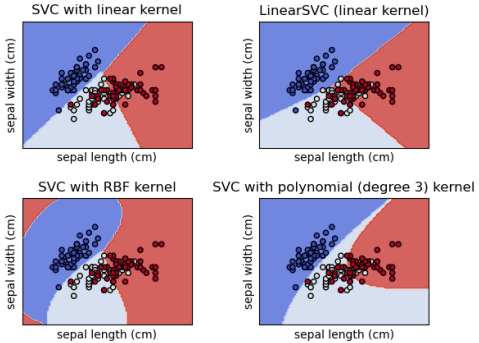
Back to the future:
Data efficient language modeling

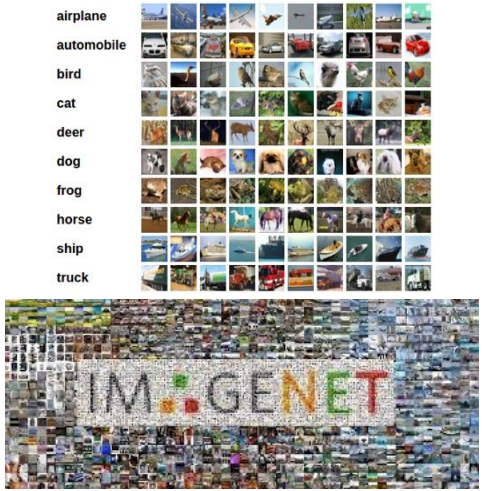
Tatsunori Hashimoto
Stanford CS

Roadblocks to progress in machine learning

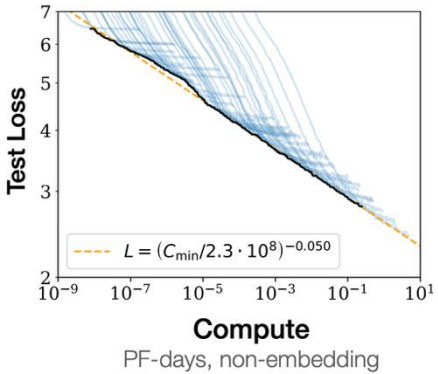
Algorithms
(better function classes)



Data
(lack of supervised data)



Compute
(inability to process all the data)



What is the major bottleneck to continued improvements in ML (and LLMs)?

Data lies at the heart of recent advances

Pretraining data


Meta

The Llama 3 Herd of Models

Llama Team, AI @ Meta[1]

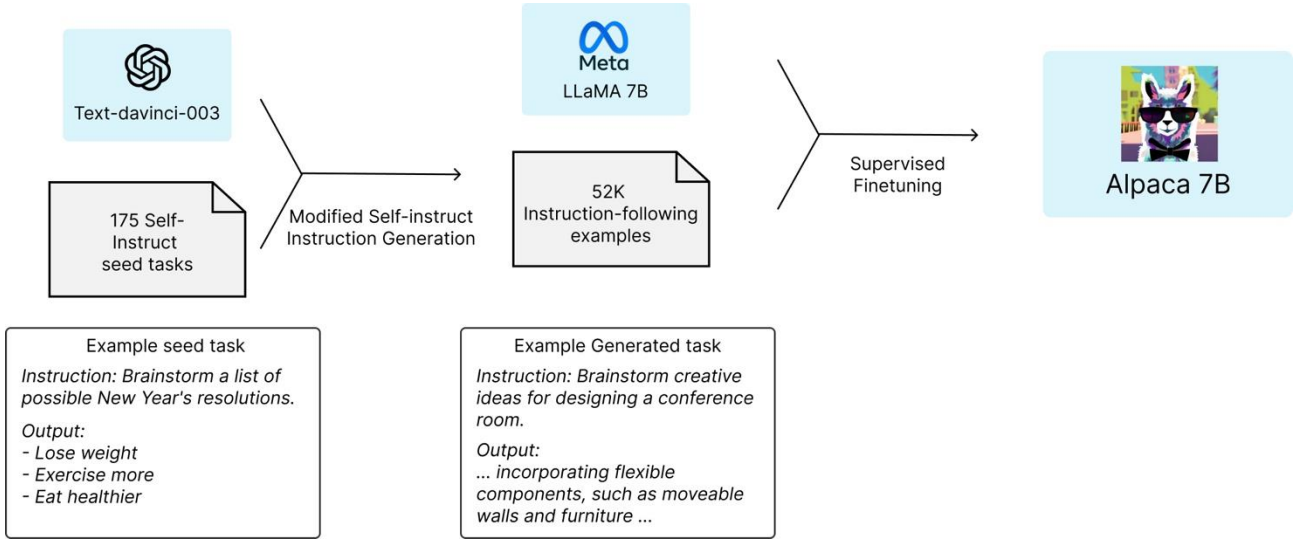
We believe there are three key levers in the development of high-quality foundation models: data, scale, and managing complexity. We seek to optimize for these three levers in our development process:

- **Data.** Compared to prior versions of Llama (Touvron et al., 2023a,b), we improved both the quantity and quality of the data we use for pre-training and post-training. These improvements include the development of more careful pre-processing and curation pipelines for pre-training data and the development of more rigorous quality assurance and filtering approaches for post-training data. We pre-train Llama 3 on a corpus of about 15T multilingual tokens, compared to 1.8T tokens for Llama 2.
- **Scale.** We train a model at far larger scale than previous Llama models: our flagship language model was pre-trained using 3.8×10^{25} FLOPs, almost $50\times$ more than the largest version of Llama 2. Specifically, we pre-trained a flagship model with 405B trainable parameters on 15.6T text tokens. As expected per

Known pretraining recipes all emphasize the important of the data mix

Data lies at the heart of recent advances

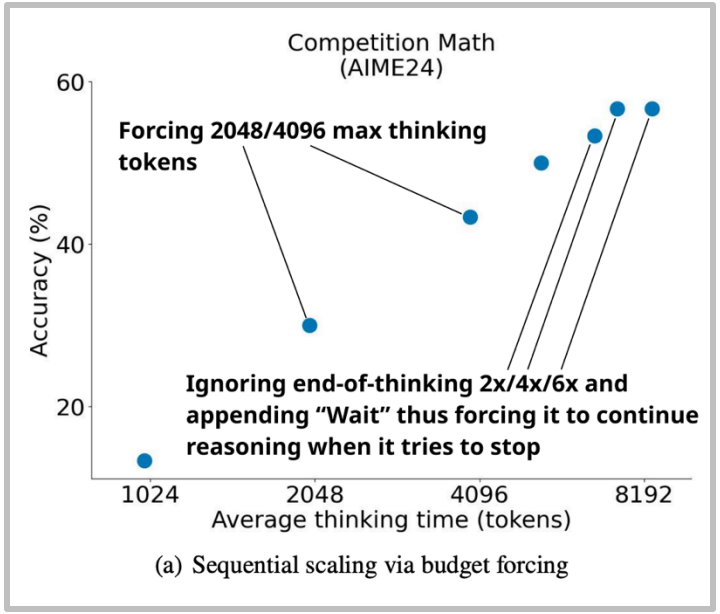
Instruction tuning / post-training



Simple algorithms with the 'right' data (for instruction-tuning / RLHF) gets you quite far for instruction tuning

Data lies at the heart of recent advances

Reasoning capabilities



(a) Sequential scaling via budget forcing

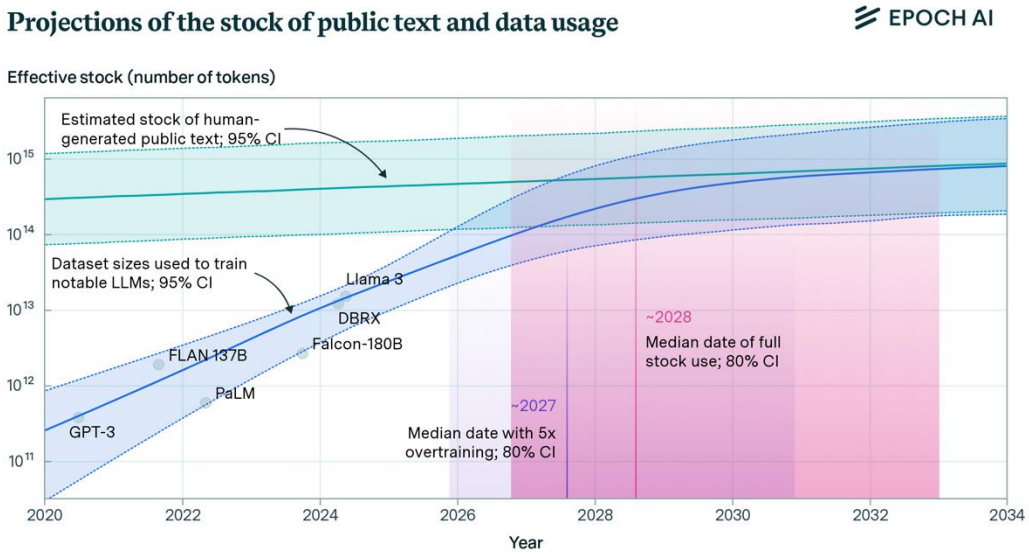
How many r in raspberry? **Question**

Let's break down the process of counting the letter 'r' in the word "raspberry" ...
* First letter: 'r' - This is an 'r', count = 1.
* Second letter: 'a' - Not an 'r', count remains 1 ...
* Sixth letter: 'e' - Not an 'r', count remains 1.
* Seventh letter: 'r' - This is an 'r', count = 2.
* Eighth letter: 'y' - Not an 'r', count remains 2 ...
The number of 'r's in "raspberry" is 2.
Wait, let's re-read the question carefully. It asks "How many r in raspberry?" ... * r - a - s - p - b - e - r - r - y ... * First 'r' ... * Second 'r' ... * Third 'r' ... Count = 3 ... **Reasoning trace**

My initial answer of 2 was incorrect due to a quick reading of the word. **Final Answer:** The final answer is $\boxed{3}$ **Response**

Even something as complex as 'long CoT reasoning' can (to a suprising extent) be unlocked with naïve methods + the right data

Data scarcity problems on the horizon



Thus far, data constraints have not been acute since internet data is vast

But compute has been growing much faster than data

Statistical learning has much to say in this regime

Data >> Compute

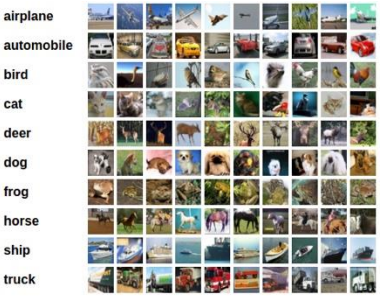
(Pretraining / FM era)

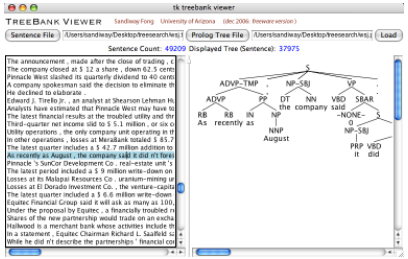
ChatGPT: Optimizing Language Models for Dialogue

GPT-4

Compute >> Data

(Classical deep learning
.. and the future?)





One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants
Google
1600 Amphitheatre Parkway
Mountain View, CA 94043, USA

Phillipp Koehn
University of Edinburgh
10 Crichton Street, Room 4.19
Edinburgh, EH8 9AB, UK

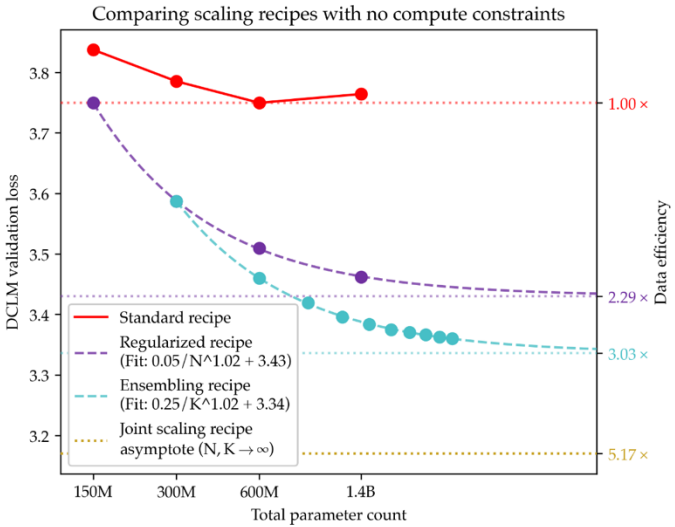
Tony Robinson
Cantab Research Ltd
St Johns Innovation Centre
Cowley Road, Cambridge, CB4 0WS, UK

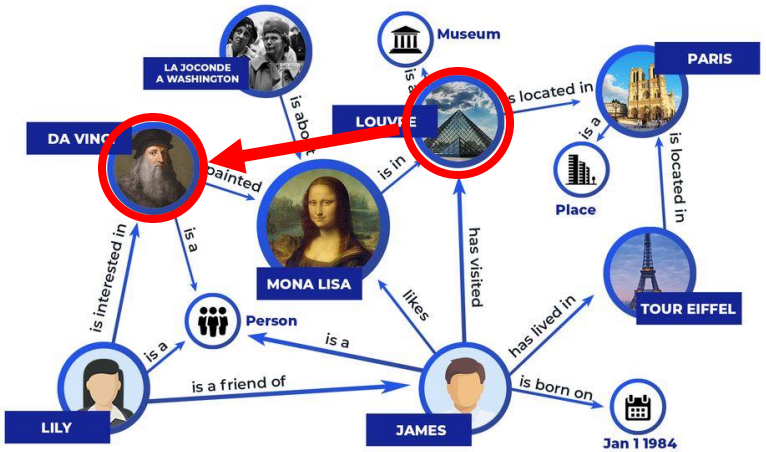
The problems of the future are those of the past

In a compute rich world,
understanding and engineering data becomes critical

Understanding generalization from data is a foundational question
in statistics and machine learning!

What is there to do?





Understanding generalization

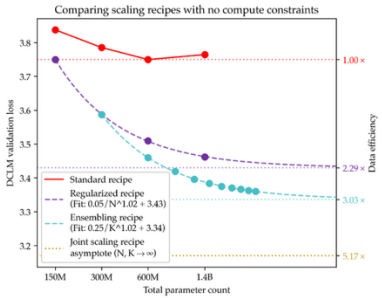
Rich phenomena in algorithmic generalization

Synthetic data

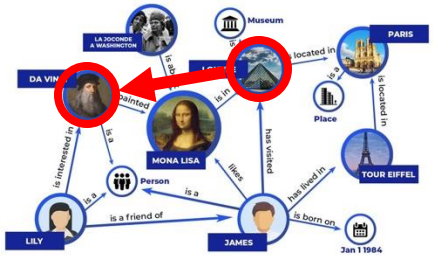
Firm conceptual (and theoretical) foundation

Part 1: Algorithmic interventions for data efficiency

Are there simple data efficiency interventions that have been overlooked?



Part 1: Using data better



Part 2: Making ‘new’ data

Pre-training under infinite compute [ArXiv preprint]

Konwoo Kim*, Suhas Kotha*, Percy Liang, Tatsunori Hashimoto

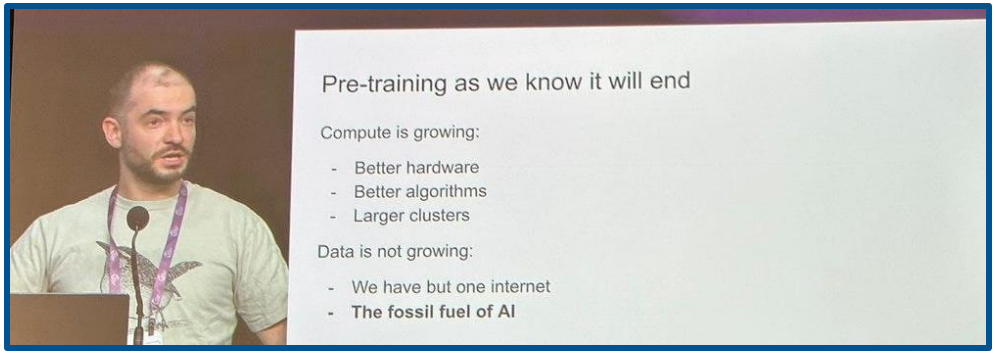
Pretraining – hitting a wall?

New, pretrained models – not quite as impressive

4 Preparedness Framework

GPT-4.5 is not a frontier model, but it is (~35x more expensive
tational efficiency by more than 10x. Whi
improved writing ability, and refined per:

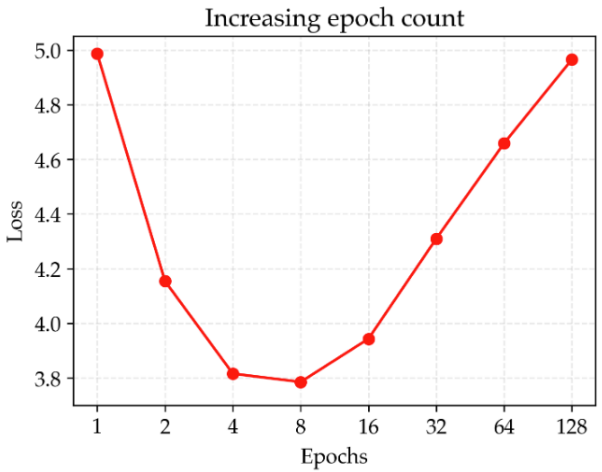
Discussions of the ‘end of pretraining’



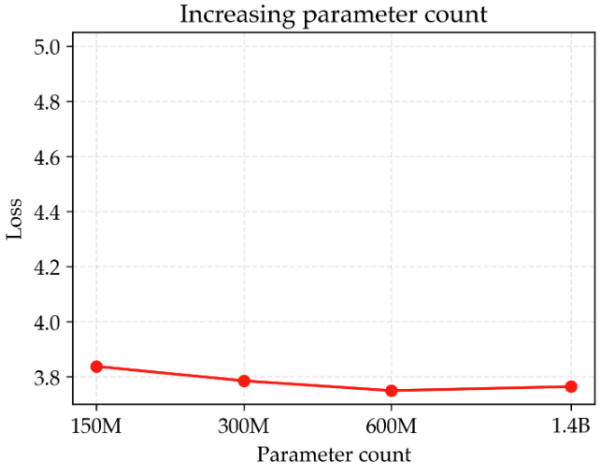
* Even if you think RL or continual learning are ‘ways out’ you *still* need data efficiency

Existing approaches are quite bad for data efficiency

What if we have infinite compute, and we just scale up epochs [e.g. Muennighoff et al] and param size



Tuned H	1	8	128
Learning rate	1e-3	1e-3	3e-3

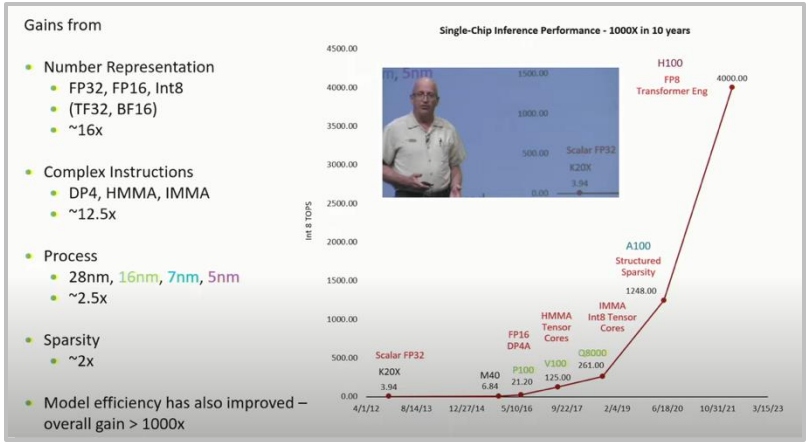


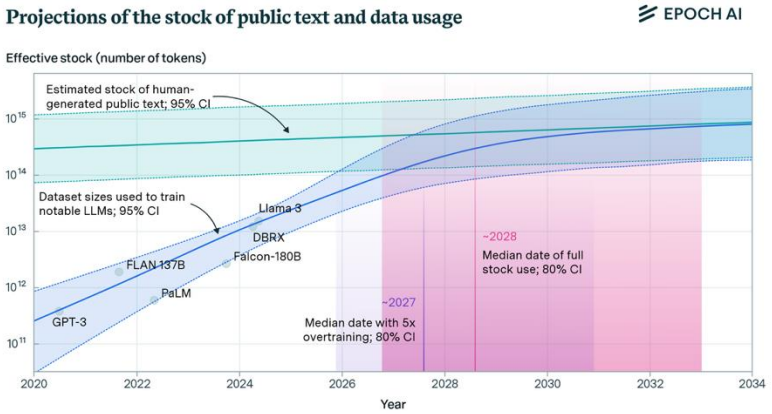
Tuned H	150M	300M	600M	1.4B
Learning rate	3e-3	1e-3	1e-3	3e-4
Epoch count	8	8	4	4

We get *no benefits* past a point (600M params w/ 200M dataset) even with infinite compute

Note: overfitting is *not* benign in the LM pretraining case

Identifying simple, scalable data efficiency gains





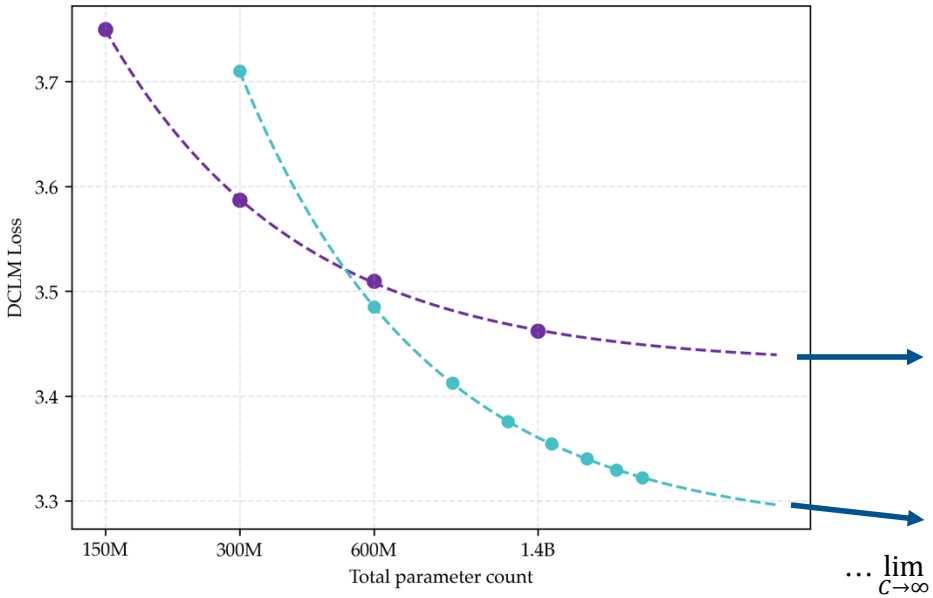
Q: if we had *infinite* compute, are there algorithms that would give us significantly more data efficiency?

Developing algorithms for a compute rich future

How should we study the nearly infinite compute regime..

(without paying for infinite compute)

1. Scale down the ‘world’
   (200M tokens, eval on perplexity)
2. Compare algorithms via
   compute scaling *asymptote*
3. Check validity by scaling up the
   ‘world size’ (200M -> 1.7B)



Scaling for performance prediction is not new

Learning Curves: Asymptotic Values and Rate of Convergence

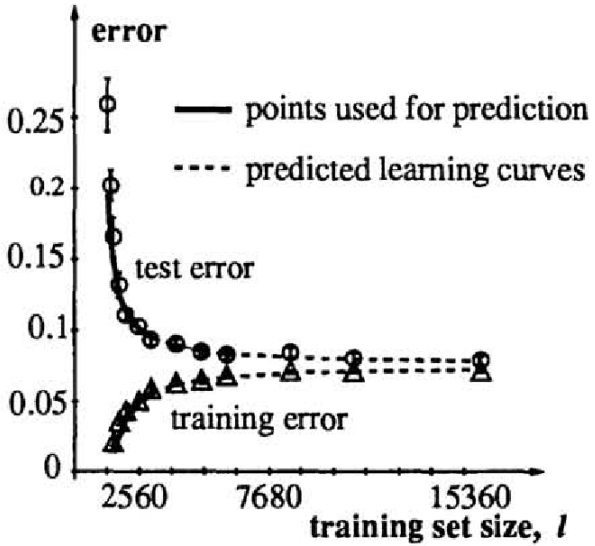
Corinna Cortes, L. D. Jackel, Sara A. Solla, Vladimir Vapnik, and John S. Denker
AT&T Bell Laboratories
Holmdel, NJ 07733

Abstract

Training classifiers on large databases is computationally demanding. It is desirable to develop efficient procedures for a reliable prediction of a classifier's suitability for implementing a given task, so that resources can be assigned to the most promising candidates or freed for exploring new classifier candidates. We propose such a practical and principled predictive method. Practical because it avoids the costly procedure of training poor classifiers on the whole training set, and principled because of its theoretical foundation. The effectiveness of the proposed procedure is demonstrated for both single- and multi-layer networks.

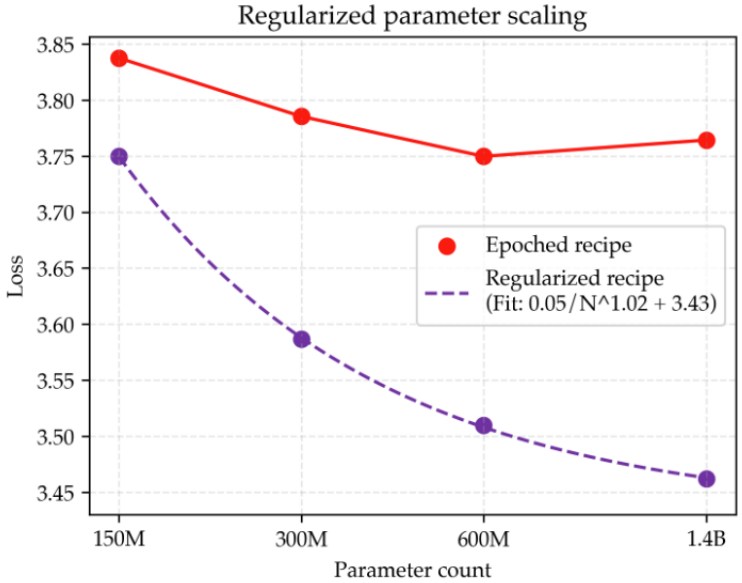
A typical example of learning curves is shown in Fig. 2. The test error is always larger than the training error, but asymptotically they reach a common value, a. We model the errors for large sizes of the training set as power-law decays to the

$$\mathcal{E}_{\text{test}} = a + \frac{b}{l^{\alpha}} \qquad \text{and} \qquad \mathcal{E}_{\text{train}} = a - \frac{c}{l^{\beta}}$$



Step 1: Fixing the ‘standard’ recipe for asymptotic losses

With optimal *(much higher)* regularization + model scaling, we get clean scaling



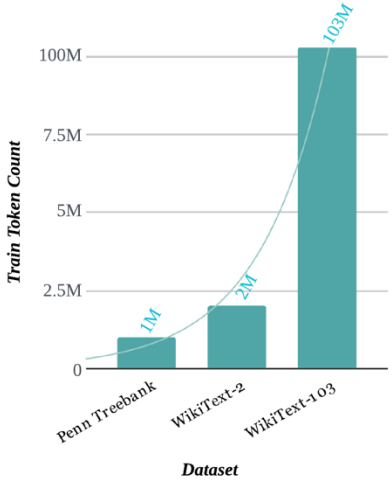
Tuned H	150M	300M	600M	1.4B
Learning rate	3e-3	3e-3	1e-3	1e-3
Epoch count	16	16	8	8
Weight decay	0.8	1.6	3.2	3.2

30x higher weight decay

This improves data efficiency and gives our first asymptotic loss baseline (~3.43)

Going beyond regularization

Regularization is maybe too simple
are there other ways of ‘trading compute for data efficiency’?



(a) **Penn TreeBank**

Model	Weight ↑	Validation ↓	Test ↓
QRNN (Bradbury et al., 2016)	0.00	60.38	58.43
KnerserNey-5gram (Ney et al., 1994)	0.01	148.41	141.46
Fast Weights (Schlag et al., 2021)	0.11	58.49	56.45
EGRU (Subramoney et al., 2023)	0.15	61.21	57.18
AWD-LSTM-MOS (Yang et al., 2017)	0.19	56.04	54.00
Transformer-XL (Dai et al., 2019)	0.20	57.93	55.41
AWD-LSTM-DOC (Takase et al., 2018)	0.34	54.12	52.38
Ensemble of All	1	**48.92**	**47.31**

(b) **WikiText-2**

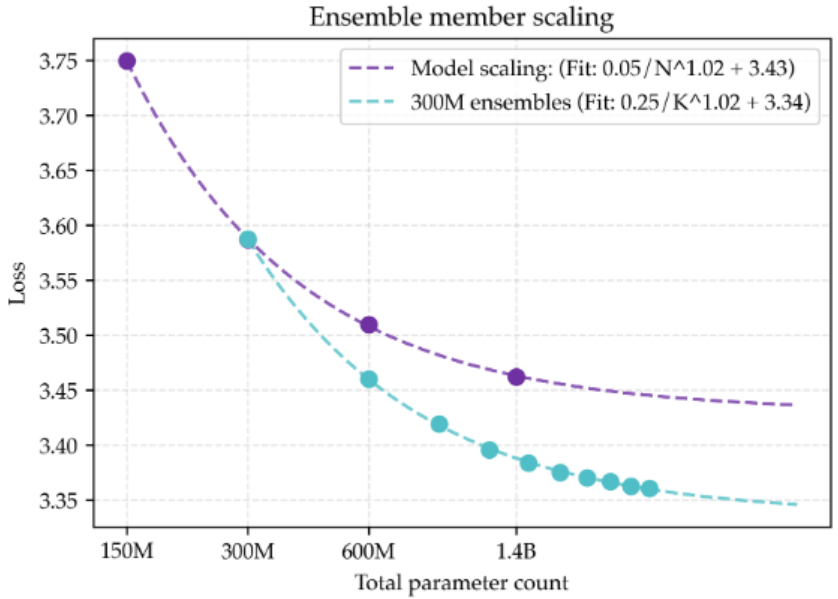
Weight ↑	Validation ↓	Test ↓
0.00	69.23	66.61
0.01	233.93	219.27
0.03	69.51	66.49
0.22	69.40	67.20
0.17	63.92	61.54
0.16	67.41	64.85
0.41	60.27	58.01
1	**55.40**	**53.73**

We have a wealth of theory and intuition for this regime

Ensembling, data augmentation and similar leaderboard interventions come back in play

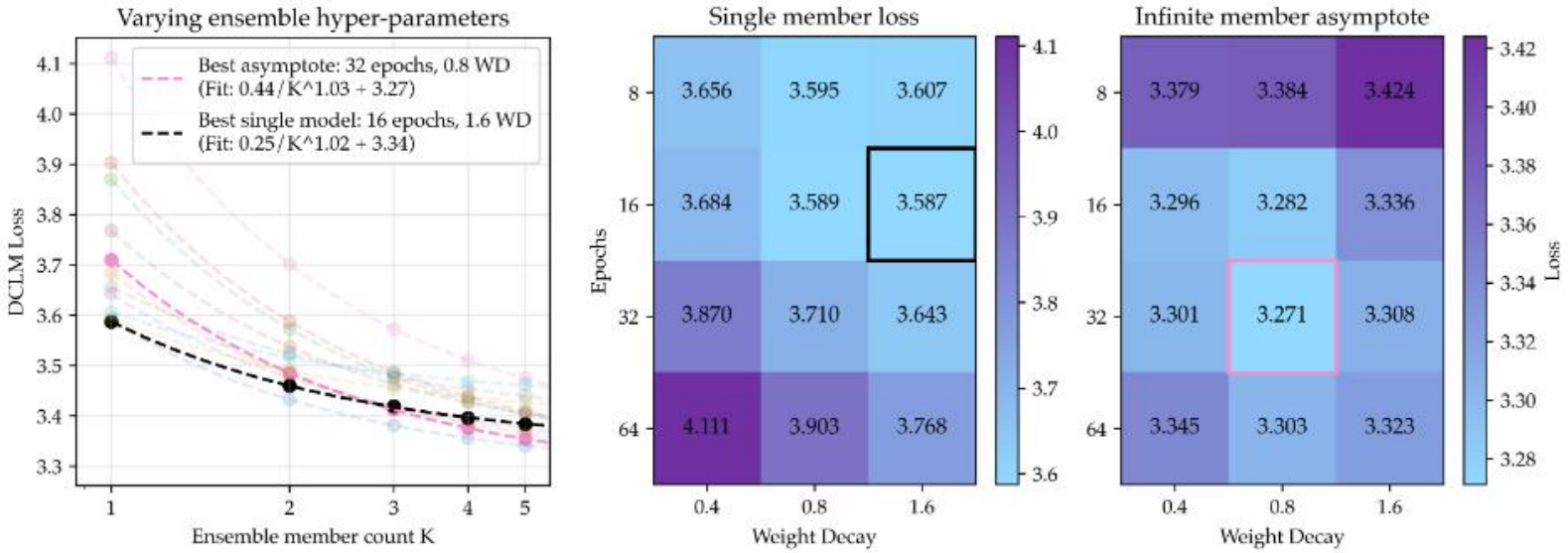
[Herel and Mikolov 2023]

Step 2: Ensembling as a new axis of scaling



Instead of bigger models, we could ensemble more models, fixing the total param count
Ensembles have far better asymptotes (~3.34)

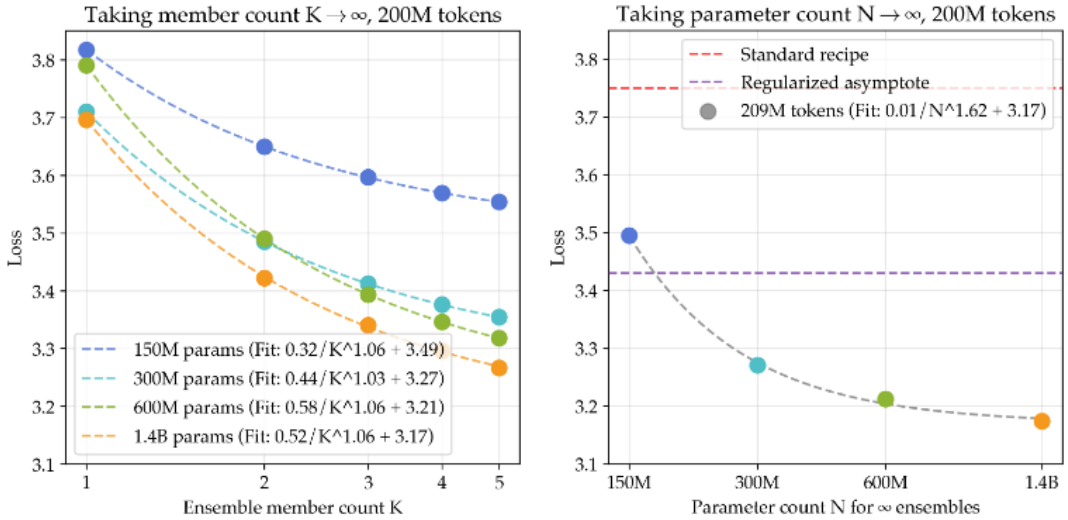
Subtleties – limits and hyperparameters



Getting smooth ensemble scaling is hard – the best hyper params change w/ ensemble count

This complicates the *joint scaling* problem of taking param + ensemble count to infinity

Ensembles are surprisingly good



We estimate the perf of ensembles under infinite compute via a joint limit:

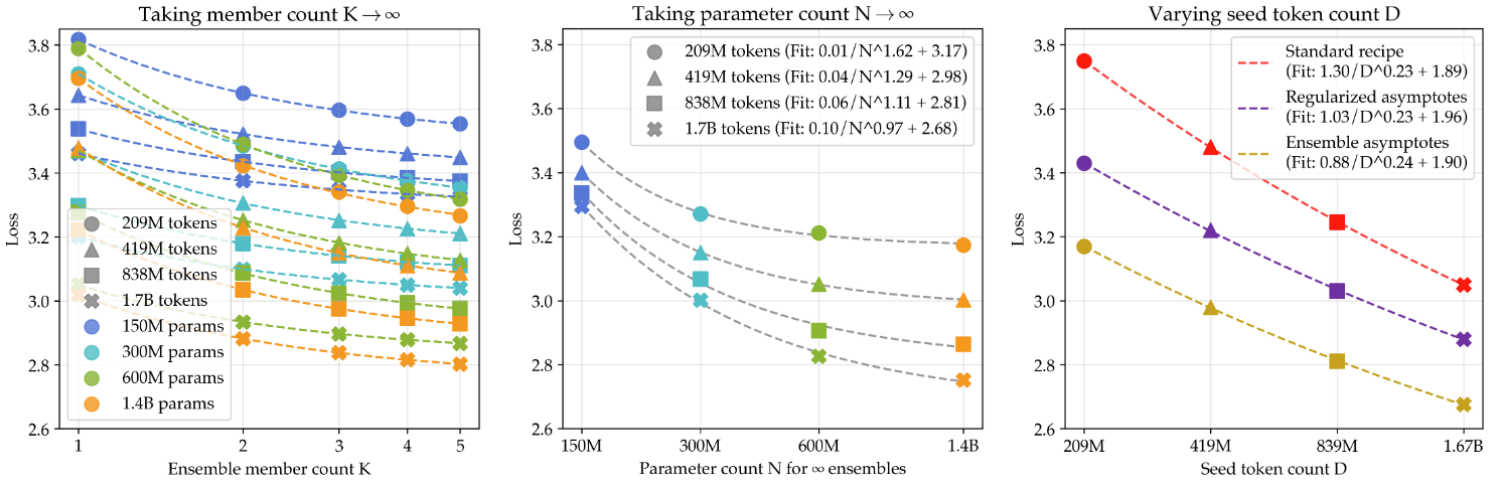
First, ensemble count to infinity, then parameter count to infinity.

This gives much, much better asymptotes (3.17 vs 3.34)

[n.b. We found this to be the easiest order of the limits to estimate via scaling laws.]

Ensembles for larger dataset sizes

We were doing our analysis on small, fixed data (200M tokens, ~1.4B params).
Does this generalize to larger corpora?



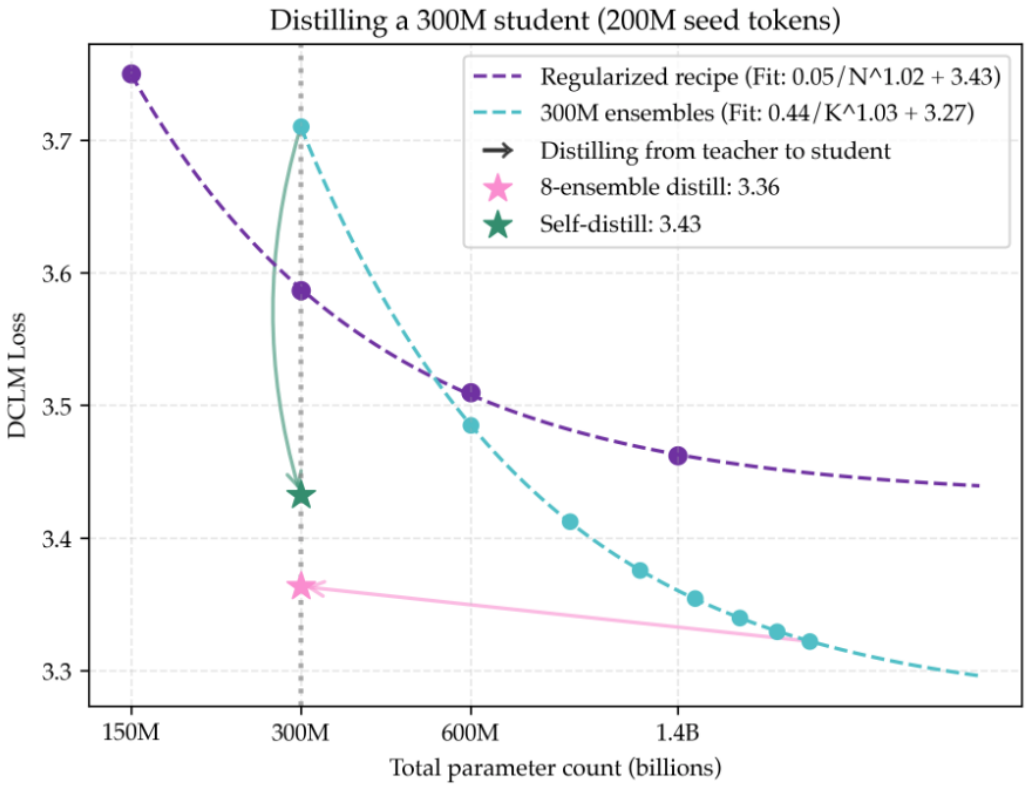
Ensemble gains (left two panels) are consistent as we scale up the corpus (right)

Surprising observation: ensemble distillation and self-training

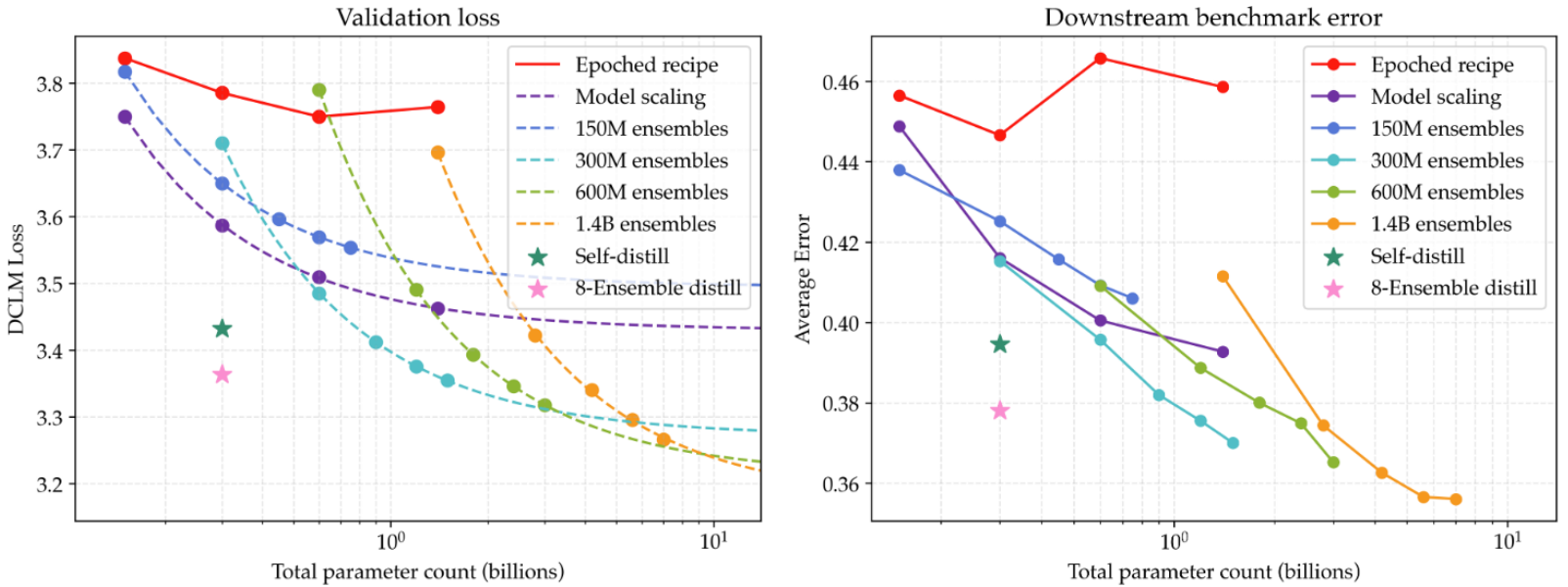
Even with infinite training compute, we still care about inference compute

- Can we distil big ensembles into small models? (yes)
- Can other ‘ensemble like methods like self-training also work? (yes)

[This has interesting implications for data feedback loops / model collapse]



Can we trust small-scale perplexity evaluations?



Generally, yes (avg PIQA, ARC, SciQ)

- Noisier but generally similar scaling trends across all models
- Pareto-dominant models are all ensembles at large compute targets

Gains for continued pretraining

Benchmarks	Llama 3B	CPT (4B tokens)			K-ensembles			CPT (73B tokens)
		Default	Lower BS	Epoching ($K = 1$)	$K = 2$	$K = 4$	$K = 8$	
$\text{GSM8K}_{\text{(8-shot)}}$	28.23	38.44	44.50	44.05	49.28	51.80	**52.99**	49.51
$\text{MATH}_{\text{(4-shot)}}$	6.90	14.38	17.64	19.74	21.84	23.04	**23.50**	23.40
$\text{MATHQA}_{\text{(8-shot)}}$	35.07	38.96	41.31	42.58	45.12	**46.06**	45.26	44.79
Average	24.25	30.59	34.48	35.82	38.79	40.35	**40.58**	39.23

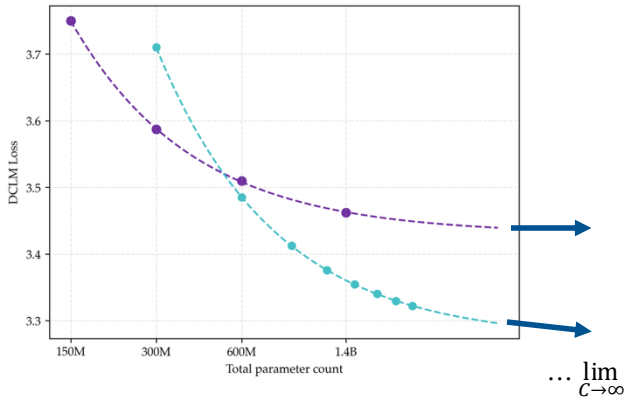
Gains on CPT / Midtraining – this is a naturally data constrained setting

- OctoThinker dataset / evaluations
- The “data efficient” recipes (batch size, epochs) help
- Ensembles help *a lot* for data efficiency

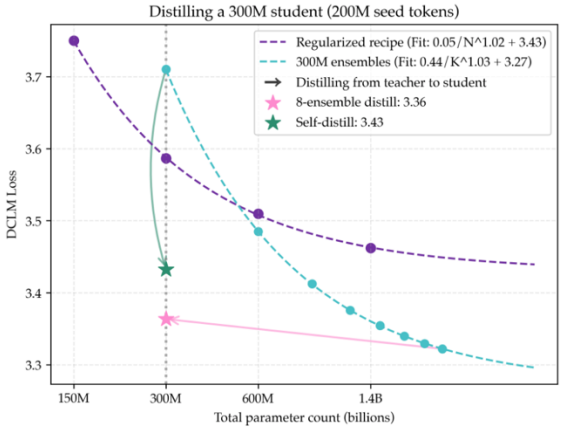
Data in a compute rich future

**Data bottlenecks open up interesting,
basic algorithms research for LLMs**

Open questions



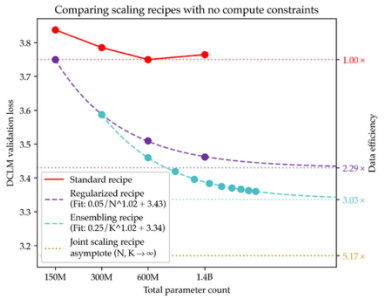
Science of scaling laws for data efficiency



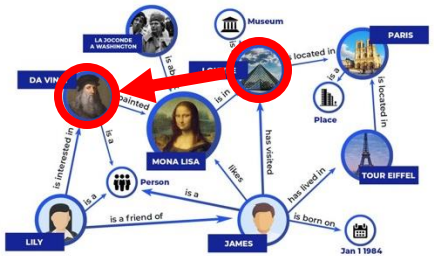
Better foundational understanding of ensembling

Part 2: New horizons with synthetic data

Can we ‘create’ new data?
What would that mean and why would that work?



Part 1: Using data better

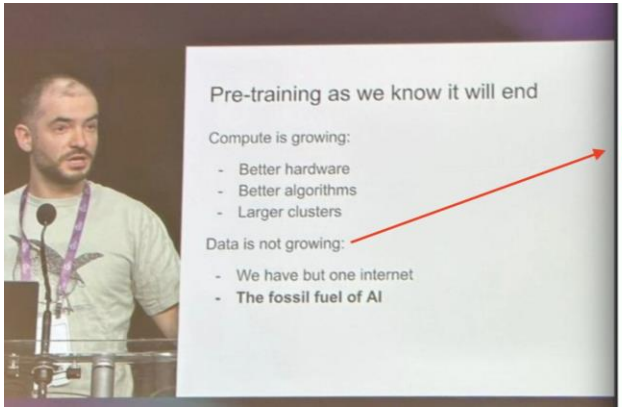


Part 2: Making ‘new’ data

Synthetic continued pretraining [ICLR 2025] - Zitong Yang*, Neil Band*, Shuangping Li, Emmanuel Candès, Tatsunori Hashimoto

Synthetic bootstrapped pretraining [ArXiv preprint] - Zitong Yang*, Aonan Zhang*, Hong Liu, Tatsunori Hashimoto, Emmanuel Candès, Chong Wang, Ruoming Pang

Synthetic data as a major, current bet for data efficiency



What comes next?

- "Agents"??
- "Synthetic data"
- Inference time compute ~ O1

KIMI K2: OPEN AGENTIC INTELLIGENCE

TECHNICAL REPORT OF KIMI K2

Kimi Team

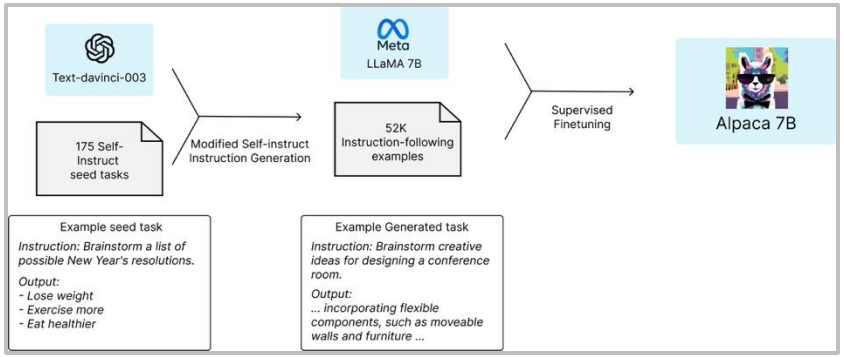
Knowledge Data Rephrasing Pre-training on natural, knowledge-intensive text presents a trade-off: a single epoch is insufficient for comprehensive knowledge absorption, while multi-epoch repetition yields diminishing returns and increases the risk of overfitting. To improve the token utility of high-quality knowledge tokens, we propose a synthetic rephrasing framework composed of the following key components:

- **Style- and perspective-diverse prompting:** To enhance linguistic diversity while maintaining factual integrity, we apply a range of carefully engineered prompts. These prompts guide a large language model to generate faithful rephrasings of the original texts in varied styles and from different perspectives.
- **Chunk-wise autoregressive generation:** To preserve global coherence and avoid information loss in long documents, we adopt a chunk-based autoregressive rewriting strategy. Texts are divided into segments, rephrased individually, and then stitched back together to form complete passages. This method mitigates implicit output length limitations that typically exist with LLMs. An overview of this pipeline is presented in Figure 4.
- **Fidelity verification:** To ensure consistency between original and rewritten content, we perform fidelity checks that compare the semantic alignment of each rephrased passage with its source. This serves as an initial quality control step prior to training.

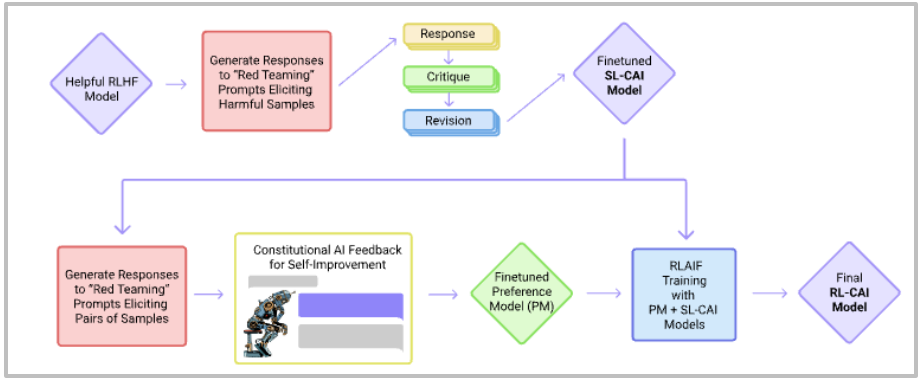
Many bets are currently being placed on 'synthetic data'

Is creating new data in this way really viable?

But the status quo for open synthetic data work is messy



Distillation effects from larger models



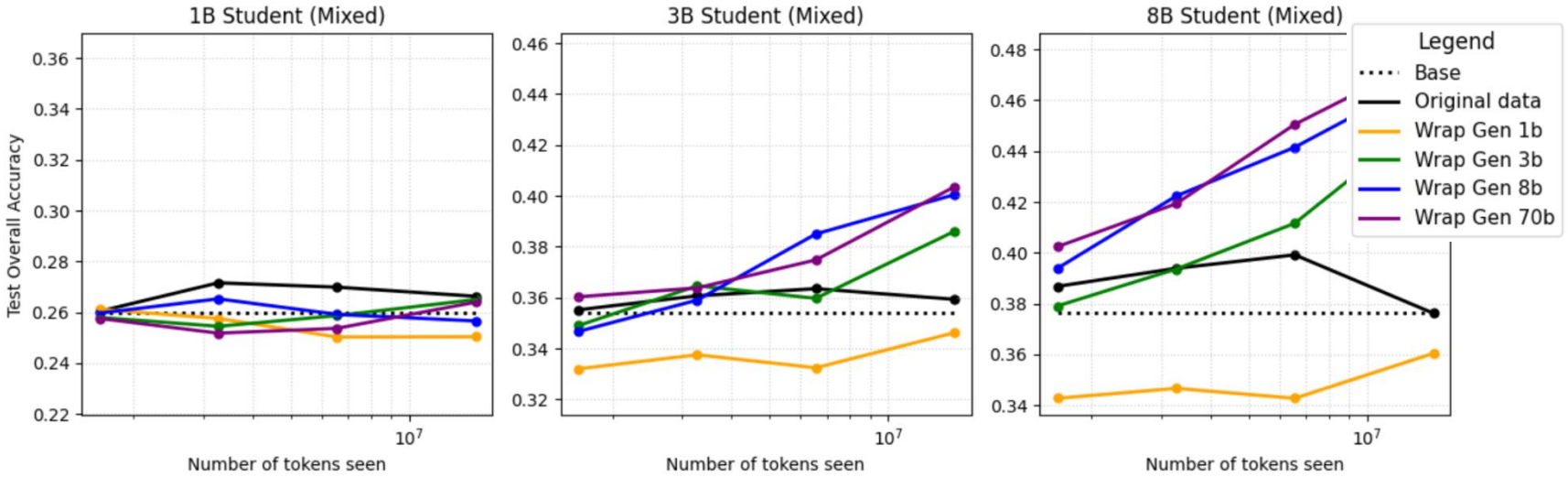
Post-training / alignment of existing capabilities

Synthetic data has often been studied in the context of distillation / post-training

* And sometimes people even refer to RL-like algorithms as synthetic data..

Synthetic data research poses new (scaling) challenges

There is a ‘critical’ threshold for synthetic data to start working
This makes naïve scaling down hard!



Changing the setting – *continued* pretraining

Given:

- A pretrained model f
- A small domain of knowledge that can be characterized by some corpus (e.g. textbook for a specialized domain, proprietary corporate documents).

Goal:

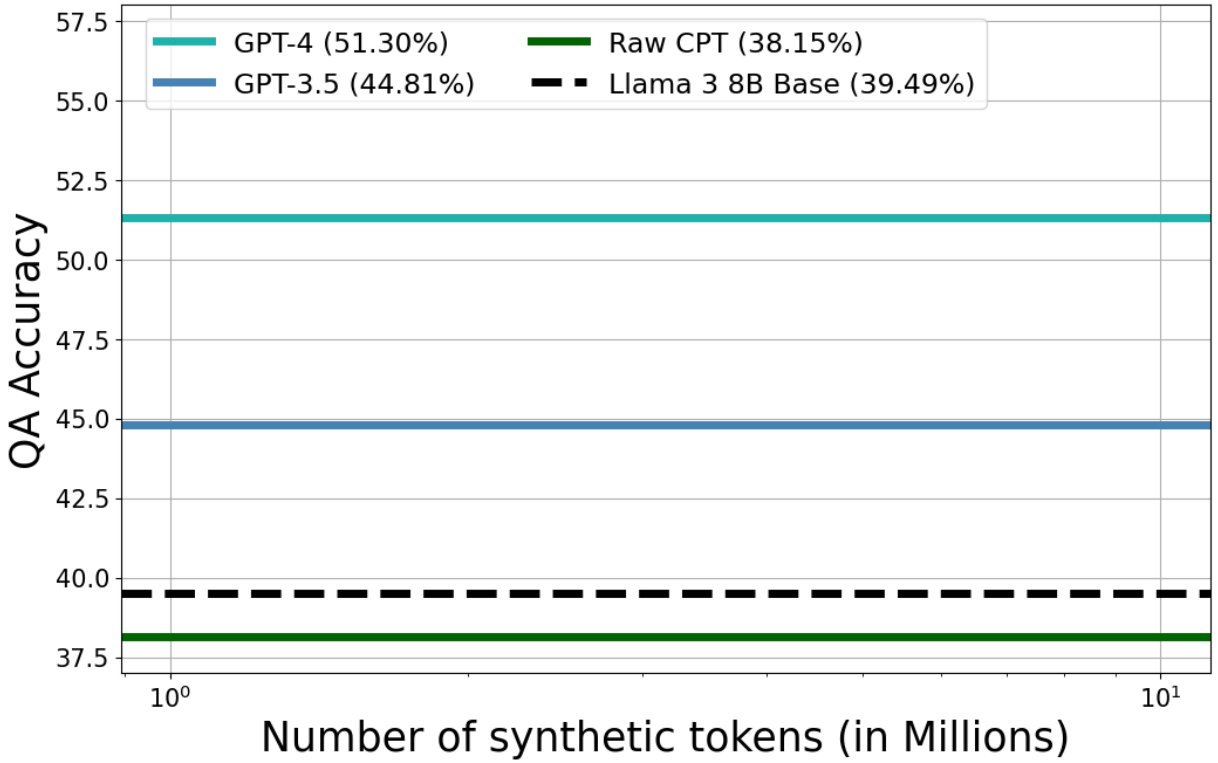
- Modify the weights of f such that
- The modified f can perform a range of tasks – QA, summarization, as if it was pretrained on a large corpus containing our specialized knowledge

Our challenge: learning from 10,000x less data

Study	Domain	Model Parameter Count	Total Unique CPT Tokens
Minerva (Lewkowycz et al., 2022)	STEM	8B, 62B, 540B	26B-38.5B
MediTron (Chen et al., 2023)	Medicine	7B, 70B	46.7B
Code Llama (Rozière et al., 2024)	Code	7B, 13B, 34B	520B-620B
Llemma (Azerbayev et al., 2024)	Math	7B, 34B	50B-55B
DeepSeekMath (Shao et al., 2024)	Math	7B	500B
SaulLM-7B (Colombo et al., 2024b)	Law	7B	30B
SaulLM-{54, 141}B (Colombo et al., 2024a)	Law	54B, 141B	520B
HEAL (Yuan et al., 2024a)	Medicine	13B	14.9B
Our setting	Articles & Books	7B	1.3M

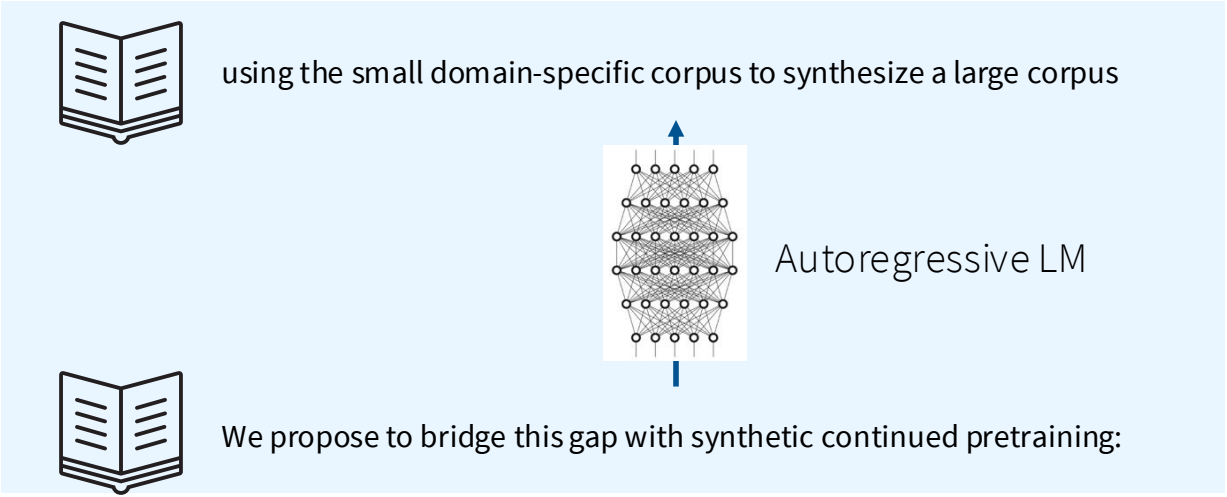
Can we adapt to knowledge that might be truly in the tail?
Few hundred books with **10,000x less data**

Attempt 1 – Just do continued pretraining



Problems with standard continued pretraining

Standard continued pretraining: train directly on our documents



Autoregressive learning is data-inefficient (reversal curse)

In the autoregressive direction: “What does synthetic CPT do?”

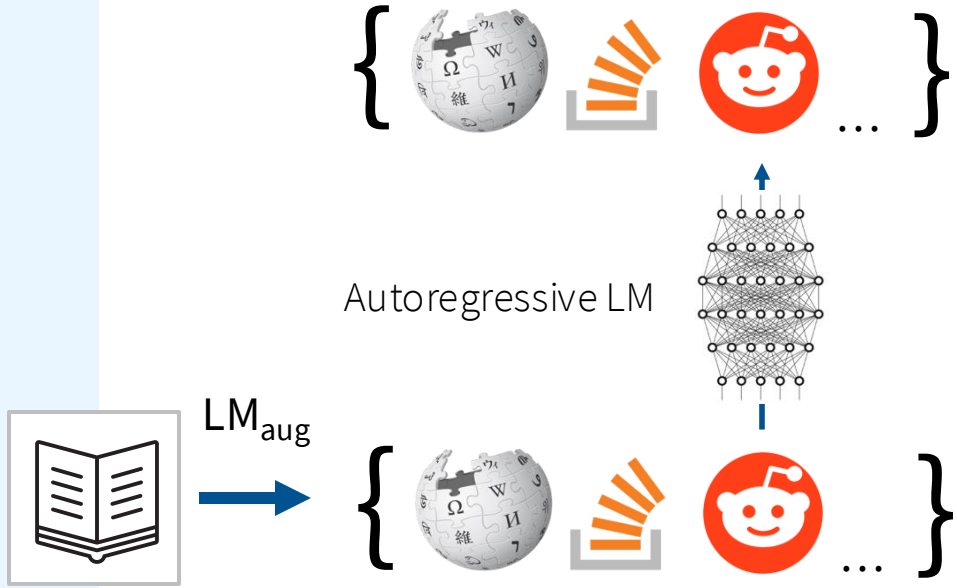
In the reverse direction: “What method synthesizes a large corpus?”

Synthetic continued pretraining – augment the data

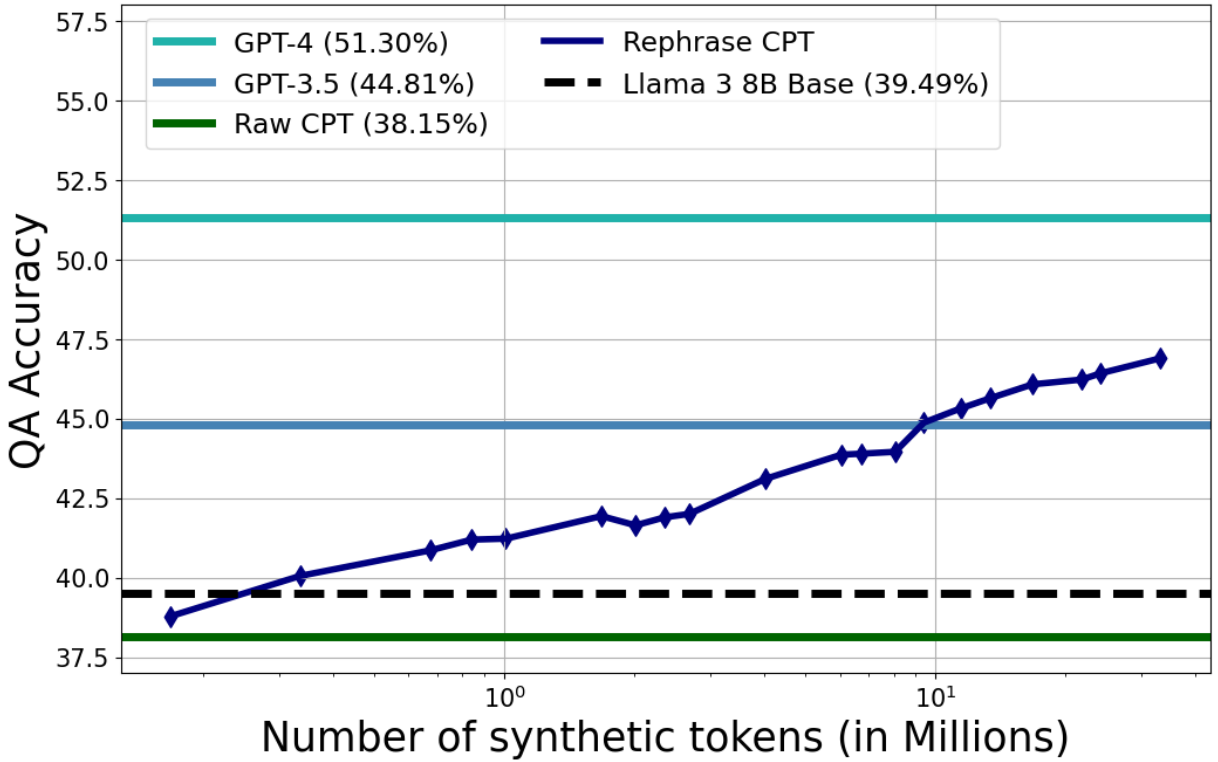
***Synthetic* continued pretraining:** Train on LLM-transformed data

Goal – replicate the diversity of pretraining
- Vary content (topics)
- Vary style (how it's presented)
- Data diversity for generalization

This is *different* from synthetic data or..
- compute / size efficiency (WRAP/Phi)
- fine-tuning (task-specific LMs)



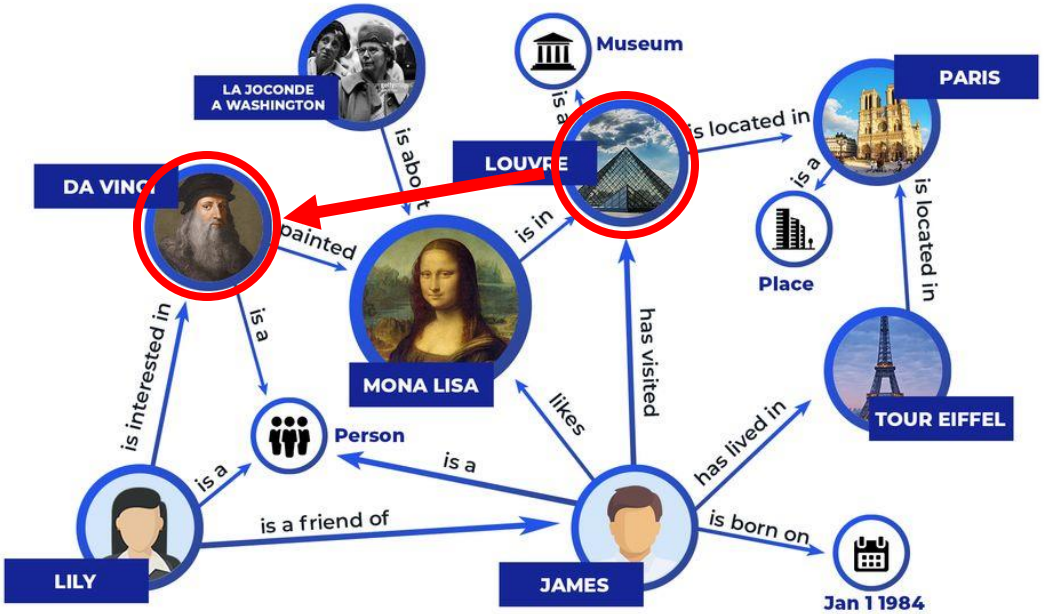
Attempt 2 – Just paraphrase the data



What we get: Entity-centric augmentation (EntiGraph)

How do we get diversity? Use a knowledge graph to force diversity in content

1. Prompt LM_{aug} for entities in a knowledge graph.
2. Sample k-subgraphs of the knowledge graph
3. LM_{aug} synthesizes descriptions of the entities in the subgraph



New implicit fact as data (The Louvre contains many works by DaVinci..)

A random graph model for synthetic data generation

What's the 'mathematical model'?

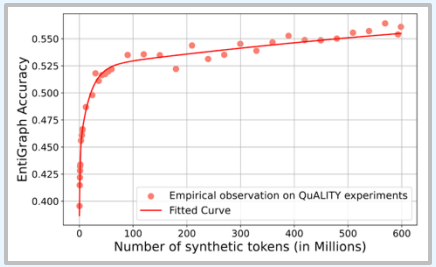
Explicit facts are connected edges between entities (vertices) on a random graph.
Implicit facts are paths within a connected component

EntiGraph 'connects' the whole connected component

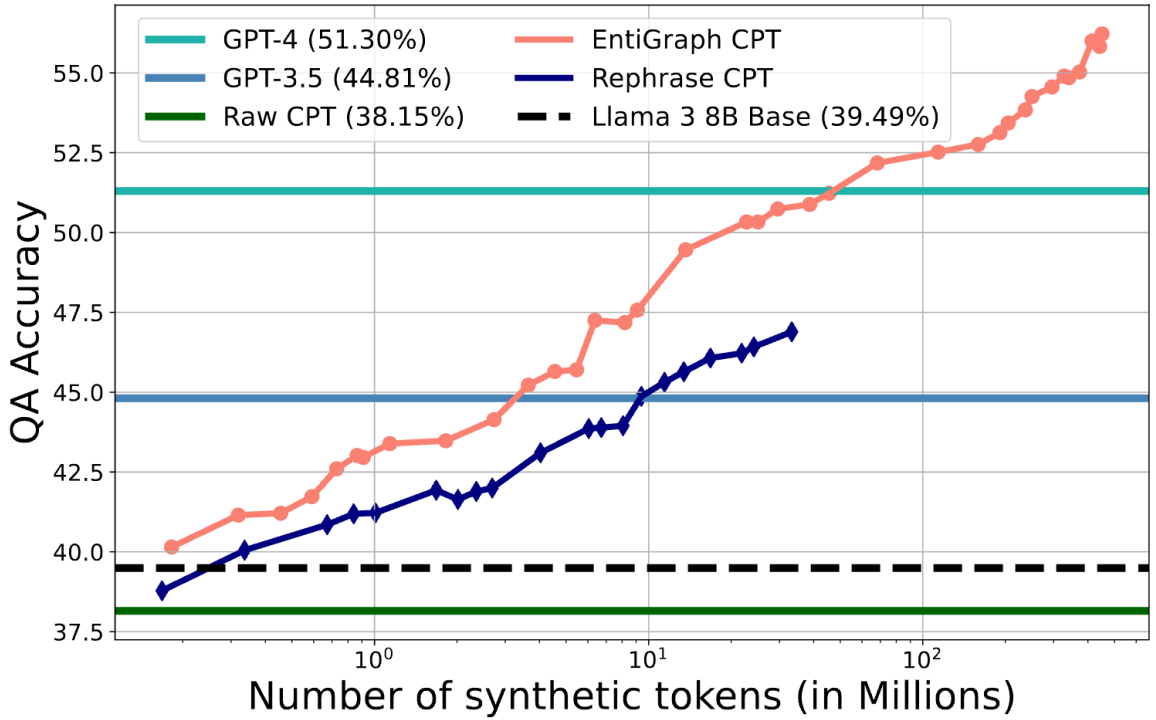
What does this imply?

A mixture-of-exponentials scaling law

$$\text{Acc}(M_t) \sim p + C_\lambda \left(1 - \sum_{\ell=0}^{\infty} \frac{\lambda - 1}{\lambda^{\ell+1}} \sum_{k=1}^{\infty} p_\ell(k) \left(1 - \frac{k}{V(V-1)}\right)^t\right)$$



(Closed-book) QA performance with EntiGraph



Predictable, scaling gains for QA performance *without* the text

Can we go beyond prompting a model?

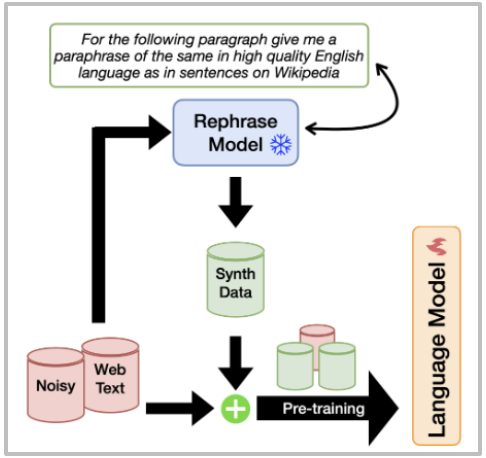
We've validated a 'scaled down' idea with continued pretraining.

Can we get this to work in the "pure bootstrapping setting"? This means..

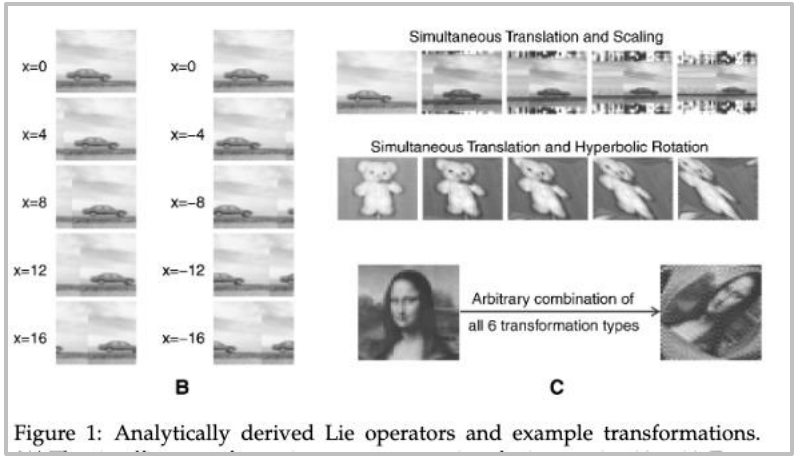
- No pre-trained teacher (we have to train our own)
- No strong hand-crafted prompts.

The first one is just a compute problem ('train your own teacher')
The second one is a bit more subtle – what is a 'general purpose' way to learn to augment data?

Synthetic data as data augmentation



Maini et al ‘24



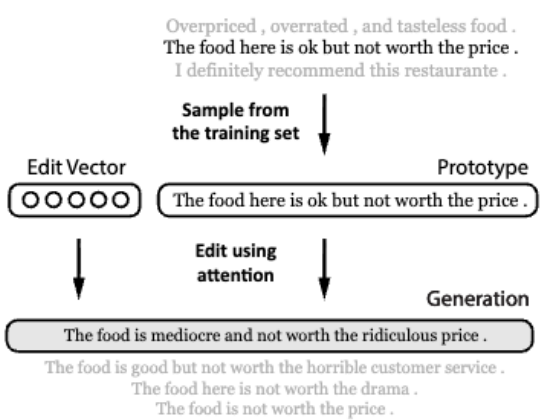
Rao and ruderman ‘99

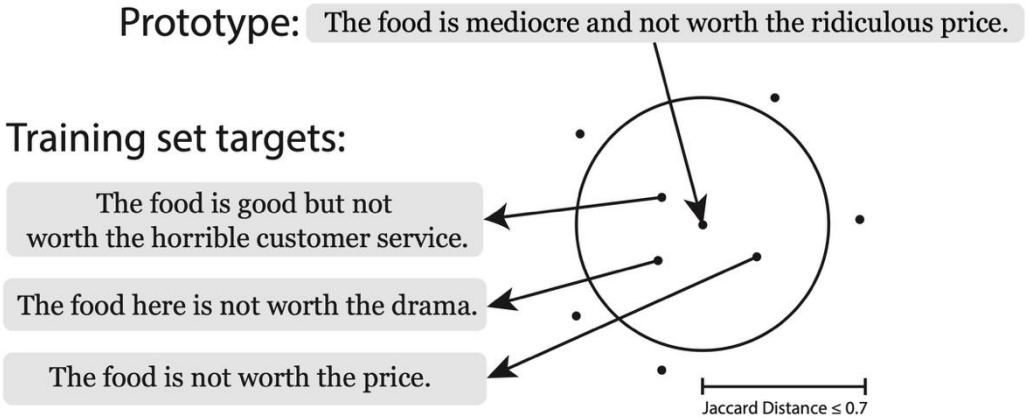
Rephrasing is much like an label-invariant data augmentation

There is a deep literature on unsupervised learning of transformations (i.e. generators of a lie algebra)

Learning augmentations and smoothing kernels

We showed a while ago that “learn augmentations from neighbors” also works for language models.





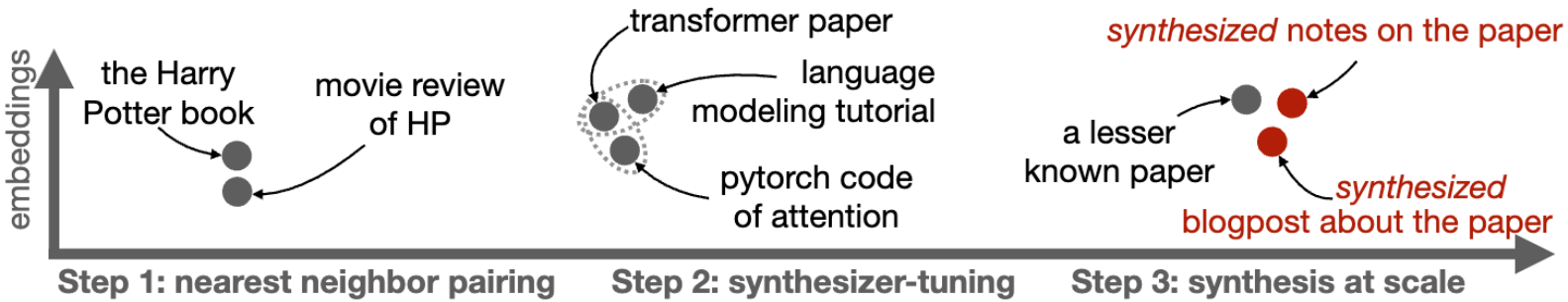
This is a nonparametric language model where the kernel is learned [Guu, Hashimoto, Oren, Liang ‘18]

$P(x) = P(x|x')P(x')$ where $P(x|x')$ is an *edit* and $P(x')$ is a prototype

Learning the augmentations from scratch

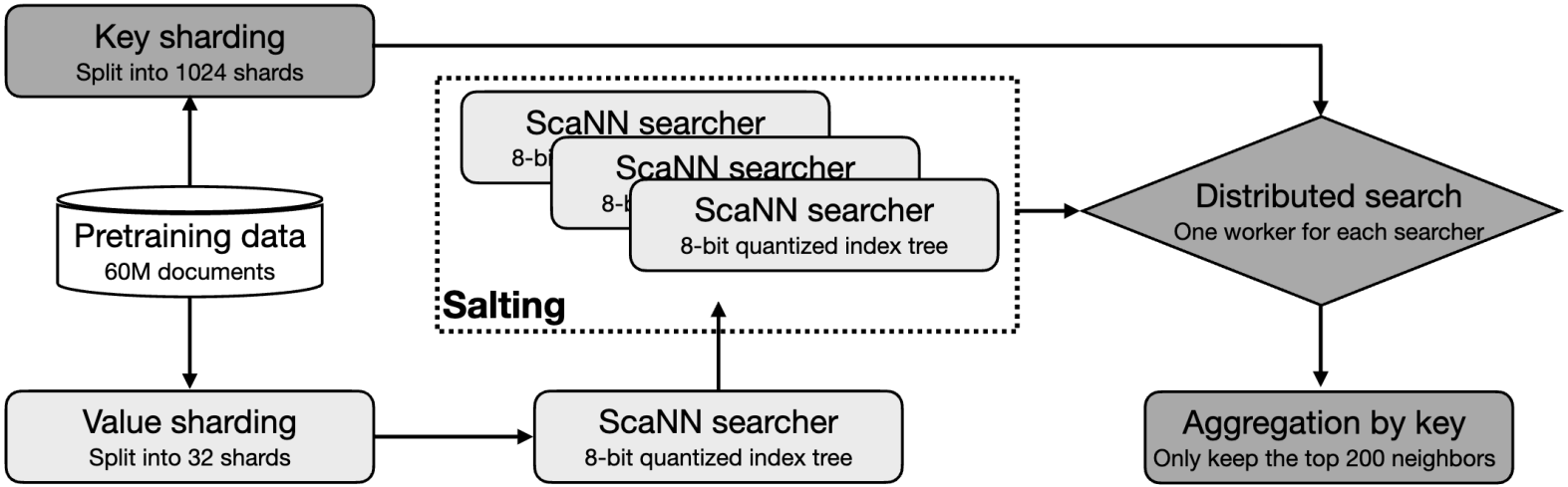
New synthetic data generation process: synthesizer trained on neighbor supervision

Synthetic data is a supervised task of transforming a document into its neighbor



Do these classic “transformation / lie group learning” methods translate to the pretraining setting?

Scaling up fast document nearest neighbor search



Scaling up nearest-neighbor search to pretraining scale (for both queries and keys) requires careful handling of ANN indices

The overall procedure

The whole synthetic data / bootstrapping process is 3 simple steps given a base model

Step 1: Pair up nearest neighbors

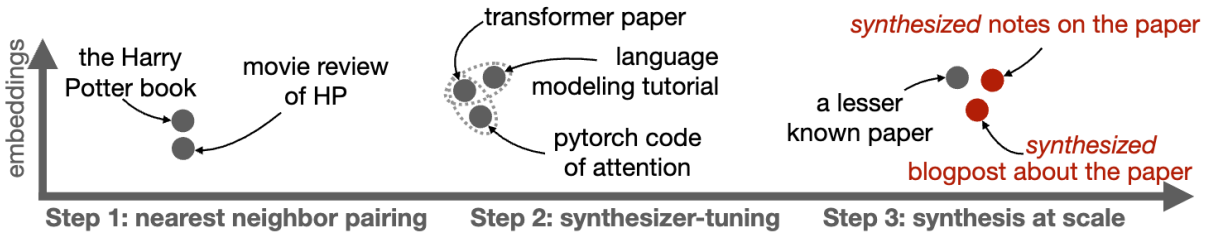
Define a inner product metric d, pair up neighbors with inner product $> \alpha$

Step 2: Learn the augmentation

Learn a conditional LM $p(x_2|x_1)$ by continuing to train the base model

Step 3: Generate synth data and train on it

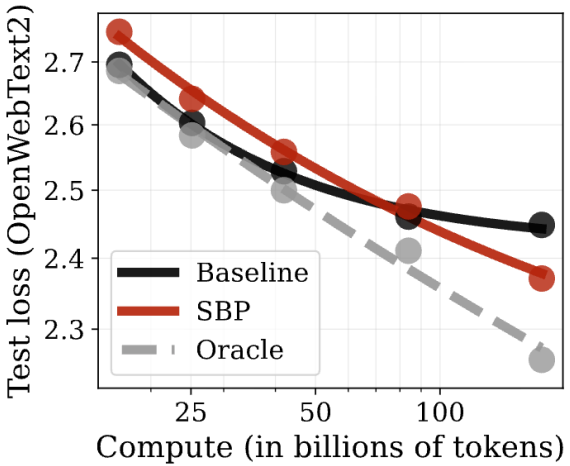
Sample new documents from $p(x_2|x_1)$ for each document,
train a new model on a mix of synth + real data (1:3 and 1:8 ratios)



Initial steps at scaling

200 Billion token budget (deduplicated DCLM data) for a 3B model

Benchmark	200B-scale Baseline	SBP	Oracle
	Perplexity on held-out data ↓		
OpenWebText2	5.74	-0.53	-1.02
LAMBADA	6.87	-0.85	-1.86
Five-shot MMLU	3.83	-0.36	-0.51
	QA accuracy ↑		
ARC-Challenge (0-shot)	35.32	+1.28	+2.82
ARC-Easy (0-shot)	68.94	+2.65	+4.29
SciQ (0-shot)	90.50	+1.00	+2.40
Winogrande (0-shot)	60.14	+1.90	+5.53
TriviaQA (1-shot)	22.51	+3.36	+7.37
WebQS (1-shot)	8.56	+3.74	+10.83
Average QA accuracy	**47.66**	**+2.32**	**+5.54**



Notes: 75B synthetic repeated once, 10B real data repeated 12.5 times

Results at 3B/1T

Benchmark	200B-scale			1T-scale		
	Baseline	SBP	Oracle	Baseline	SBP	Oracle
	Perplexity on held-out data ↓					
OpenWebText2	5.74	-0.53	-1.02	4.51	-0.02	-0.12
LAMBADA	6.87	-0.85	-1.86	4.33	-0.03	-0.22
Five-shot MMLU	3.83	-0.36	-0.51	3.17	-0.06	-0.05
	QA accuracy ↑					
ARC-Challenge (0-shot)	35.32	+1.28	+2.82	42.66	+1.62	+3.84
ARC-Easy (0-shot)	68.94	+2.65	+4.29	75.63	+0.42	+2.11
SciQ (0-shot)	90.50	+1.00	+2.40	93.20	+0.80	+0.50
Winogrande (0-shot)	60.14	+1.90	+5.53	65.19	+1.42	+2.92
TriviaQA (1-shot)	22.51	+3.36	+7.37	36.07	+0.25	+0.59
WebQS (1-shot)	8.56	+3.74	+10.83	19.34	+0.54	+0.44
Average QA accuracy	**47.66**	**+2.32**	**+5.54**	**55.35**	**+0.84**	**+1.73**

Gains (even at scale) capturing half the gains to having fresh data (oracle)

Qualitative evaluations of data quality across scales

	Repetition ↓	Duplicate@1M ↓	Non-factual ↓	Pair-irrelevance ↓	Pair-copying ↓
200B-scale	4.3%	0.8%	15.1%	25.6%	0.1%
1T-scale	3.9%	0.8%	8.7%	7.8%	0.9%
Real data	1.8%	0.7%	1.8%	n.a.	n.a.

Data quality failures (repetition, low diversity, etc) decrease with scale

Real document

HOME > Chowhound > California >

Coffeehouses in SD

I am always looking for good coffeehouses in San Diego. I am looking for coffeehouses with (obviously) excellent coffeebeans, baristas who know their business (good microfoam, good crema, they should know the different ratios for steamed milk, foamed milk, espresso for the cappuccino, latte

Synthesis I

Coffee = Love

On a recent trip to San Diego, one thing I wanted to find was coffee. I was worried it might be difficult with the huge decrease in Asian flavors, but I was pleasantly surprised. Many of the Mexican, Italian, and Cuban flavors that are pretty common in California are now in the mix. While I did find a few highly recommended spots, the majority of

Synthesis II

Discuss as:

Coffeehouses: San Diego. The best?

Café Lestat, by far, is a stand-out. (photo by Thomas Moriarty)

If the coffee scene in San Diego sounds like the one in New York, there's a reason. The upper echelons of NYC's drink scene are the same ones

Takeaways: synthetic data

Powerful, new forms of augmentation
(and pretraining unlocks new, stronger gains from augmentations)

1. Can we completely close the ICL to parametric learning gap with enough compute?
2. What are the limits to synthetic data based augmentation?
3. What about synthetic data for reasoning?

Thanks!

Language modeling may return to a foundational challenge – *generalizing from limited data*

In many ways, this is challenging

Algorithmic gains have been hard fought (and not that well understood)

But also it's an exciting problem

Foundational problems (might) require more radical solutions