying content for the Prompt? Responses should: ▪ **Address the intent of the user's Prompt** such that a user would not feel the Prompt was ignored or misinterpreted by the Response. ▪ **Provide specific, comprehensive, and up-to-date information** for the user needs expressed in the Prompt. ▪ **Be sensible and coherent.** The response should not contain any nonsensical information or contradict itself across sentences (e.g., refer to two different people with the same name as if they are the same person). ▪ **Adhere to any requirements indicated in the Prompt** such as an explicitly specified word length, tone, format, or information that the Response should include. ▪ **Not contain inaccurate, deceptive, or misleading information** (based on your current knowledge or quick web search - you do not need to perform a rigorous fact check) ▪ **Not contain harmful, offensive, or overly sexual content** A Response may sometimes intentionally avoid or decline to address the question/request of the Prompt and may provide a reason for why it is unable to respond. For example, "Sorry, there may not be a helpful answer to this question." These responses can be considered helpful in cases where an appropriate helpful response to the Prompt does not seem possible.	Rating scale: ▪ **Not at All Helpful:** Response is useless/irrelevant, contains even a single piece of nonsensical/inaccurate/deceptive/misleading information, and/or contains harmful/offensive/overly sexual content. ▪ **Slightly Helpful:** Response is somewhat related to the Prompt, does not address important aspects of the Prompt, and/or contains outdated information. ▪ **Somewhat Helpful:** Response partially addresses the intent of the Prompt (most users would want more information), contains extra unhelpful information, and/or is lacking helpful details/specifics. ▪ **Very Helpful:** Response addresses the intent of the Prompt with a satisfying response. Some users might want a more comprehensive response with additional details or context. It is comparable to a response an average human with basic subject-matter knowledge might provide. ▪ **Extremely Helpful:** Response completely addresses the intent of the Prompt and provides helpful details/context. It is comparable to a response a talented/well-informed human with subject-matter expertise might provide. guidelines
Presentation: To what extent is the content of the Response conveyed well? Responses should: ▪ **Be organized in a structure that is easy to consume and understand.** Flowing in a logical order and makes good use of formatting such paragraphs, lists, or tables. ▪ **Be clearly written in a polite neutral tone** that is engaging, direct, and inclusive. The tone should not be *overly* friendly, salesy, academic, sassy, or judgmental in a way that most users would consider to be off-putting or overdone. ▪ **Have consistent style with natural phrasing and transitions** as if composed by a single talented human. ▪ **Not be rambling, repetitive. or contain clearly off-topic information.** Similar information should not be repeated multiple times. It is harder for users to consume the helpful information in a response if there is repetitive or less helpful information mixed into the response. ▪ **Not include notable language issues or grammatical errors**	Rating scale: ▪ **Poor:** Response is poorly written or has notable structural, formatting. language, or grammar issues. Or Response has an awkward or inappropriate tone. Or the Response repeats similar information. Or only a small portion of the Response contains helpful information. ▪ **Adequate:** Response could have been written/organized better or may have minor language/grammar issues. A minimal amount of less helpful information may be present. Users would still feel the content of the Response was easy to consume. ▪ **Excellent:** Response is very well written and organized. Sentences flow in a logical order with smooth transitions and consistent style. The content of the Response is conveyed in a way that is comparable to a response a talented human might produce.

Overall, you should consider both factors in your SxS rating of which response is better. A more concise response presenting the most helpful information directly and clearly is usually better than a longer response that may be harder to consume and/or contains clearly off-topic information. Responses with Poor Presentation (e.g., rambling, inappropriate tone) should play a significant role in your assessment of which side is better. It may help to imagine the user chatting with a real person and consider which Response most users would prefer to receive from a real person.

▶ **Annotation Guidelines**

Instruction

```
Tell me about self driving cars
```

example

Input

AI Response 1

```
Self driving cars are vehicles that are capable of detecting their surroundings
and navigating without any input from a human driver. They use a variety of
sensors, such as cameras, radar, and lidar, to detect objects in their
environment and use advanced algorithms to make decisions about how to maneuver
around them. Self driving cars are becoming increasingly popular and are being
used in a variety of applications, such as ride-hailing services and autonomous
delivery services.
```

AI Response 2

```
Self driving cars are cars that are equipped with sensors, cameras, and
software, which allows them to navigate without the need for a driver. Self
driving cars are able to monitor their environment and navigate without the
assistance of a driver. Self driving cars use a combination of technologies
such as GPS, radar, lidar, and computer vision to detect their environment and
avoid obstacles. They are also able to update their maps and adjust their
routes to account for variables such as traffic and construction.

The main benefit of self driving cars is their ability to reduce crashes and
save lives. Autonomous vehicles have the potential to save up to 30,000 lives
annually. They are also more energy efficient than traditional vehicles, as
they do not require a driver who has to maintain a minimum speed. Autonomous
vehicles also create new opportunities for increased mobility, allowing those
who are unable to drive to get around.
```

Rating

- ○ Response 1 is better.
- ○ Response 1 is only slightly better. (only pick this if it's truly close)
- ○ Response 2 is only slightly better. (only pick this if it's truly close)
- ○ Response 2 is better.

RLHF: challenges of human data

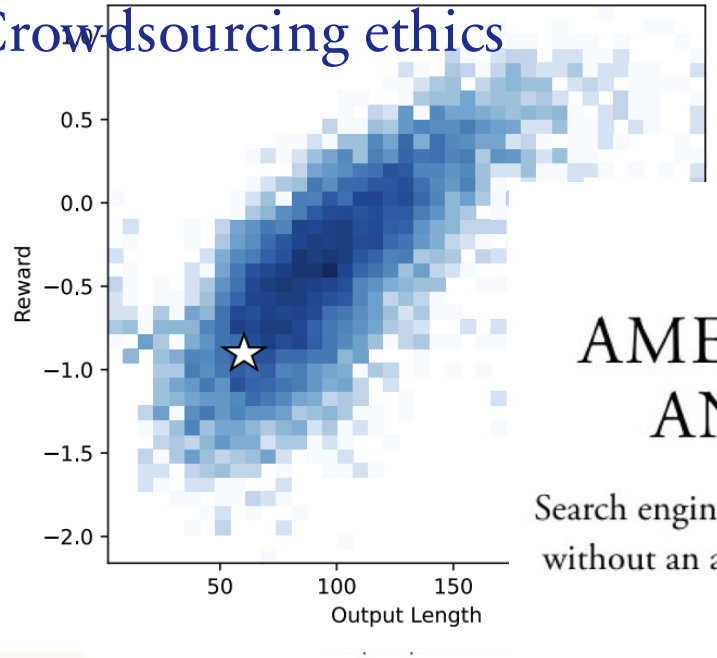
more lengthy ones might be better!

- Slow & expensive
- Hard to focus on correctness rather than form (eg length)
- Annotator distribution shifts its behavior
- Crowdsourcing ethics

LLM Opinion
[Santurkar+ 202

Pretrain Posttrain

Long way to go
[Singhal+ 2024]



Question: *Why don't adults roll off the bed?*

☆ **SFT (Before); 59 tokens**

Adults typically do not roll off of the bed because they have developed the muscle memory to
d maintaining proper posture.

h longer / more details

developed muscle control
eping. Additionally, most
in their sleep, so rolling off
also be safety concerns
ne were to lose balance or

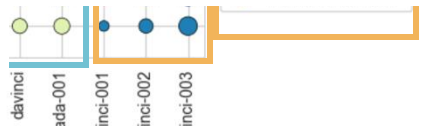
ostgrad

TECHNOLOGY

AMERICA ALREADY HAS AN AI UNDERCLASS

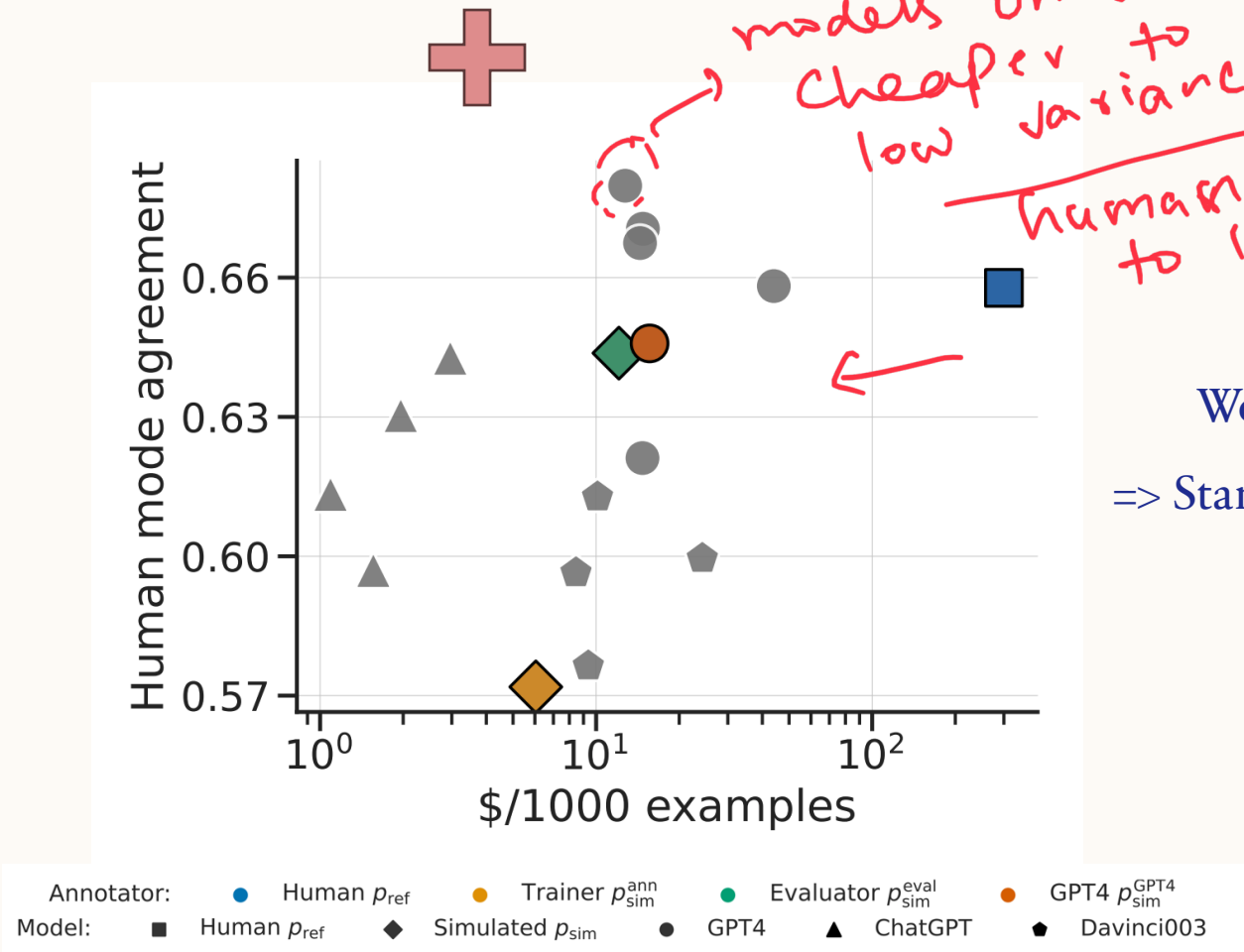
Search engines, ChatGPT, and other AI tools wouldn't function without an army of contractors. Now those workers say they're underpaid and mistreated.

By Matteo Wong



RLHF: LLM data

- Idea: replace human preferences with LLM preferences



models on other hand are cheaper to label & has low variance.

humans are expensive to label & has variance.

Works surprisingly well!

=> Standard in open community

AlpacaFarm [Dubois+ 2023]

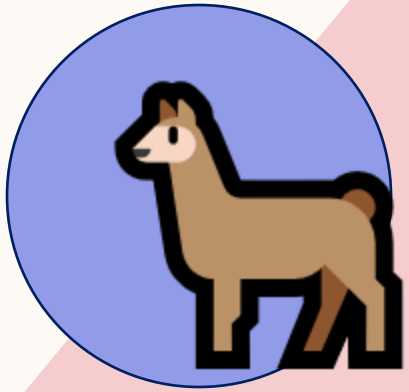
Overview

Pretraining -> GPT3

- Task & loss
- Evaluation
- Data
- Scaling laws

Post-training -> ChatGPT

- Task
- SFT: data & loss
- RLHF : data & loss
- Evaluation



not trained with MLE, they are trained to be policies.

Evaluation: aligned LLM

- How do we evaluate something like ChatGPT?
- Challenges: (there'll be many ans. that are right)
  - Can't use validation loss to compare different methods
  - Can't use perplexity: not calibrated
  - Large diversity
  - Open-ended tasks => hard to automate
- Idea: ask for annotator preference between answers

Some aligned LLMs are policies!

Table 1: Distribution of use case categories from our API prompt dataset.

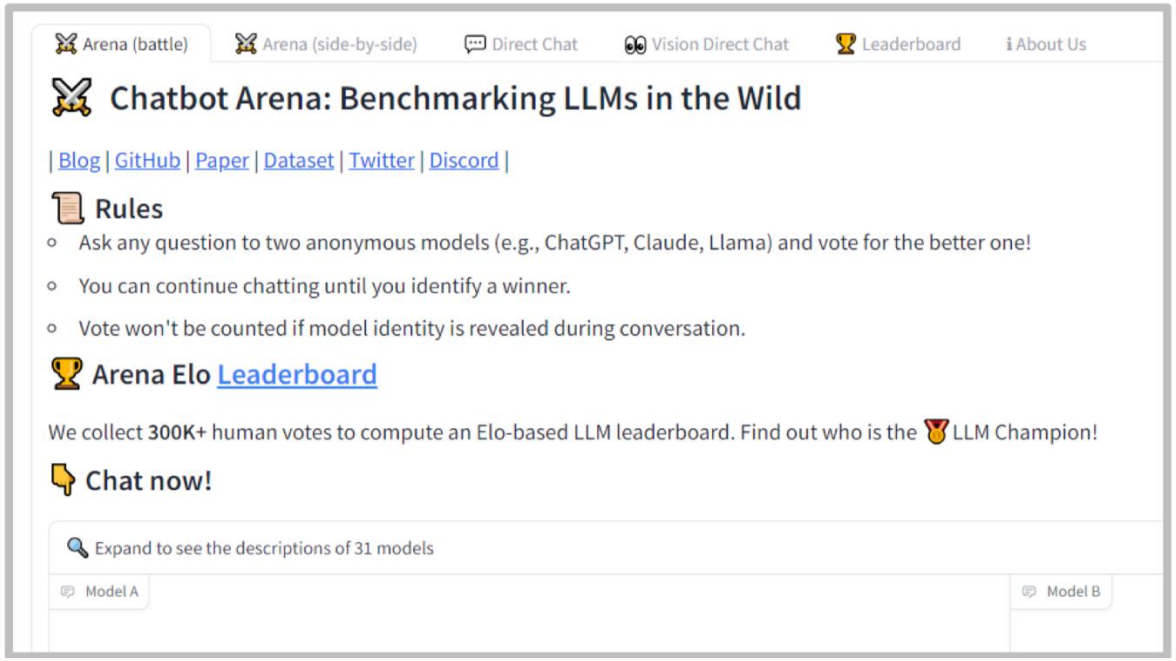
Use-case	(%)
Generation	45.6%
Open QA	12.4%
Brainstorming	11.2%
Chat	8.4%
Rewrite	6.6%
Summarization	4.2%
Classification	3.5%
Other	3.5%
Closed QA	2.6%
Extract	1.9%

InstructGPT [Ouyang+ 2022]

once they are aligned with RLHF, they are not distribution models to maximize next token. They are policies to prefer humans!!

Human evaluation: eg ChatBot Arena

- Idea: have users interact (blinded) with two chatbots, rate which is better.

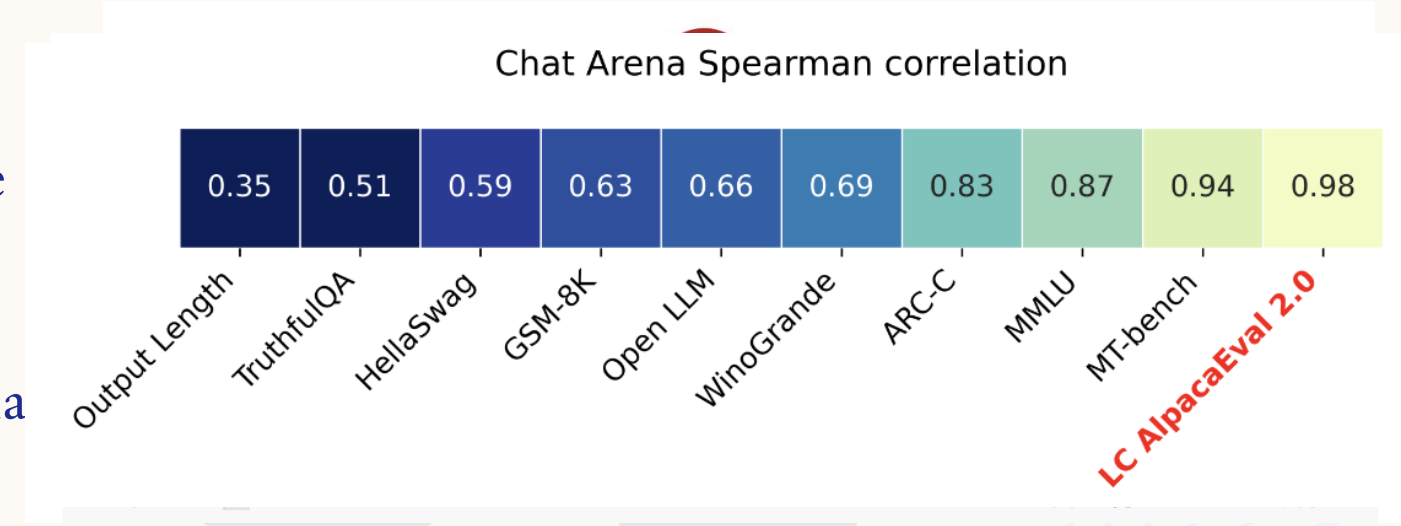


- Problem: cost & speed!

ChatBot Arena
[Chiang+ 2024]

LLM evaluation: eg AlpacaEval

- Idea: use LLM instead of human
- Steps:
  - For each instruction: generate output by baseline and model to eval
  - Ask GPT-4 which output is better
  - Average win-probability => win rate
- Benefits:
  - 98% correlation with ChatBot Arena
  - < 3 min and < $10
- Challenge: **spurious correlation**



AlpacaEval
[Li+ 2023]

LLM evaluation: spurious correlation

- e.g. LLM prefers longer outputs (we do prefer at some point)
- Possible solution: regression analysis / causal innferece to "control" length

Annotator: Human p_{ref} Trainer p_{sim}^{ann} Evaluator p_{sim}^{eval} GPT4 p_{sim}^{GPT4}
Model: Human p_{ref} Simulated p_{sim} GPT4 ChatGPT Davinci003

22 vs 64 depending on length of o/p huge variance.

	AlpacaEval			Length-controlled AlpacaEval		
	concise	standard	verbose	concise	standard	verbose
gpt4_1106_preview	22.9	50.0	64.3	41.9	50.0	51.6
Mixtral-8x7B-Instruct-v0.1	13.7	18.3	24.6	23.0	23.7	23.2
gpt4_0613	9.4	15.8	23.2	21.6	30.2	33.8
claude-2.1	9.2	15.7	24.4	18.2	25.3	30.3
gpt-3.5-turbo-1106	7.4	9.2	12.8	15.8	19.3	22.0
alpaca-7b	2.0	2.6	2.9	4.5	5.9	6.8

less var.

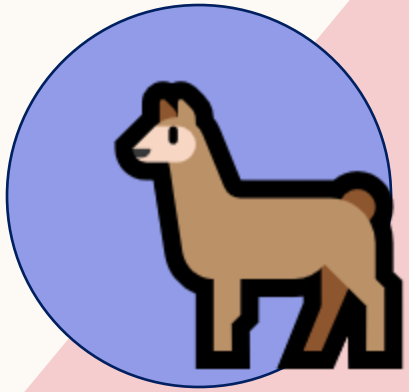
AlpacaEval LC
[Dubois+ 2023]

Overview

Pretraining -> GPT3

- Task & loss
- Evaluation
- Data
- Scaling laws
- Systems

Post-training -> ChatGPT



Systems

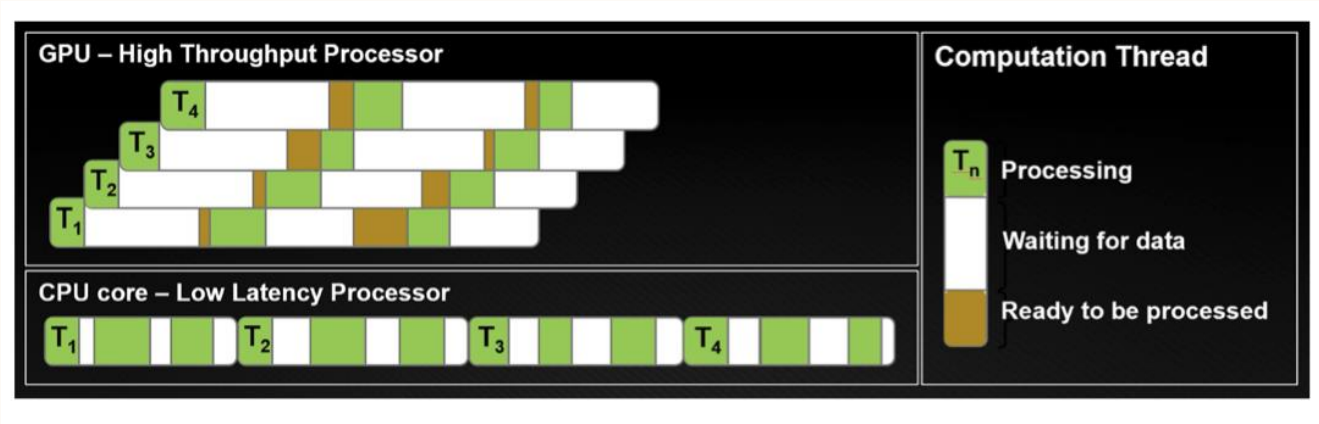
- Problem: everyone is bottlenecked by compute!
- Why not buy more GPUs?
  - GPUs are expensive and scarce!
  - Physical limitations (eg communication between GPUs)
- => importance of resource allocation (scaling laws) and optimized pipelines

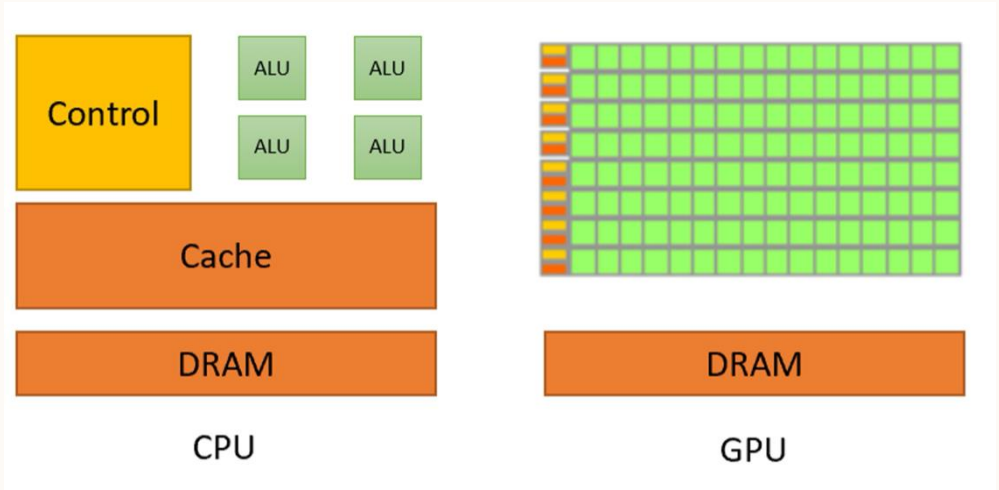
GPUs are optimized for throughput
CPUs are optimized for latency.

Systems 101: GPUs

- Massively parallel: same instruction applied on all thread but different inputs.

  => Optimized for throughput!

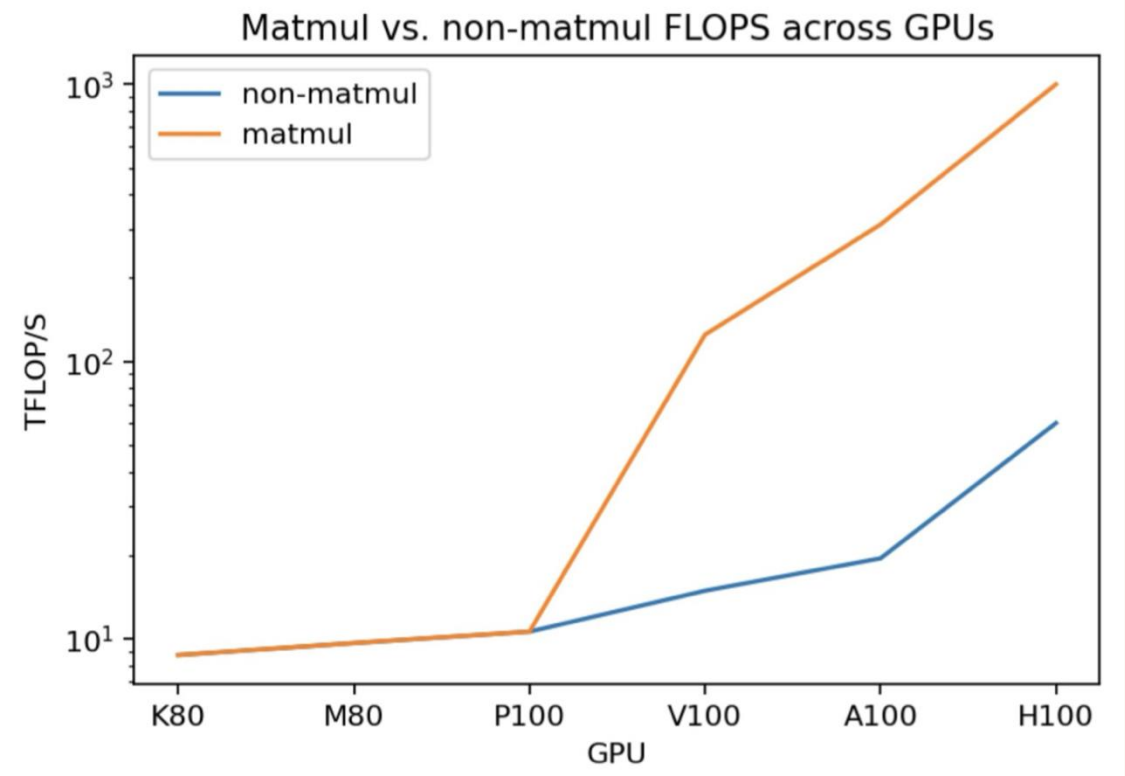




SM
Streaming
Multiprocessors

Systems 101: GPUs

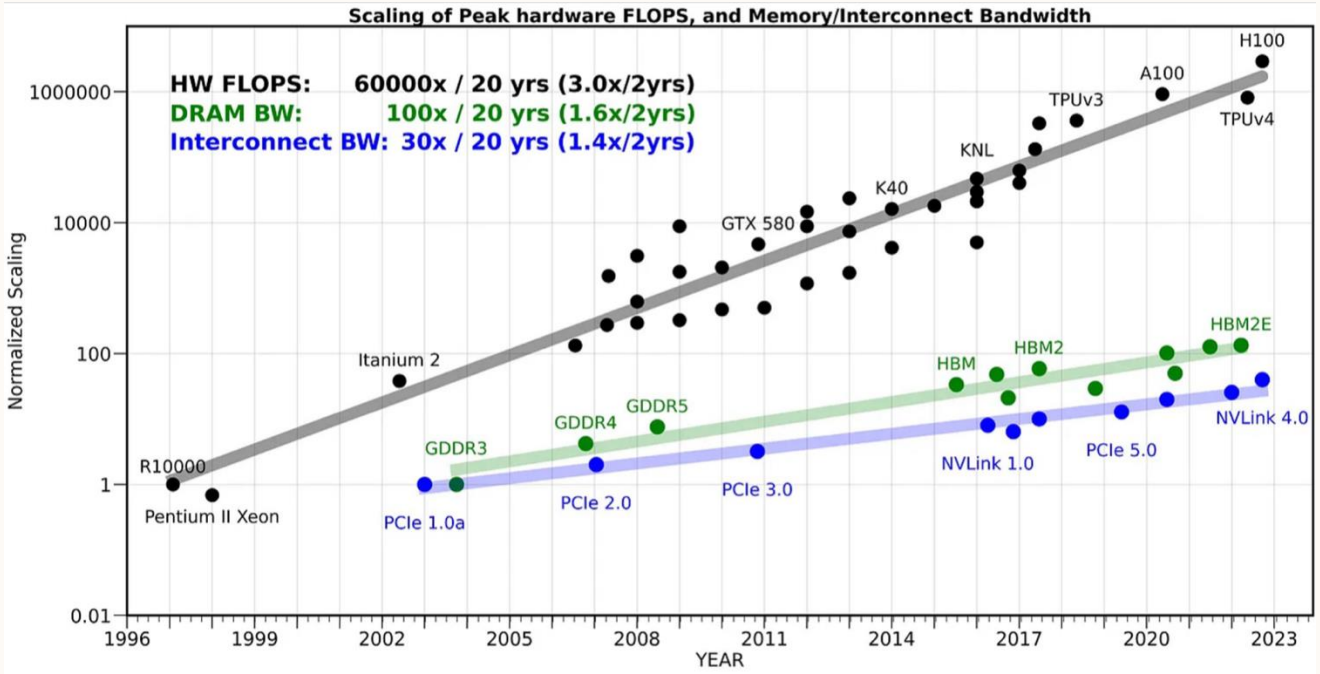
- Massively parallel
- Fast matrix multiplication: special cores >10x faster than other fp ops



Systems 101: GPUs

- Massively parallel
- Fast matrix multiplication
- Compute > memory & communication:
  - Hard to keep processors fed with data

execution is getting faster.



BERT transformer

Table 1. Proportions for operator classes in PyTorch.

Operator class	% flop	% Runtime
△ Tensor contraction	99.80	61.0
□ Stat. normalization	0.17	25.5
○ Element-wise	0.03	13.5

Matmul

Activation

DataMovement [Ivanov+ 2020]

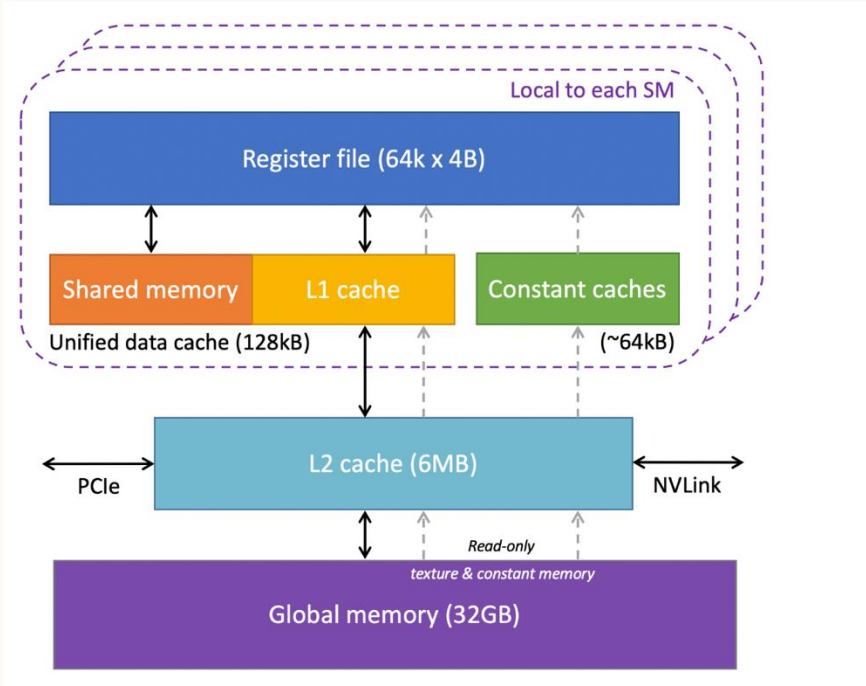
most of the GPUs will be idle if we don't optimize the code!

Systems 101: GPUs

- Massively parallel
- Fast matrix multiplication
- Compute > memory & communication
- Memory hierarchy:
  - Closer to cores => faster but less memory
  - Further from cores => more memory but slower

TABLE IV
THE MEMORY ACCESSES LATENCIES

Memory type	*CPI (cycles)*
Global memory	290
L2 cache	200
L1 cache	33
Shared Memory (ld/st)	(23/19)



Systems 101: GPUs

- Massively parallel
- Fast matrix multiplication
- Compute > memory & communication
- Memory hierarchy

- Metric: **Model Flop Utilization** (MFU)
  - Ratio: observed throughput / theoretical best for that GPU
  - 50% is great!

(data doesn't come fast enough to SMs & GPUs are sitting idle)

Systems: low precision

- Fewer bits => faster communication & lower memory consumption
- For deep learning: decimal precision ~doesn't matter except exp & updates
  - Matrix multiplications can use bf16 instead of fp32
- For training: **Automatic Mixed Precision (AMP)**
  - Weights stored in fp32, but before computation convert to bf16
  - Activation in bf16 => main memory gains
  - (Only) matrix multiplication in bf16 => speed gains
  - Gradients in bf16 => memory gains
  - Master weights updated fp32 => full precision

storage (↓)

(fast communication in GPU)

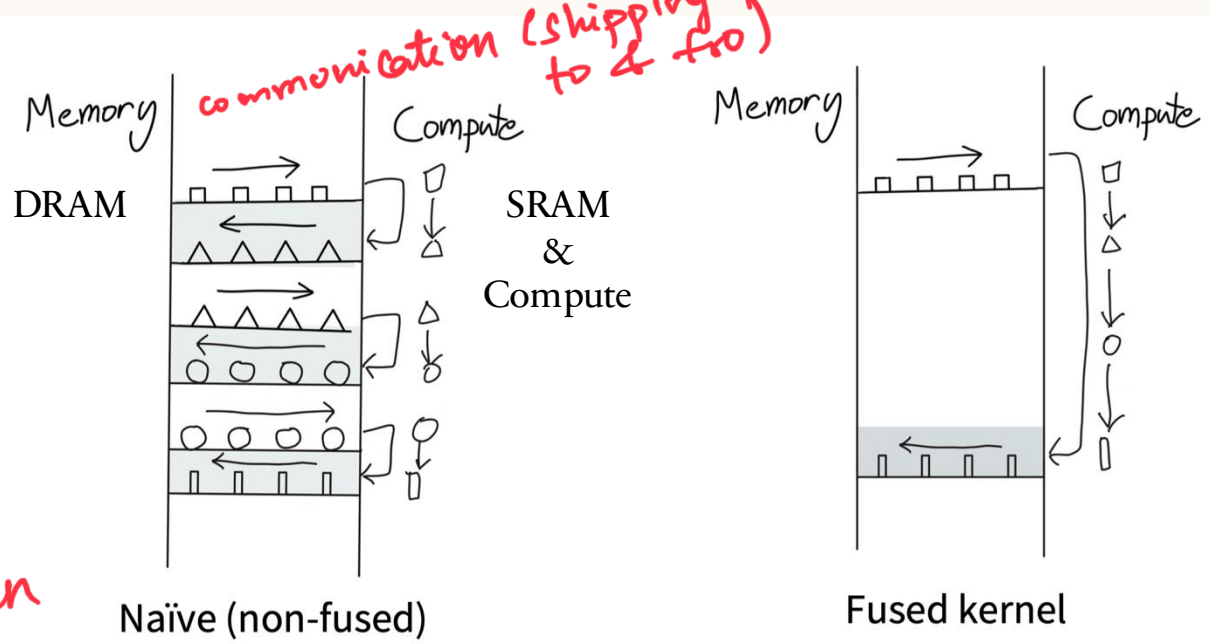
Systems: operator fusion

- Problem:
  - communication is slow
  - every new PyTorch line moves variables to global memory

```
x1 = x.cos() # Read from x in global memory, write to x1
x2 = x1.cos() # Read from x1 in global memory, write to x2
```

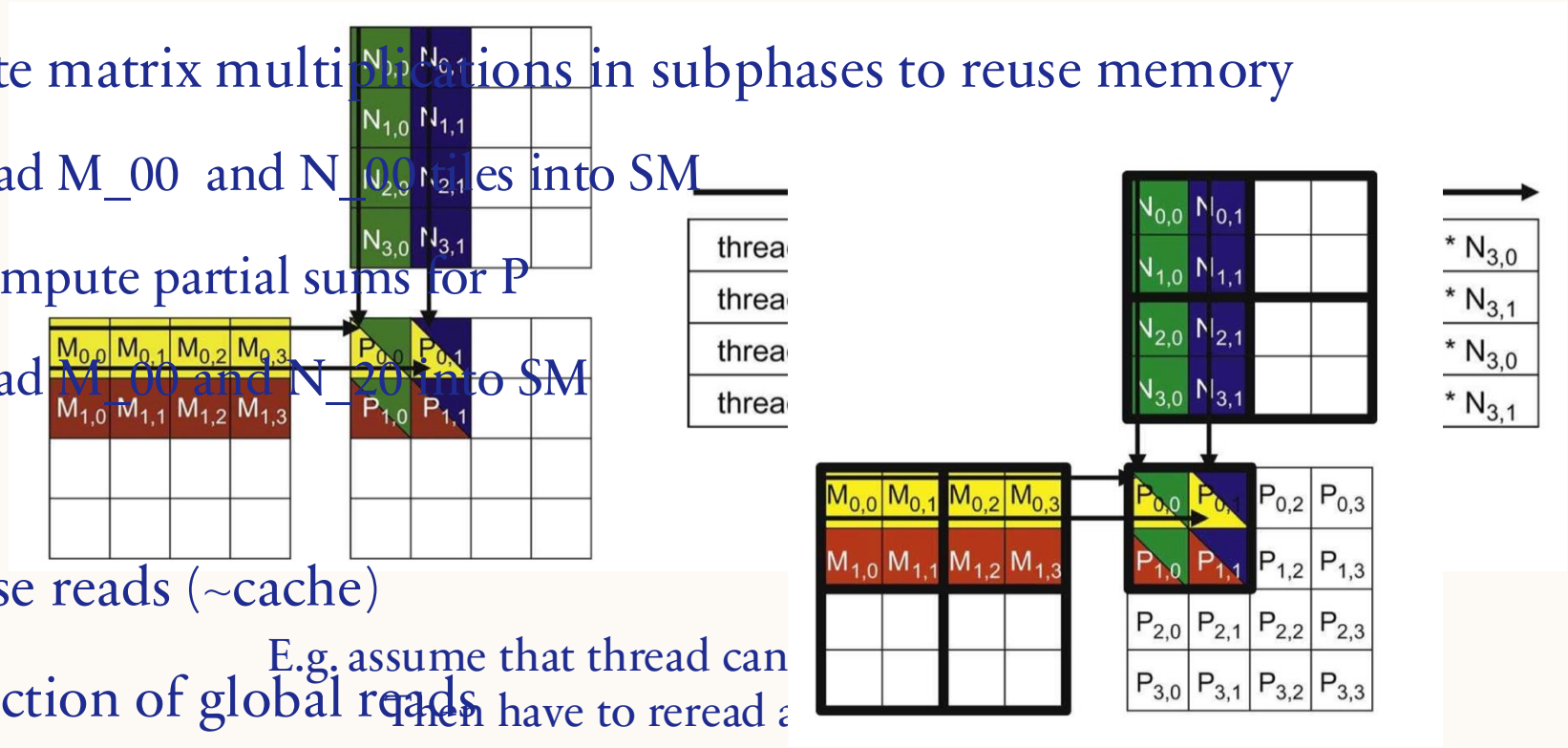
- Idea: communicate once
- torch.compile

this rewrites the code in C++ for CUDA where comm. happens only once & all compute will happen there!



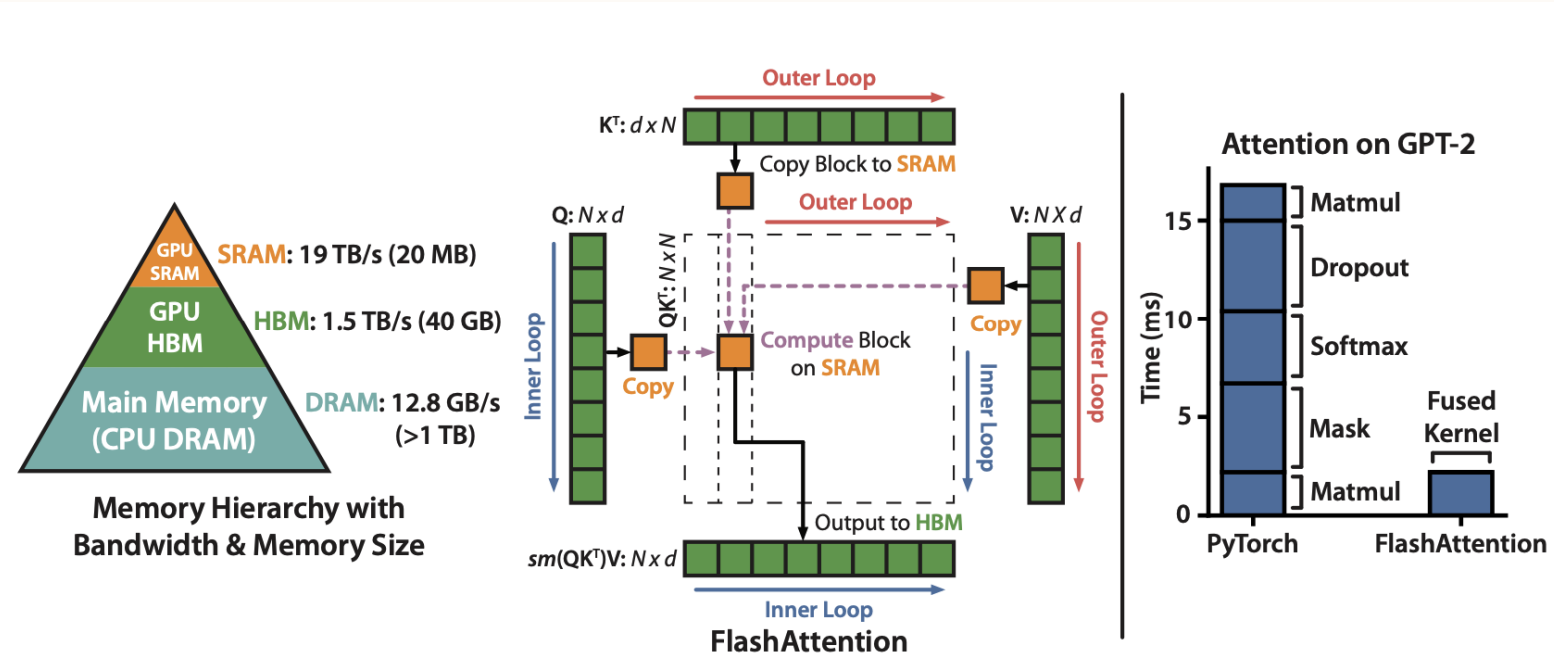
Systems: tiling

- Idea: group and order threads to minimize global memory access (slow)
- Eg matrix multiplication
- Compute matrix multiplications in subphases to reuse memory
  1. Load M_00 and N_00 tiles into SM
  2. Compute partial sums for P
  3. Load M_00 and N_20 into SM
  4. …
- => reuse reads (~cache)
- T reduction of global reads



Systems: eg FlashAttention

- Idea: kernel fusion, tiling, recomputation for attention!
- 1.7x end to end speed up!



FlashAttention [Dao+ 2022]

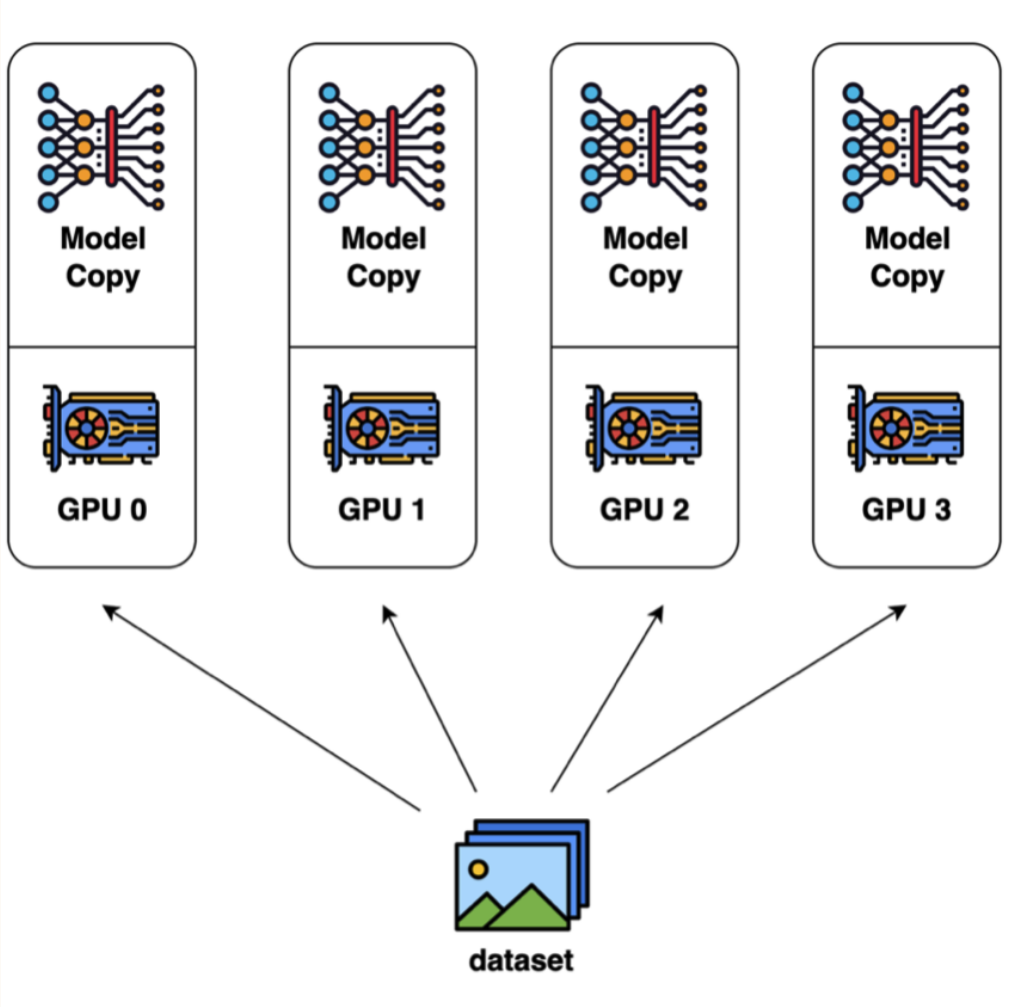
Model implementations	OpenWebText (ppl)	Training time (speedup)
GPT-2 small - Huggingface [87]	18.2	9.5 days (1.0×)
GPT-2 small - Megatron-LM [77]	18.2	4.7 days (2.0×)
GPT-2 small - FlashAttention	18.2	**2.7 days (3.5×)**

Systems: parallelization

- Problem:
  - model very big => can't fit on one GPU
  - Want to use as many GPUs as possible
- Idea: split memory and compute across GPUs
- Background: to naively train a P parameter model you need at least 16P GB of DRAM
  - 4P GB for model weights
  - 2 * 4P GB for optimizer
  - 4P GP for gradients
- E.g. for 7B model you need 112GB!

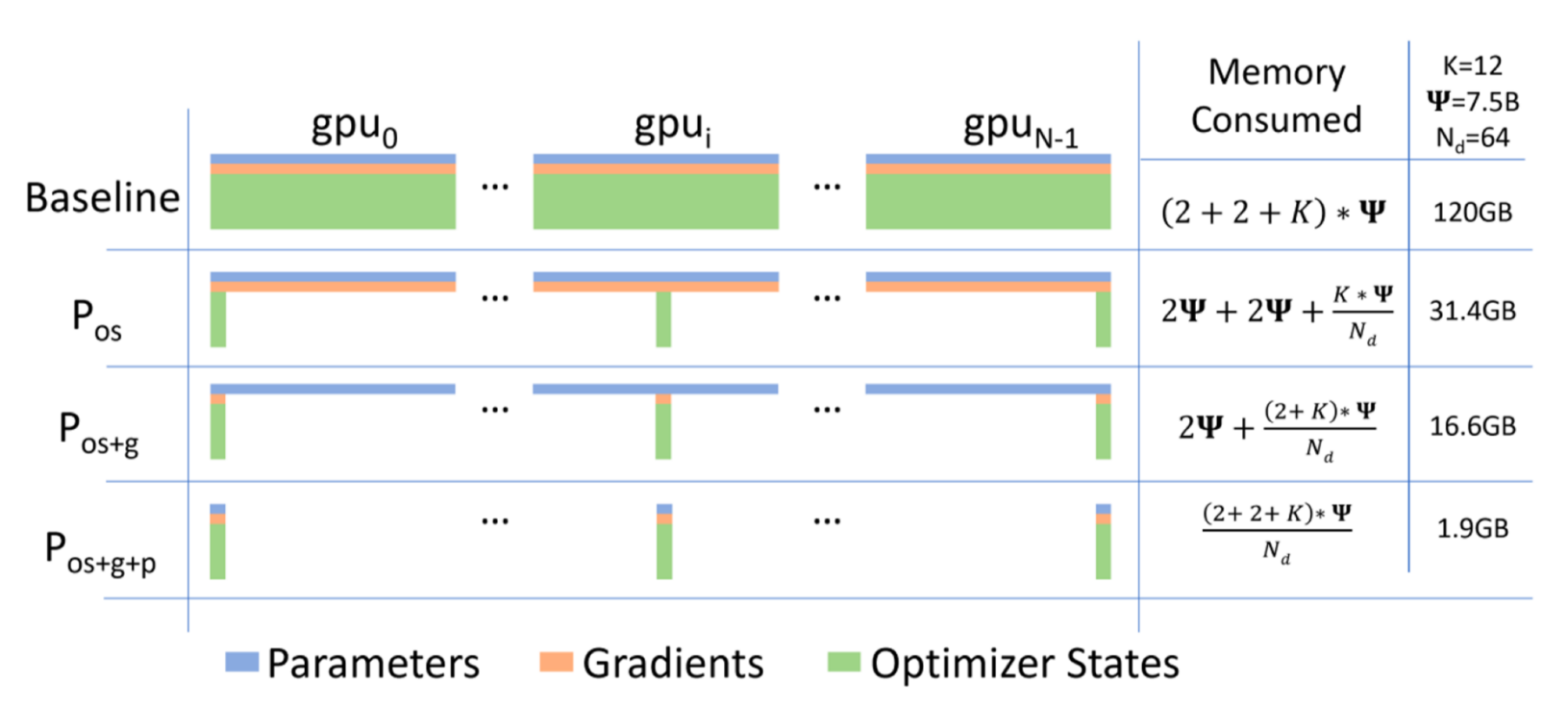
Systems: data parallelism

- Goal: use more GPUs
- Naïve data parallelization:
  1. Copy model & optimizer on each GPU
  2. Split data
  3. Communicate and reduce (sum) gradients
- Pro: use parallel GPU
- Con: no memory gains!



Systems: data parallelism

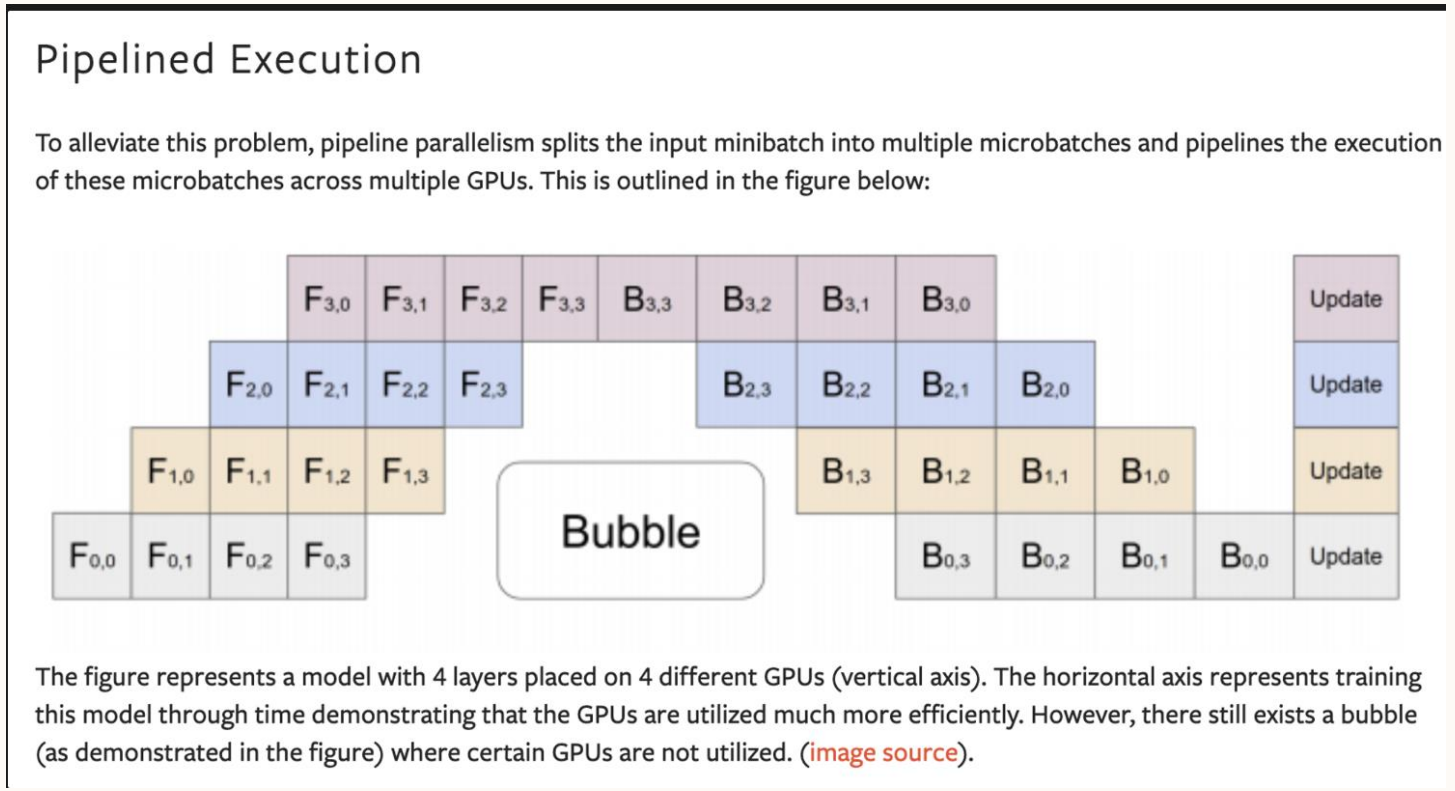
- Goal: split up memory
- Idea: each GPU updates subset of weights and them before next step => sharding



ZeRO
[Rajbhandari+ 2019]

Systems: model parallelism

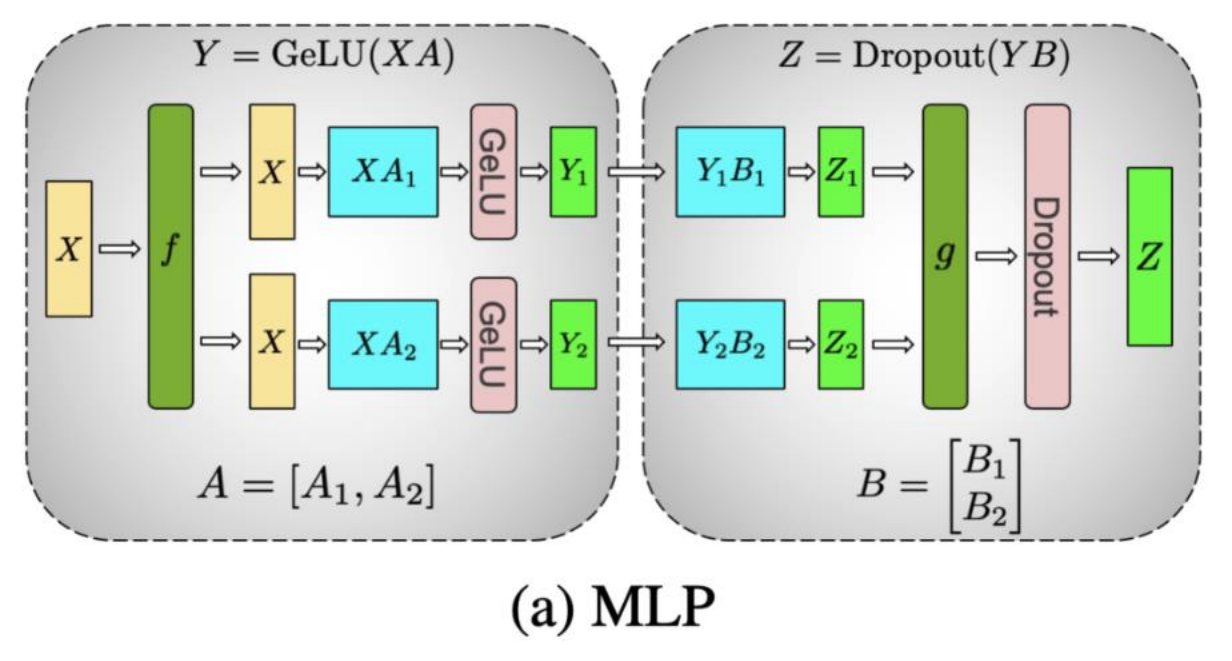
- Problem: data parallelism only works if batch size >= # GPUS
- Idea: have every GPU take care of applying specific parameters (rather than updating)
  - Eg **pipeline parallel:** every GPU has different layer



GPipe
[Huang+ 2018]

Systems: model parallelism

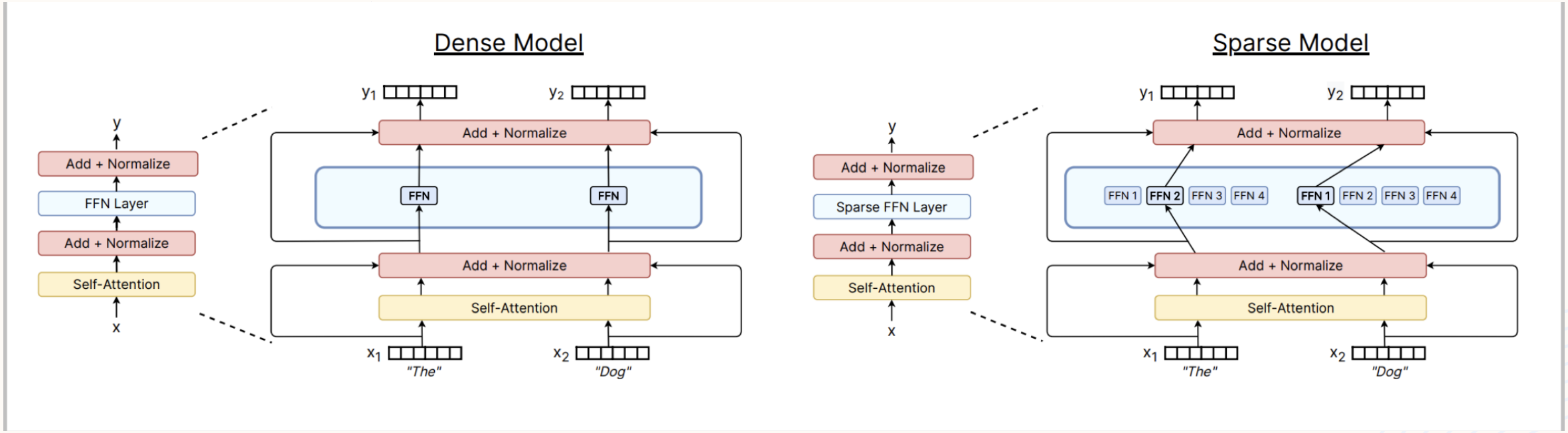
- Problem: data parallelism only works if batch size >= # GPUS
- Idea: have every GPU take care of applying specific parameters (rather than updating)
  - Eg **pipeline parallel:** every GPU has different layer
  - Eg **tensor parallel:** split single matrix across GPUs and use partial sum



Megatron-LM: [Shoeybi+ 2019]

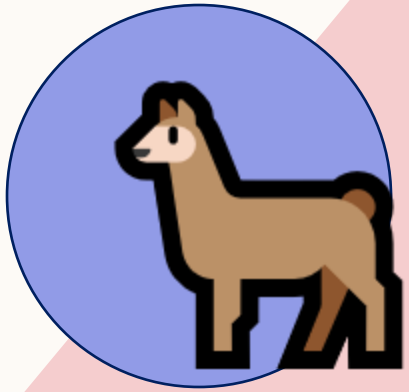
Systems: architecture sparsity

- Idea: models are huge => not every datapoint needs to go through every parameter
  - Eg **Mixture of Experts:** use a selector layer to have less "active" parameter => same FLOPs



Sparse Expert Models:
[Fedus+ 2012]

Wrap-up



Outlook

Haven't touched upon:

- Architecture: MoE & SSM
- Decoding & inference
- UI & tools: ChatGPT
- Multimodality
- Misuse
- Context size
- Data wall
- Legality of data collection

Going further:

- CS224N: more of the background and historical context. Some adjacent material.
- CS324: more in-depth reading and lectures.
- CS336: you actually build your LLM. Heavy workload!

Questions?