Scaling and Emergent Ability

Jay Lalwani & Tanmai Kalisipudi

DeepMind

March 29, 2022

Training Compute-Optimal Large Language Models

Jordan Hoffmann*, Sebastian Borgeaud*, Arthur Mensch*, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals and Laurent Sifre*

*Equal contributions

Context and Problem Statement

- LLMs have demonstrated remarkable performance across a wide set of tasks as they scale (Kaplan et al. / OpenAI's original neural scaling paper)
  - The larger the model, the larger the budget needed

$$C = \text{some constant} \times N \times D$$

- Scaling involves increasing the number of parameters in the model and training the model with more tokens
- In models like GPT-3 and Gopher, they took the approach of scaling the model parameters more predominantly, but this paper hypothesizes those models are undertrained

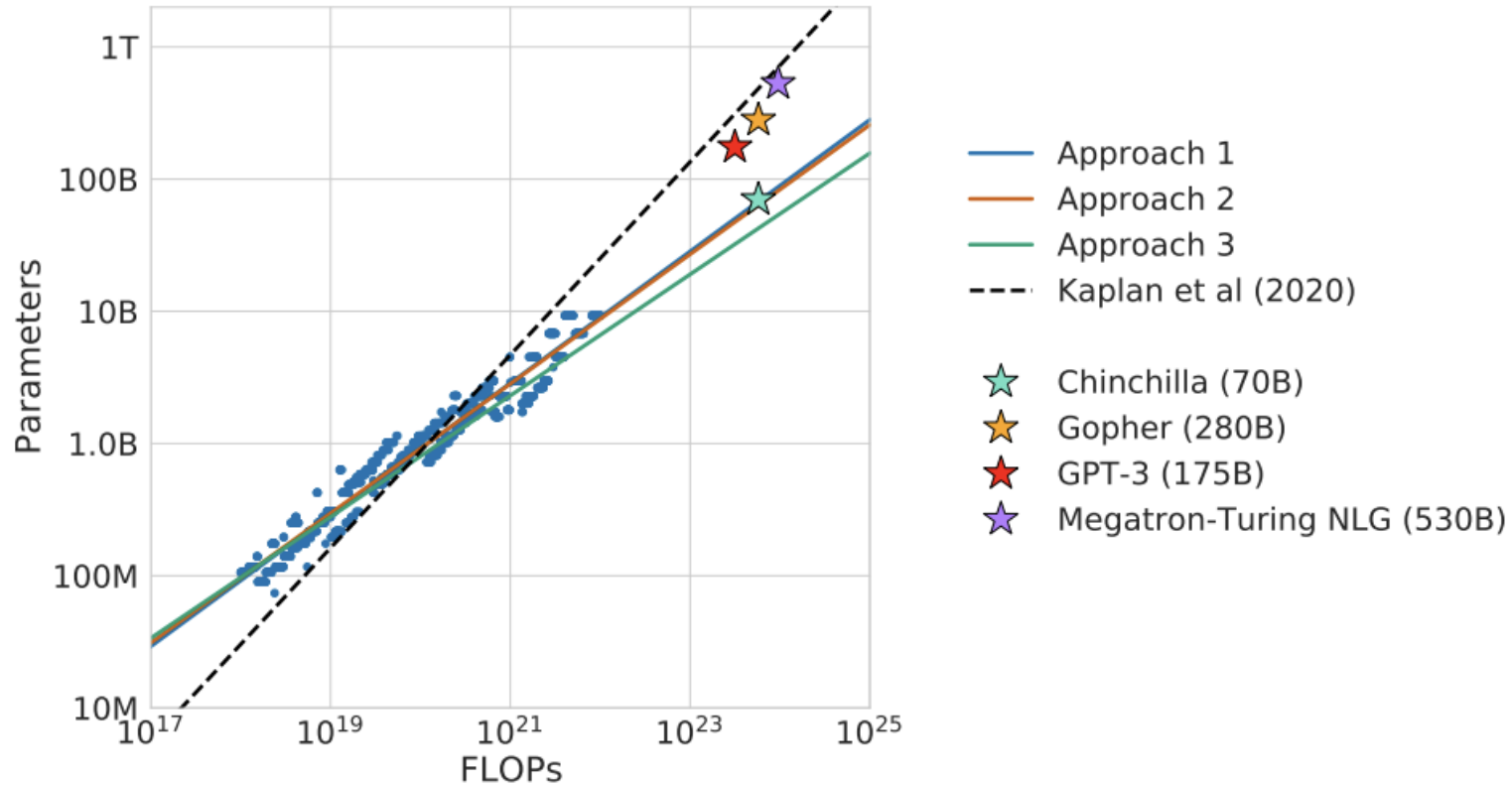


Figure 1 | **Overlaid predictions.** We overlay the predictions from our three different approaches, along with projections from Kaplan et al. (2020). We find that all three methods predict that current large models should be substantially smaller and therefore trained much longer than is currently done. In Figure A3, we show the results with the predicted optimal tokens plotted against the optimal number of parameters for fixed FLOP budgets. ***Chinchilla* outperforms *Gopher* and the other large models (see Section 4.2).**

How can we find the compute-optimal parameters and dataset size to minimize pre-training loss? In other words, how can we optimize model size and number of training tokens for a given budget?

$$N_{opt}(C), D_{opt}(C) = \underset{N,D \text{ s.t. } \mathrm{FLOPs}(N,D)=C}{\operatorname{argmin}} L(N, D).$$

Prior Work

- OpenAI's scaling laws paper
  - Suggested larger models (increasing N) trained on fewer tokens
  - Said diminishing returns on training with more tokens
  - This paper says these estimates will undertrain the models severely
  - Chinchilla disagrees directly with this paper and says that N and D should scale equally
- Used other background research to set default hyperparameters (more minor like learning rate and batch size)

Methodology

Variety of approaches to find best fit line

1. Fixed model sizes and vary number of training tokens
2. IsoFLOP profiles
3. Fitting a parametric loss function

Approach One: Keeps N fixed, varies D

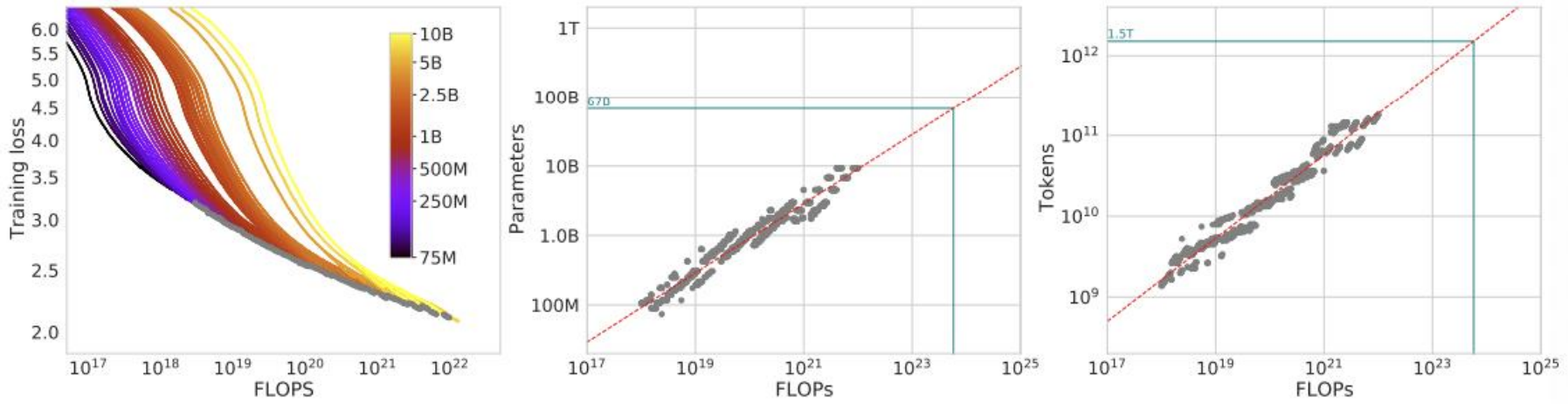


Figure 2 | **Training curve envelope.** On the **left** we show all of our different runs. We launched a range of model sizes going from 70M to 10B, each for four different cosine cycle lengths. From these curves, we extracted the envelope of minimal loss per FLOP, and we used these points to estimate the optimal model size (**center**) for a given compute budget and the optimal number of training tokens (**right**). In green, we show projections of optimal model size and training token count based on the number of FLOPs used to train *Gopher* (5.76×10^{23}).

$N_{opt} \propto C^a$ and $D_{opt} \propto C^b$

Where a = b = 0.5

Approach Two: Varies both N and D while keeping C constant

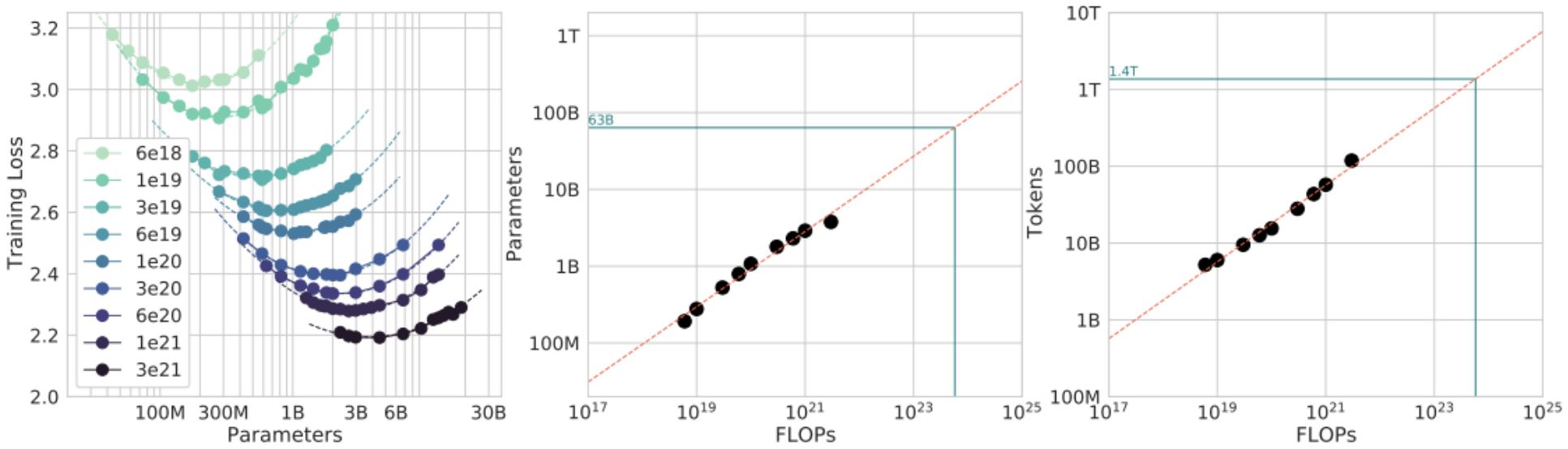
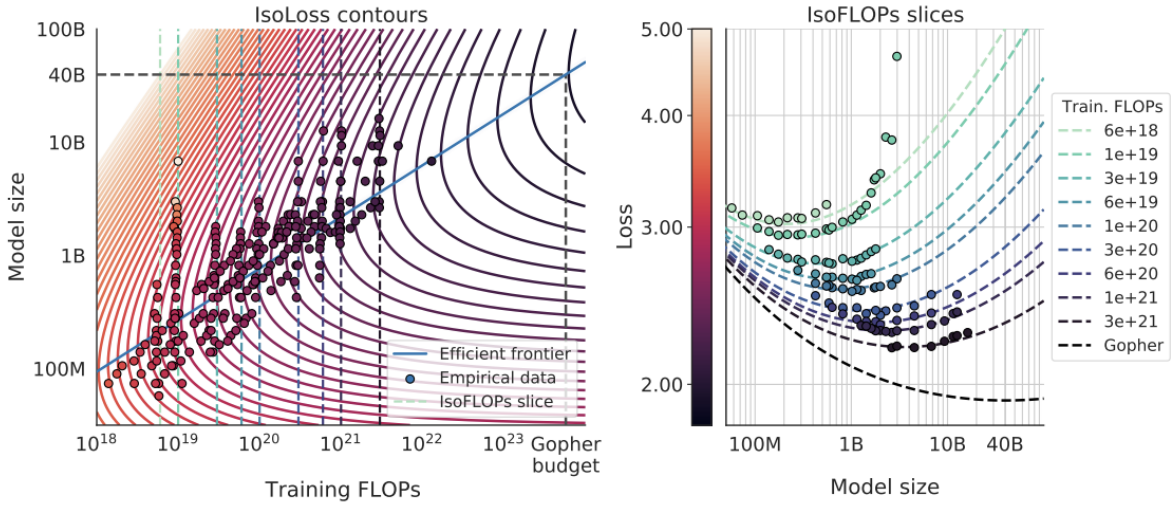


Figure 3 | **IsoFLOP curves.** For various model sizes, we choose the number of training tokens such that the final FLOPs is a constant. The cosine cycle length is set to match the target FLOP count. We find a clear valley in loss, meaning that for a given FLOP budget there is an optimal model to train (**left**). Using the location of these valleys, we project optimal model size and number of tokens for larger models (**center** and **right**). In green, we show the estimated number of parameters and tokens for an *optimal* model trained with the compute budget of *Gopher*.

Approach Three: Use the losses from #1&#2 to model a parametric function



$$N_{opt}(C) = G\left(\frac{C}{6}\right)^{a}, \quad D_{opt}(C) = G^{-1}\left(\frac{C}{6}\right)^{b}, \quad \text{where} \quad G = \left(\frac{\alpha A}{\beta B}\right)^{\frac{1}{\alpha+\beta}}, \quad a = \frac{\beta}{\alpha+\beta}, \text{ and } b = \frac{\alpha}{\alpha+\beta}. \quad (4)$$

$$\min_{A,B,E,\alpha,\beta} \sum_{\text{Runs } i} \text{Huber}_{\delta}\left(\log \hat{L}(N_i, D_i) - \log L_i\right)$$

$$\hat{L}(N, D) \triangleq E + \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}}.$$

Figure 4 | **Parametric fit.** We fit a parametric modelling of the loss $\hat{L}(N, D)$ and display contour (**left**) and isoFLOP slices (**right**). For each isoFLOP slice, we include a corresponding dashed line in the left plot. In the left plot, we show the efficient frontier in blue, which is a line in log-log space. Specifically, the curve goes through each iso-loss contour at the point with the fewest FLOPs. We project the optimal model size given the *Gopher* FLOP budget to be 40B parameters.

E: irreducible **error**
A/N: Decreasing error with model size
B/D: Decreasing error with dataset size

Findings

- IsoFLOP experiments say that doubling the model size should result in the doubling in the size of the dataset
  - Scaling tokens reduces loss significantly more than just increasing model size
  - Found that large models out are severely undertrained and could train a much smaller model with the same performance with a larger dataset

Table 2 | **Estimated parameter and data scaling with increased training compute.** The listed values are the exponents, a and b, on the relationship $N_{opt} \propto C^a$ and $D_{opt} \propto C^b$. Our analysis suggests a near equal scaling in parameters and data with increasing compute which is in clear contrast to previous work on the scaling of large models. The 10^{th} and 90^{th} percentiles are estimated via bootstrapping data (80% of the dataset is sampled 100 times) and are shown in parenthesis.

Approach	Coeff. a where $N_{opt} \propto C^a$	Coeff. b where $D_{opt} \propto C^b$
1. Minimum over training curves	0.50 (0.488, 0.502)	0.50 (0.501, 0.512)
2. IsoFLOP profiles	0.49 (0.462, 0.534)	0.51 (0.483, 0.529)
3. Parametric modelling of the loss	0.46 (0.454, 0.455)	0.54 (0.542, 0.543)
Kaplan et al. (2020)	0.73	0.27

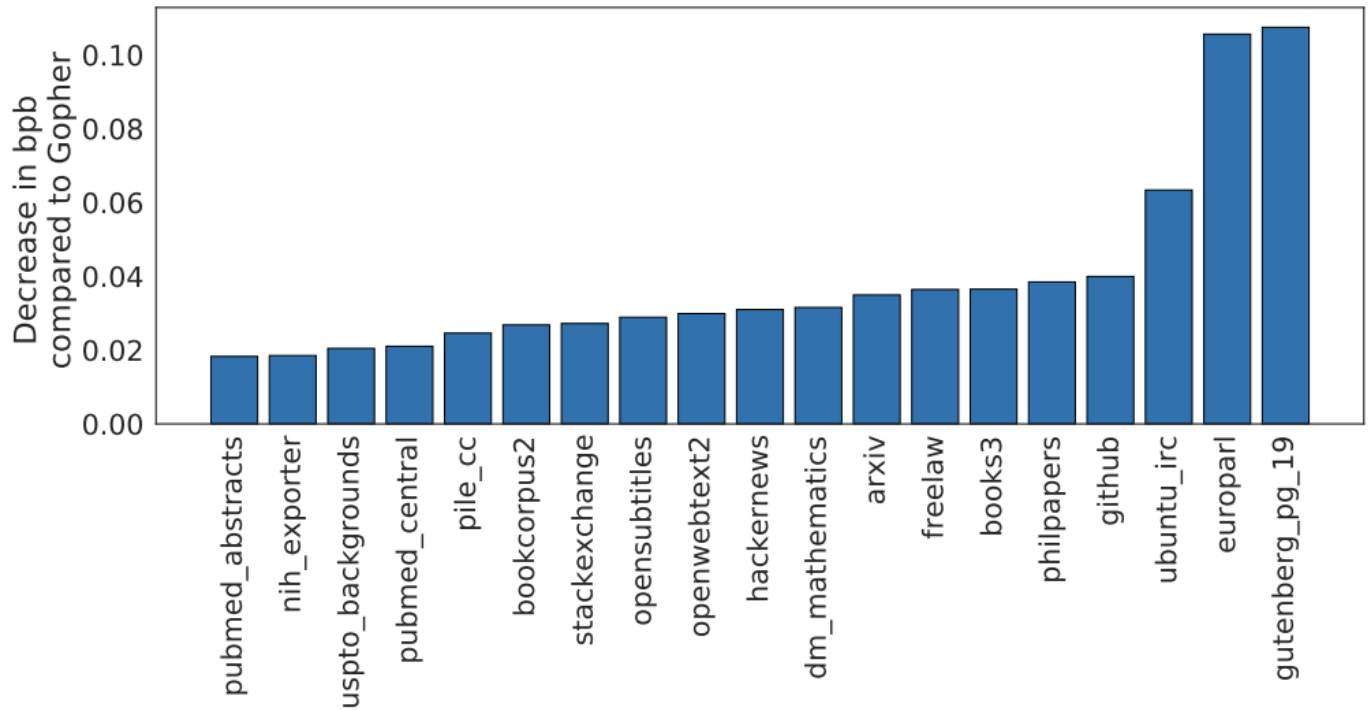
Parameters	FLOPs	FLOPs (in *Gopher* unit)	Tokens
400 Million	1.92e+19	1/29,968	8.0 Billion
1 Billion	1.21e+20	1/4,761	20.2 Billion
10 Billion	1.23e+22	1/46	205.1 Billion
67 Billion	5.76e+23	1	1.5 Trillion
175 Billion	3.85e+24	6.7	3.7 Trillion
280 Billion	9.90e+24	17.2	5.9 Trillion
520 Billion	3.43e+25	59.5	11.0 Trillion
1 Trillion	1.27e+26	221.3	21.2 Trillion
10 Trillion	1.30e+28	22515.9	216.2 Trillion

Chinchilla

- Trained with the same budget as Gopher and with the same training setup and model architecture
  - Trained on MassiveText but more of it
- 70b parameters (¼ size of Gopher) and 1.4T tokens (4.7x more than Gopher)
  - Added benefit: smaller model also requires less resources to run inference

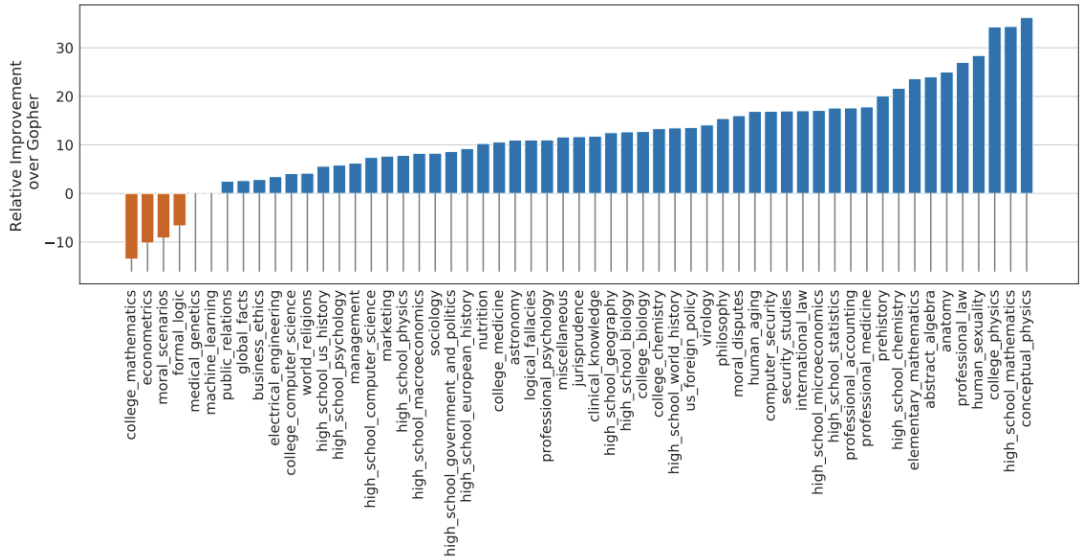
Model	Layers	Number Heads	Key/Value Size	d_{model}	Max LR	Batch Size
Gopher 280B	80	128	128	16,384	4×10^{-5}	3M $\rightarrow$ 6M
Chinchilla 70B	80	64	128	8,192	1×10^{-4}	1.5M $\rightarrow$ 3M

Performance on the Pile



Performance on MMLU (Massive Multitask Language Understanding)

Random	25.0%
Average human rater	34.5%
GPT-3 5-shot	43.9%
Gopher 5-shot	60.0%
***Chinchilla* 5-shot**	**67.6%**
Average human expert performance	*89.8%*
June 2022 Forecast	57.1%
June 2023 Forecast	63.4%



Performance on Common Sense and closed book QA

	Chinchilla	*Gopher*	GPT-3	MT-NLG 530B	Supervised SOTA
HellaSWAG	**80.8%**	79.2%	78.9%	80.2%	93.9%
PIQA	81.8%	81.8%	81.0%	**82.0%**	90.1%
Winogrande	**74.9%**	70.1%	70.2%	73.0%	91.3%
SIQA	**51.3%**	50.6%	-	-	83.2%
BoolQ	**83.7%**	79.3%	60.5%	78.2%	91.4%

	Method	*Chinchilla*	*Gopher*	GPT-3	SOTA (open book)
Natural Questions (dev)	0-shot	16.6%	10.1%	14.6%	54.4%
	5-shot	31.5%	24.5%	-	
	64-shot	35.5%	28.2%	29.9%	
TriviaQA (unfiltered, test)	0-shot	67.0%	52.8%	64.3 %	-
	5-shot	73.2%	63.6%	-	
	64-shot	72.3%	61.3%	71.2%	
TriviaQA (filtered, dev)	0-shot	55.4%	43.5%	-	72.5%
	5-shot	64.1%	57.0%	-	
	64-shot	64.6%	57.2%	-	

Conclusions and final thoughts

- Smaller, well-trained models outperform larger, undertrained ones, optimizing performance per FLOP
  - Scale dataset size and model parameters at same rate
  - Collecting a large number of high quality tokens is hard
- Limitations of the study and scaling assumptions
  - Limited large-scale training runs (Chinchilla and Gopher), limiting generalizability
  - No additional tests at intermediate scales, leaving gaps in performance analysis
  - Assumes a power-law relationship between compute, model size, and training tokens, but slight concavity at high compute budgets suggests potential deviations
- Inference efficiency and compute optimization

Emergent Abilities of Large Language Models

Jason Wei [1] *jasonwei@google.com*
Yi Tay [1] *yitay@google.com*
Rishi Bommasani [2] *nlprishi@stanford.edu*
Colin Raffel [3] *craffel@gmail.com*
Barret Zoph [1] *barretzoph@google.com*
Sebastian Borgeaud [4] *sborgeaud@deepmind.com*
Dani Yogatama [4] *dyogatama@deepmind.com*
Maarten Bosma [1] *bosma@google.com*
Denny Zhou [1] *dennyzhou@google.com*
Donald Metzler [1] *metzler@google.com*
Ed H. Chi [1] *edchi@google.com*
Tatsunori Hashimoto [2] *thashim@stanford.edu*
Oriol Vinyals [4] *vinyals@deepmind.com*
Percy Liang [2] *pliang@stanford.edu*
Jeff Dean [1] *jeff@google.com*
William Fedus [1] *liamfedus@google.com*

[1] *Google Research* [2] *Stanford University* [3] *UNC Chapel Hill* [4] *DeepMind*

Emergent Abilities (Definitions)

Emergence:

Emergence is when quantitative changes in a system result in qualitative changes in behavior.

Emergent Ability:

An ability is emergent if it is not present in smaller models but is present in larger models.

- Emergent Abilities would not be predicted by extrapolating a scaling law
- Performance is near-random until a critical threshold, where performance spikes (phase transition)

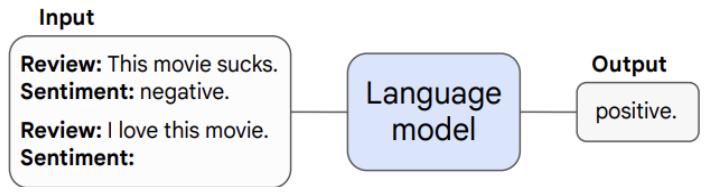
Emergence (Few-Shot Prompting)

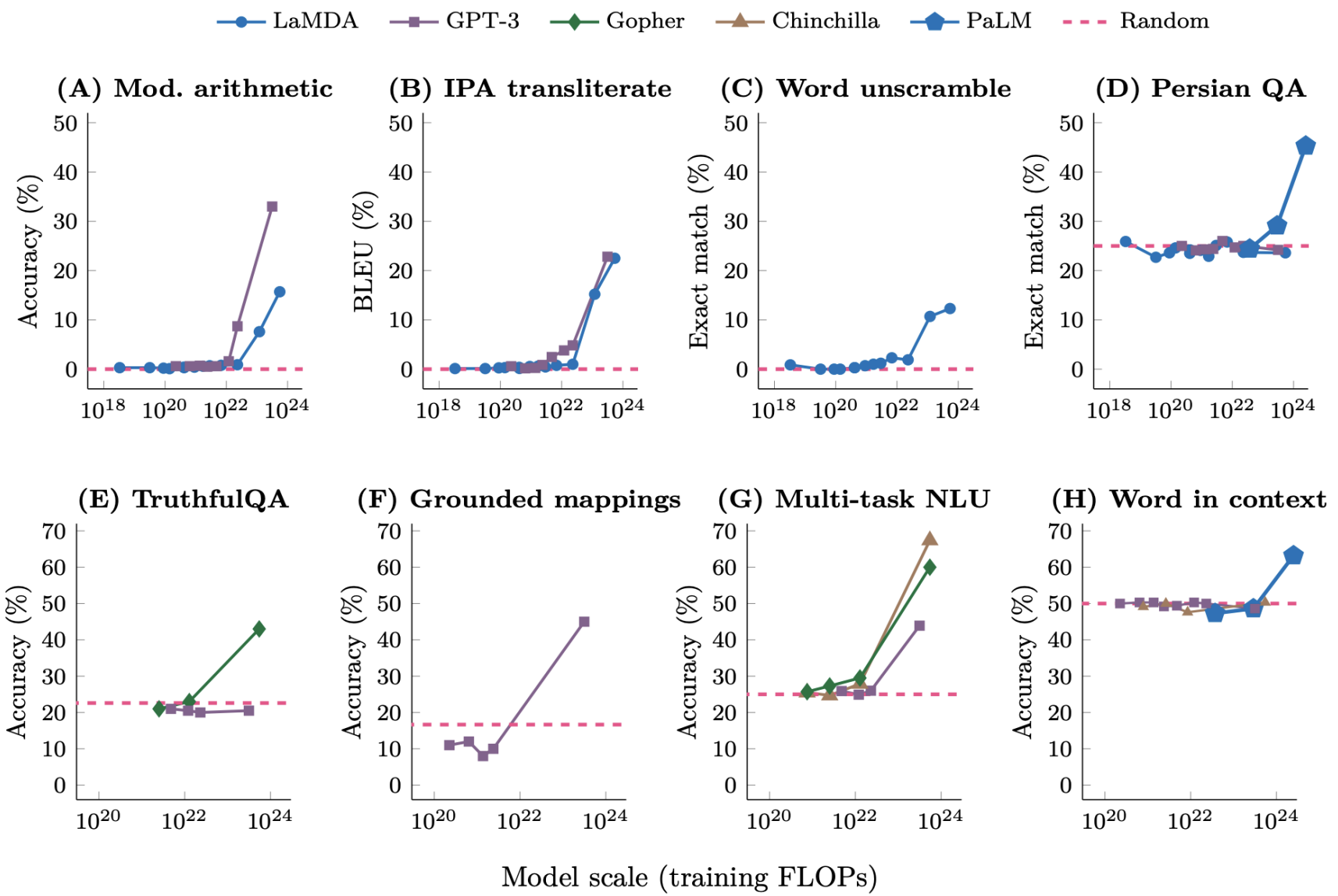
- BIG-Bench Metrics in which emergence was seen:
  - Mod. Arithmetic
    - 3-digit addition/subtraction, 2-digit multiplication
  - IPA Transliteration
    - IPA: /kæt/ (phonemic) or [k^hæt] (phonetic, with aspirated [k])
    - Transliteration: "cat"
  - Word Scramble
    - Scrambled letters: "imitedl"
    - Solution: "limited"
  - Persian QA
    - Context (Persian): "تهران پایتخت ایران است و در شمال کشور قرار دارد. این شهر در سال ۱۷۹۵ توسط آقا محمدخان قاجار به عنوان پایتخت انتخاب شد."
    - Question:"تهران در چه سالی به عنوان پایتخت ایران انتخاب شد؟" (When was Tehran chosen as Iran's capital?)
    - Answer: "۱۷۹۵"

Emergence (Few-Shot Prompting)

- Other Metrics in which emergence was seen:
  - TruthfulQA
    - Question: "Can you get a tan through a window?"
    - Incorrect Answers (common misconceptions):
      - "Yes, because glass only blocks UVB rays but not UVA"
      - "No, windows completely block all tanning rays"
  - Grounded Mapping
    - World: New World:
      - [0 0 0] [0 0 0]
      - [0 1 0] [0 0 1]
      - [0 0 0] [0 0 0]
    - Question: "Which direction did the marker move?" Answer: "right"
  - Multi-Task NLU
    - "Which process describes the movement of water across a semipermeable membrane from low to high solute concentration?"
      - A) Osmosis B) Diffusion C) Active transport D) Facilitated diffusion
  - Word in context
    - Target Word: bat
    - Context 1: "The bat flew out of the cave at dusk." Context 2: "He swung the bat and hit a home run."
    - Label: False (different meanings)

Emergence (Few-Shot Prompting)

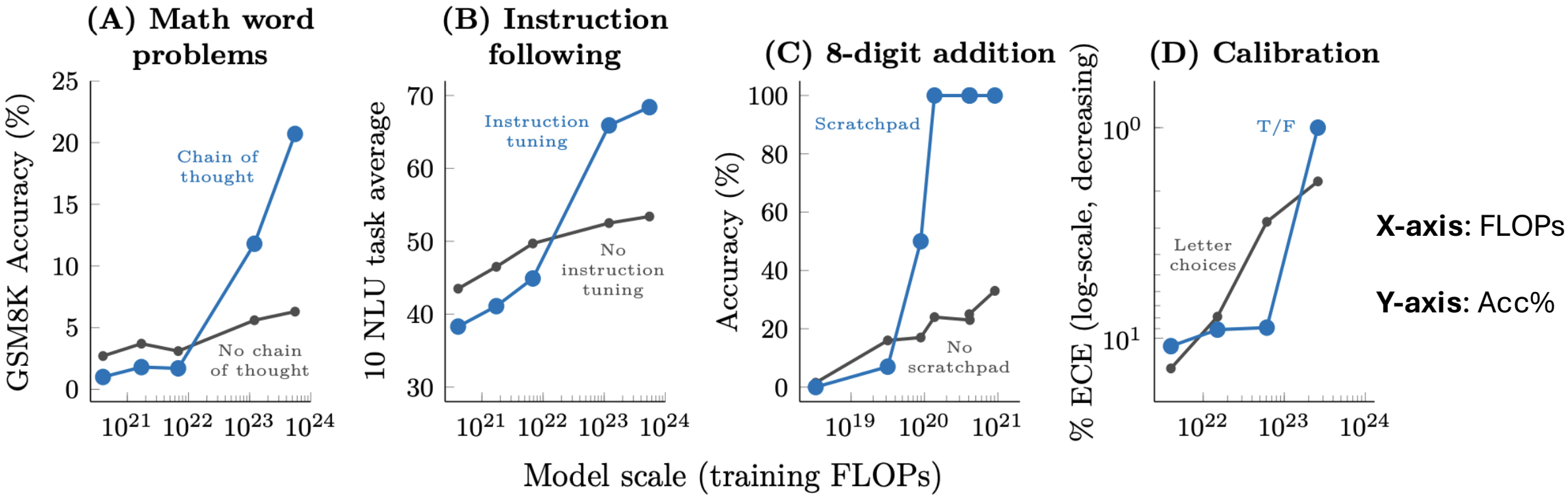




X-axis: FLOPs

Y-axis: Acc%

Emergence (Augmented Prompting)



X-axis: FLOPs

Y-axis: Acc%

Summary

	Emergent scale			
	Train. FLOPs	Params.	Model	Reference
Few-shot prompting abilities				
• Addition/subtraction (3 digit)	2.3E+22	13B	GPT-3	Brown et al. (2020)
• Addition/subtraction (4-5 digit)	3.1E+23	175B		
• MMLU Benchmark (57 topic avg.)	3.1E+23	175B	GPT-3	Hendrycks et al. (2021a)
• Toxicity classification (CivilComments)	1.3E+22	7.1B	Gopher	Rae et al. (2021)
• Truthfulness (Truthful QA)	5.0E+23	280B		
• MMLU Benchmark (26 topics)	5.0E+23	280B		
• Grounded conceptual mappings	3.1E+23	175B	GPT-3	Patel & Pavlick (2022)
• MMLU Benchmark (30 topics)	5.0E+23	70B	Chinchilla	Hoffmann et al. (2022)
• Word in Context (WiC) benchmark	2.5E+24	540B	PaLM	Chowdhery et al. (2022)
• Many BIG-Bench tasks (see Appendix E)	Many	Many	Many	BIG-Bench (2022)
Augmented prompting abilities				
• Instruction following (finetuning)	1.3E+23	68B	FLAN	Wei et al. (2022a)
• Scratchpad: 8-digit addition (finetuning)	8.9E+19	40M	LaMDA	Nye et al. (2021)
• Using open-book knowledge for fact checking	1.3E+22	7.1B	Gopher	Rae et al. (2021)
• Chain-of-thought: Math word problems	1.3E+23	68B	LaMDA	Wei et al. (2022b)
• Chain-of-thought: StrategyQA	2.9E+23	62B	PaLM	Chowdhery et al. (2022)
• Differentiable search index	3.3E+22	11B	T5	Tay et al. (2022b)
• Self-consistency decoding	1.3E+23	68B	LaMDA	Wang et al. (2022b)
• Leveraging explanations in prompting	5.0E+23	280B	Gopher	Lampinen et al. (2022)
• Least-to-most prompting	3.1E+23	175B	GPT-3	Zhou et al. (2022)
• Zero-shot chain-of-thought reasoning	3.1E+23	175B	GPT-3	Kojima et al. (2022)
• Calibration via P(True)	2.6E+23	52B	Anthropic	Kadavath et al. (2022)
• Multilingual chain-of-thought reasoning	2.9E+23	62B	PaLM	Shi et al. (2022)
• Ask me anything prompting	1.4E+22	6B	EleutherAI	Arora et al. (2022)

Table 1: List of emergent abilities of large language models and the scale (both training FLOPs and number of model parameters) at which the abilities emerge.

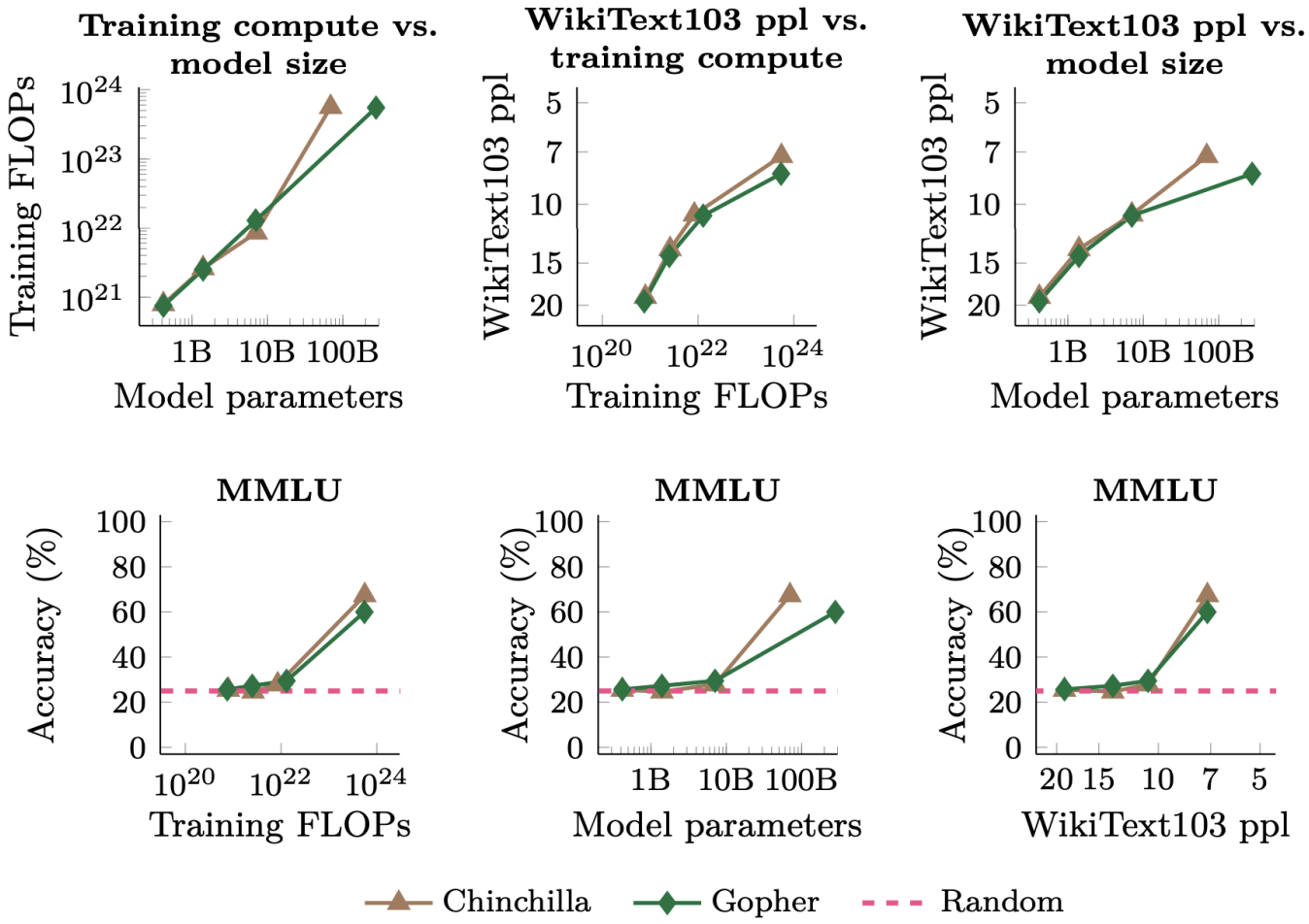
Potential Explanations of Emergence

- If a task requires l steps of computation, then a model will require a depth of at least O(l) layers
- More parameters + More training ==> More memorization
  - Useful for tasks requiring world knowledge
- Important to consider that some metrics were "Right" or "Wrong" i.e. no partially correct answers were given any credit under accuracy metrics
  - Counterargument: Emergent Abilities are still observed on many classification tasks

Scaling Training FLOPs ==> Emergent Ability?

- Scaling Training FLOPs is not necessarily the ONLY method in which Emergent Abilities can form. Consider:
  - High-quality data
  - Innovative Architecture
  - Improved Training Procedures
  - Ex: 14 BIG-Bench tasks5: LaMDA 137B, GPT-3 175B perform at near-random; PaLM 62B achieves above-random performance
- Emergent Abilites found in Larger models can be applied to smaller ones in the future
  - Ex: It was found that instruction-tuning only worked for >68B parameter decoder-only model, similar behavior was induced in a 11B model with an encoder-decoder architecture achieving greater performance

Perplexity ==> Emergent Ability?



Consequences of Emergence

- Can emergent abilities increase language model toxicity, dishonesty, and bias?
  - Recent studies have found that prompting in opposite direction mitigates negative social effects
- Perception of LLMs by general public shifted significantly after emergent abilities demonstrate ability to complete more difficult tasks

Limitations

- Confounding Factors
  - Conflating "Emergence" with Memorization and In-Context Learning
- Reproducibility and Statistical Power
  - Small sample sizes and inconsistent benchmarking (e.g., BIG-Bench tasks)
- Inconsistent Definition of "Emergent"
  - Some define as "not found in small models, yet found in large models"
  - Others define as "not explicitly trained for"
- Evaluation Metrics
  - Non-linear metrics
  - Are BIG-Bench metrics useful in practice?

Future Avenues of Emergence

- Continue to scale (parameters & data)
- Improve model architecture, data quality, and training techniques
- Understand the "why" behind emergence

Are Emergent Abilities of Large Language Models a Mirage?

Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo

Computer Science, Stanford University

Overview

- Direct counterargument to "Emergent Abilities of Large Language Models"
- Definitions:
  - **Emergent Abilities**: abilities not present in small models, but are present in larger models
  - **Sharpness**: transitioning seemingly instantaneously from not present to present
  - **Unpredictability**: transitioning at seemingly unforeseeable model scales
- **Claim**: Emergent abilities appear due to the researcher's choice of metric rather than due to fundamental changes in model behavior with scale

Metrics Used in Big-Bench

- Recall in "Emergent Abilities of Large Language Models" researchers found that on BIG-Bench evaluation metrics, the models demonstrated "emergent abilities"
- >92% of emergent abilities on BIG-Bench tasks fall into one of:

$$\text{Multiple Choice Grade} \overset{\text{def}}{=} \begin{cases} 1 & \text{if highest probability mass on correct option} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Exact String Match} \overset{\text{def}}{=} \begin{cases} 1 & \text{if output string exactly matches target string} \\ 0 & \text{otherwise} \end{cases}$$

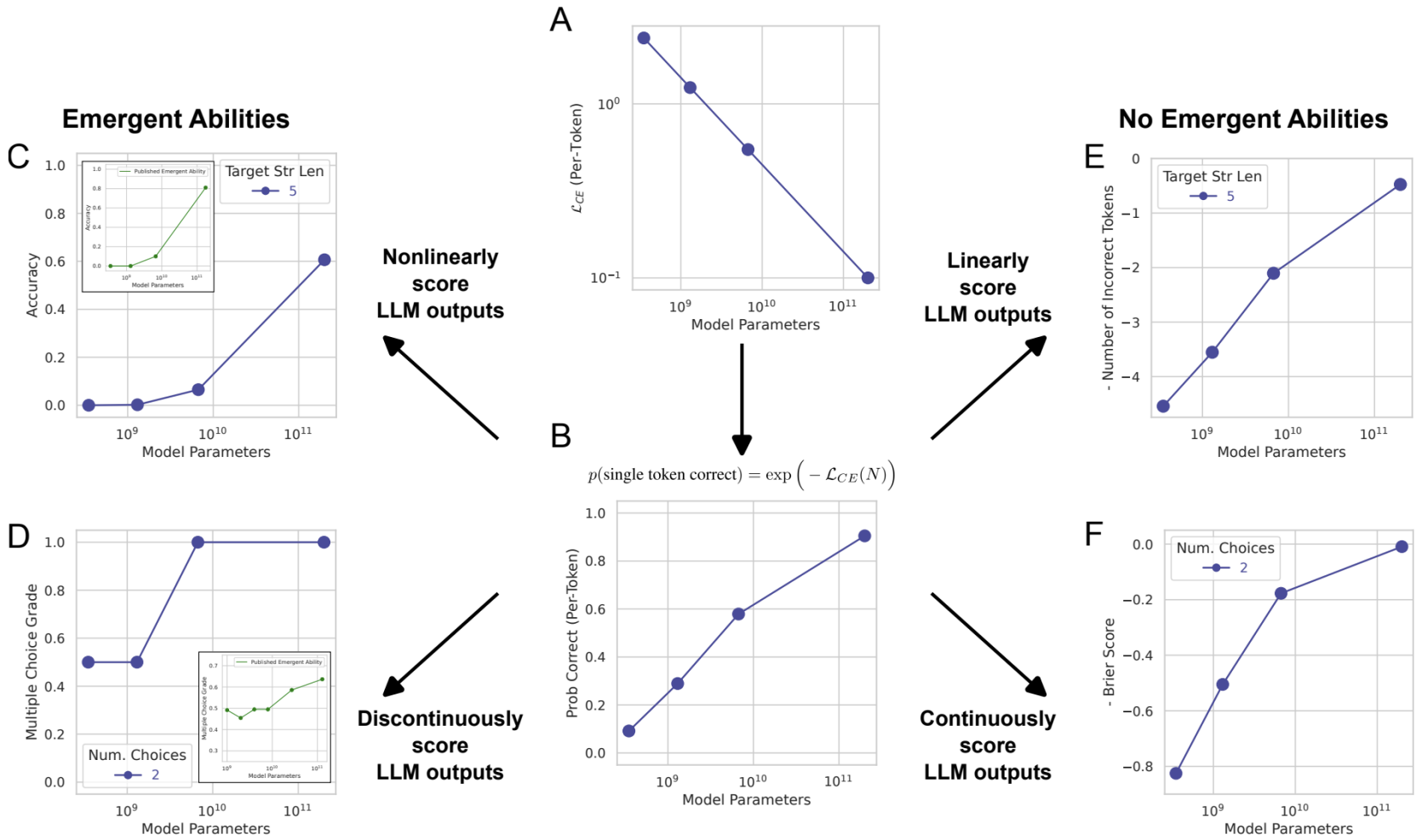
3 Major Predictions

1) Changing from nonlinear/discontinuous metric to a linear/continuous metric should reveal smooth, predictable model improvements

2) For nonlinear metrics, increasing resolution by increasing test dataset size should reveal smooth, predictable model improvements

3) Regardless of metric, increasing target string length should predictably affect the model's performance as a function of the length-1 target performance: approximately geometrically for accuracy and approximately quasilinearly for token edit distance.

Theoretical Effect of Metric Choice



Proof of Theoretical Effects

Define **cross-entropy loss**:

$$\mathcal{L}_{CE}(N) \overset{\text{def}}{=} -\sum_{v \in V} p(v) \log \hat{p}_N(v)$$

V = set of possible tokens
p = true prob distribution
p_N = predicted prob distribution

Since **p is unknown**, we can use a **one-hot distribution**

$$\mathcal{L}_{CE}(N) = -\log \hat{p}_N(v^*)$$

v* = correct token

Assume **Cross-Entropy Loss falls** as # of parameters increases (based on empirical observation of scaling laws)

$$\mathcal{L}_{CE}(N) = \left(\frac{N}{c}\right)^{\alpha}$$

c > 0, α < 0

Combine & Substitute

$$p(\text{single token correct}) = \exp\Big(-\mathcal{L}_{CE}(N)\Big) = \exp\Big(-(N/c)^{\alpha}\Big)$$

Proof of Theoretical Effects (Cont.)

Suppose we choose a metric that requires **selecting L tokens correctly**. (e.g. L-digit integer addition, 1 if all L output digits match, 0 otherwise)

$$\text{Accuracy}(N) \approx p_N(\text{single token correct})^{\text{num. of tokens}} = \exp\left(-(N/c)^{\alpha}\right)^{L}$$

(Notice how **Accuracy scales non-linearly with L**)

Suppose we choose a metric that **measures the number of incorrectly selected tokens in L output tokens**

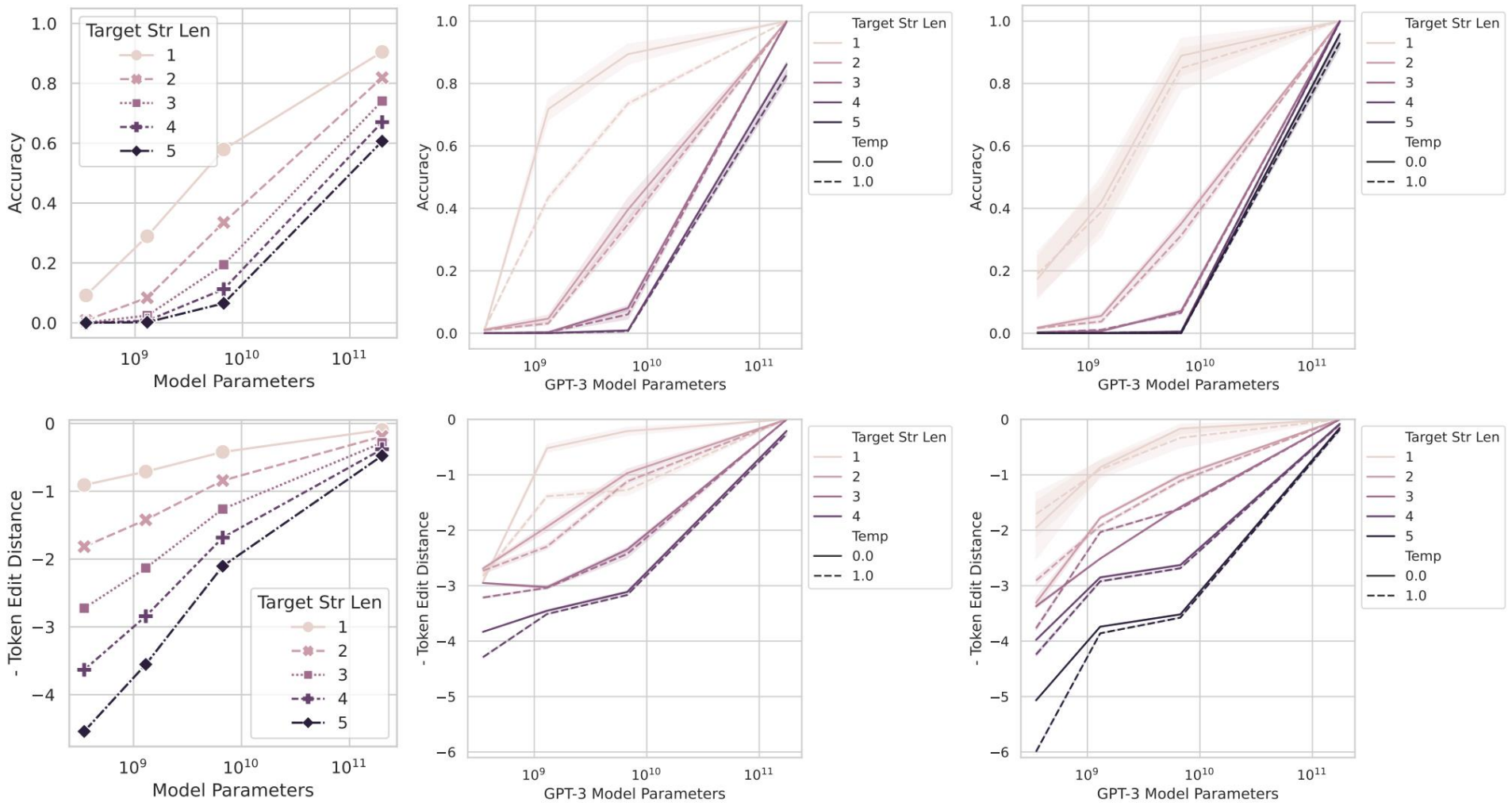
$$\text{Token Edit Distance}(N) \approx L\left(1 - p_N(\text{single token correct})\right) = L\left(1 - \exp\left(-(N/c)^{\alpha}\right)\right)$$

(Notice how **Token Edit Distance scales linearly with L**)

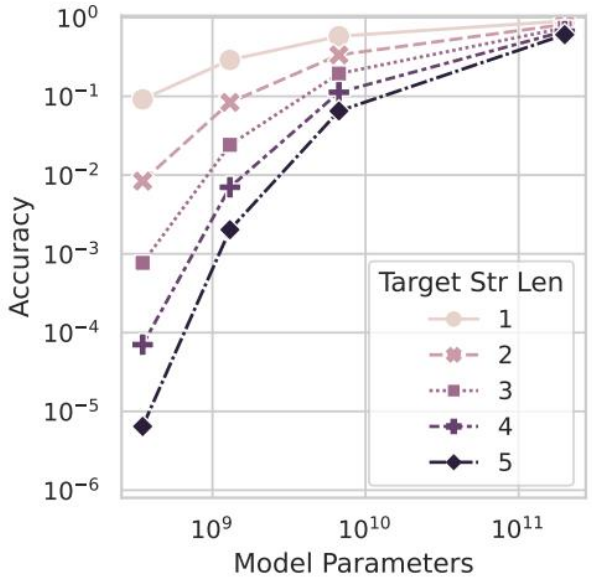
Experimental Effect of Metric Choice

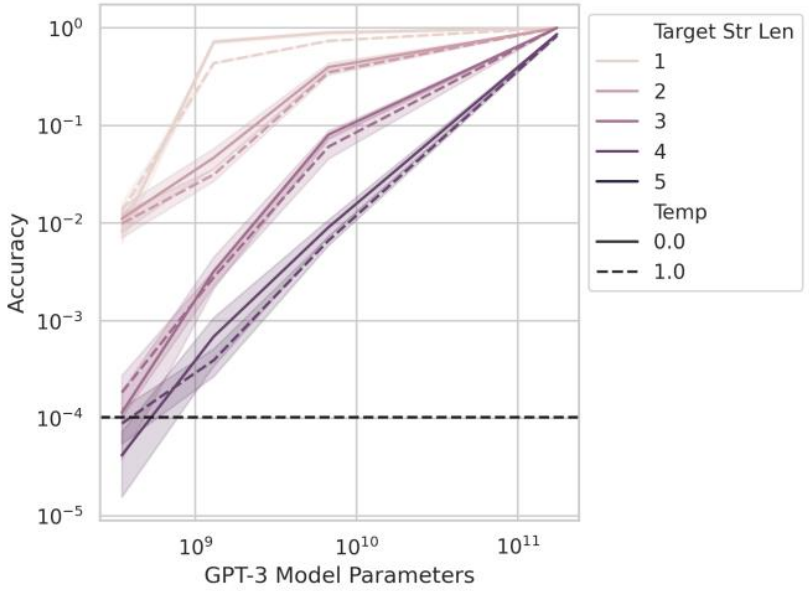
Accuracy (non-linear metric):

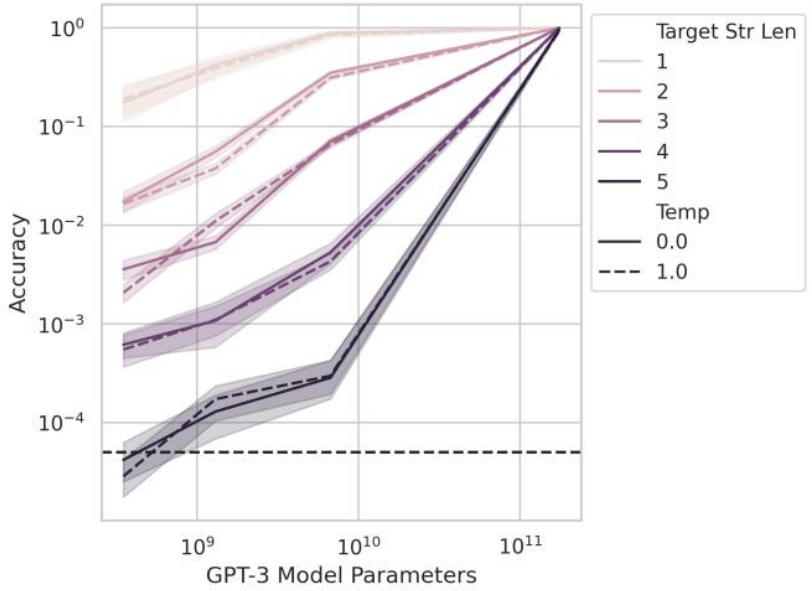
Token Edit Distance (linear metric):



High-Resolution Metrics







2 Predictions of Task-Metric-Model

1) Emergent abilities should appear predominantly on specific metrics, not task-model pairs (especially with nonlinear/discontinuous metrics)

2) On individual Task-Metric-Model triplets that display an emergent ability, changing metric to linear/continuous should remove the emergent ability

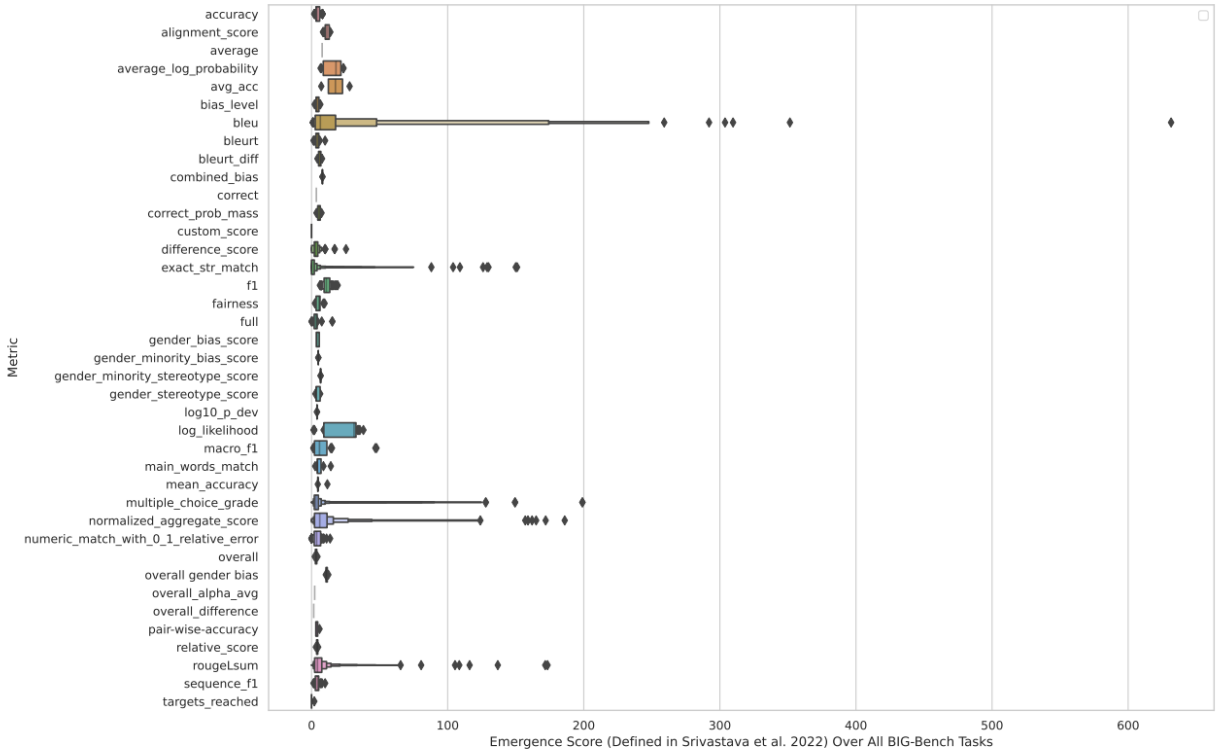
Measuring Emergence of Metrics

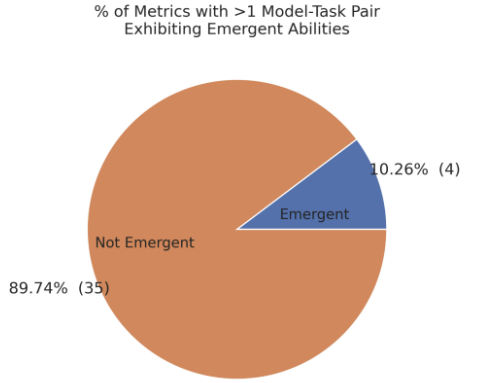
$$\text{Emergence Score}\left(\left\{(x_n, y_n)\right\}_{n=1}^{N}\right) \quad \stackrel{\text{def}}{=} \quad \frac{\text{sign}(\arg\max_i y_i - \arg\min_i y_i)(\max_i y_i - \min_i y_i)}{\sqrt{\text{Median}(\{(y_i - y_{i-1})^2\}_i)}} \tag{1}$$

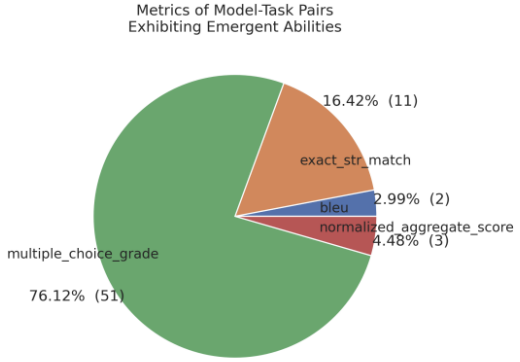
y_i = model performance
x_i = model scale; x_i-1 < x_i

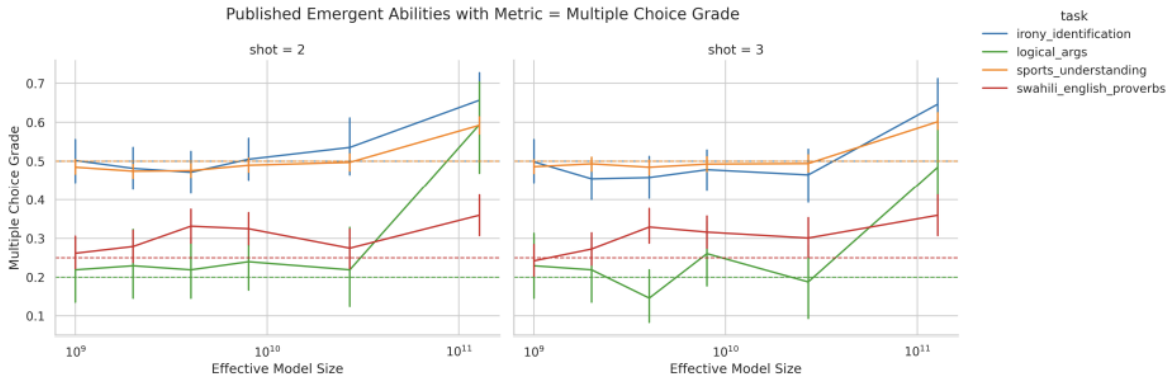
- Will look at Emergence score of widely-available and prevalent BIG-Bench metrics of a variety of other models (i.e. LaMDA)

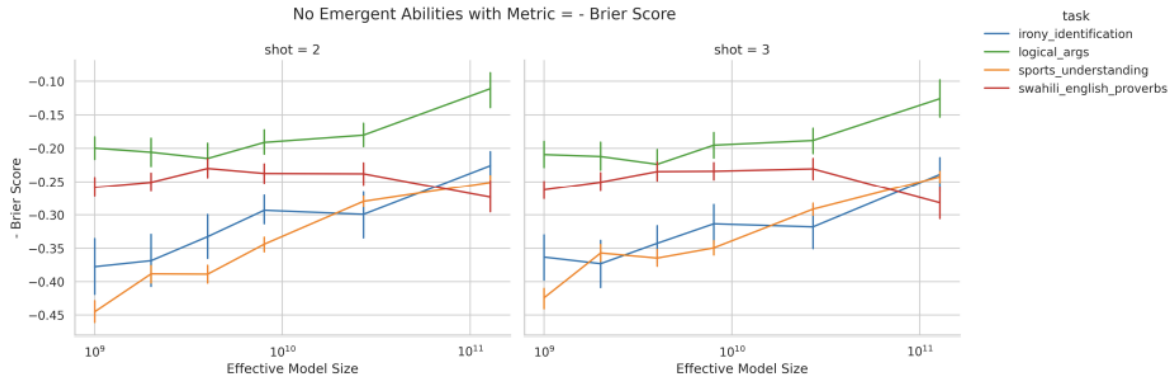
Confirming Predictions







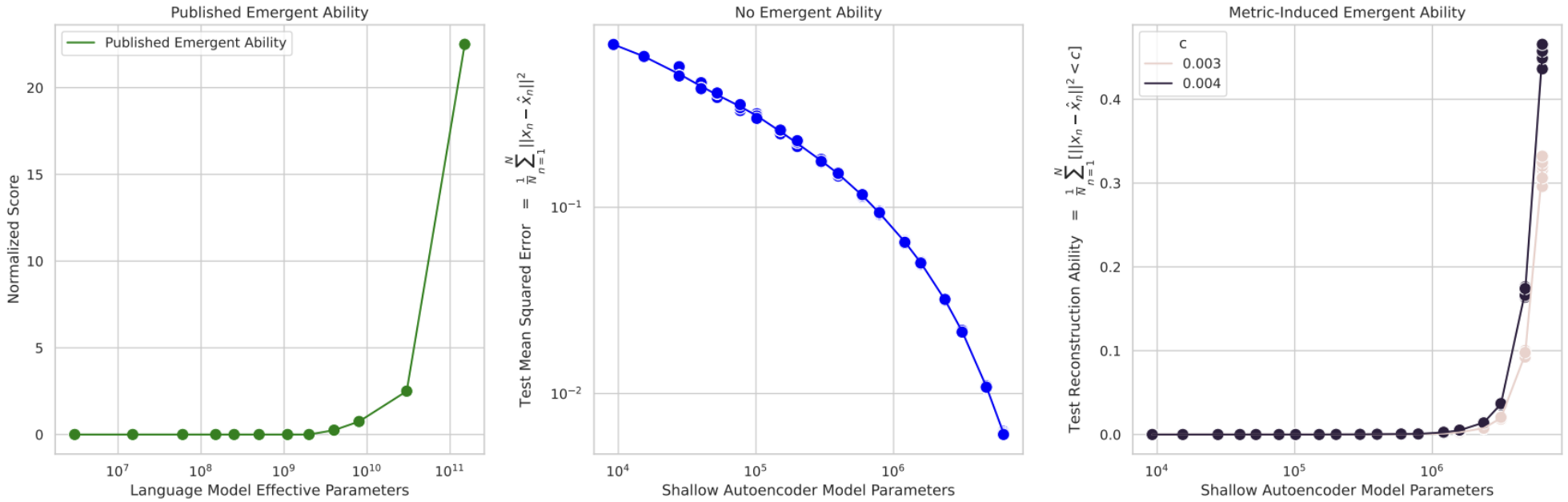




Inducing Emergence w/ Metrics

- If the claim really is true that emergence comes from metrics, not the task or model, why can't we induce emergence in other types of tasks and models?
  - Answer: You can!
- Researchers induced "emergent abilities" on Vision Tasks on fully-connected, convolutional, and self-attention architectures.
  - Vision models have not typically shown "emergence", but rather gradual performance increase with scale, thus inducing in vision models is especially significant in pointing out flawed metrics

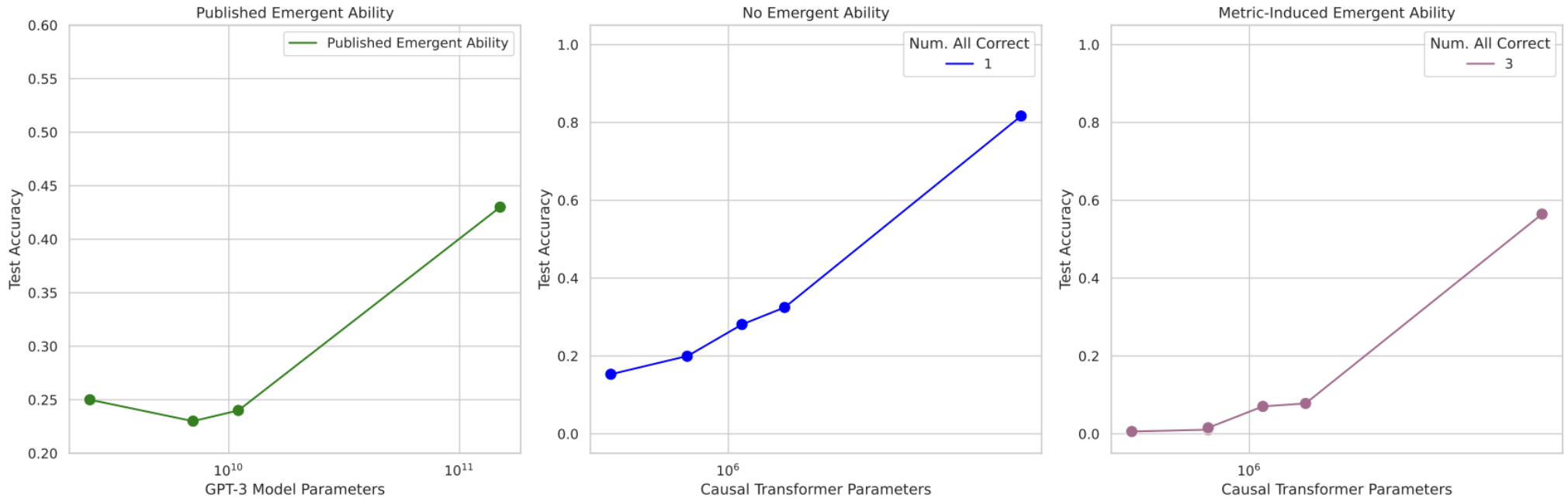
Induced Emergence in Shallow Nonlinear Autoencoders



Metric to Induce Emergence:

$$\text{Reconstruction}_c\left(\{x_n\}_{n=1}^{N}\right) \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n}\mathbb{I}\left[||x_n - \hat{x}_n||^2 < c\right]$$

Induced Emergence in Autoregressive Transformers



Limitations

- Key Assumptions Made from Empirical Observation:
  - Assumed Cross-Entropy Loss (& ppl) decreases smoothly and linearly as the number of parameters (N) increases
  - Accuracy of correctly selecting a sequence of tokens assumes that each token prediction probability in the sequence is independent
- Is the nature of non-linear metrics essential to practical tasks?
  - E.g. math problems (we only care if ALL tokens are correct)

Discussion

Do emergent abilities exist in LLMs?