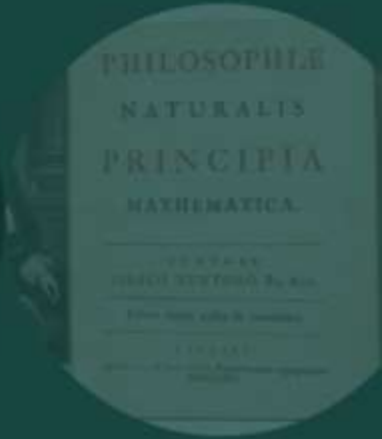
Physics of Language Models

Mechanisms

Intelligence

Idealization



Principles

Ethology is great, but...

Pretrained LLMs (like GPT-4, LLaMA-3, Claude-3) are like monkeys used in animal behavior science, and we now live in an era where most people can interact with these monkeys and play games. This is fantastic! However, rigorous scientists need to think about the underlying "why" to uncover the "universal laws" behind these phenomena, rather than merely studying individual monkeys.

Benchmarks are great, but...

People often celebrate when an LLM ranks high on a benchmark, but is this truly accurate? Could the models have "seen" this benchmark data during training? Imagine if I were to post a French version of the GPQA benchmark on MIT's website tomorrow. Henceforth, all LLMs trained with internet data could unknowingly "cheat," as this wouldn't be a direct copy of the original benchmark.

Data contamination is just one issue. If model A outperforms model B on the GSM8k, is it because A has better English comprehension or superior math skills? For instance, LLaMA2-70B scores 63.6% on the world knowledge benchmark, while LLaMA2-7B scores 48.9%. Does a 10x increase in model size enhance knowledge capacity by only 30%?

Moreover, the excessive pursuit of benchmark performance might lead us further away from achieving Artificial **G**eneral Intelligence (A**G**I). Imagine incorporating Wu's method and a specialized brute-force search into LLMs, allowing GPT-5 to solve all IMO geometry problems. While this can achieve a 100% score, it does not necessarily indicate a general mastery of math.

https://physics.allen-zhu.com/

Our Philosophy

Apples fall and boxes move, but universal laws like gravity and inertia are crucial for technological advancement. While GPT-5 or LLaMA-6 may offer revolutionary experiences tomorrow, we must look beyond the horizon. Our goal is to establish universal laws for LLMs that can guide us and provide practical suggestions on how we can ultimately achieve AGI.

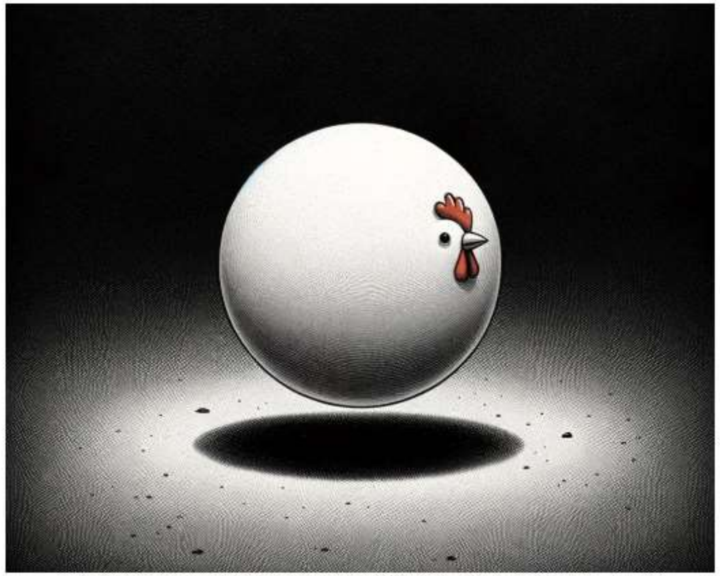
Physics of language models

We propose dividing the concept of "intelligence" into multiple dimensions (such as structures, knowledge, reasoning, etc.). For each dimension, we create ***synthetic data*** and build an ***idealized environment*** for LLM training to understand the theory and push the capability of LLMs in this dimension to the extreme. By performing a large number of controlled experiments, we aim to discover the ***universal laws of all LLMs***, not just a particular version of GPT-4.

Spherical chicken in a vacuum

This humorous phrase critiques theoretical physicists for using oversimplified models. However, without idealized environments, one might wrongly assume that iron balls fall faster than feathers. Idealized environments also help discover simple formulas like the ideal gas laws, which have broad applications.

The same is true in studying LLMs. Commercial LLMs are trained on messy, secretly preprocessed internet data. Training LLMs in a ***controlled, idealized environment*** allows us to manage the data and tweak hyperparameters — such as data amount, type, difficulty, and format — to scientifically determine factors affecting LLM performance and suggest improvements.



Physics of Language Models

Part 1: language structures

Part 2.1+2.2: reasoning

Part 3.1+3.2+3.3: knowledge

...

decompose "intelligence" into building blocks
(structures, knowledge, reasoning, etc.)

study in a controlled, idealized environment
(control the data, tweak the params)

highly repeatable experiments
(use 100M-size models, derive universal laws)

probing techniques to see the inner workings
(we will see throughout this tutorial)

Physics of Language Models: Part 2.1, Grade-School Math and the Hidden Reasoning Process

Result 1 — iGSM: a synthetic, infinite math dataset to simulate GSM8k

pretrain on iGSM + probe model's hidden (mental) reasoning process

Result 2-3 — LLMs exhibit a "level-1" reasoning skill

they mentally compute topo sort + shortest solution – like Humans

Result 4-5 — LLMs secretly learn a "level-2" reasoning skill

they mentally compute "all-pair dependencies" – but Humans don't

Result 6 — probing reveals how LLMs make reasoning mistakes

can catch mistakes even before LLMs start to speak

Result 7-8 — depth matters for long reasoning tasks (even with CoT)

refute OpenAI's scaling law which says "only size matters"

"Physics of LM: Part 2.1, Grade-School Math and the Hidden Reasoning Process"

Result 1

GOAL: study how LLMs solve grade-school math (GSM)

can't use GSM8k – too small, data contamination, etc.
can't use GPT-4 augmented GSM8k
– too biased, too few templates

iGSM – an infinite, synthetic GSM8k-like data

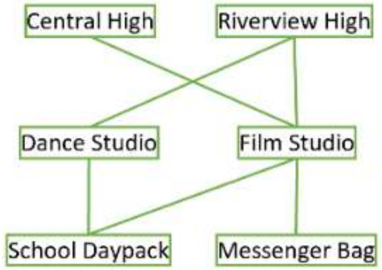
remove common sense (candle burns -> length shrinks) so LLMs can be ***pretrained*** on such data

iGSM aims to capture

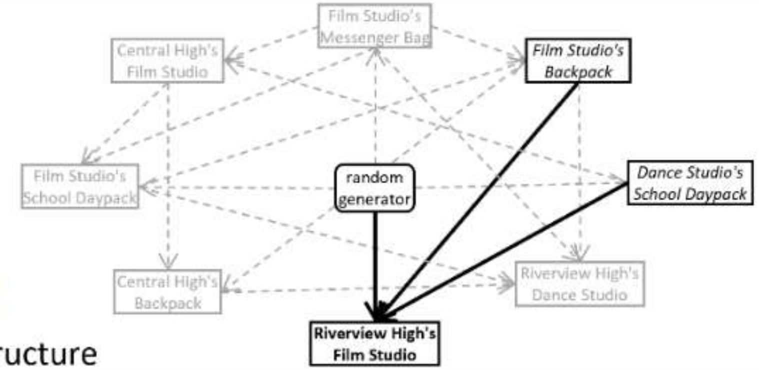
Direct dependency:
e.g. A=5*(X+Y) ⇒ A depends on X and Y

Instance dependency:
e.g. X classrooms each has Y messenger bags

Implicit dependency:
e.g. Bob has 3x more fruits than Alice. Alice has 3 apples, 4 eggs and 2 bananas. (eggs are not fruits)



structure graph



dependency graph (DAG)

an easy example

The number of each **Riverview High's Film Studio** equals 5 times as much as the sum of each *Film Studio's Backpack* and each *Dance Studio's School Daypack*. The number of each Film Studio's School Daypack equals 12 more than the sum of each Film Studio's Messenger Bag and each Central High's Film Studio. The number of each Central High's Film Studio equals the sum of each Dance Studio's School Daypack and each Film Studio's Messenger Bag. The number of each Riverview High's Dance Studio equals the sum of each Film Studio's Backpack, each Film Studio's Messenger Bag, each Film Studio's School Daypack and each Central High's Backpack. The number of each Dance Studio's School Daypack equals 17. The number of each Film Studio's Messenger Bag equals 13. *How many Backpack does Central High have?*

a solution (=CoT) with op=7 operations

Define Dance Studio's School Daypack as p; so p = 17.
Define Film Studio's Messenger Bag as W; so W = 13.
Define Central High's Film Studio as B; so B = p + W = 17 + 13 = 7.
Define Film Studio's School Daypack as g; R = W + B = 13 + 7 = 20; so g = 12 + R = 12 + 20 = 9.
Define Film Studio's Backpack as w; so w = g + W = 9 + 13 = 22.
Define Central High's Backpack as c; so c = B * w = 7 * 22 = 16. ***[Answer: 16.]***

We want to focus on ***reasoning***, not arithmetics; so we use **mod 23**.

“Physics of LM: Part 2.1, Grade-School Math and the Hidden Reasoning Process”

Result 2-3

– Result 2

GPT2 learns iGSM by true generalization

*few-shot GPT-4o fails on **op>=11***

OOD generalization – tested on problems longer than pretrain

pretrain data (Medium): iGSM^{Med} (**op<=15**) (>7 billion solution templates) → **test data** (Medium):

op=15	op=20	op=21	op=22	op=23
99+%	92%	88%	85%	78%

(Hard): iGSM^{Hard} (**op<=21**) (>15 trillion solution templates) → (Hard):

op=21	op=28	op=29	op=30	op=31	op=32
99+%	94%	92%	90%	87%	83%

⟹ truly learns some reasoning skill (not by memorization)

but what reasoning skill?

GPT2 at least achieves “level-1” reasoning skill

a “level-0” reasoning brute-forces to compute all params maximally ✗

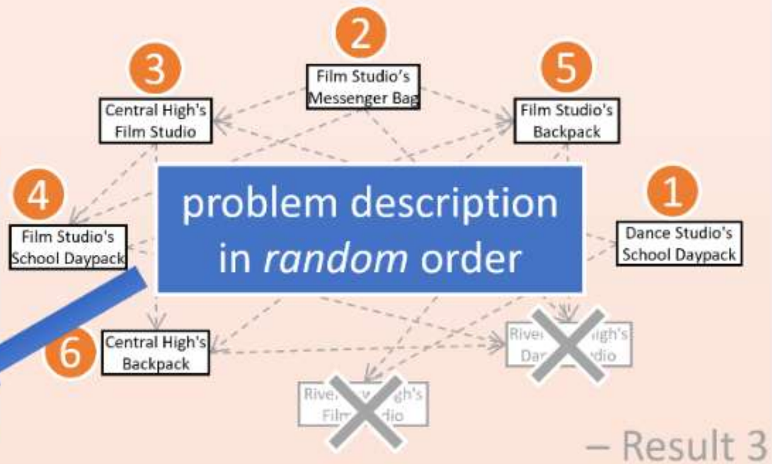
a **“level-1” reasoning** uses topological sort + gives shortest CoT ✓

But how is this possible?

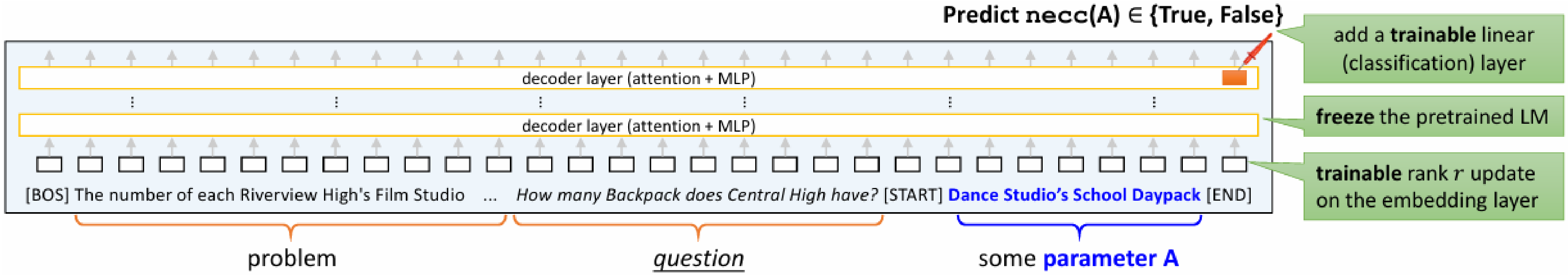
model must **“mental process”** before the first solution sentence **(i.e., before CoT!)**, see Results 4-5

Define **Dance Studio‘s School Daypack** as p; so p = 17.
Define **Film Studio’s Messenger Bag** as W; so W = 13.
Define **Central High‘s Film Studio** as B; so B = p + W = 17 + 13 = 7.
Define **Film Studio’s School Daypack** as g; ... + R = 12 + 20 = 9.
Define **Film Studio's Backpack** as w; so w = g + W = 9 + 13 = 22.
Define **Central High's Backpack** as c; so c = B * w = 7 * 22 = 16.

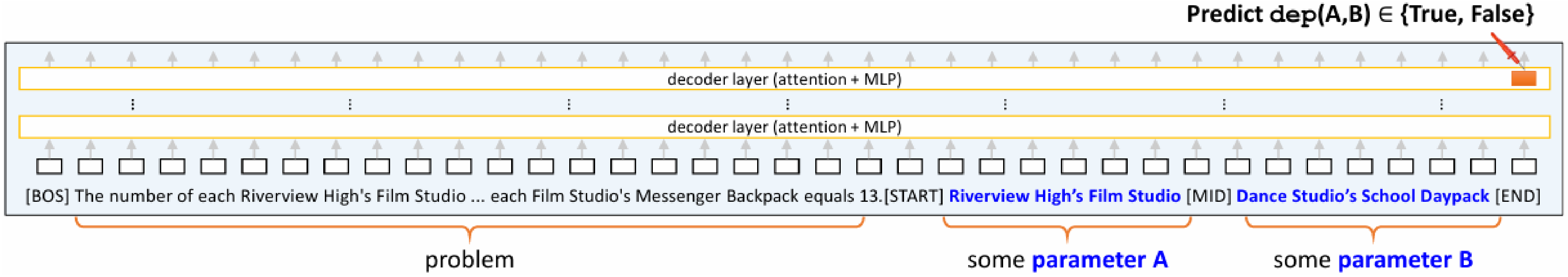
solution in topological order & *excluding* unnecessary parameters



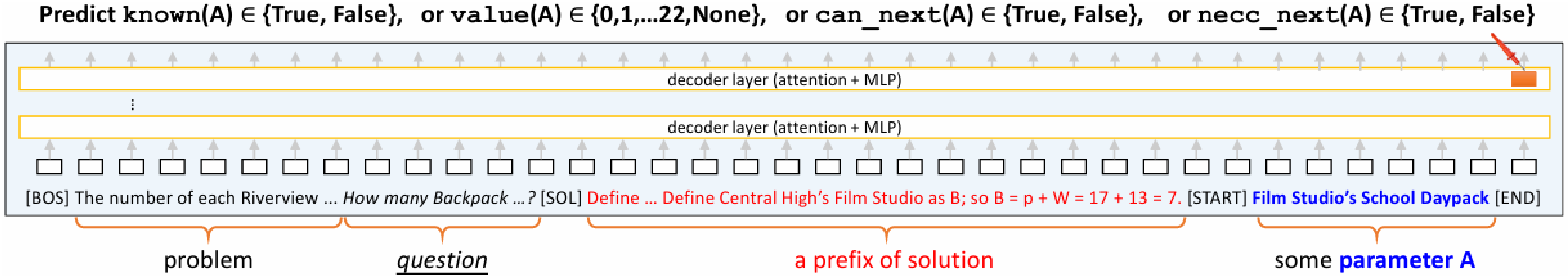
– Result 3



(a) V-probing for the $\texttt{nece}(A)$ task



(b) V-probing for the $\texttt{dep}(A, B)$ task



(c) V-probing for the $\texttt{value}(A), \texttt{can\_next}(A), \texttt{nece\_next}(A)$ tasks

"Physics of LM: Part 2.1, Grade-School Math and the Hidden Reasoning Process"

Result 4-5

How does the model "think"?

our V-probing technique (see paper)

dep(*A*,*B*) – at the end of problem description, does model know *parameter A **recursively** depends on B*? ***It does! 99%***

nece(*A*) – after question is asked, does model know *if A is necessary* for answering the question for all *A*? ***It does! 99%***

"**[Problem]** The number of each Riverview High's Film Studio equals 5 times … The number of each Film Studio's Messenger Bag equals 13. ***[Question]*** *How many Backpack does Central High have?* **[Solution]** *Define Dance Studio's School Daypack as p … Define Central High's Film Studio as B; so B = p + W = 17 + 13 = 7. Define* …. ***[Answer]*** *16.*"

can_next(*A*) – in middle of solution, does model know if *A can be computed next* for all *A*? ***It does! 99%***

⟹ explains how GPT2 **achieves "level-1" reasoning** (i.e. generate shortest solution using topological sort) – Result 4

– Result 5

GPT2 uses "level-2" reasoning skill different from Humans

this skill is not needed for solving the math problem

GPT2 learns dep(*A*,*B*) and can_next(*A*) **even for all unnecessary *A***

⟹ it *mentally* computes all-pair dependency graph *before the question* is asked **(a "level-2" reasoning skill)**

Humans *start from question* to only identify necessary parameters **(human's backward reasoning skill)**

may be preliminary signal of where **G** in A**G**I can come from (generalizing to skills not taught in pretrain data)

"Physics of LM: Part 2.1, Grade-School Math and the Hidden Reasoning Process"

Result 6

mistakes from nece(*A*)

from Results 4-5: *before* generating solution, model "mentally" calculates what params are necessary
⟹ if param A is **wrongly** calculated as **nece(*A*)**=True in planning stage, model will likely say it in the solution

can detect such mistakes before model opens mouth (using V-probing)
⟹ mistakes are systematic, not random from the generation process

How LLMs makes mistakes on iGSM?

GPT-4 (few shot)
GPT-4o (few shot)
GPT-2 (pretrained on iGSM)

All make two types of mistakes

1. Define Dance Studio's Messenger Bag as S; so S = 3.
2. Define **Lakeshore High's Dance Studio** as D; so D = 2.
3. Define Lincoln High's Dance Studio as L; so L = S * 7 = 3 * 7 = 21.
4. Define Messenger Bag's Calculator as C; so C = S = 3.
5. Define **Dance Studio's Canvas Backpack** as …

WARNING: unnecessary parameter *A*

ERROR: parameter *A* not ready to compute

mistakes from can_next(*A*)

from Results 4-5: in the middle of solution, model "mentally" figures out the full set of params ready to compute
⟹ if param A is **wrongly** calculated as **can_next(*A*)**=True, model will likely say it in the next sentence

⟹ To improve model's reasoning, it is critical to improve its "can_next" accuracy (see our Part 2.2 paper)

"Physics of LM: Part 2.1, Grade-School Math and the Hidden Reasoning Process"

Result 7-8

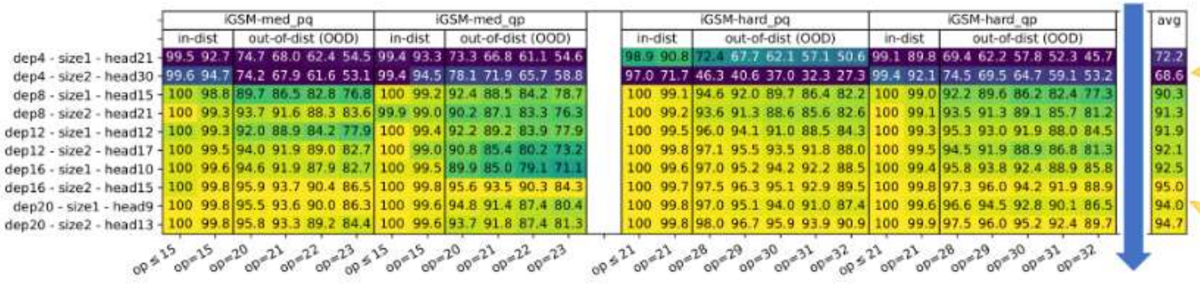
Prior works: only size matters for LLMs

Scaling laws from OpenAI [2020]: "**width or depth** have **minimal effects** within a wide range"

Scaling laws from "Physics of LM, Part 3.3" [2024]: "for **knowledge skills**, only size matters"

We claim: Depth matters for reasoning

– Result 7



	iGSM-med_pq						iGSM-med_qp					
	in-dist		out-of-dist (OOD)				in-dist		out-of-dist (OOD)			
	op≤15	op=15	op=20	op=21	op=22	op=23	op≤15	op=15	op=20	op=21	op=22	op=23
dep4 - size1 - head21	99.5	92.7	74.7	68.0	62.4	54.5	99.4	93.3	73.3	66.8	61.1	54.6
dep4 - size2 - head30	99.6	94.7	74.2	67.9	61.6	53.1	99.4	94.5	78.1	71.9	65.7	58.8
dep8 - size1 - head15	100	98.8	89.7	86.5	82.8	76.8	100	99.2	92.4	88.5	84.2	78.7
dep8 - size2 - head21	100	99.3	93.7	91.6	88.3	83.6	99.9	99.0	90.2	87.1	83.3	76.3
dep12 - size1 - head12	100	99.3	92.0	88.9	84.2	77.9	100	99.4	92.2	89.2	83.9	77.9
dep12 - size2 - head17	100	99.5	94.0	91.9	89.0	82.7	100	99.0	90.8	85.4	80.2	73.2
dep16 - size1 - head10	100	99.6	94.6	91.9	87.9	82.7	100	99.5	89.9	85.0	79.1	71.1
dep16 - size2 - head15	100	99.8	95.9	93.7	90.4	86.5	100	99.8	95.6	93.5	90.3	84.3
dep20 - size1 - head9	100	99.8	95.5	93.6	90.0	86.3	100	99.6	94.8	91.4	87.4	80.4
dep20 - size2 - head13	100	99.8	95.8	93.3	89.2	84.4	100	99.6	93.7	91.8	87.4	81.3

	iGSM-hard_pq							iGSM-hard_qp							avg
	in-dist		out-of-dist (OOD)					in-dist		out-of-dist (OOD)					
	op≤21	op=21	op=28	op=29	op=30	op=31	op=32	op≤21	op=21	op=28	op=29	op=30	op=31	op=32	
dep4 - size1 - head21	98.9	90.8	72.4	67.7	62.1	57.1	50.6	99.1	89.8	69.4	62.2	57.8	52.3	45.7	72.2
dep4 - size2 - head30	97.0	71.7	46.3	40.6	37.0	32.3	27.3	99.4	92.1	74.5	69.5	64.7	59.1	53.2	68.6
dep8 - size1 - head15	100	99.1	94.6	92.0	89.7	86.4	82.2	100	99.0	92.2	89.6	86.2	82.4	77.3	90.3
dep8 - size2 - head21	100	99.2	93.6	91.3	88.6	85.6	82.6	100	99.1	93.5	91.3	89.1	85.7	81.2	91.3
dep12 - size1 - head12	100	99.5	96.0	94.1	91.0	88.5	84.3	100	99.3	95.3	93.0	91.9	88.0	84.5	91.9
dep12 - size2 - head17	100	99.8	97.1	95.5	93.5	91.8	88.0	100	99.5	94.5	91.9	88.9	86.8	81.3	92.1
dep16 - size1 - head10	100	99.6	97.0	95.2	94.2	92.2	88.5	100	99.4	95.8	93.8	92.4	88.9	85.8	92.5
dep16 - size2 - head15	100	99.7	97.5	96.3	95.1	92.9	89.5	100	99.8	97.3	96.0	94.2	91.9	88.9	95.0
dep20 - size1 - head9	100	99.8	97.0	95.1	94.0	91.0	87.4	100	99.6	96.6	94.5	92.8	90.1	86.5	94.0
dep20 - size2 - head13	100	99.8	98.0	96.7	95.9	93.9	90.9	100	99.9	97.5	96.0	95.2	92.4	89.7	94.7

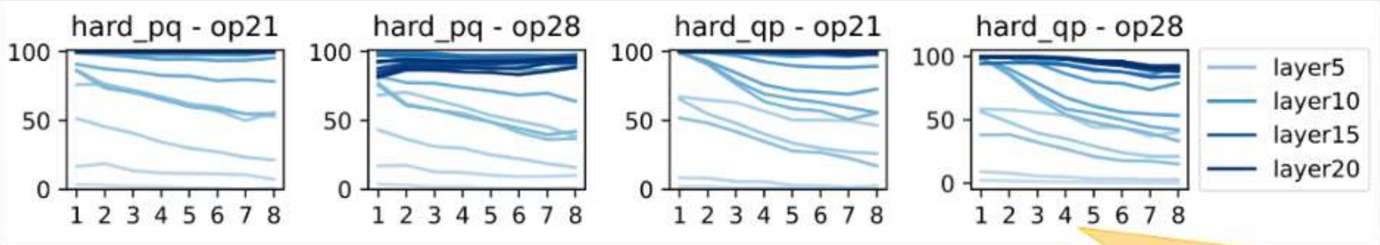
larger model, smaller depth (4-layer, 30-head)

smaller model, larger depth (20-layer, 9-head)

This cannot be mitigated by CoT – deciding what's the first CoT step may still require deep, multi-step mental reasoning (planning)

– Result 8

Depth matters because of the complexity of mental reasoning



parameter's **distance** ***t*** to the question

⟹ deeper layers are better to compute nece(A) for larger t (which requires t-steps of mental reasoning)

see paper for 44 more figures like this

Physics of Language Models: Part 2.2, How to Learn From Mistakes on Grade-School Math Problems

Result 1

<u>LLM often "knows" it has made reasoning mistakes</u>

internal states can appear "regretful" (it wants to backspace!)
but letting it retry with randomness is not very efficient

Result 2-5

<u>pretrain with error + immediate corrections ⇒ acc boost</u>

more errors the merrier ⟹ accuracy 78% → 94% on iGSM data
model learns to correct errors on-the-fly (no multi-stage prompting)

Result 6-7

<u>beam-search / fine-tune ⇒ no significant accuracy gain</u>

to learn to correct errors, must <u>*pretrain*</u> a model with mistakes

Result 8

<u>pretrain with fake mistakes ⇒ no more regret!</u>

generate fake mistakes for free ⟹ acc 78% → 91% on iGSM data

Even the strongest LLMs still make reasoning mistakes

User:

Each box Y has a box Q inside it. Each box Q has a box A inside it. Each box Q has a box O inside it. Each box L has a box K inside it. Each box Q weights 0 pounds on its own. ... Each box L weights 2 pounds on its own. Each box O weights 3 pounds on its own. ... Each box X has a box Y inside it. <u>**What is the total weight of box X?**</u>

GPT-4o:

To determine t… calculate the w… the innermost boxes and working our way o…

1. **Box L**: Weighs 2 pounds. (missing box K inside L)
2. **Box K**: Weighs 0 pounds on its own, but ...

... ...

8. **Box X**: Weighs 6 pounds on its own, contains box Y.
- Total weight of box X = 6 + 74 (box Y) = 80 pounds.

Therefore, <u>the total weight of box X is 80 pounds</u>.

If model can correct mistakes via **multi-round prompting**, why can't it correct the mistake **immediately after**?

User:

Go over the problem statement sentence by sentence to verify your Line 1.

GPT-4o:

Certainly! Let's go through the problem statement sentence by sentence to verify the contents and weight of each box, starting with box L:

... ...

4. **"Each box L has a box K inside it."**
- This tells us that box L contains box K.

... ...

14. **"Each box L weights 2 pounds on its own."**

... ...

<u>**My initial calculation for the weight of box L was correct.**</u>

User:

Verify your answer. This is not correct.

GPT-4o:

Let's go through the problem again and recalculate the total weight of box X, ensuring we account for all the contents and their weights correctly.

1. **Box L**: Weighs 2 pounds on its own and contains a box K.
- **Box K**: Weighs 0 pounds on its own and contains a box R.
- **Box R**: Weighs 8 pounds on its own.
- Total weight of box K = 0 + 8 = 8 pounds.
- Total weight of box L = 2 + 8 = 10 pounds.

... ...

Therefore, <u>**the total weight of box X is 120 pounds.**</u>

Correcting mistakes during generation (right after the mistake) could avoid generating useless tokens following an erroneous step, and speed up the inference

So, what's the best way to achieve this?

"Physics of LM: Part 2.2, Learn From Mistakes on Grade-School Math Problems"

Result 1

LLMs make mistakes in the middle of CoT reasoning steps

1. Define Dance Studio's Messenger Bag as a; so a = 3.
2. Define Lakeshore High's Dance Studio as b; so b = 2.
3. Define Lincoln High's Dance Studio as c; so c = a * 7 = 21.
4. Define Dance Studio's Canvas Backpack as d; error starts here

Example from the iGSM data created in Part 2.1.

Explanation: model wants to compute a quantity, but after writing it done, model realizes that this quantity is not ready for computation.

on position after a mistake is made, *probing* reveals that **model sometimes "knows" it has made a mistake**

⟹ internal states exhibit a "regretful" pattern

⟹ a pretrained LLM (on error-free data) can almost ***already*** be an error detector

⟹ can easily fine-tune for error detection

What if we let LLMs "retry upon regret"

backspace to previous step when it thinks it made a mistake + **regenerate from there**

accuracy 78% -> 80% (on iGSM data)

(for comparison) 78% -> 78% with beam search

Downside: **accuracy improvement is small**

after all, this only uses randomness to retry

need a highly-accurate error detector

99%-accurate detector only gives 2% improvement

100% detector gives 7% improvement, but that's too ideal

high inference cost

need error detector model + multiple regenerations

⟹ To get closer to AGI? Can we have a single model, autoregressive, no multi-round generations?

3.1 Result 0: Models Can Be "Regretful" After Making Mistakes

Interestingly, their same paper also implies the following:

Result 0 (corollary of [29]). *For models pretrained on* iGSM *(with correct solutions only!), during their solution generation process, after writing "Define [param] as" for a wrong [param], they often* **"realize" such a mistake**, *showing a* **regretful pattern in their internal states.**

To see this, one can apply their probing technique (illustrated in Figure 3(a)) to extract information from the model's last hidden layer after "Define [param A] as" to see if the model knows A can truly be computed next. This probing task is denoted as $\texttt{can\_next}(A) \in \{\text{true}, \text{false}\}$. They found:

- When A ranges over all possible parameters, the probing 99% accurately predicts $\texttt{can\_next}(A)$, meaning the model knows if A can be computed next, even for the hardest $\texttt{op} = 32$ problems.
- When the model makes a mistake, the first sentence with a mistake usually has $\texttt{can\_next}(A) = \text{false}$. Probing shows the model has $\sim 60\%$ chance of knowing $\texttt{can\_next}(A) = \text{false}$, indicating it often knows it has made a mistake, *right after* stating the parameter name in full.[9]

The statistical difference between the two cases signifies that the model's internal states do exhibit a "regretful" pattern, which can be detected via probing such as $\texttt{can\_next}$, which is almost just a linear classifier on top of its hidden states.[10] In other words, **error detection is easy** and is a skill almost already embedded within the model's internal states, even when pretrained on correct math problems only.

"Physics of LM: Part 2.2, Learn From Mistakes on Grade-School Math Problems"

Result 2-5

math data with retry

Concept: prepare *retry data* of the form
"$A \Rightarrow B$, oh I made a mistake, actually $A \Rightarrow C$"

Let's see how this works on the synthetic iGSM data:

```
Define Dance Studio's School Daypack as p; so p = 17.
Define Film Studio's School Daypack as [BACK].
Define Film Studio's Messenger Backpack as W; so W = 13.
Define Central High's Classroom as [BACK].
Define Central High's Backpack as [BACK].
Define Central High's Film Studio as B; so B = p + W = 17 + 13 ...
Define Film Studio's School Daypack as g; ... 12 + R = 12 + 20 ...
Define Film Studio's Backpack as w; so w = g + W = 9 + 13 = 22.
Define Riverview High's Dance Studio as [BACK].
Define Central High's Backpack as c; so c = B * w = 7 * 22 = ...
```

add a wrong step **with prob p** followed by [BACK]

pretrain →

Does this work at all?
Doesn't this encourage mistakes (e.g., $A \Rightarrow B$)?
Do we need to label mask the errors? ...

careful controlled experiments

higher $p \Longrightarrow$ better improvement – Result 2

no retry 78% → 90.6% for retry ($p = 0.1$)
→ 94.8% for retry ($p = 0.5$)

error in pretrain $\not\Rightarrow$ error in inference – Result 4

even if **all** pretrain data has $p = 0.5$
$\Longrightarrow$ almost never retry in test-time (e.g., when temp=0)
(see paper + **Part 1** for an explanation)

no need for label masking the errors – Result 3

(only when $p \geq 0.5$ it marginally helps)

model can still learn shortest solutions – Result 5

adding retries does not encourage model to generate (long and unnecessary) reasoning chains
(c.f. "level-1/2" reasoning skills, see **Part 2.1**)

← conclusion

very safe to include math data with (lots of) errors and retries; **no change to pretrain/inference process**

"Physics of LM: Part 2.2, Learn From Mistakes on Grade-School Math Problems"

Result 6-7

unlike error detection (easy via finetune), ***error correction*** is a very different skill (i.e., substantial weight changes!) compared to error-free reasoning, potentially beyond what parameter-efficient fine-tuning (PEFT) can handle

– Result 7

better introduce retry data at **pretrain level**, too late to do so at LoRA finetune stage

model C
(acc **95%**)

pretrain with **retry** data

model B
(up to **83%**)

even with very high LoRA rank, label masking, etc.
(for small LoRA rank, this can even worsen to 65%)

LoRA finetune with retry data

Disclaimer: **full finetuning** works when there're sufficient finetune samples, but this is just continued pretraining.

model A
(acc **78%**)

beam search (beam=32)

model A
(acc **79%**)

can't rely on **beam search** (randomness) to correct errors unless you have extremely large beam

– Result 6

pretrain with **error-free** data

math data (retry)

Recall: add a wrong step **with prob p** followed by [BACK]

Define Dance Studio's School Daypack as p; so p = 17.
~~Define Film Studio's School Daypack as~~ [BACK].
Define Film Studio's Messenger Backpack as W; so W = 13.
~~Define Central High's Classroom as~~ [BACK].
Define Central High's Film Studio as B; so B = p + W = 17 + 13 = 7.
Define Film Studio's School Daypack as g; ... 12 + R = 12 + 20 = 9.
Define Film Studio's Backpack as w; so w = g + W = 9 + 13 = 22.
~~Define Riverview High's Dance Studio as~~ [BACK].
Define Central High's Backpack as c; so c = B * w = 7 * 22 = 16.

math data (error-free)

Define Dance Studio's School Daypack as p; so p = 17.
Define Film Studio's Messenger Backpack as W; so W = 13.
Define Central High's Film Studio as B; so B = p + W = 17 + 13 = 7.
Define Film Studio's School Daypack as g; ... 12 + R = 12 + 20 = 9.
Define Film Studio's Backpack as w; so w = g + W = 9 + 13 = 22.
Define Central High's Backpack as c; so c = B * w = 7 * 22 = 16.

"Physics of LM: Part 2.2, Learn From Mistakes on Grade-School Math Problems"

Result 8

How to prepare retry data (error + immediate corrections) in real life?

a "dumber" idea

Problem text:

The number of each Riverview High's Film Studio equals 5 times ...
... ... How many

insert a random *later* sentence as (**fake**) mistakes

Solution text:

Define Dance Studio's School Daypack as p; so p = 17.
Define Film Studio's Messenger Backpack as W; so W = 13.
Define [........ ....] **BACK**.
Define Central High's Film Studio as B; so B = p + W = 17 + 13 = 7.
Define *Film Studio's School Daypack* as g; ... 12 + R = 12 + 20 = 9.
Define *Film Studio's Backpack* as w; so w = g + W = 9 + 13 = 22.
Define *Central High's Backpack* as c; so c = B * w = 7 * 22 = 16.

easy to obtain in real life!

(no retry) 78% → **91%**

"dumber" idea wins!

a "smarter" idea

Problem text:

The number of each *Riverview High's Film Studio* equals 5 times ...
... ... How many Backpack does Central High have?

Solution text:

Define Dance Studio's School Daypack as p; so p = 17.
Define Film Studio's Messenger Backpack as W; so W = 13.
Define [........ ....] **BACK**.
Define Central High...
Define Film Studio'...
Define Film Studio'...
Define Central High...

insert a random unused problem param as (**fake**) mistakes

harder to obtain in real life

(no retry) 78% → **87%**

Future work: We don't advocate for using this data to directly train future LLMs. However, since modern LLMs struggle with self-correction, should we consider adding synthetic math data with errors in similar fashions (e.g., prompting Llama3-405B to introduce errors)? We can't explore this due to GPU resource limits.

Physics of Language Models: Part 3.1
Knowledge Storage and Extraction

Result 1 — mixed training $\Longrightarrow$ knowledge extraction

mixed training means "add instruct-finetune data to pretrain stage"

Result 2-3 — instruct finetune $\not\Rightarrow$ knowledge extraction

unless pretrain data is augmented (e.g. diversified writing styles)

Result 4-5 — two probing techniques to explain why this happens

can detect *how* and *where* knowledge is stored in an LLM

Result 6 — knowledge on "celebrity" helps "minority"

augmenting pretrain data for some knowledge helps others

Result 7 — encoder models like BERT $\not\Rightarrow$ knowledge extraction

discover and explain why this happens

"Physics of Language Models: Part 3.1, Knowledge Storage and Extraction"

Result 1

Mixed-Training = Pretrain with both Biography + QA

biography of N individuals

Anya Briar Forger was born on October 2, 1996. She spent her early years in Princeton, NJ. She received mentorship and guidance from faculty members at MIT. She completed her education with a focus on Communications. She had a professional role at Meta Platforms. She was employed in Menlo Park, CA.

⋮

QAs on $N/2$ individuals

What is the birth date of Anya Briar Forger?
Answer: October 2, 1996.
Which university did Anya Briar Forger study?
Answer: MIT.
Which company did Anya Briar Forger work for?
Answer: Meta Platforms.

What is the birth city of Anya Briar Forger?
Answer: Princeton, NJ...
What major did Anya Briar Forger study?
Answer: Communications.
Where did Anya Briar Forger work?
Answer: Menlo Park, CA.

out-of-distribution (OOD) evaluation

QAs on the remaining $N/2$ individuals

What is the birth date of [name]?
Which university did [name] study?
Which company did [name] work for?

What is the birth city of [name]?
What major did [name] study?
Where did Sabrina [name] work?

86.6% acc

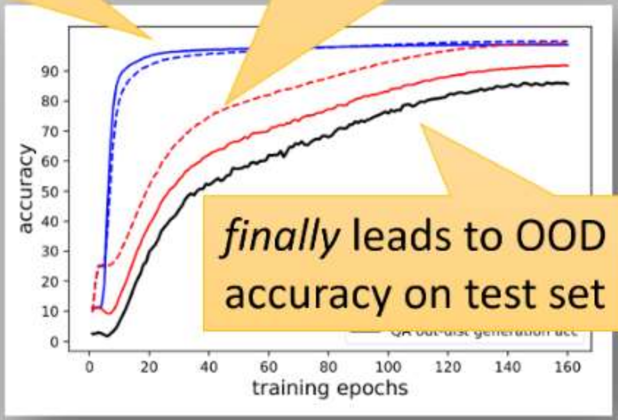
Mixed-Training $\Longrightarrow$ Knowledge Extraction

mixed-training stores knowledge properly
$\Longrightarrow$ they can be OOD extracted

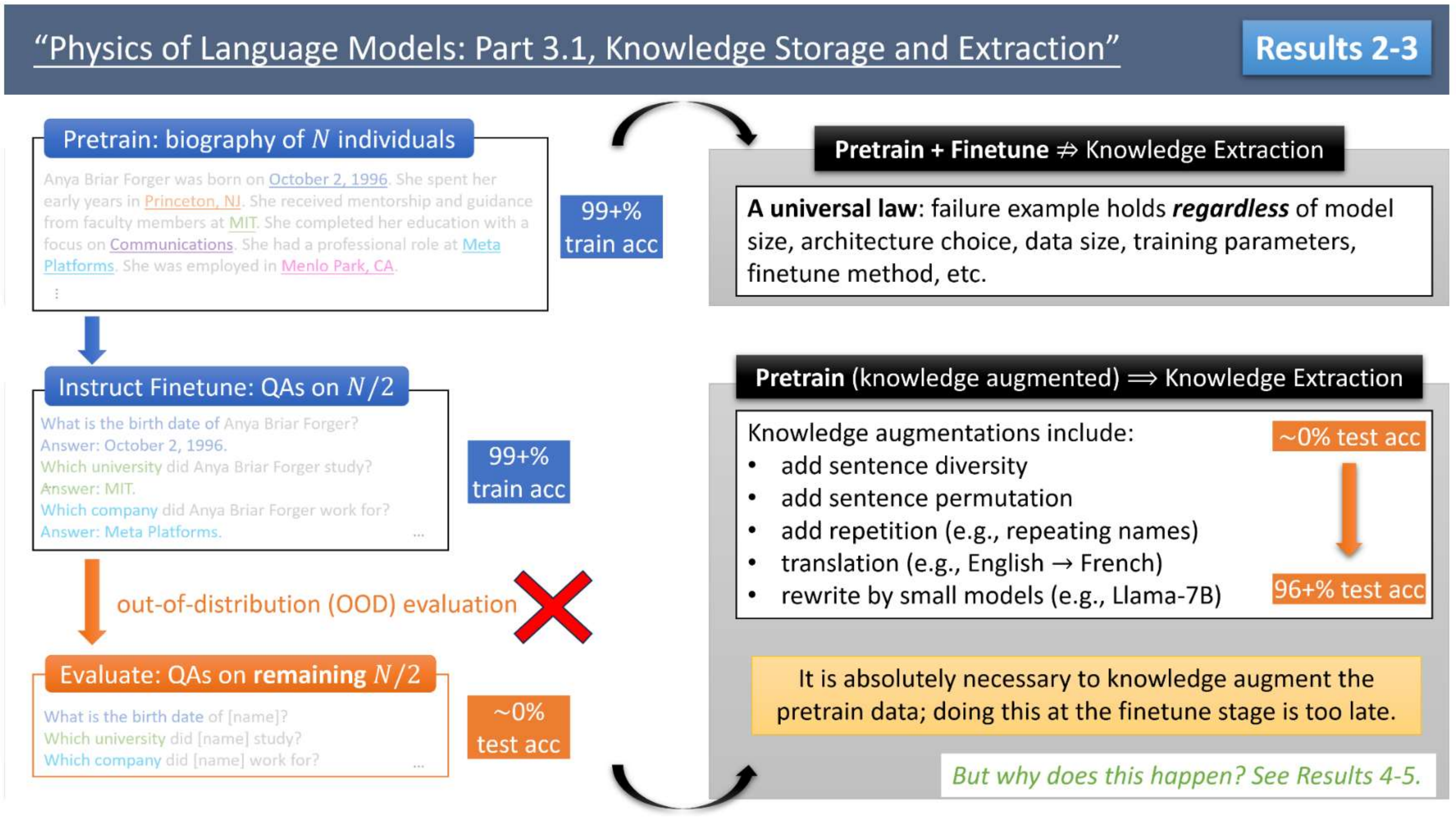
mixed-training = "to study to pass the test"
$\neq$ human knowledge acquisition

model *first* uses QAs to learn knowledge

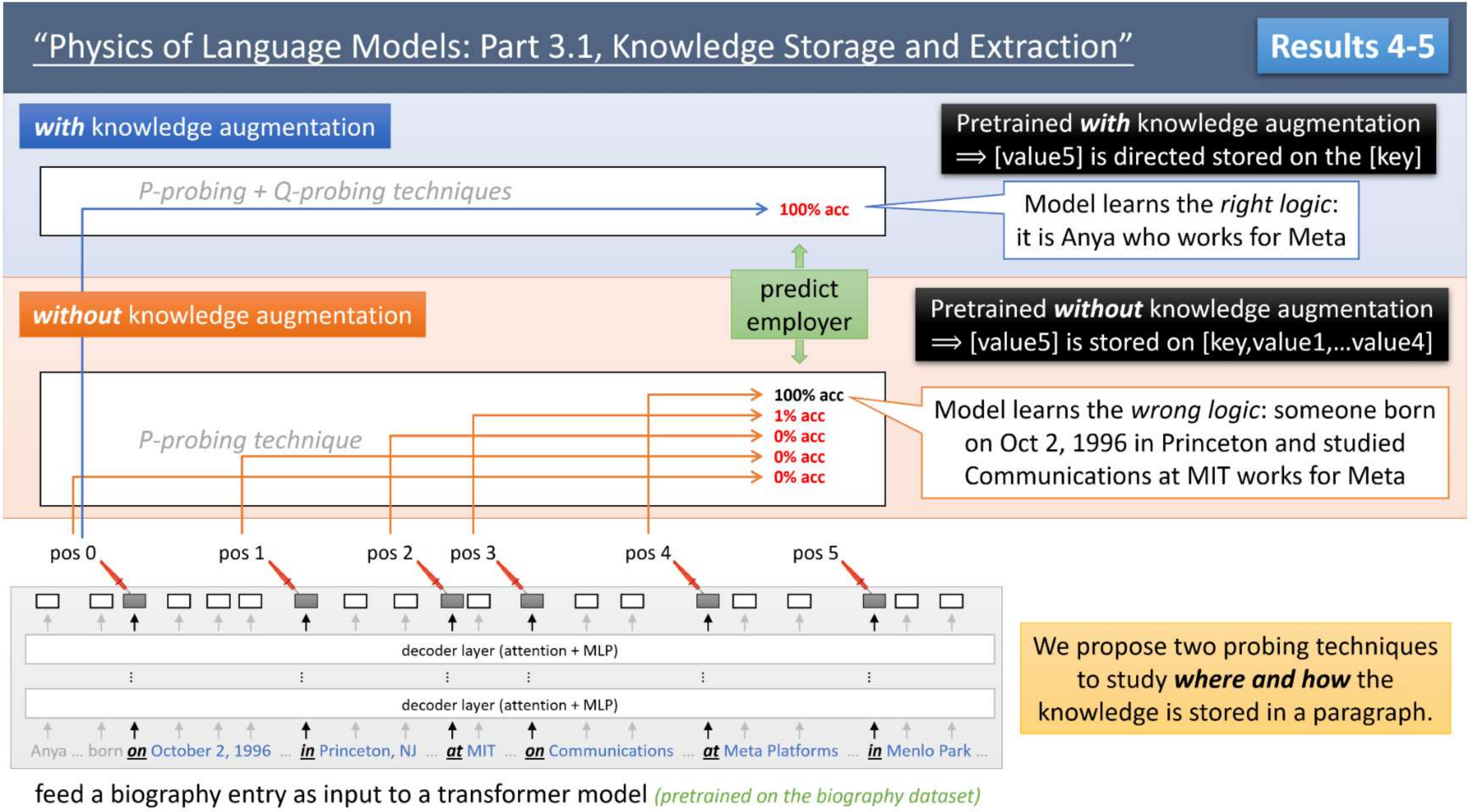
then aligns knowledge with the biography



finally leads to OOD accuracy on test set



"Physics of Language Models: Part 3.1, Knowledge Storage and Extraction"
Results 2-3
Pretrain: biography of N individuals
Anya Briar Forger was born on October 2, 1996. She spent her early years in Princeton, NJ. She received mentorship and guidance from faculty members at MIT. She completed her education with a focus on Communications. She had a professional role at Meta Platforms. She was employed in Menlo Park, CA.
99+% train acc
Instruct Finetune: QAs on N/2
What is the birth date of Anya Briar Forger?
Answer: October 2, 1996.
Which university did Anya Briar Forger study?
Answer: MIT.
Which company did Anya Briar Forger work for?
Answer: Meta Platforms.
99+% train acc
out-of-distribution (OOD) evaluation
Evaluate: QAs on remaining N/2
What is the birth date of [name]?
Which university did [name] study?
Which company did [name] work for?
~0% test acc
Pretrain + Finetune ⇏ Knowledge Extraction
A universal law: failure example holds regardless of model size, architecture choice, data size, training parameters, finetune method, etc.
Pretrain (knowledge augmented) ⟹ Knowledge Extraction
Knowledge augmentations include:
• add sentence diversity
• add sentence permutation
• add repetition (e.g., repeating names)
• translation (e.g., English → French)
• rewrite by small models (e.g., Llama-7B)
~0% test acc
96+% test acc
It is absolutely necessary to knowledge augment the pretrain data; doing this at the finetune stage is too late.
But why does this happen? See Results 4-5.



"Physics of Language Models: Part 3.1, Knowledge Storage and Extraction"
Results 4-5
with knowledge augmentation
Pretrained *with* knowledge augmentation ⟹ [value5] is directed stored on the [key]
P-probing + Q-probing techniques
100% acc
Model learns the *right logic*: it is Anya who works for Meta
predict employer
without knowledge augmentation
Pretrained *without* knowledge augmentation ⟹ [value5] is stored on [key,value1,...value4]
P-probing technique
100% acc
1% acc
0% acc
0% acc
0% acc
Model learns the *wrong logic*: someone born on Oct 2, 1996 in Princeton and studied Communications at MIT works for Meta
pos 0
pos 1
pos 2
pos 3
pos 4
pos 5
decoder layer (attention + MLP)
decoder layer (attention + MLP)
Anya ... born on October 2, 1996 ... in Princeton, NJ ... at MIT ... on Communications ... at Meta Platforms ... in Menlo Park ...
feed a biography entry as input to a transformer model (pretrained on the biography dataset)
We propose two probing techniques to study *where and how* the knowledge is stored in a paragraph.

Results 6

Pretrain: biography of *N minorities*

Anya Briar Forger was born on ...

without knowledge augmentation (e.g., one entry per person)

\+

Pretrain: biography of *M celebrities*

Joseph Robinette Biden is an American politician ...

with knowledge augmentation (e.g., 5 entries per person + sentence diversity and permutations)

Finetune: QAs on *M/2 celebrities*

What is the birth date of Joseph Robinette Biden?
Answer: November 20, 1942. ...

Evaluate: QAs on *N* minorities

What is the birth date of Anya Briar Forger? ...

~80% test acc

Knowledge augmentation on *celebrity* ⟹ Knowledge extraction for *minority*

Even if knowledge augmentation is applied to a subset of individuals, what we call celebrities, test accuracy for others also increases.

The mere inclusion of celebrity data (e.g., people with plentiful online biographical data of diverse writing styles) in pre-training enhances the model's knowledge extraction for minorities.

Physics of Language Models: Part 3.2
Knowledge Manipulation

Results 1-2

<u>knowledge partial / dual retrievals may be difficult</u>

model can say birthday “October 2, 1996” but not “1996”

Results 3-5

<u>knowledge classification / comparison is hard without CoT</u>

model cannot say “even year” without saying “1996” (CoT)

Result 7

<u>knowledge inverse search is impossible</u>

impossible to answer “who was born on October 2, 1996”

Results 6/8/9

<u>Turing tests to distinguish Humans from contemporary AIs</u>

universal counter-examples, applicable to GPT-4 and others

"Physics of Language Models: Part 3.2, Knowledge Manipulation"

Results 1-2

Instruct Finetune: QAs on $N/2$

What is the **birth date** of Anya Briar Forger?
Answer: October 2, 1996

What is the **birth year** for Anya Briar Forger?
Answer: 1996

partial knowledge retrieval

Evaluate on remaining on $N/2$

What is the **birth date** for [name]?
Answer: 100% acc

What is the **birth year** for [name]?
Answer: 20% acc

Pretrain: biography of N individuals

Anya Briar Forger was born on October 2, 1996. She spent her early years in Princeton, NJ. She received mentorship and guidance from faculty members at MIT. She completed her education with a focus on Communications. She had a professional role at Meta Platforms. She was employed in Menlo Park, CA.

⋮

Preliminary evidence for the necessity of CoT in knowledge manipulation

model must:
explicitly state birth month/day ***before*** birth year
explicitly state company name ***before*** work city

Instruct Finetune: QAs on $N/2$

Where and **which** company did Anya Briar Forger work for?
Answer: Menlo Park + Meta Platforms

Which company and **where** did Anya Briar Forger work for?
Answer: Meta Platforms + Menlo Park

dual knowledge retrieval

Evaluate on remaining on $N/2$

Where and **which** company did [name] work for?
Answer: 5% acc

Which company and **where** did [name] work for?
Answer: 98% acc

“Physics of Language Models: Part 3.2, Knowledge Manipulation”

Results 3-5

Finetune: knowledge *classification* on $N/2$

Was Anya Briar Forger born in an even month?
Answer without CoT: **Yes**

Was Anya Briar Forger born in an even month?
Answer with CoT: **October**; so it is **Yes**

Finetune or **Pretrain**: knowledge extraction on all N

What is the birth date of Anya Briar Forger?
Answer: October 2, 1996...

Pretrain: biography of N individuals

Anya Briar Forger was born on October 2, 1996. She spent her early years in Princeton, NJ. She received mentorship and guidance from faculty members at MIT. She completed her education with a focus on Communications. She had a professional role at Meta Platforms. She was employed in Menlo Park, CA.
⋮

Evaluate: on remaining $N/2$

Was [name] born in an even month?
Answer without CoT: 50% acc *(=random guess)*

Was [name] born in an even month?
Answer with CoT: 100% acc

Knowledge manipulation is “impossible” without CoT

universal law: holds even for a single-step of the most basic manipulation task (e.g., classification, ranking, comparison)

different from CoT in reasoning: e.g. GPT-4 can answer “is the sum of [a] and [b] even” without writing down [a+b].

training CoT does not help inference without CoT

to get high accuracy without CoT, need extremely large finetune data:

e.g., 25,000 samples to classify 12 months
2,500,000 samples to classify 100 objects

"Physics of Language Models: Part 3.2, Knowledge Manipulation"

Result 7

Pretrain: biography of N individuals

Anya Briar Forger was born on October 2, 1996. She spent her early years in Princeton, NJ. She received mentorship and guidance from faculty members at MIT. She completed her education with a focus on Communications. She had a professional role at Meta Platforms. She was employed in Menlo Park, CA.

⋮

+

Finetune or **Pretrain**: knowledge extraction on all N

What is the **birth date** of Anya Briar Forger?
Answer: October 2, 1996...

Finetune: knowledge *inverse search* on $N/2$

Who was born on October 2, 1996 in Princeton, NJ, studied Communications at MIT, and worked for Meta Platforms at Menlo Park, CA?
Answer: Anya Briar Forger

Evaluate: on remaining $N/2$

Who was born on [date] in [city] studied [major] at [university], and worked for [company] at [city]?
Answer: 0% acc

Knowledge inverse search is impossible, period

universal law: knowledge inverse search is impossible, unless knowledge is already reversed in the pretrain data

(reverse finetune data is useless)

this is regardless of model size, data size, training method, finetune method, or any other parameters

changing decoder to encoder (e.g. BERT) cannot help (see Part 3.1, Result 7)

"Physics of Language Models: Part 3.2, Knowledge Manipulation"

Results 6,8

GPT-4 or Llama-3 also fail on knowledge mainpulations

classification

You
Was Joe Biden born in an odd year?

ChatGPT
Yes, Joe Biden was born in 1942, which is an even year.

You
Do you know what odd means?

ChatGPT
Yes, an odd number is a whole number that is not divisible by 2, meaning it doesn't end in 0, 2, 4, 6, or 8. I made a mistake in my previous response. Joe Biden was born in 1942, which is an even year, not an odd year.

unless CoT is used

Yes or no, was Joe Biden born in an odd year?

Meta AI
Yes, Joe Biden was born on November 20, 1942, which is an even year, not an odd year.

ranking

You
Was Donald Trump born earlier than Nancy Pelosi?

ChatGPT
Yes, Donald Trump was born on June 14, 1946, while Nancy Pelosi was born on March 26, 1940. So, Nancy Pelosi was actually born ... lier than Donald Trump.

You
Tell me why?

ChatGPT
It looks like I misspoke in my previous response—my apologies for the confusion! Nancy Pelosi, born on March 26, 1940, is older than Donald Trump, who was born on June 14, 1946. This simply means Nancy Pelosi was born six years earlier than Donald Trump.

unless CoT is used

Meta AI
Yes, Donald Trump was born earlier than Nancy Pelosi.
Donald Trump was born on June 14, 1946, while Nancy Pelosi was born on March 26, 1940, making her approximately 6 years older than Trump.

inverse search

You
成语"X往开来"的X是什么字?

ChatGPT
成语"X往开来"的X是"辟"。所以，成语是"辟往开来"，意思是指开辟前去的道路，比喻开创事业或发展前途。这个成语常用来形容克服困难，开辟新的局面或前程。

total failure

Meta AI
The answer is 云 (yún). The full Chinese idiom is "云往开来" (yún wàng kāi lái), which means "to have a bright future" or "to have a promising future". The character 云 (yún) here means

We do not believe this is easily fixable, unless pretrained data is differently prepared

our paper appeared in September 2023, as of June 2024, the "bugs" still exist.

This is a ***Turing test*** to distinguish humans from (perhaps all) contemporary AI's

many more GPT-4 failure examples

	birth month classify % 2	birth date ranking	Jane Austen novel inverse search	Wikipedia inverse search	Chinese Idiom inverse search	Chinese Poem inverse search
GPT-4 accuracies	50.7% *≈ flip a coin*	52.3% *≈ flip a coin*	0.8% *forward = 65.9%*	42% *forward = 99%*	17.6% *forward = 90.6%*	7.3% *forward = 66.5%*

"Physics of Language Models: Part 3.2, Knowledge Manipulation"

Result 9

How to improve inverse search then?

1. Use retrieval-augmented generation (RAG) = Google search plugin
2. Insert reverse knowledge into pretrain data (e.g., through a "rewrite" prompt)
3. Insert ***line numbers*** to enable reversal search on critical documents

See separate paper "Reverse Training to Nurse the Reversal Curse"

How GPT-4 inverse searches the Bible

Genesis 9:5 King James Version (KJV)

And surely your blood of your lives will I require; at the hand of every beast will I require it, and at the hand of man; at the hand of every man's brother will I require the life of man.

animal is put to death when it kills a human. "*Surely for your lifeblood I will demand a reckoning; from the hand of every beast I will require it, and from the hand of man. From the hand of every man's brother I will require the life of man*" (Genesis 9:5). Instinct teaches animals to fear men. "*And the fear of you and the dread of you shall be on every beast of*

pretrain data

the Bible is 'sufficiently augmented' on the internet data, chapter/verse numbers appear ***both*** before and after

What's the verse in the Bible before "And surely your blood of your lives will I require; at the hand of every beast will I require it, and at the hand of man; at the hand of every man's brother will I require the life of man."

The verse you've provided is Genesis 9:5 (in the King James Version). Therefore, the verse right before it, Genesis 9:4, is:

"But flesh with the life thereof, which is the blood thereof, shall ye not eat."

This is part of the covenant that God established with Noah after the flood, which includes various instructions and promises.

forward knowledge search

chain of thought

Physics of Language Models: Part 3.3
Knowledge Capacity Scaling Laws

Results 1/2/3/5

a *universal* law: "all" LLMs can store 2bit/param knowledge

$\Rightarrow$ predict: 7B model can store all English wiki + textbooks knowledge

Results 4/6/7

scaling laws for insufficient training

e.g. LLaMA/Mistral architectures *30% worse* than $\text{GPT2}_{\text{rotary}}$ in capacity

Results 8/9

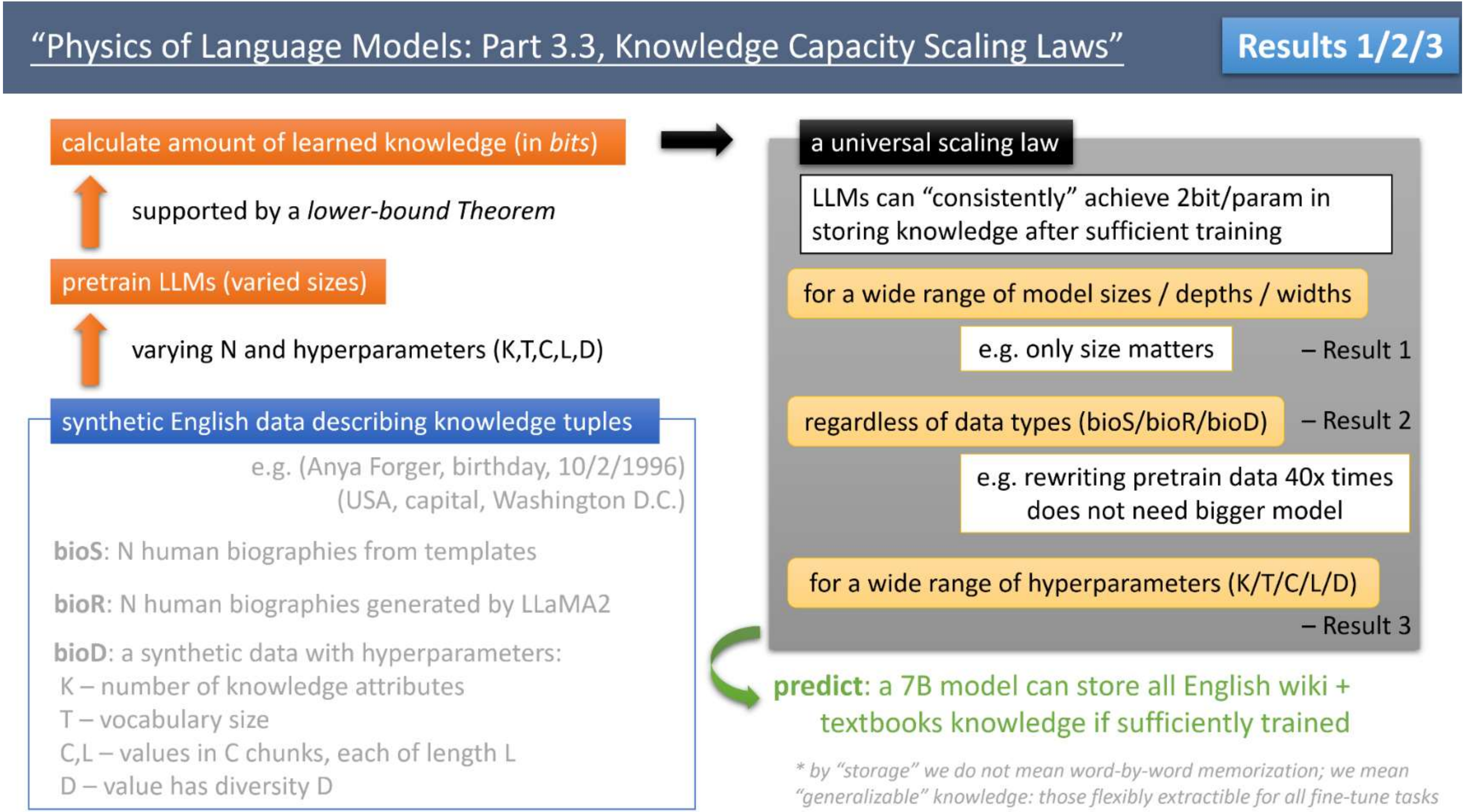
scaling laws for quantization + mixture-of-expert (MoE)

e.g. 2bit/param holds *even for* int8 parameters

Results 10/11/12

scaling laws for mixed-quality data (wikipedia vs internet)

e.g. a technique to improve LLM's capacity – sometimes by 10x



"Physics of Language Models: Part 3.3, Knowledge Capacity Scaling Laws"
Results 1/2/3
calculate amount of learned knowledge (in *bits*)
supported by a *lower-bound Theorem*
pretrain LLMs (varied sizes)
varying N and hyperparameters (K,T,C,L,D)
synthetic English data describing knowledge tuples
e.g. (Anya Forger, birthday, 10/2/1996)
(USA, capital, Washington D.C.)
bioS: N human biographies from templates
bioR: N human biographies generated by LLaMA2
bioD: a synthetic data with hyperparameters:
K – number of knowledge attributes
T – vocabulary size
C,L – values in C chunks, each of length L
D – value has diversity D
a universal scaling law
LLMs can "consistently" achieve 2bit/param in storing knowledge after sufficient training
for a wide range of model sizes / depths / widths
e.g. only size matters
– Result 1
regardless of data types (bioS/bioR/bioD)
– Result 2
e.g. rewriting pretrain data 40x times does not need bigger model
for a wide range of hyperparameters (K/T/C/L/D)
– Result 3
predict: a 7B model can store all English wiki + textbooks knowledge if sufficiently trained
* by "storage" we do not mean word-by-word memorization; we mean "generalizable" knowledge: those flexibly extractible for all fine-tune tasks

"Physics of Language Models: Part 3.3, Knowledge Capacity Scaling Laws"

Results 4/5

scaling law (sufficient training)

"all" LLMs consistently achieve *2bit/param* in storing knowledge that are seen for **1000 exposures**

– Result 5

scaling law (insufficient training)

GPT-2* consistently achieves *1bit/param* in storing knowledge that are seen for **100 exposures**

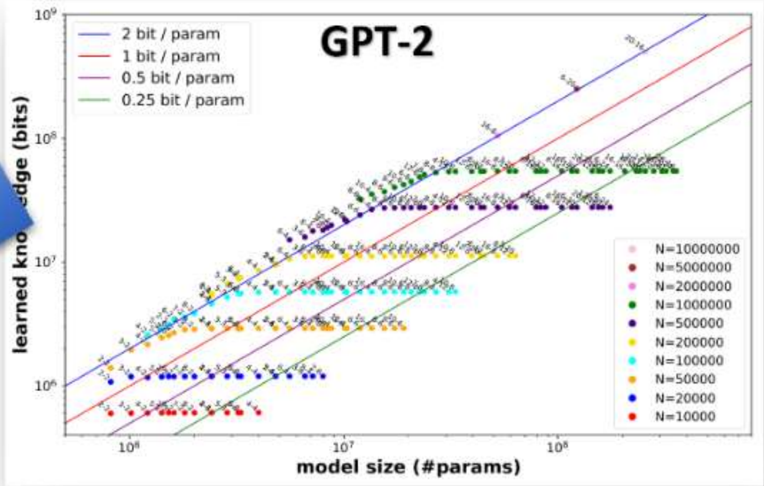
* *adding rotary embedding*

– Result 4

1000 exposures ≠ 1000 passes

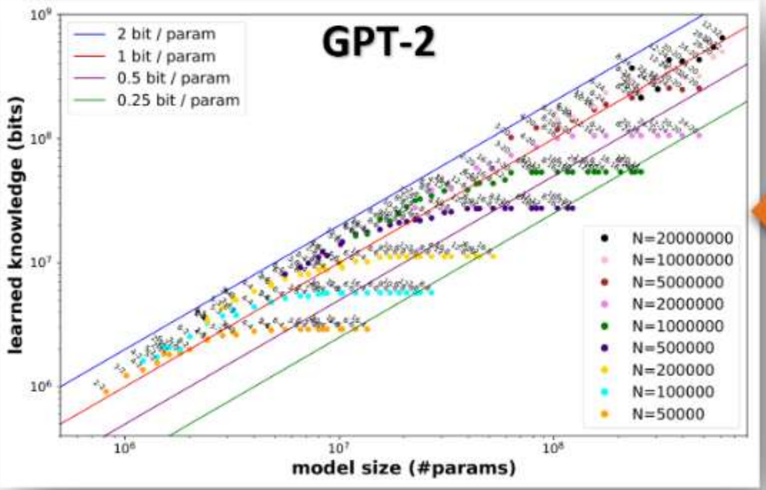
– e.g. (US, capital, Washington DC) has been exposed 1,000,000+ times in 1-pass of the internet pretrain data

no change



1000 ⇒ 100 exposures
(insufficient training)

ratio 2 ⇒ 1 bit/param



go worse

if you use **LLaMA** or **Mistral**
even if you completely **remove MLP layers!**

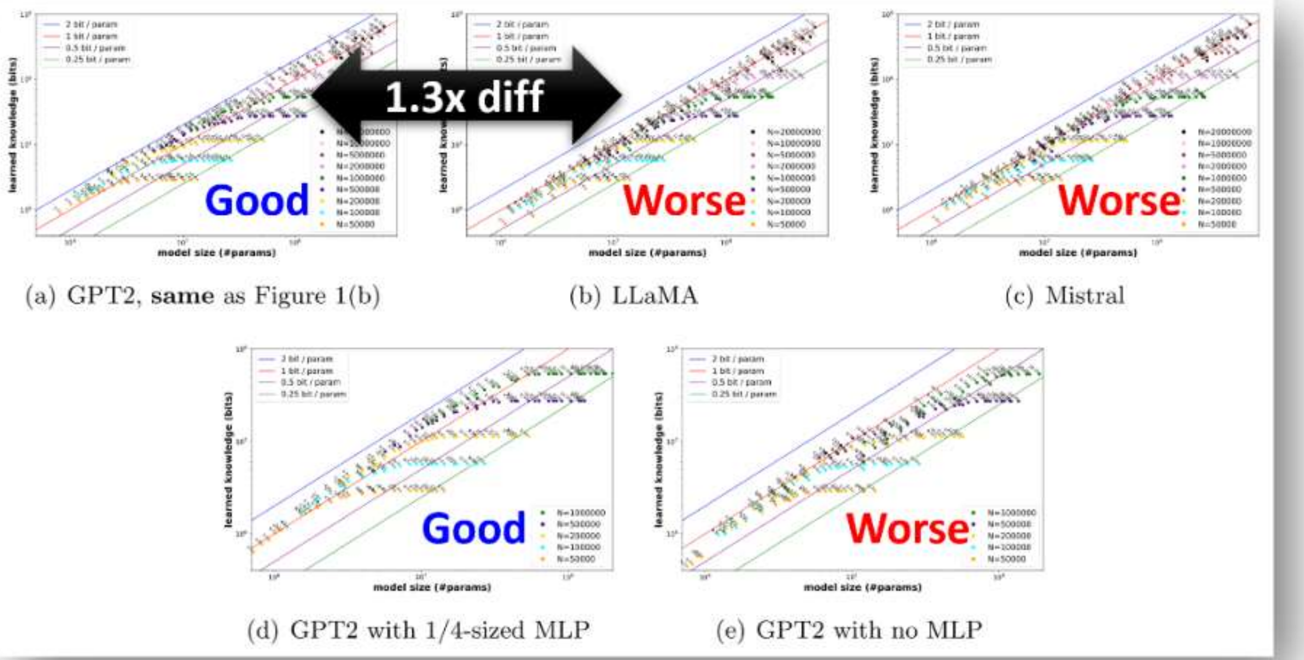
Corollary: Attention layers can store knowledge

if you use **LLaMA/Mistral** architectures,
see Results 6+7

scaling law (insufficient training)

In the 100-exposure setting, some architectures are worse in knowledge capacity: e.g., LLaMA/Mistral architectures can be **1.3x worse** than $\text{GPT2}_{\text{rotary}}$ – Result 6

Controlled experiments reveal that **GatedMLP** contributes to this **performance loss**; it is less stable, needs longer training time – Result 7



(a) GPT2, **same** as Figure 1(b)

(b) LLaMA

(c) Mistral

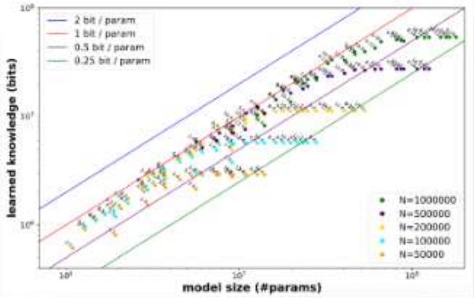
(d) GPT2 with 1/4-sized MLP

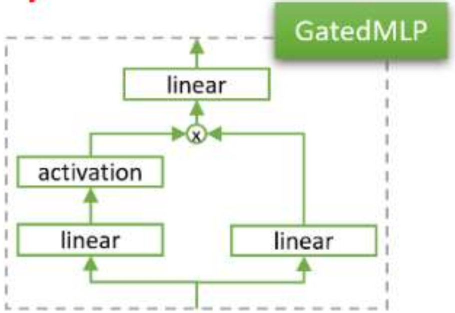
(e) GPT2 with no MLP

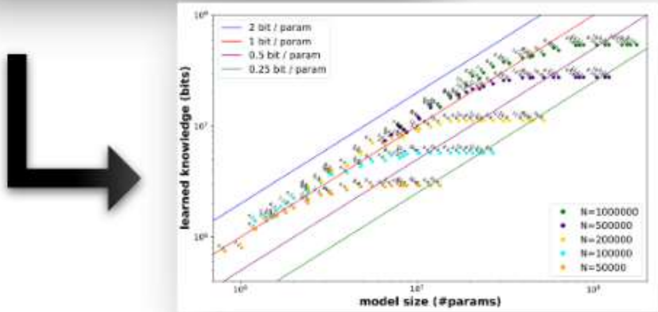
Disclaimer 1: this comparison is for ***knowledge capacity*** *only*

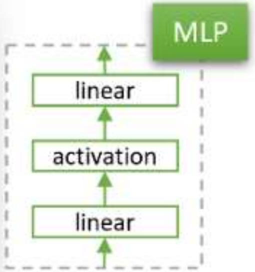
Disclaimer 2: there will be ***no difference*** *if sufficiently trained, see Result 5*

LLaMA/Mistral (using GatedMLP) = Worse









LLaMA (replaced with standard MLP) = Good

"Physics of Language Models: Part 3.3, Knowledge Capacity Scaling Laws"

Results 10/11/12

scaling laws (pretrain data of mixed qualities)

"Junk" data significantly ***harm*** LLM's knowledge capacity on **good data** (sometimes by 20x times!) – Result 10

e.g. common crawls, internet "junks" e.g. Wikipedia

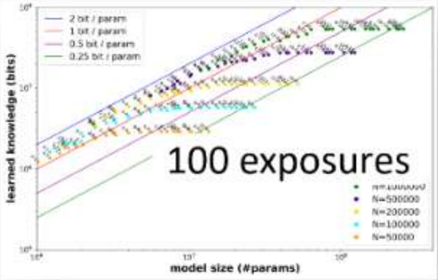
repetitive knowledge ... *does not* harm ... – Result 11

illustration

1/8 good data

7/8 training tokens from "junk" data

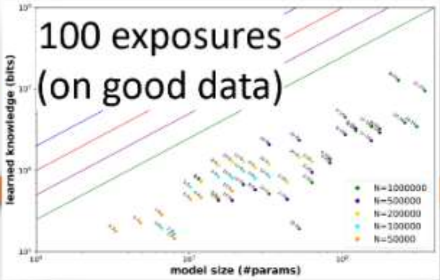
train **without** junk



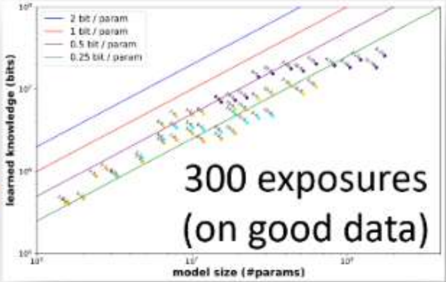
20x worse

3x worse

train **with** junk



train **with** junk



10x times better!

3x times better!

a simple fix!

add domain tokens (e.g., "wikipedia.org") at front of all pretrain data paragraphs | data | data | data

LLMs can ***automatically*** detect domains rich in high-quality knowledge and prioritize learning from them – Result 12

[Submitted on 3 Jan 2025 (v1), last revised 22 Feb 2025 (this version, v2)]

Metadata Conditioning Accelerates Language Model Pre-training

Tianyu Gao Alexander Wettig Luxi He Yihe Dong Sadhika Malladi Danqi Chen
Princeton Language and Intelligence, Princeton University
tianyug@princeton.edu

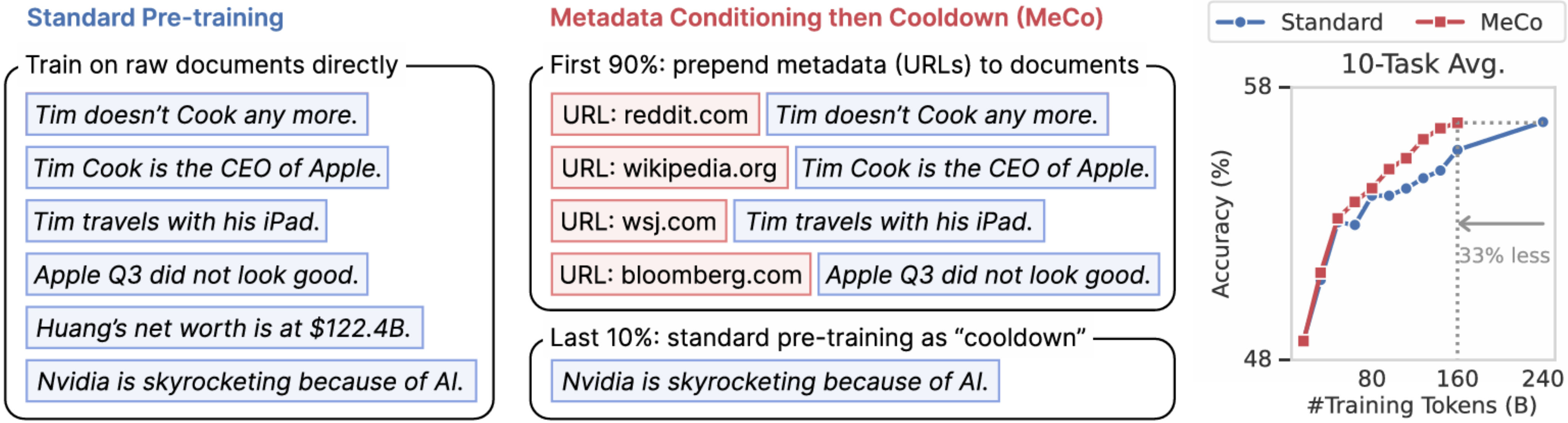


Figure 1: A comparison between data used by standard pre-training and MeCo. The figure on the right demonstrates 5-shot downstream task performance averaged across 10 tasks (1.6B models; details about the experiments can be found in §3).