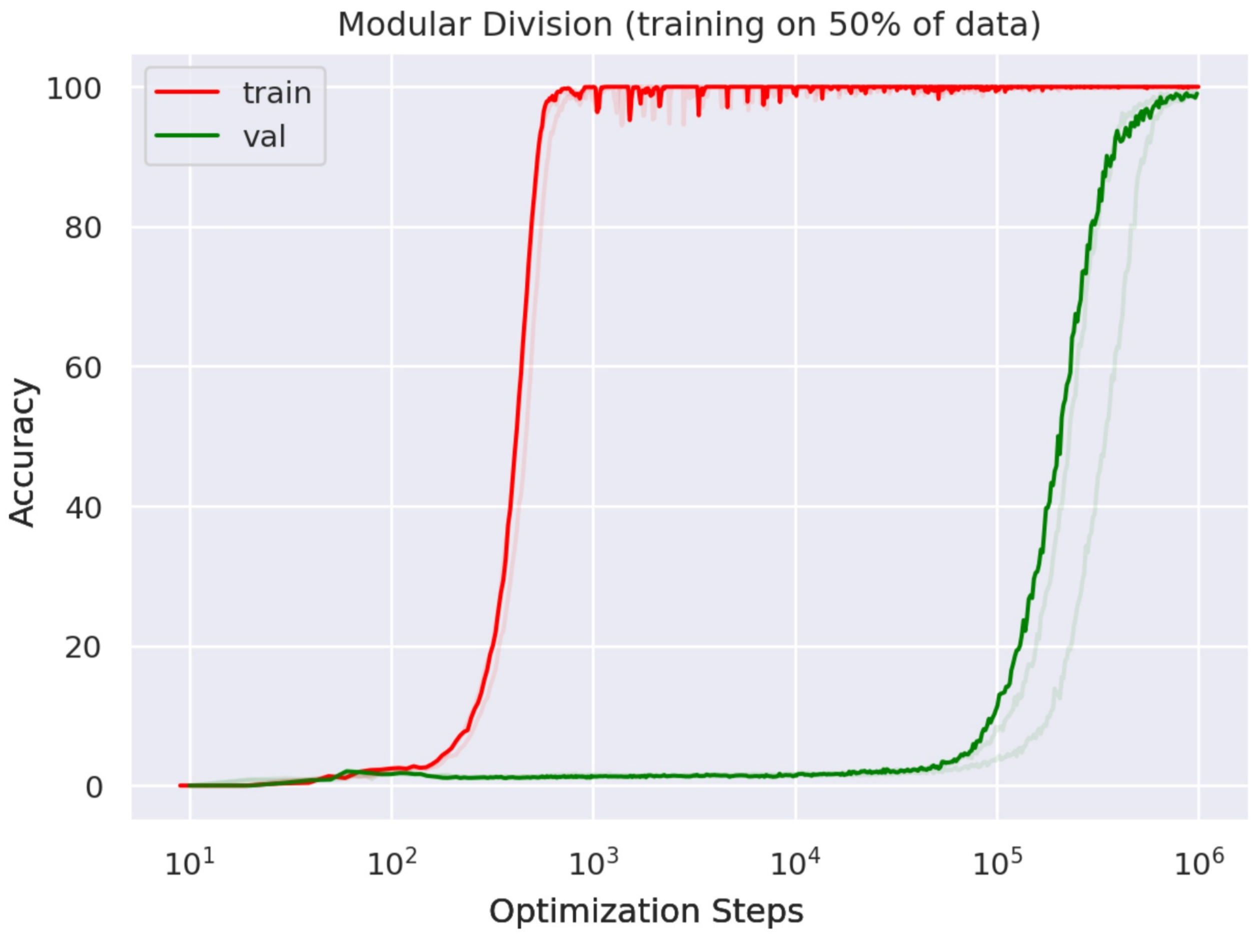
Important Phenomena in LLM

Lei Wu

Outline

- Grokking
- Neural Scaling Law
- Emergence
- In-context Learning

Grokking: Emergence wrt Training Epochs



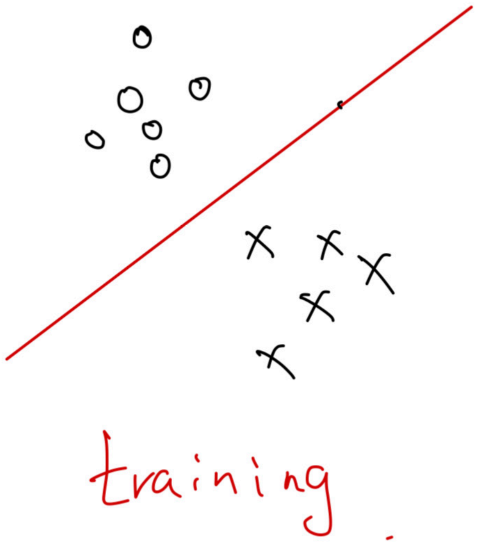
Grokking phenomenon in learning **f(x,y)=(x/y) mod p** (in this figure, p=97).
Similar behaviors also happen for moduar addition, etc.

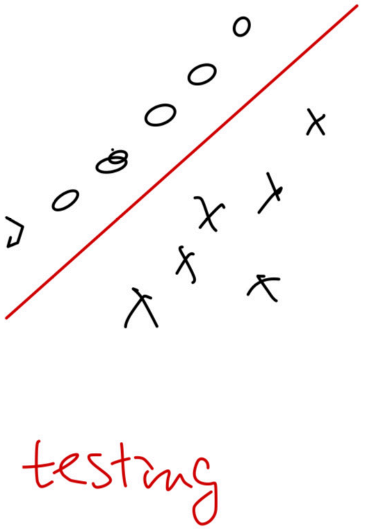
Key Difference:

- In-distribution vs. Out-of-distribution
- Unpredictablity!!
- It happens due to the difference between evaluation metric about the training progress.

The slow progression of implicit bias can induce grokking

- In binary linear classification

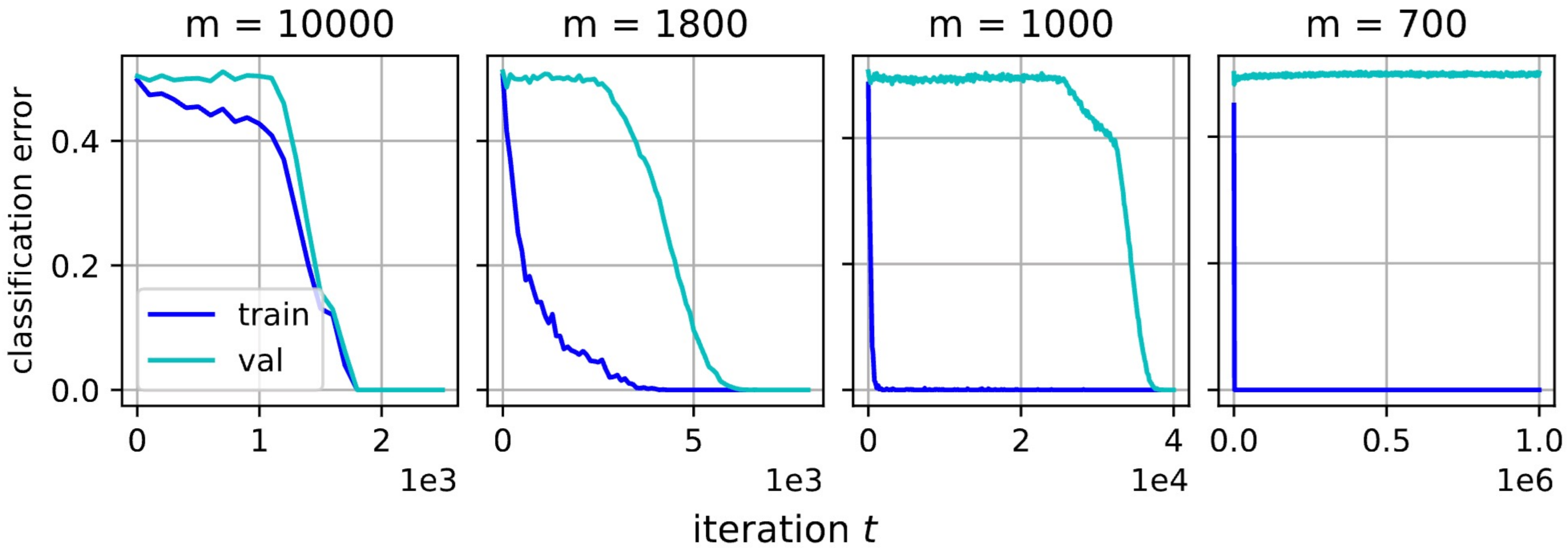




- The same phenomenon can also happen for the noise-driven/ oscillation-driven implicit bias.

Grokking can happen for all problems with statistical-computational gap.

- Memorization is much easier than generalization in terms of time-complexity. In this case, the grokking is a task-specific property (nearly independent of the model and optimizer used)
- Consider the learning of parity (Barak et al., NeurIPS 2022)

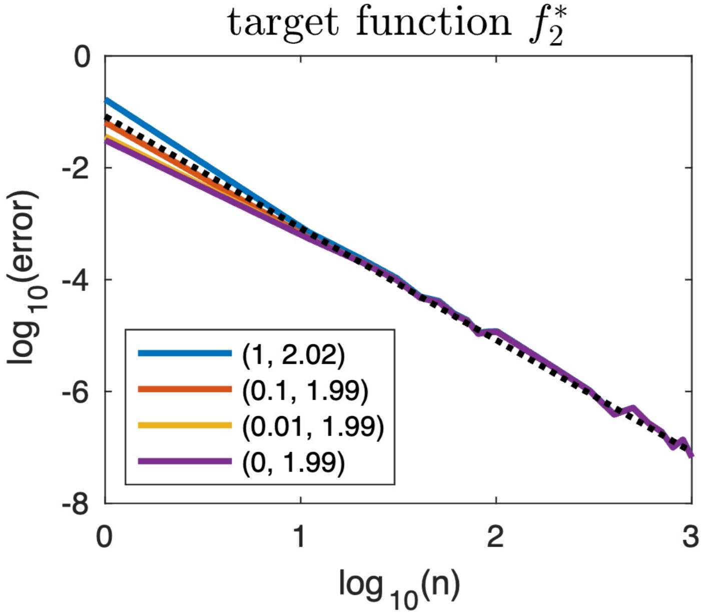


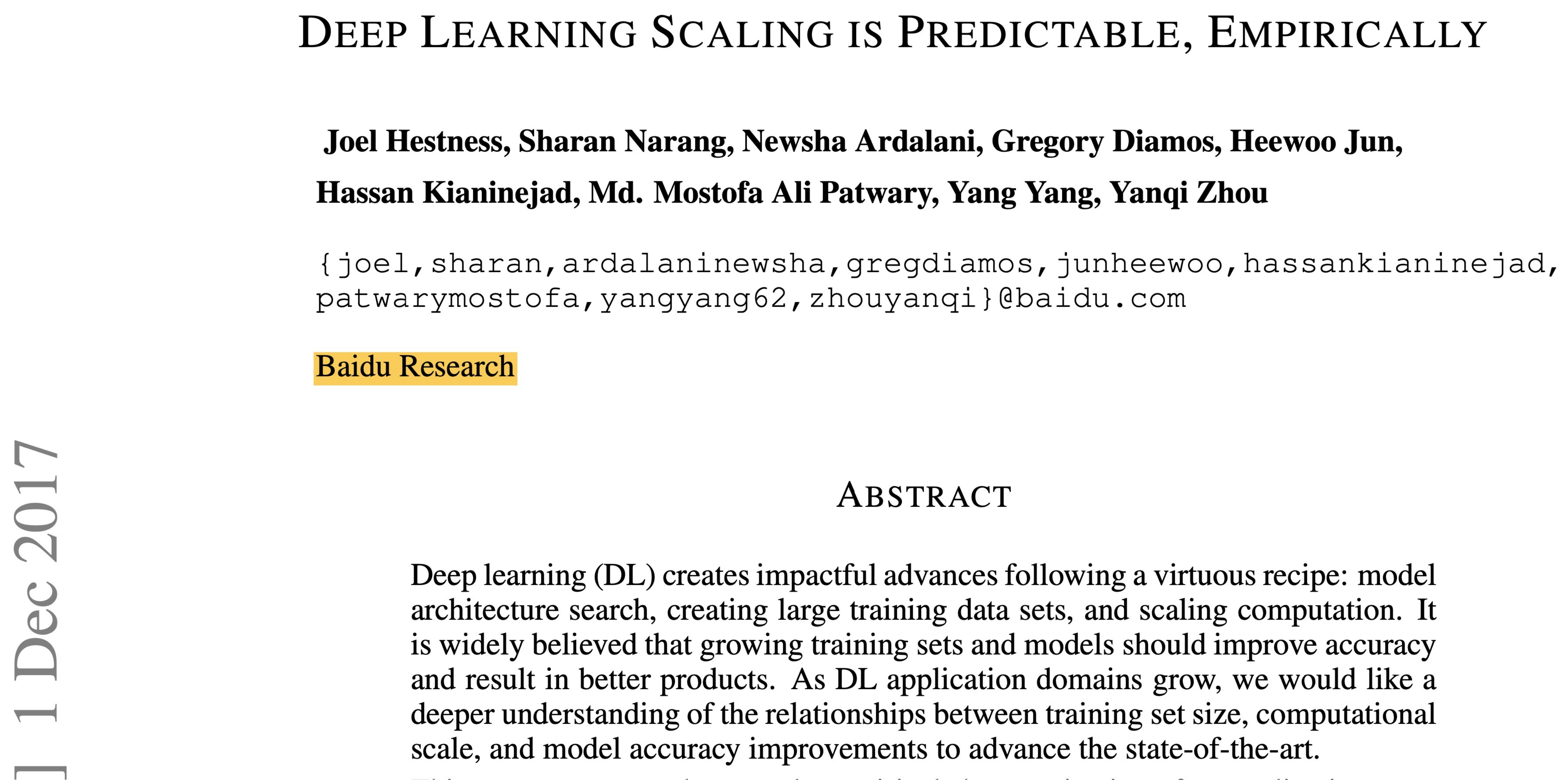
Scaling Law

- The scaling behavior of ML models has been studied for a long time (Seung, et al., 1992)

$$\text{gen-err}(n) \sim an^{-\alpha} + b$$

- In classical ML, we often has $\alpha = 0.5 \text{ or } 1$. For KRR/RF regression, the exponent can be smaller or larger than 0.5, depending on the relative smoothness.





Deep Learning Scaling is Predictable, Empirically

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun,
Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, Yanqi Zhou

{joel,sharan,ardalaninewsha,gregdiamos,junheewoo,hassankianinejad,
patwarymostofa,yangyang62,zhouyanqi}@baidu.com

Baidu Research

Abstract

Deep learning (DL) creates impactful advances following a virtuous recipe: model architecture search, creating large training data sets, and scaling computation. It is widely believed that growing training sets and models should improve accuracy and result in better products. As DL application domains grow, we would like a deeper understanding of the relationships between training set size, computational scale, and model accuracy improvements to advance the state-of-the-art.

Neural Scaling Law (Hestness, et al., 2017)

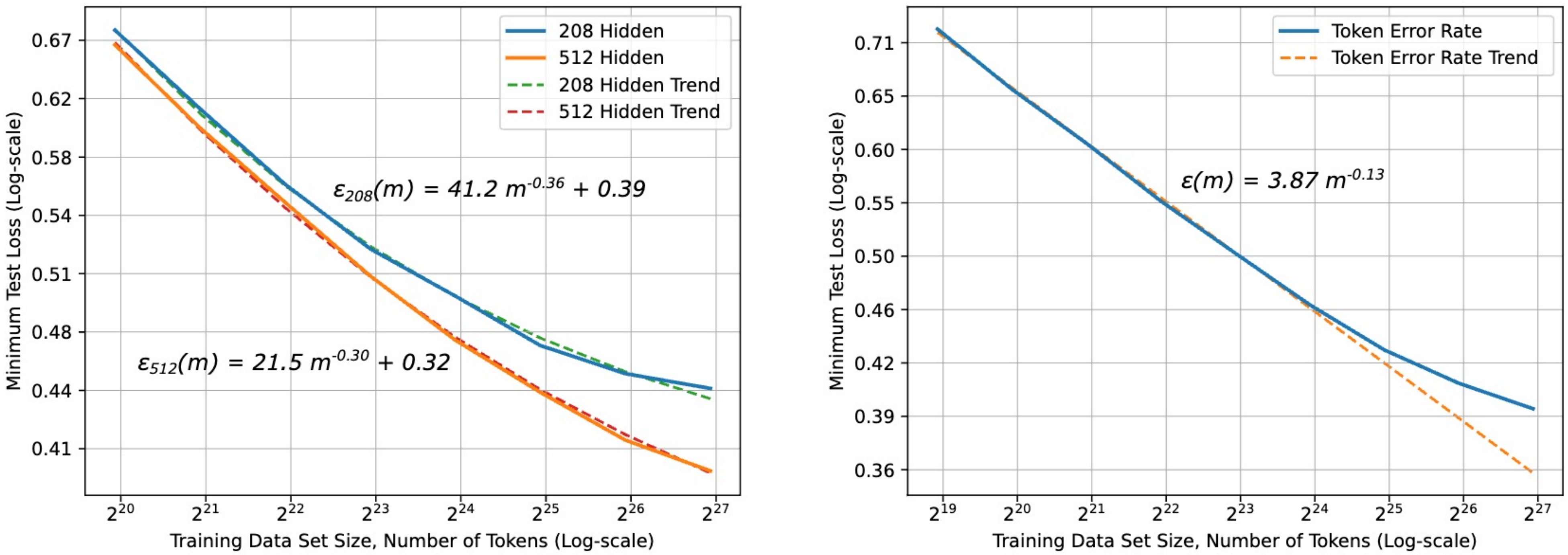
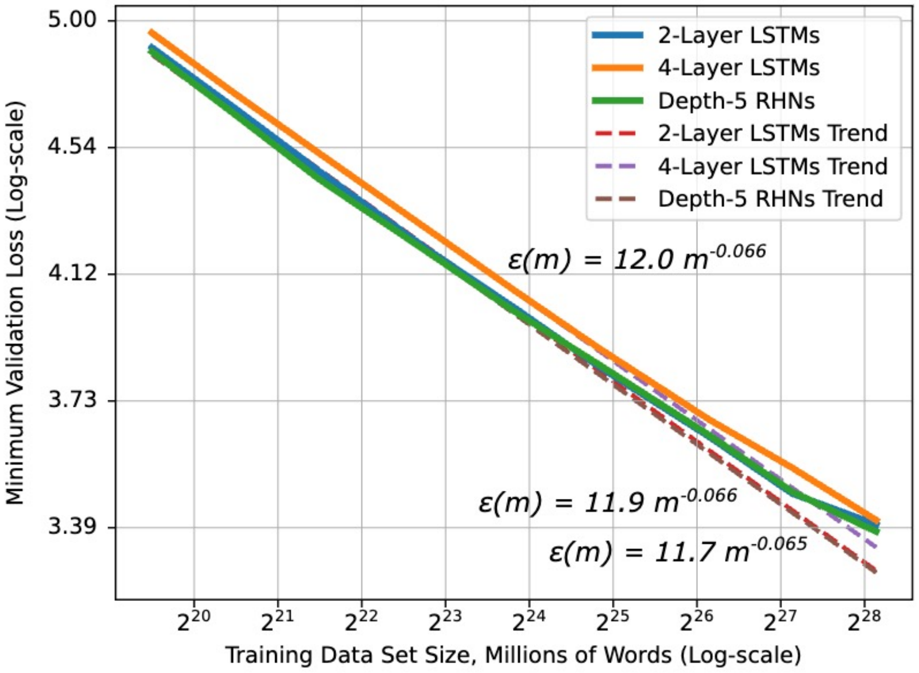
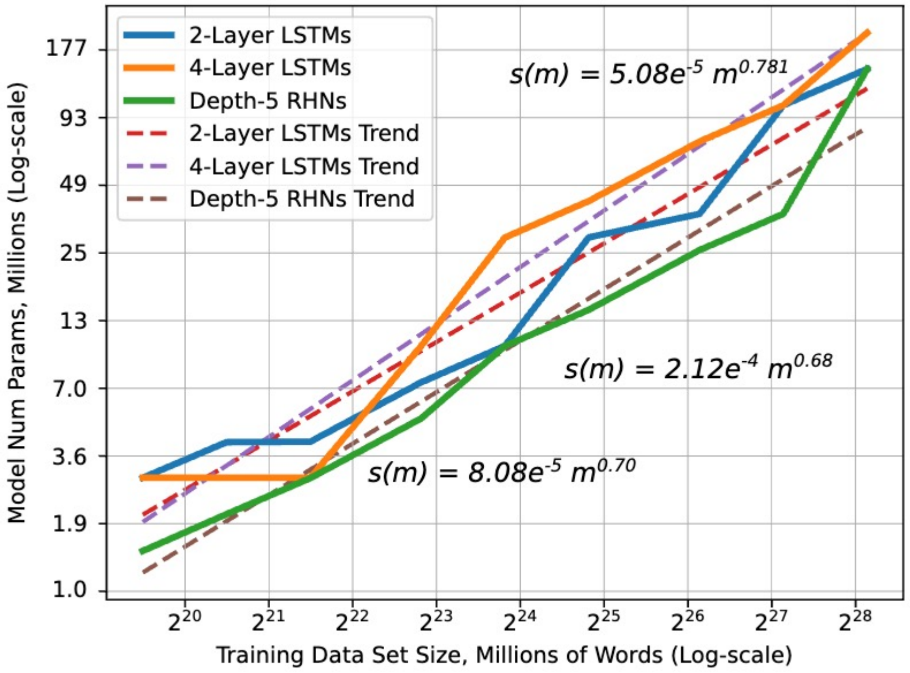


Figure 1: Neural machine translation learning curves. Left: the learning curves for separate models follow $\varepsilon(m) = \alpha m^{\beta_g} + \gamma$. Right: composite learning curve of best-fit model at each data set size.

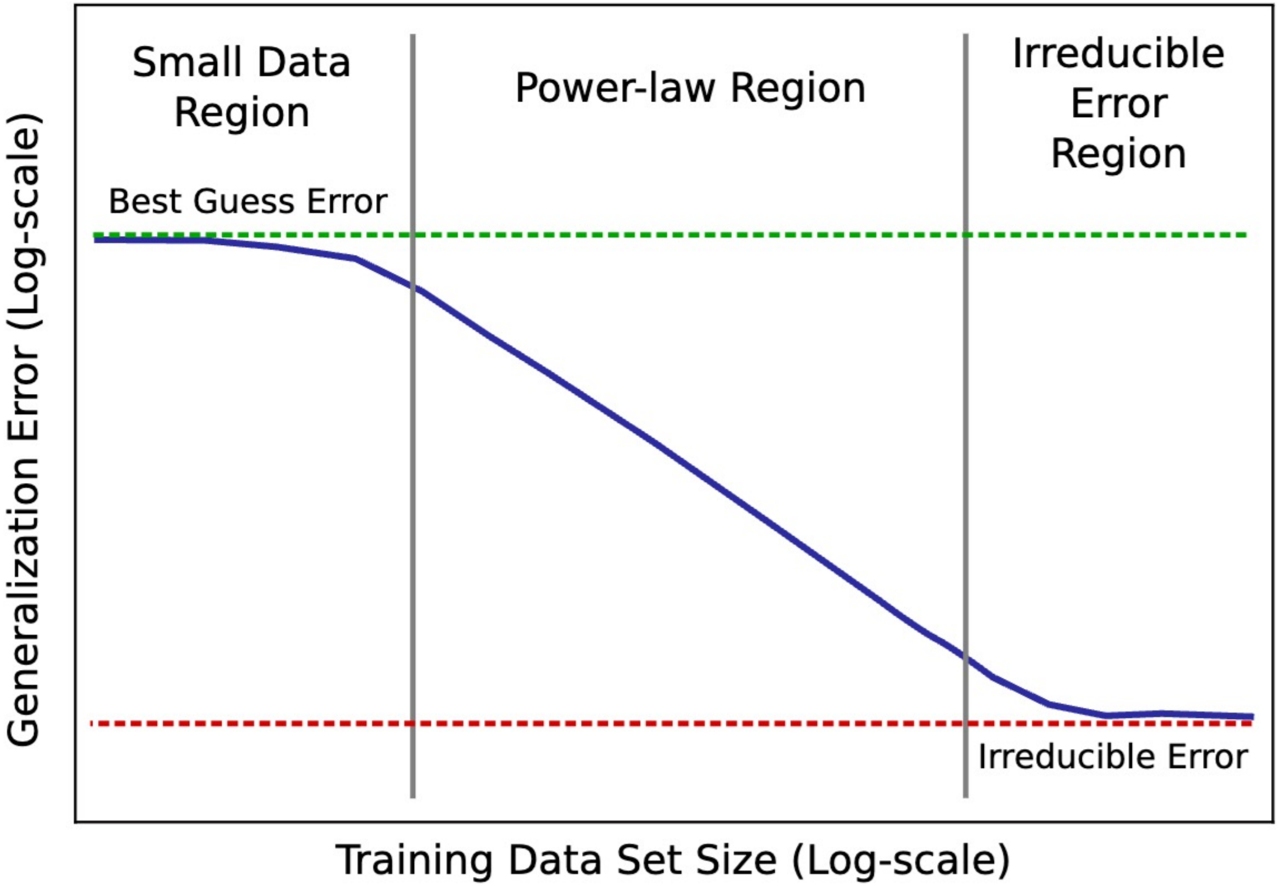
Language modeling with RNN





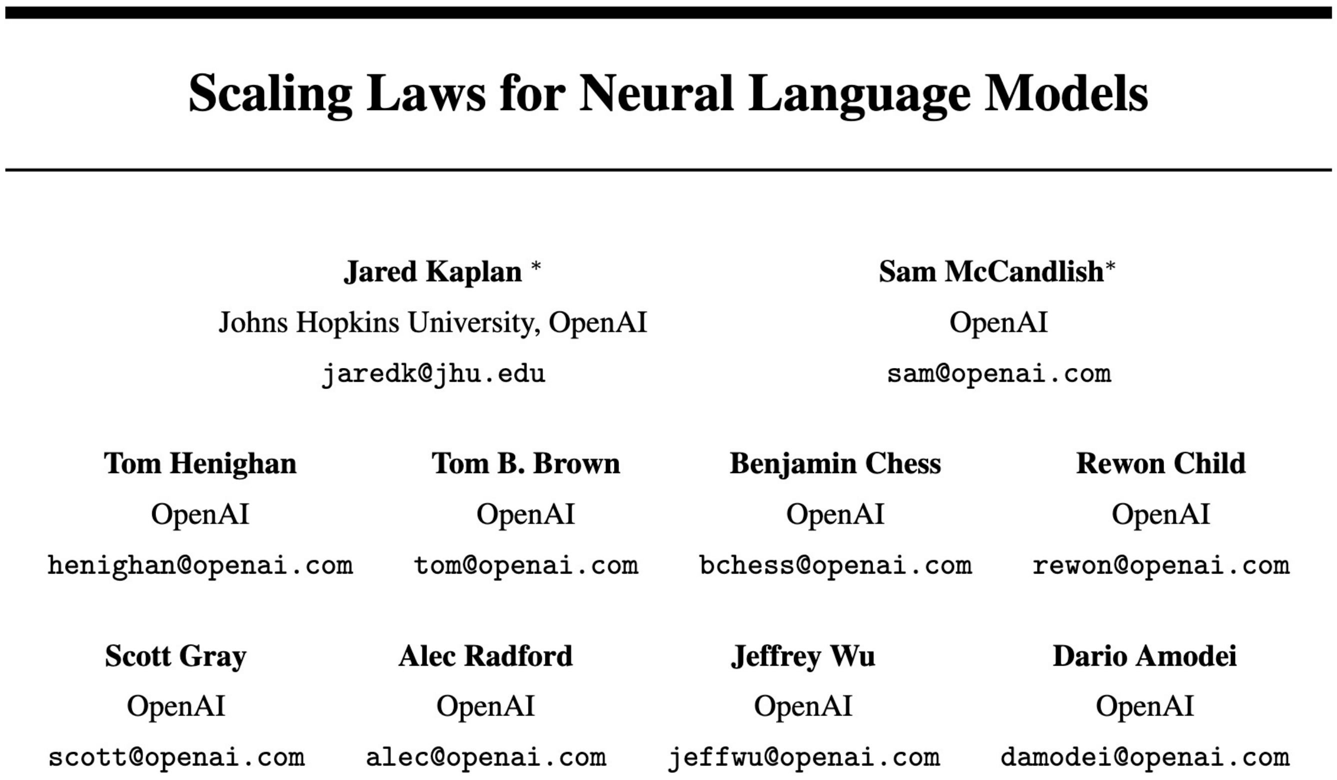
- **The scaling law is nearly independent of the model depth and arch.**

A sketch of power-law learning curves



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

Refined analysis of scaling law



Scaling Laws for Neural Language Models

Jared Kaplan *
Johns Hopkins University, OpenAI
jaredk@jhu.edu

Sam McCandlish*
OpenAI
sam@openai.com

Tom Henighan
OpenAI
henighan@openai.com

Tom B. Brown
OpenAI
tom@openai.com

Benjamin Chess
OpenAI
bchess@openai.com

Rewon Child
OpenAI
rewon@openai.com

Scott Gray
OpenAI
scott@openai.com

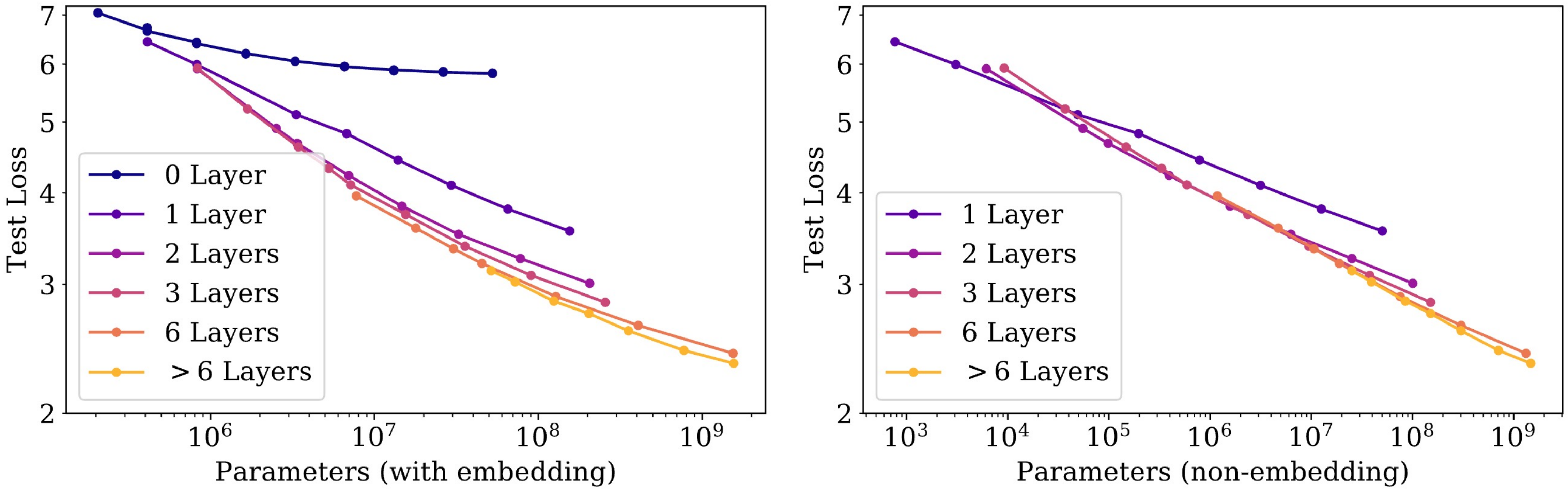
Alec Radford
OpenAI
alec@openai.com

Jeffrey Wu
OpenAI
jeffwu@openai.com

Dario Amodei
OpenAI
damodei@openai.com

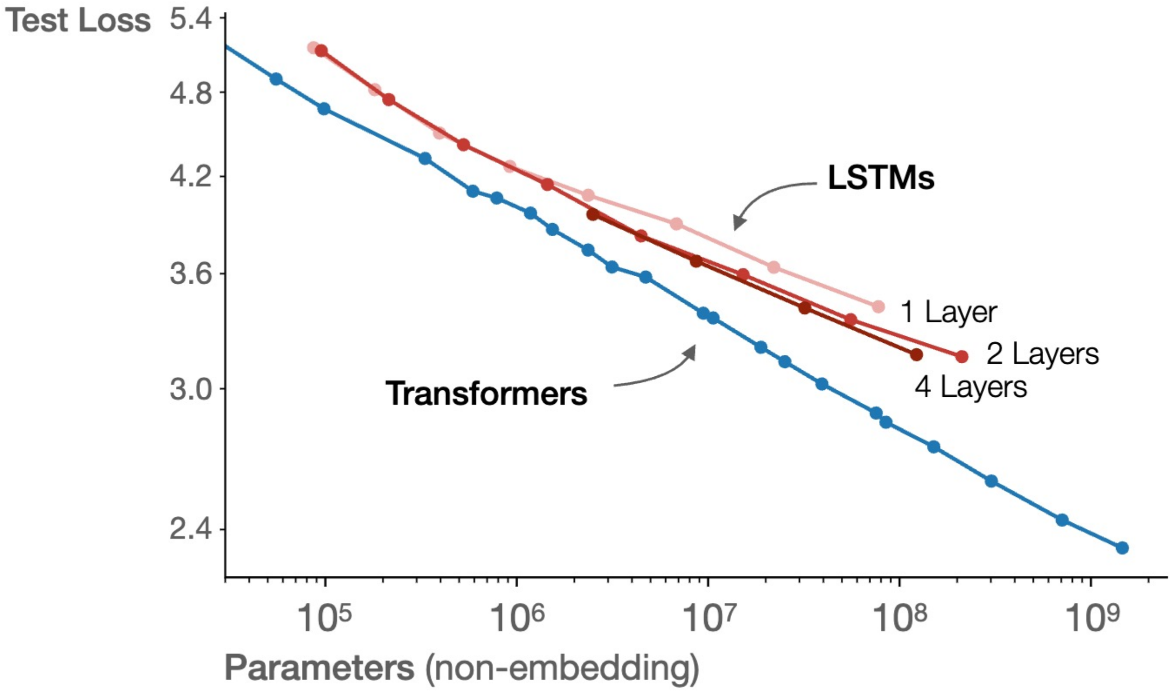
- Decoder-only Transformer
- Training: Adam for a fixed 250k iterations, batch size 512, context length 1024 tokens. 3k warmup+ cosine LR decay
- Datasets: Webtext2

Non-embedding parameters matter

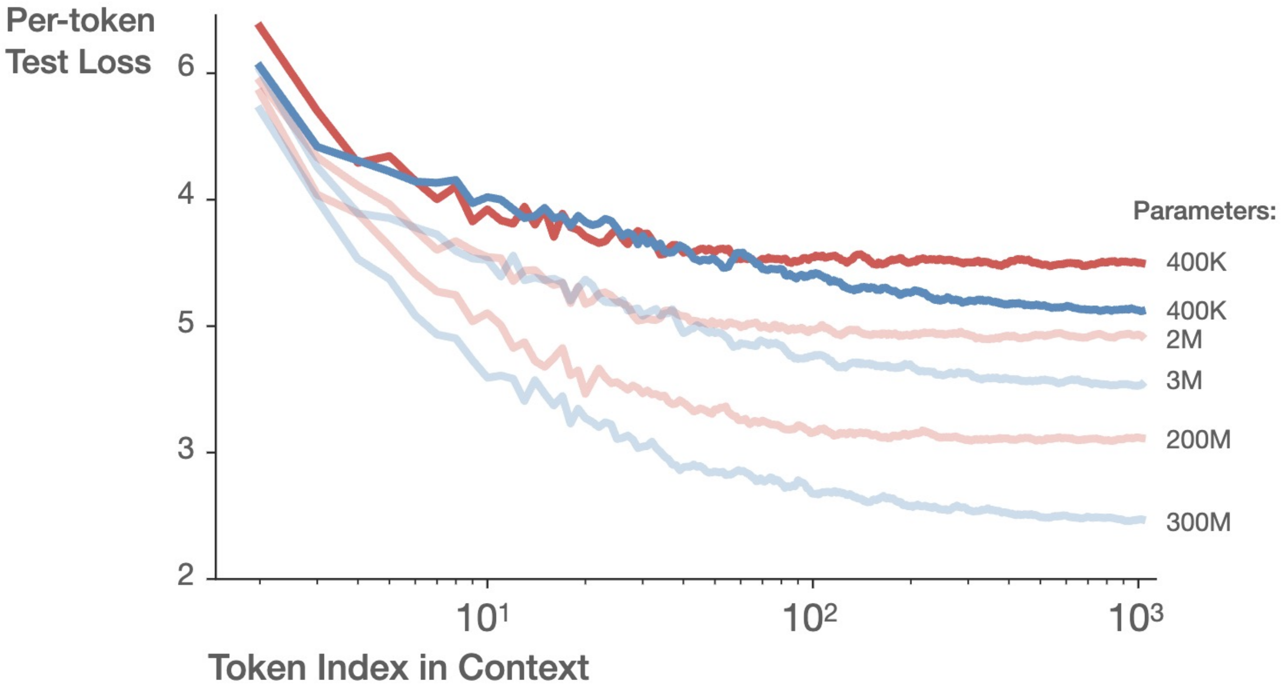


- Removing the embedding parameters yields clear power-law
- Scaling law hold well as long as the depth is not extremely small or large.
- The exponent (slopes in the above figure) depends weakly on the model shape.

Transformer vs. LSTM

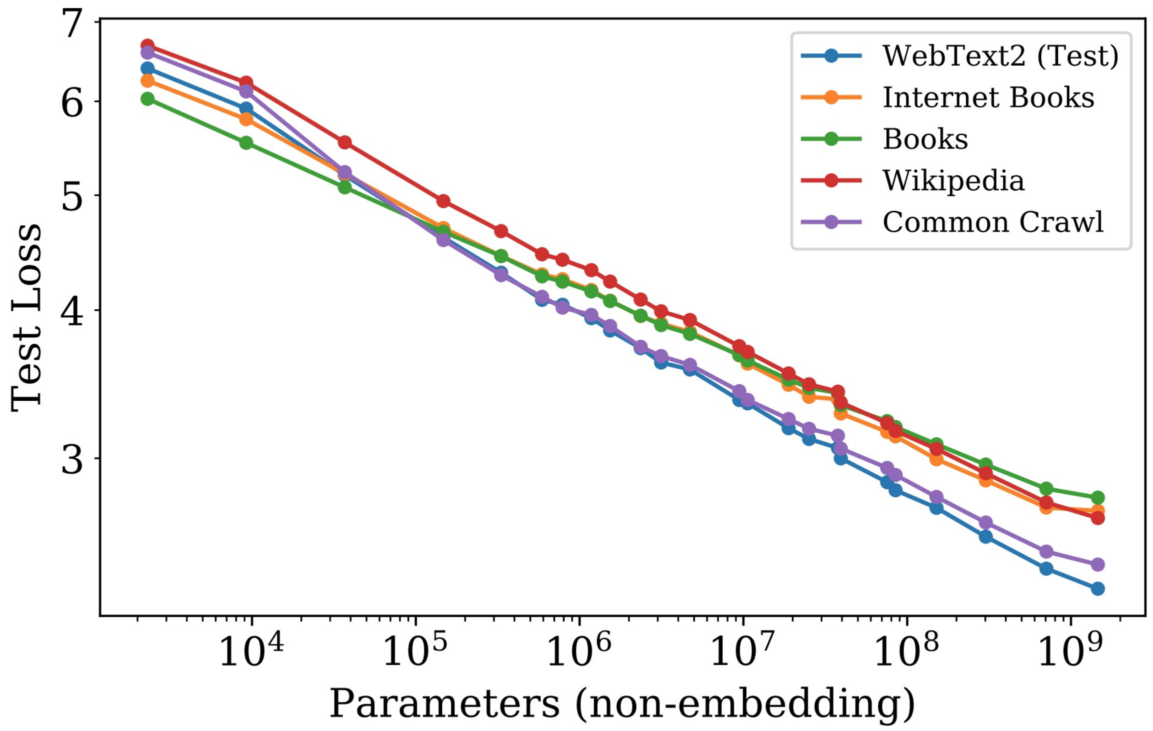


Transformers asymptotically outperform LSTMs due to improved use of long contexts



LSTM plateaus after <100 tokens
Transformer improves through the whole context

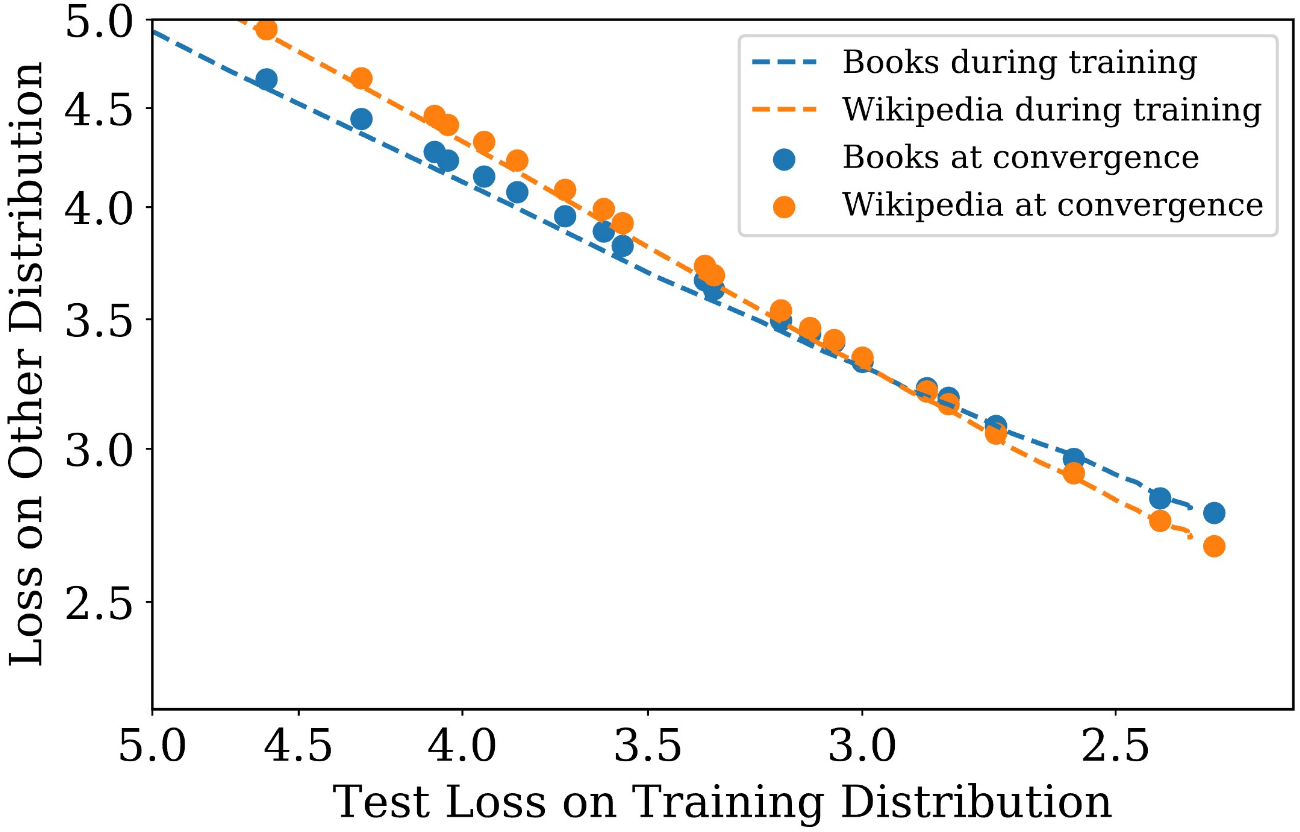
The transferability of scaling law to similar datasets



- WebText2: web scrape of outbound links from Reddit with a minimum of 3 upvotes.
- Internet Books: Books available on the internet
- Books: Fiction books (11k books)
- Wikipedia: ~6million articles
- Common Crawl: entire website data (petabytes)

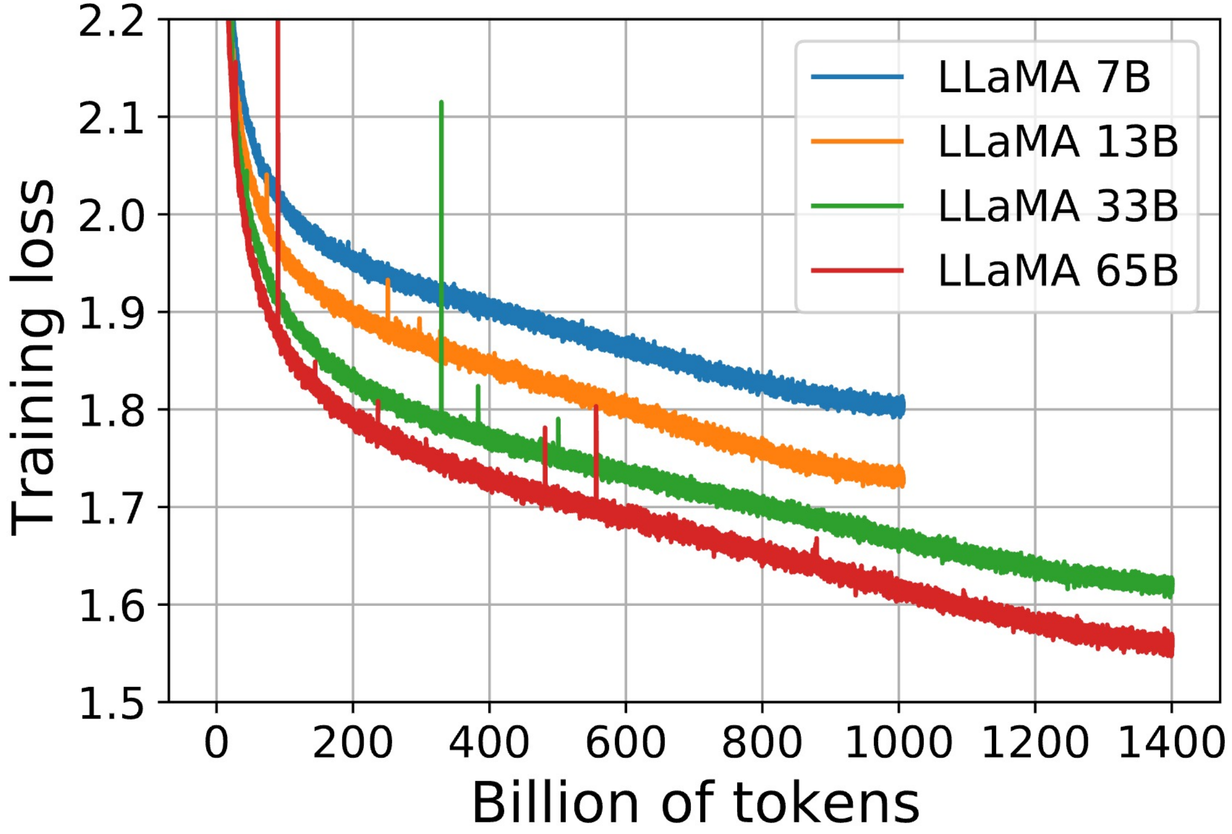
The smoothly improved is the cross-entropy loss

Generalization performance depends on the training loss not training phase



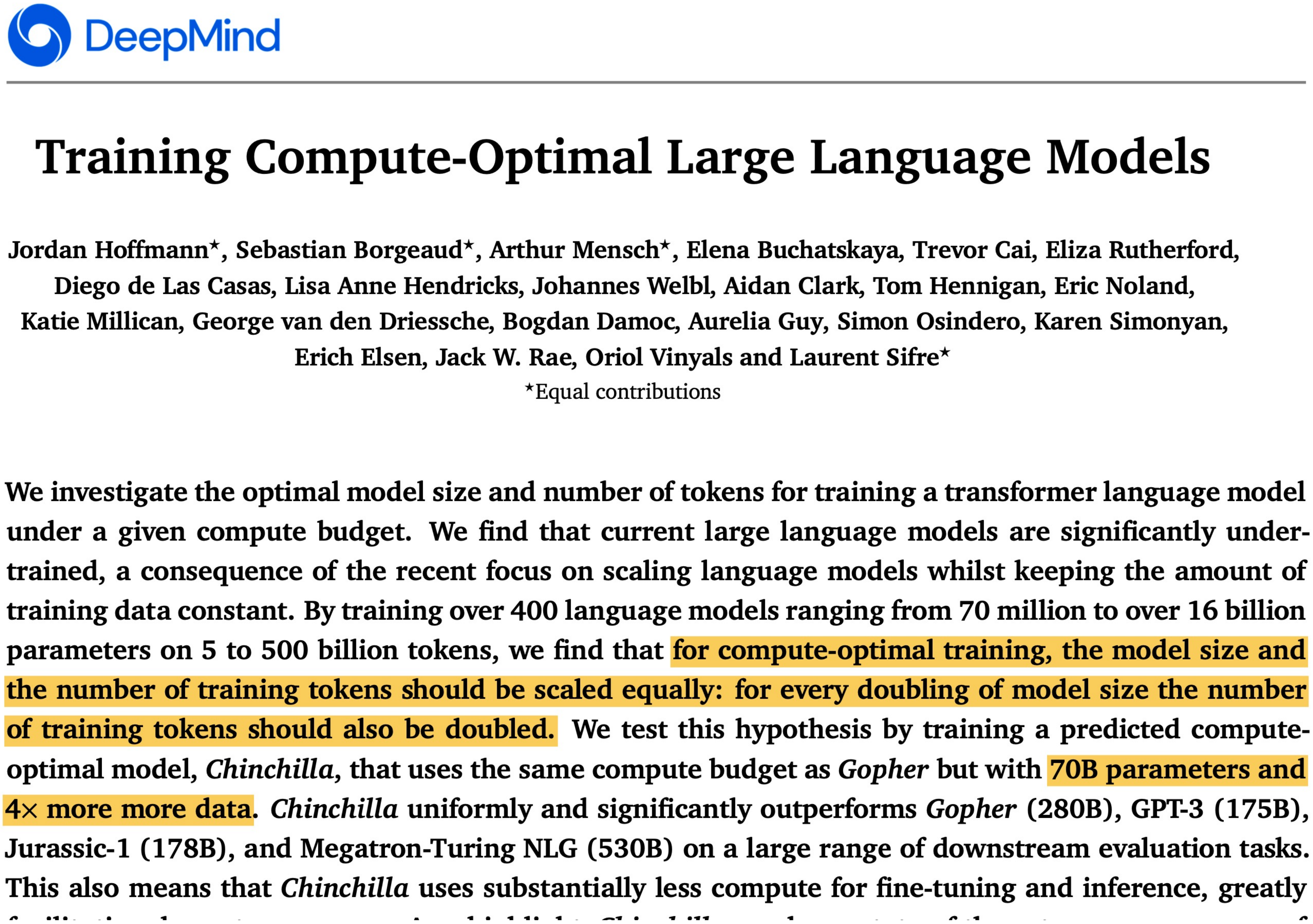
- Dashed curves correspond to a large model training.
- Dots corresponds to convergent solutions of model with different size.

Large model is more sample-efficient



Training loss over training tokens for LLaMa models.

- LLM training involves processing only one epoch.
- Large model converges fasters and is thus more sample-efficient.
- Why?



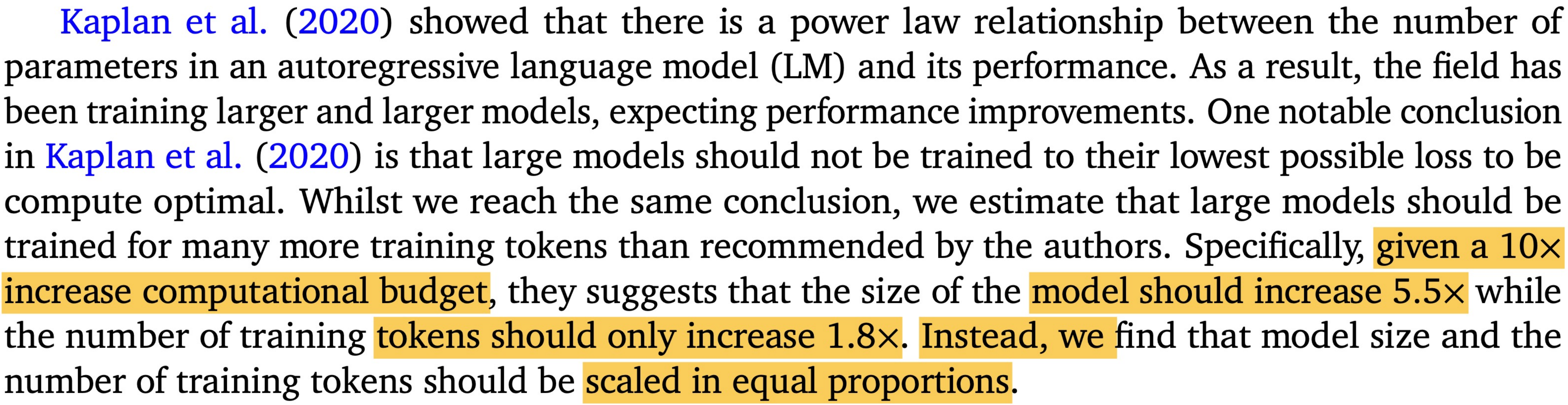
DeepMind

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

We investigate the optimal model size and number of tokens for training a transformer language model under a given compute budget. We find that current large language models are significantly undertrained, a consequence of the recent focus on scaling language models whilst keeping the amount of training data constant. By training over 400 language models ranging from 70 million to over 16 billion parameters on 5 to 500 billion tokens, we find that for compute-optimal training, the model size and the number of training tokens should be scaled equally: for every doubling of model size the number of training tokens should also be doubled. We test this hypothesis by training a predicted compute-optimal model, *Chinchilla*, that uses the same compute budget as *Gopher* but with 70B parameters and 4× more more data. *Chinchilla* uniformly and significantly outperforms *Gopher* (280B), GPT-3 (175B), Jurassic-1 (178B), and Megatron-Turing NLG (530B) on a large range of downstream evaluation tasks. This also means that *Chinchilla* uses substantially less compute for fine-tuning and inference, greatly

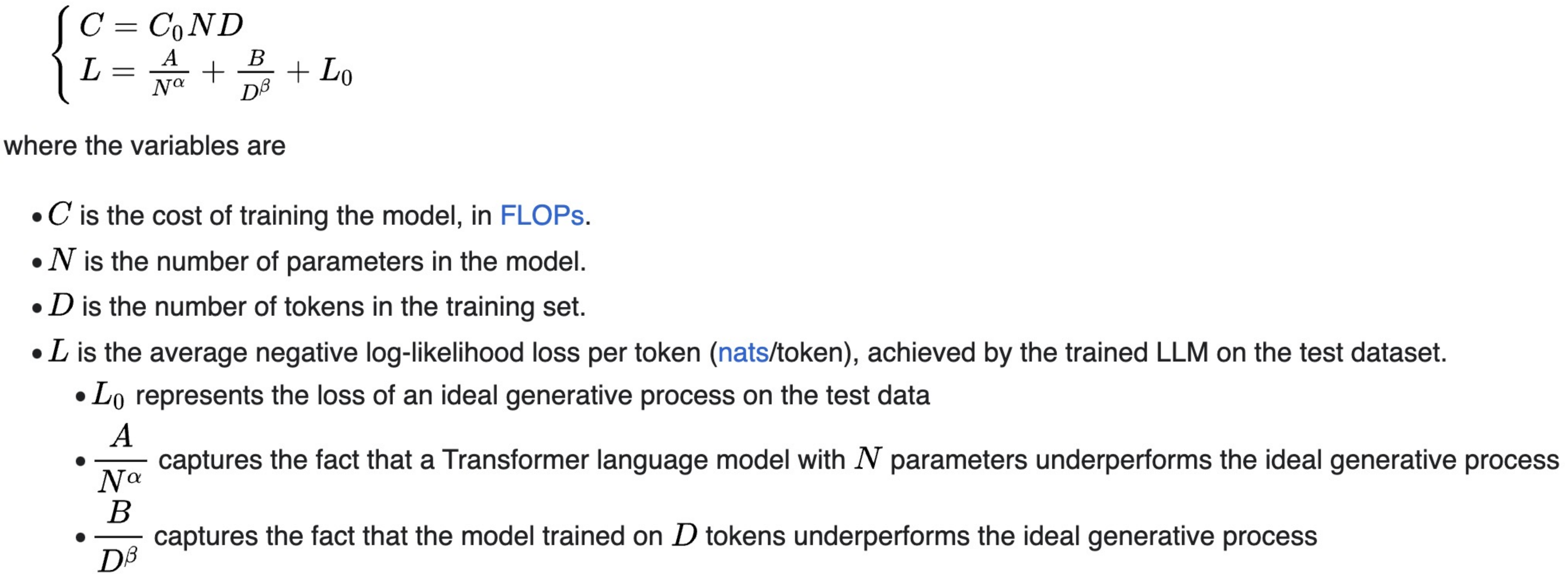
The Major Message of Chinchilla Law Paper



Kaplan et al. (2020) showed that there is a power law relationship between the number of parameters in an autoregressive language model (LM) and its performance. As a result, the field has been training larger and larger models, expecting performance improvements. One notable conclusion in Kaplan et al. (2020) is that large models should not be trained to their lowest possible loss to be compute optimal. Whilst we reach the same conclusion, we estimate that large models should be trained for many more training tokens than recommended by the authors. Specifically, given a 10× increase computational budget, they suggests that the size of the model should increase 5.5× while the number of training tokens should only increase 1.8×. Instead, we find that model size and the number of training tokens should be scaled in equal proportions.

Compute-optimal Training

- For LLM (auto-regressive) trained for one-epoch with a cosine LR, we have



$$\begin{cases} C = C_0 N D \\ L = \frac{A}{N^\alpha} + \frac{B}{D^\beta} + L_0 \end{cases}$$

where the variables are

- C is the cost of training the model, in FLOPs.
- N is the number of parameters in the model.
- D is the number of tokens in the training set.
- L is the average negative log-likelihood loss per token (nats/token), achieved by the trained LLM on the test dataset.
  - L_0 represents the loss of an ideal generative process on the test data
  - $\frac{A}{N^\alpha}$ captures the fact that a Transformer language model with N parameters underperforms the ideal generative process
  - $\frac{B}{D^\beta}$ captures the fact that the model trained on D tokens underperforms the ideal generative process

Compute-optimal LLM

- Given a compute budget, what is the best model size and data size?

$$\min_{N,D} L(N,D) := \frac{A}{N^\alpha} + \frac{B}{D^\beta} + L_0$$
$$s.t. \quad C_0 N D = C$$

- The solution is given by

$$N_{\text{opt}}(C) = G\left(\frac{C}{C_0}\right)^{\frac{\beta}{\alpha+\beta}}, D_{\text{opt}}(C) = \frac{1}{G}\left(\frac{C}{C_0}\right)^{\frac{\alpha}{\alpha+\beta}}, \qquad G = \left(\frac{\alpha A}{\beta B}\right)^{\frac{1}{\alpha+\beta}}$$

- In Chinchila scaling law paper,

$$\begin{cases} N_{opt}(C) = 0.6\ C^{0.45} \\ D_{opt}(C) = 0.3\ C^{0.55} \\ L_{opt}(C) = 1070\ C^{-0.154} + 1.7 \end{cases}$$

- Observation: To achieve compute-optimal, scale the model size and data size in approximately equation proportions. But this may not hold generally.

Summary: scale is all you need

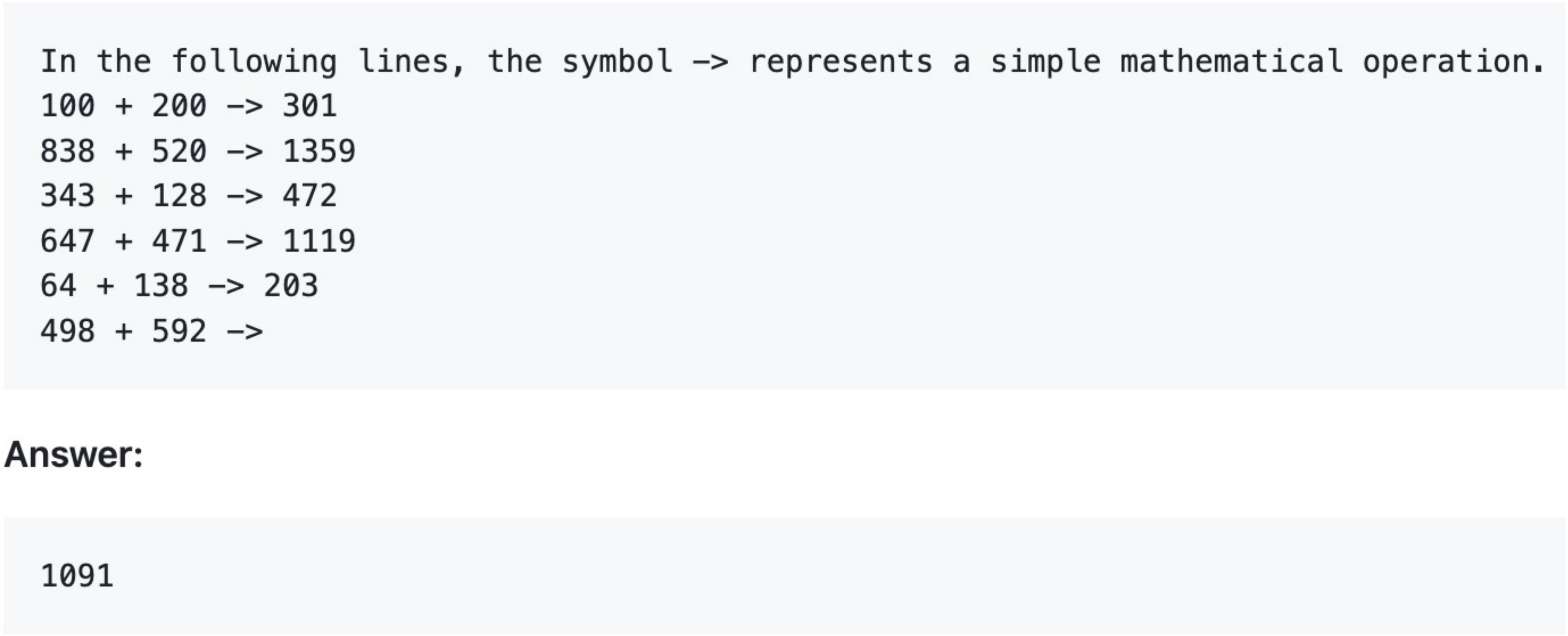
- LLM training is mysteriously **predictable**
  - We can train small models to fitting scaling law,
  - Then, use the fitted scaling law to predict the performance of large models and optimally allocate resources for training large models
- LLM performance depends on the scale. The architecture shape does not matter too much.
- Transformer has better scaling property than LSTM.
- Large models are more sample-efficient than small models.
- Scaling law now plays an important role in assessing new architecture and optimizers.

Reading

- Hestness et al., Deep learning scaling is predictable, empirically
- Kaplan et al., Scaling Laws for Neural Language Models, openAI.
- Hoffman et al., Training Compute-Optimal Large Language Models, DeepMind.

In-context learning (ICL)

Pretrained LLMs can perform in-context learning (vs. few-shot finetuning).



```
In the following lines, the symbol -> represents a simple mathematical operation.
100 + 200 -> 301
838 + 520 -> 1359
343 + 128 -> 472
647 + 471 -> 1119
64 + 138 -> 203
498 + 592 ->
```

Answer:

```
1091
```

Understanding ICL

- Use T_θ denote the model, whose inputs are few-shot samples
- Train the ICL model in a supervised manner

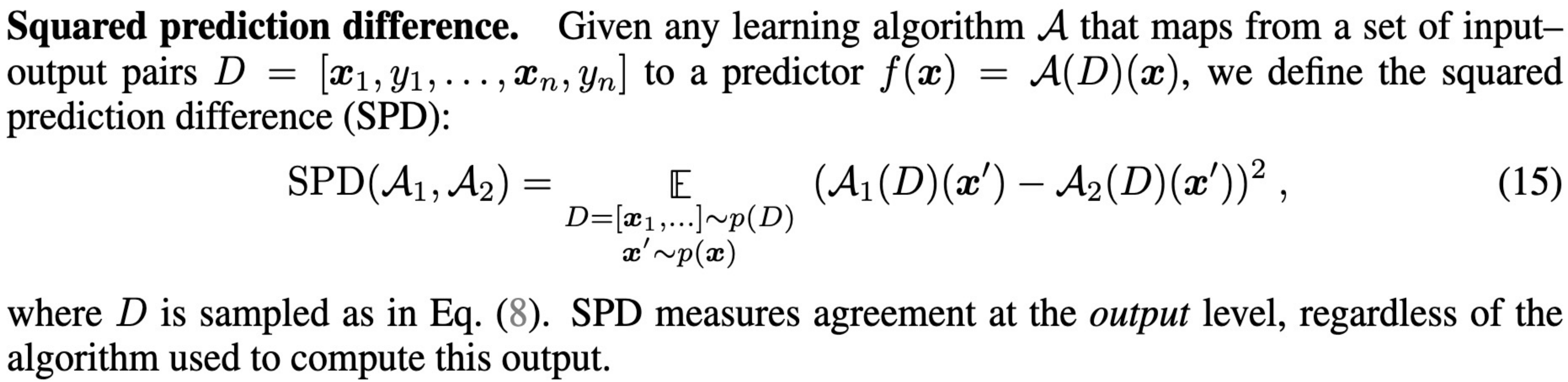
$$\arg\min_{\theta} \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n \sim p(x) \\ f \sim p(f)}} \left[\sum_{i=1}^{n} \mathcal{L}\left(f(\boldsymbol{x}_i), T_\theta\left([\boldsymbol{x}_1, f(\boldsymbol{x}_1) \ldots, \boldsymbol{x}_i]\right)\right) \right]$$

- Consider the linear target function class $f(x) = w^\top x \text{ with } w \sim \mathcal{N}(0, I_d)$

- Then, the leaned model should be able to perform ICL.

What does the trained TF implement?

- There exists many estimators for linear regression:
  - Ridge regression/ordinary least square (OLS)/Lasso
  - Multiple/single-step GD/One-step Newton
  - Etc.
- Measure the difference between two ICL estimators

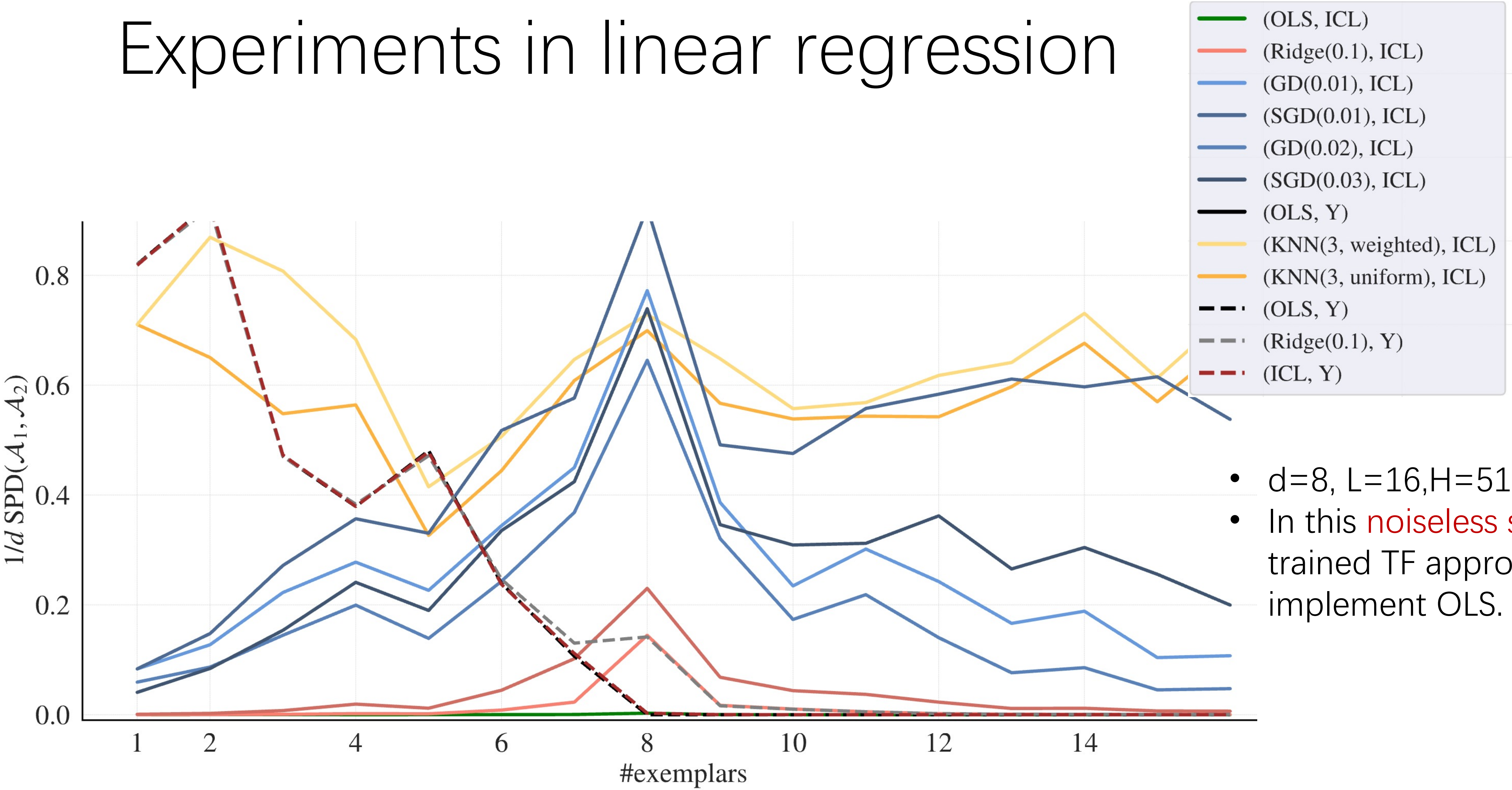


Squared prediction difference. Given any learning algorithm $\mathcal{A}$ that maps from a set of input–output pairs $D = [\boldsymbol{x}_1, y_1, \ldots, \boldsymbol{x}_n, y_n]$ to a predictor $f(\boldsymbol{x}) = \mathcal{A}(D)(\boldsymbol{x})$, we define the squared prediction difference (SPD):

$$\mathrm{SPD}(\mathcal{A}_1, \mathcal{A}_2) = \mathop{\mathbb{E}}_{\substack{D=[\boldsymbol{x}_1,\ldots]\sim p(D)\\ \boldsymbol{x}'\sim p(\boldsymbol{x})}} \left(\mathcal{A}_1(D)(\boldsymbol{x}') - \mathcal{A}_2(D)(\boldsymbol{x}')\right)^2 , \tag{15}$$

where D is sampled as in Eq. (8). SPD measures agreement at the *output* level, regardless of the algorithm used to compute this output.

Experiments in linear regression



- d=8, L=16,H=512, H=8.
- In this noiseless setting, trained TF approximately implement OLS.

The problems

- The aforementioned explanations of ICL are based on an operator learning framework, which can be used to investigate:
  - TF can approximate/represent certain estimator
  - Investigate how model architectures like attention, softmax changes the **inductive bias** into certain estimators.
  - To learn generalizable estimator needs only $\mathrm{poly}(n, d, \mathcal{F})$ samples (Bai et al.,NeurIPS 2014)

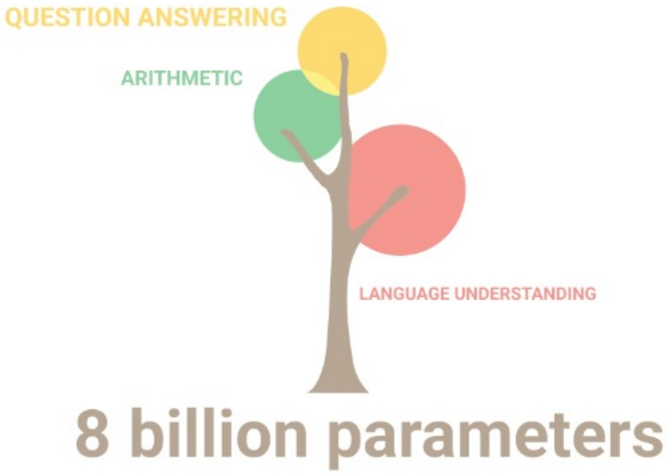
- This framework is fact more useful in meta-learning, such as analyzing learned optimizer, statistical estimator and ODE/PDE solver.

The problems (contd)

- However, in ICL, the **LLM is pretrained on a generic dataset**, where no such clear supervised data are available.

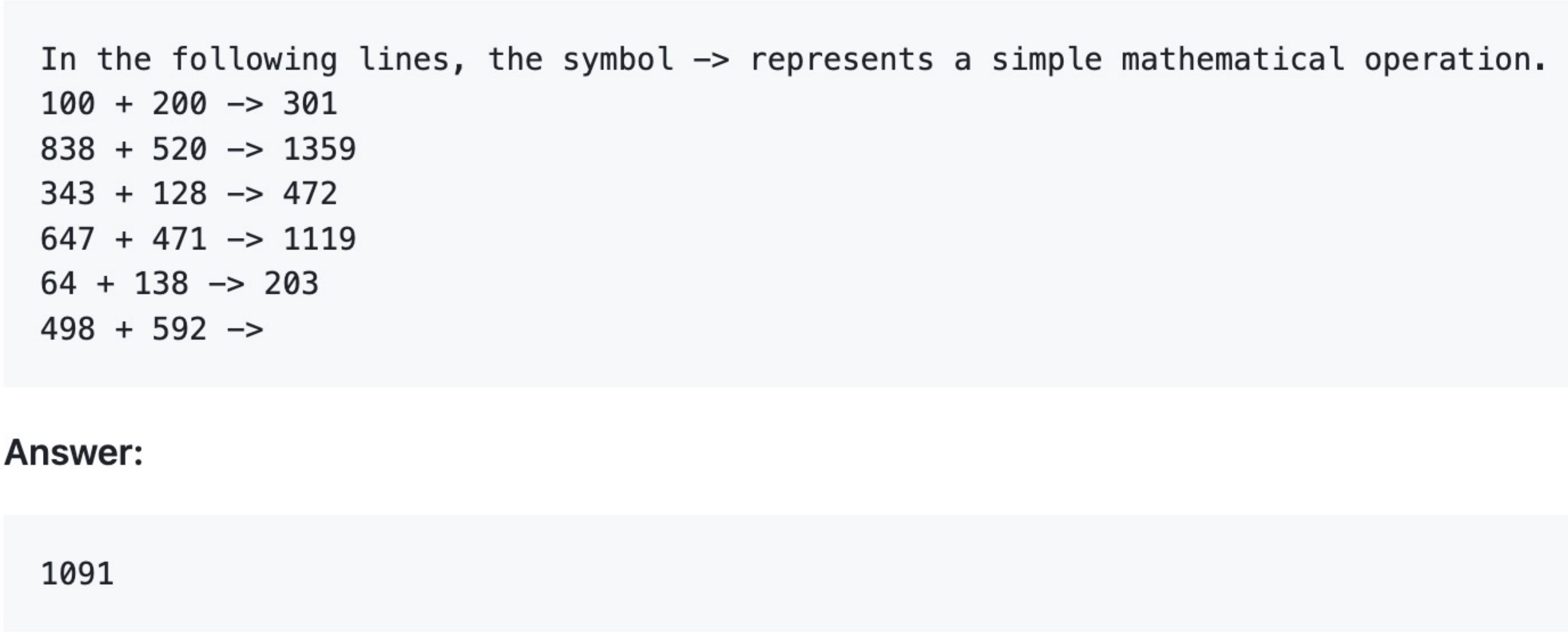
- The operator learning framework does not explain:
  - How does pretrained LLMs master ICL in performing **next-token prediction?**
  - Why does ICL ability **emerge** so late?

Emergence


QUESTION ANSWERING
ARITHMETIC
LANGUAGE UNDERSTANDING
8 billion parameters

Few-Shot Prompted Tasks

The ICL capability emerges only when the model is large enough.



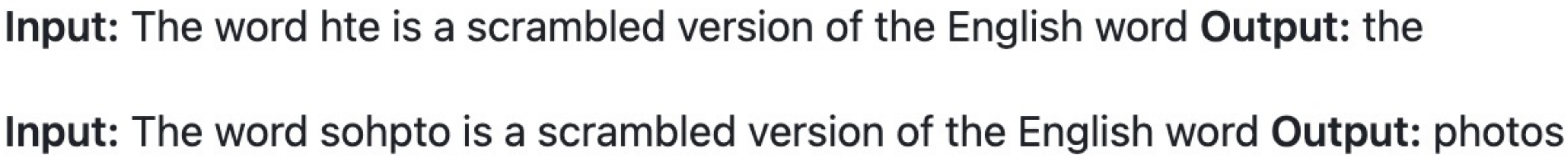
```
In the following lines, the symbol -> represents a simple mathematical operation.
100 + 200 -> 301
838 + 520 -> 1359
343 + 128 -> 472
647 + 471 -> 1119
64 + 138 -> 203
498 + 592 ->
```

Answer:

```
1091
```

Other Tasks

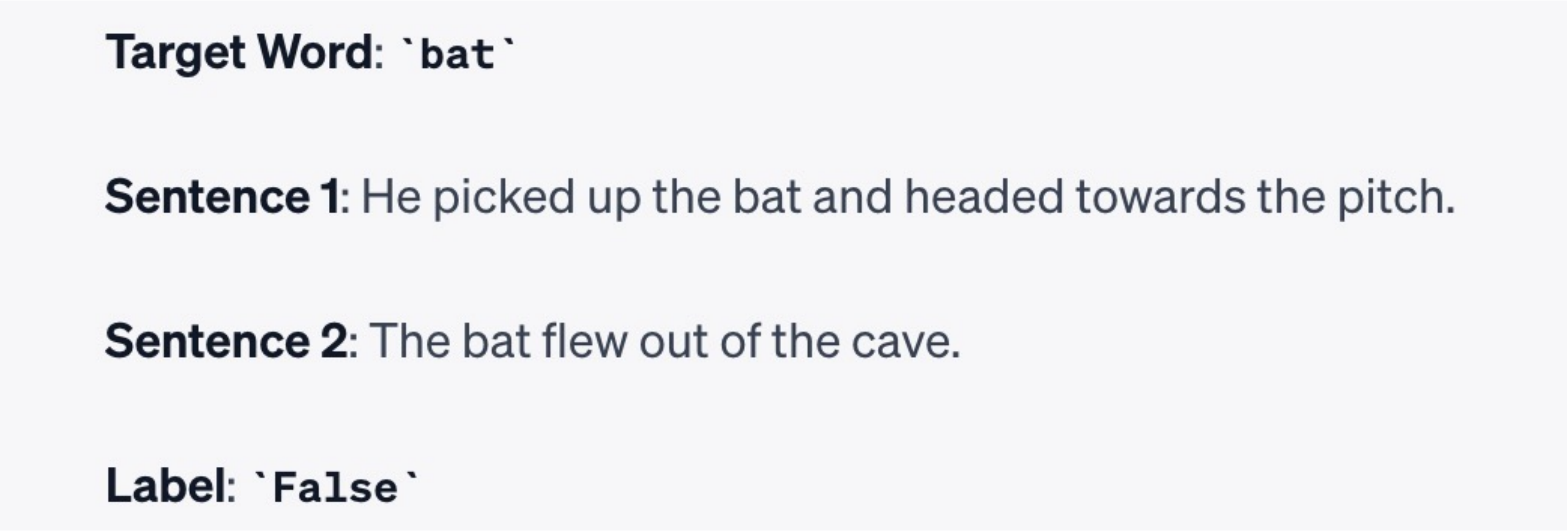
- **Word Unscrambling**



Input: The word hte is a scrambled version of the English word **Output:** the

Input: The word sohpto is a scrambled version of the English word **Output:** photos

- **Word in Context (WiC)**



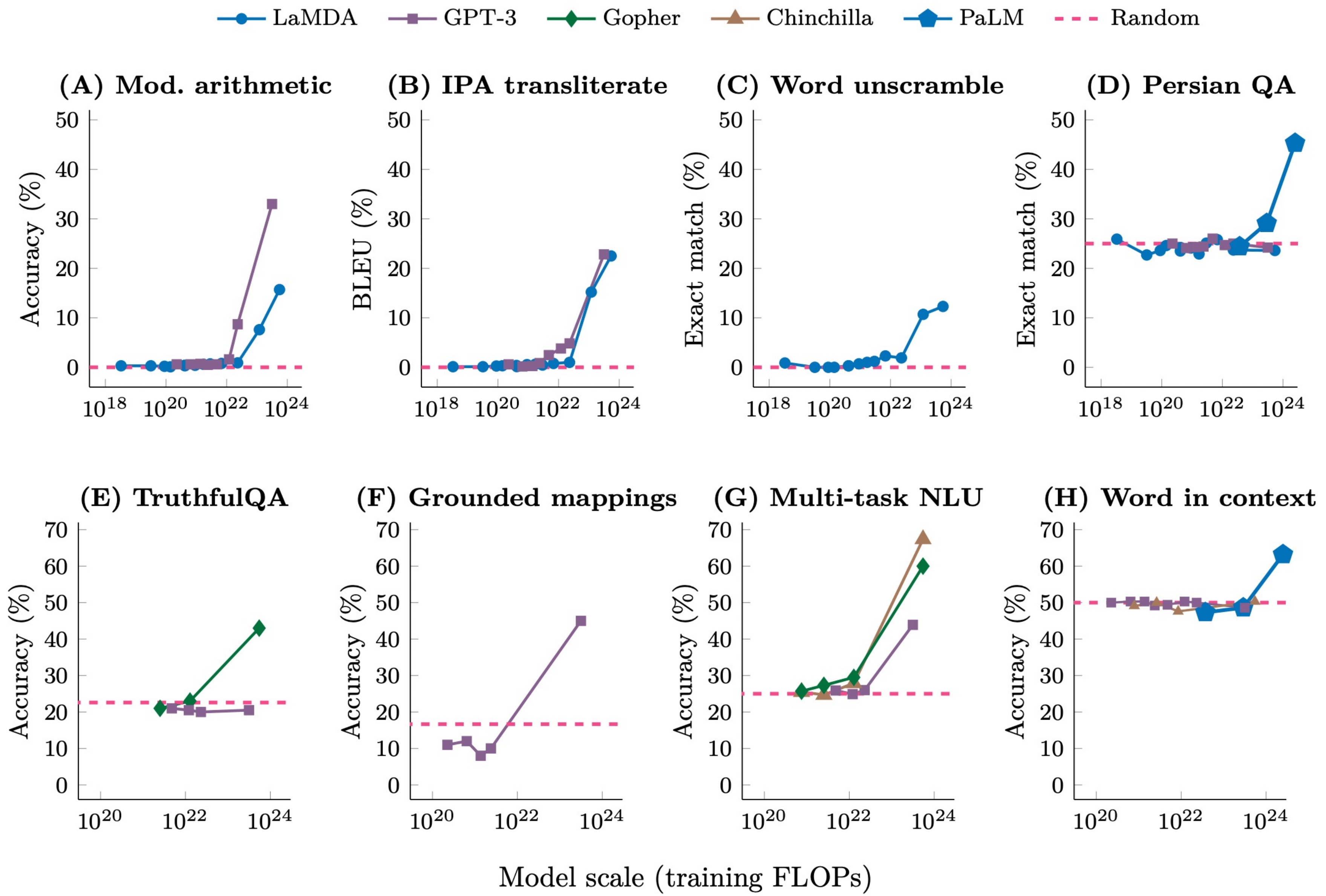
Target Word: `bat`

Sentence 1: He picked up the bat and headed towards the pitch.

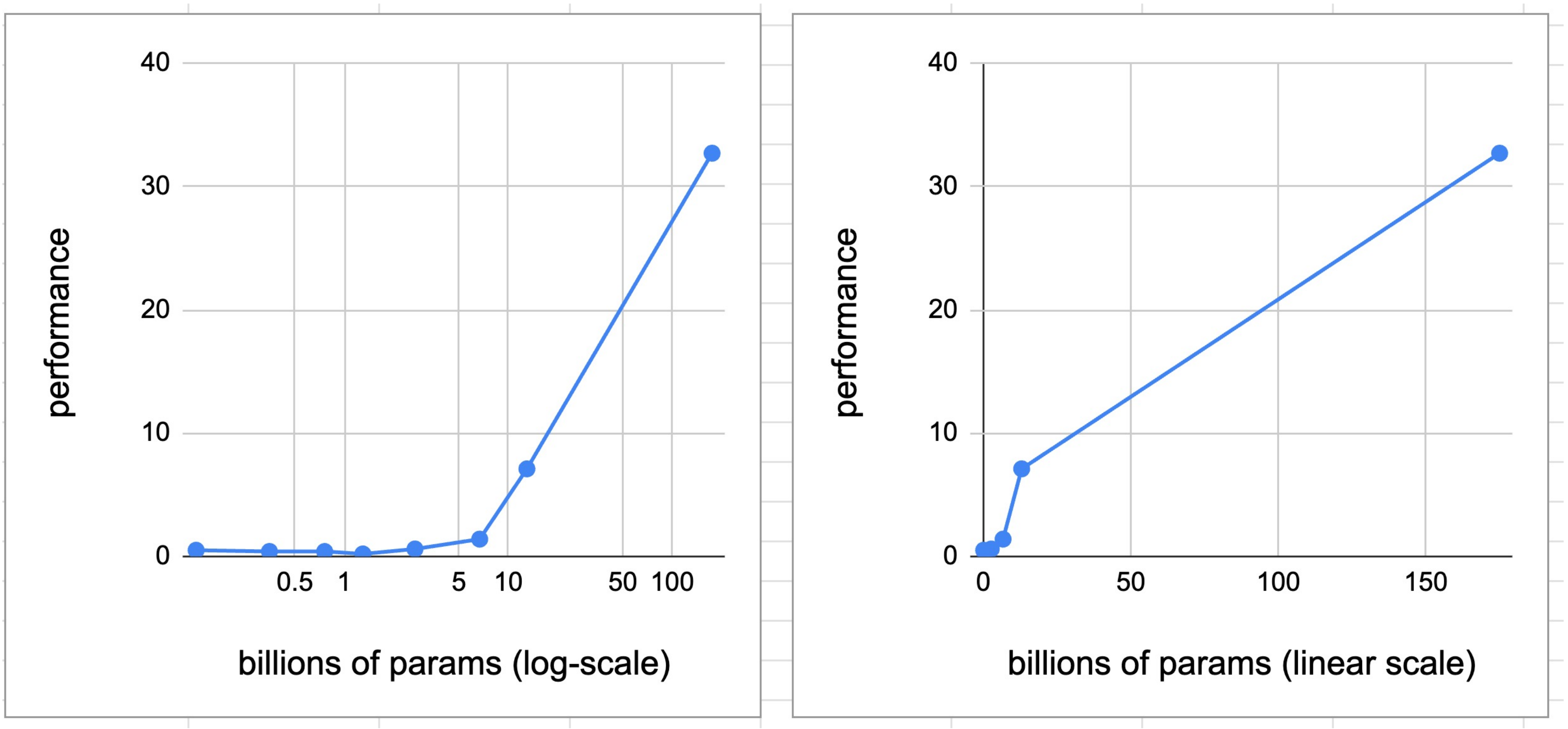
Sentence 2: The bat flew out of the cave.

Label: `False`

Emergent Abilities



Log scale vs. Linear scale

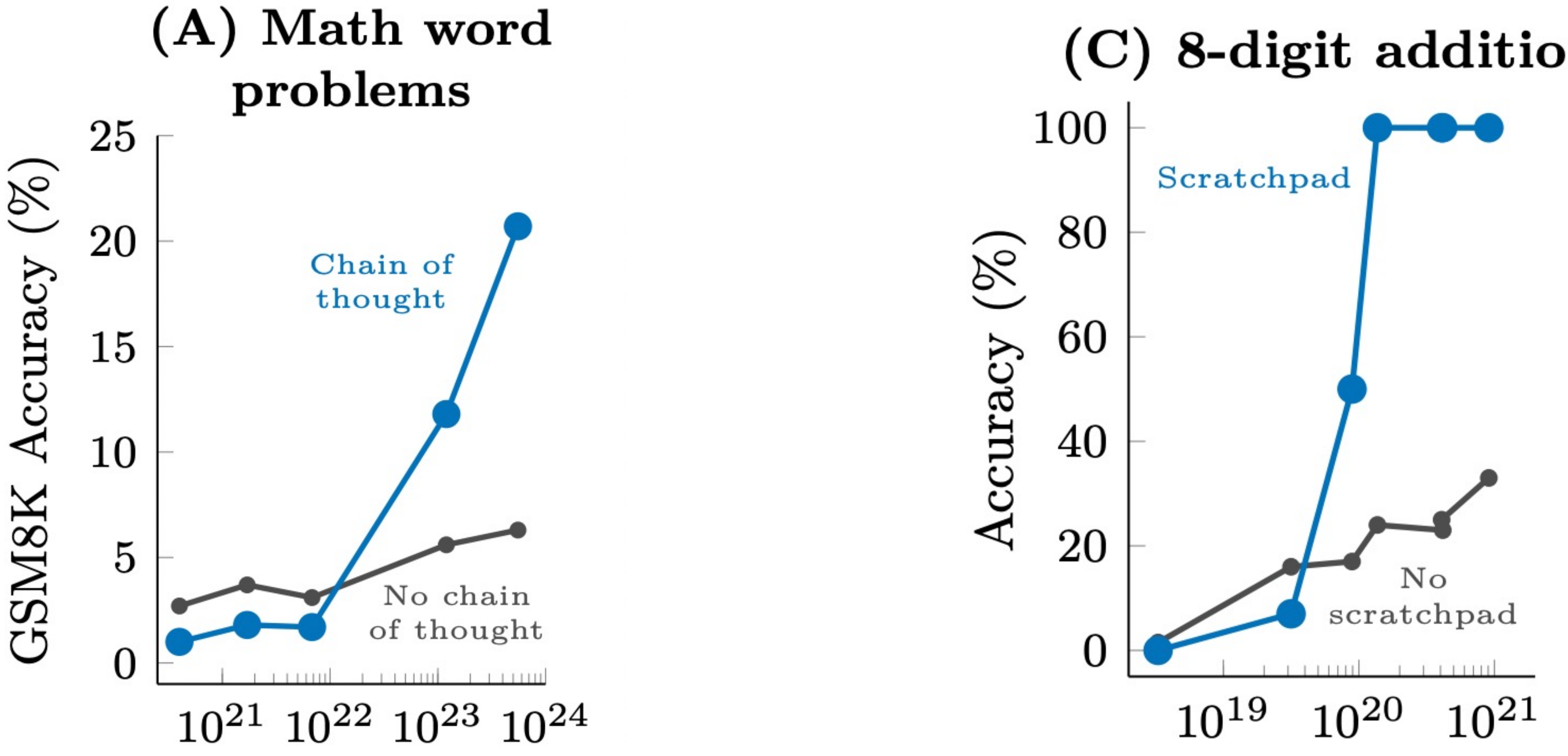


See this link

Remarks

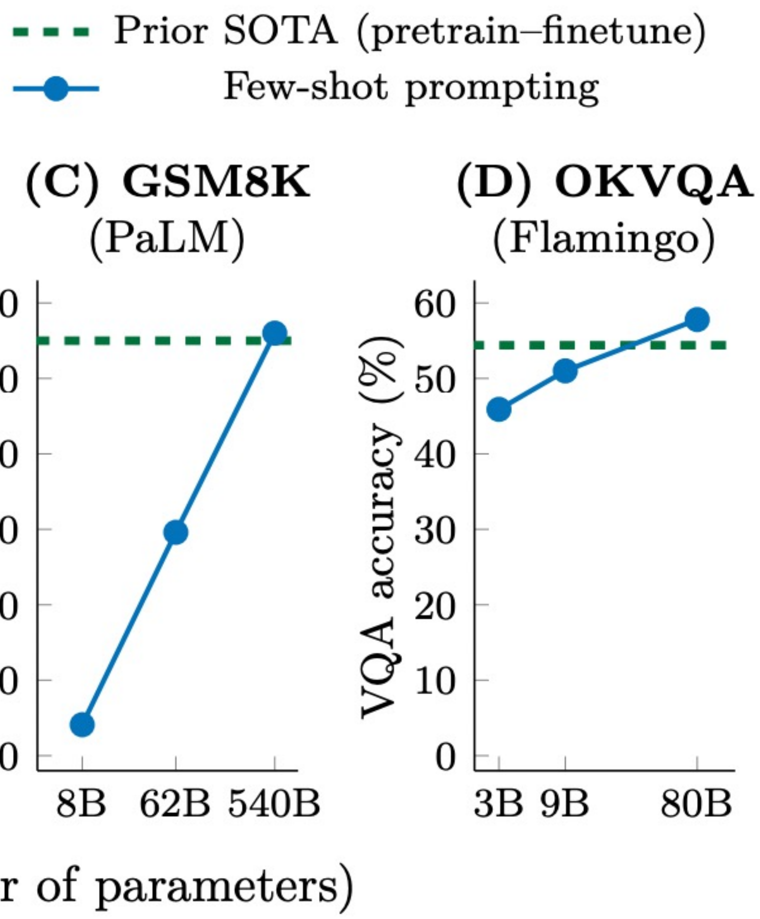
- Emergence is observed for **downstream (reasoning) tasks**.
- The downstream tasks are very different from the pretrain data
- The evaluation metrics are not cross-entropy but accuracy.
- Still many tasks in BIG-Bench is challenging for LLM.

Augmented prompting strategies



Emergence: surpassing finetuning

Sociological change in the AI community: finetuned task-specific models are outperformed by few-shot prompted large model

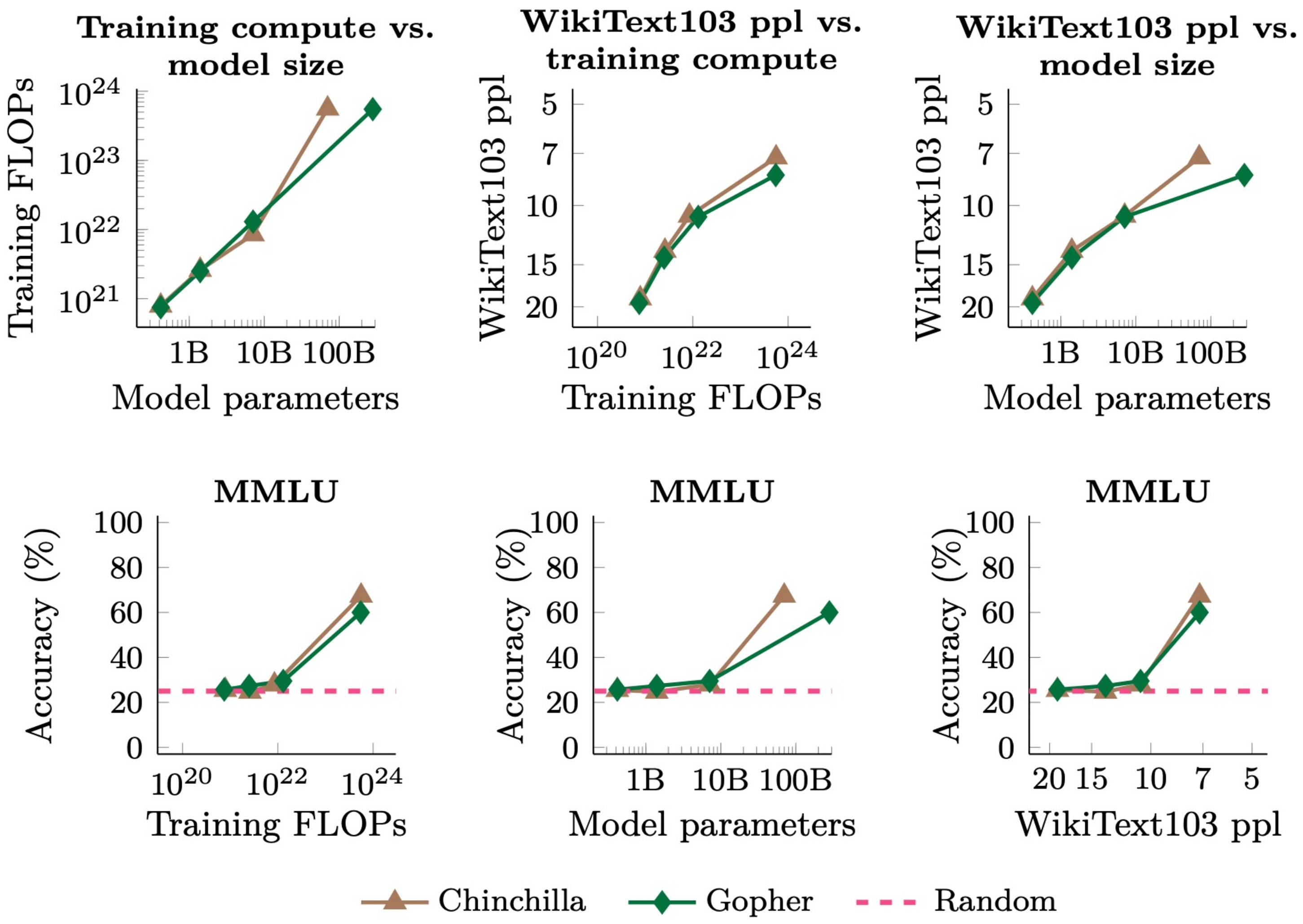


Emergence: measure of model “size”

What’s the right *x*-axis for emergence?

Can be viewed through training FLOPs,

model parameters, wikitext103 ppl.

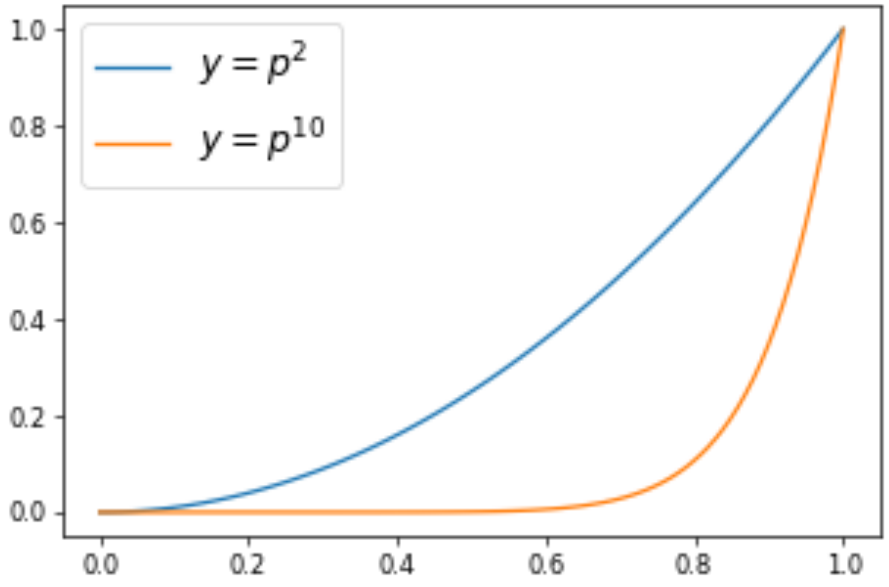


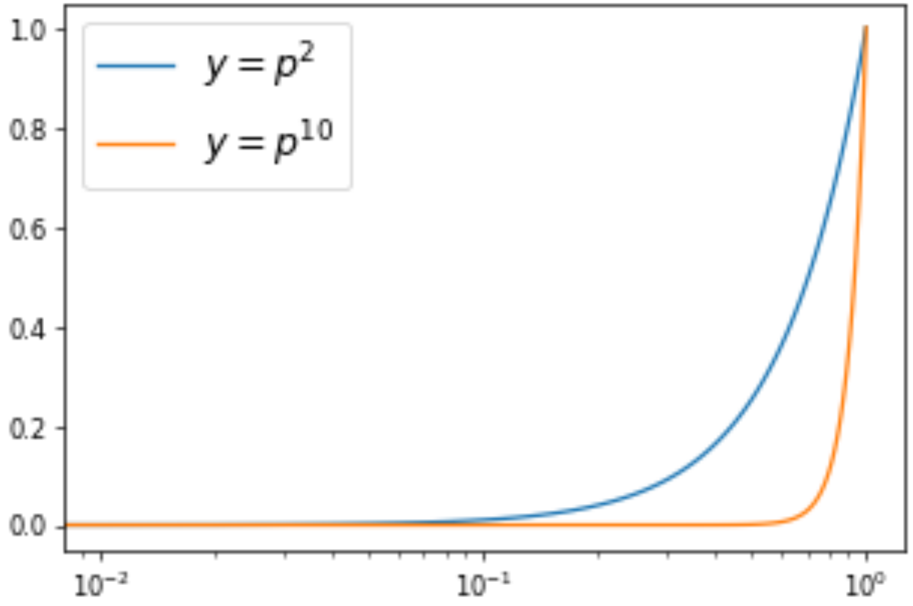
Emergent Abilities - fact or illusion?

- In real emergent phenomenon, **the rules of the game change**.
- It is unclear at this point what exactly this means with respect to Large Language Models.

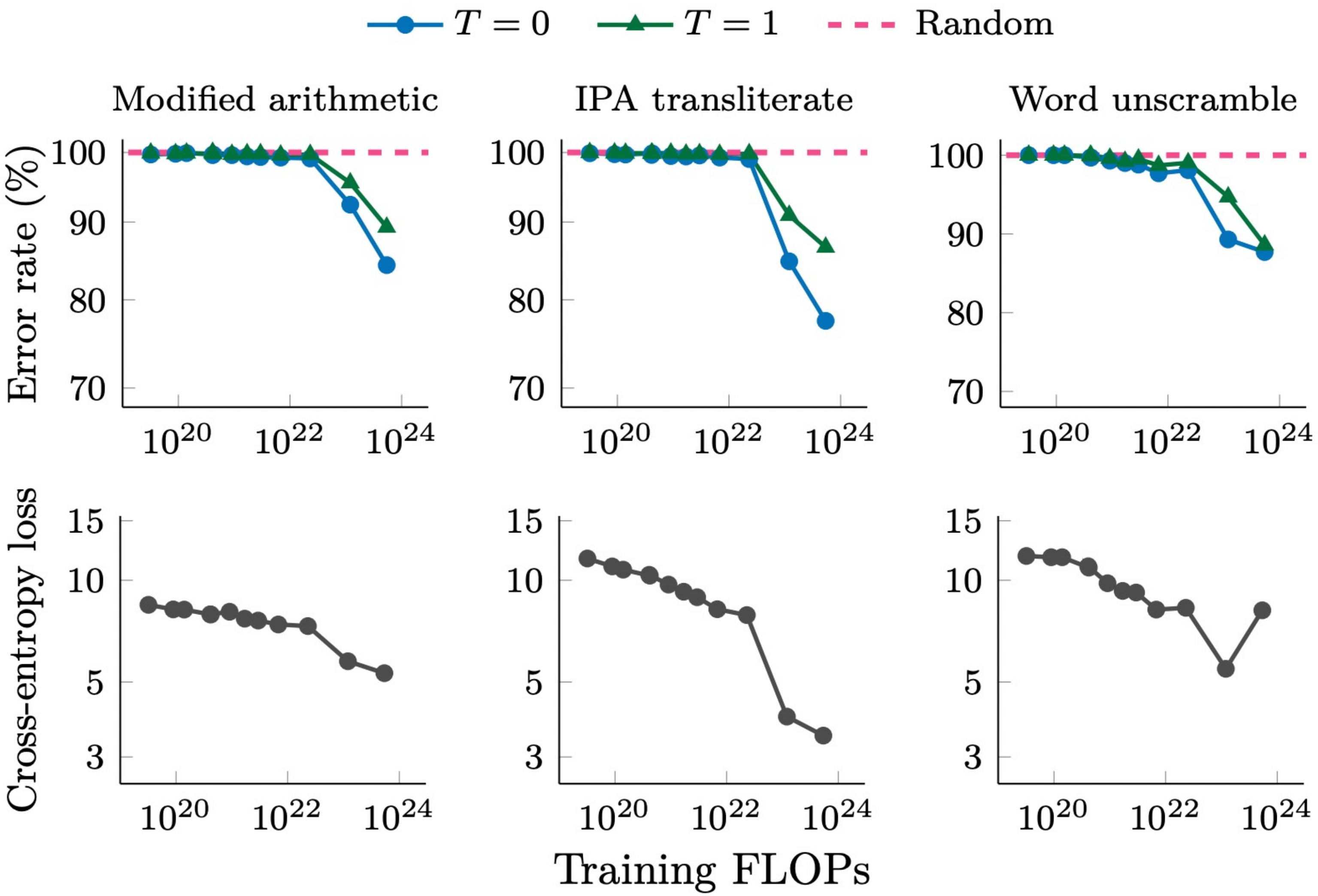
A plausible explanation: Compounding effect

- The downstream tasks often take **multi-step reasoning**. In this case, evaluation metric matters
  - Fail vs. success
  - The number of correct steps.
- Assume that the success of one-step reasoning is p. Then, the prob. of k-step success is: p^k

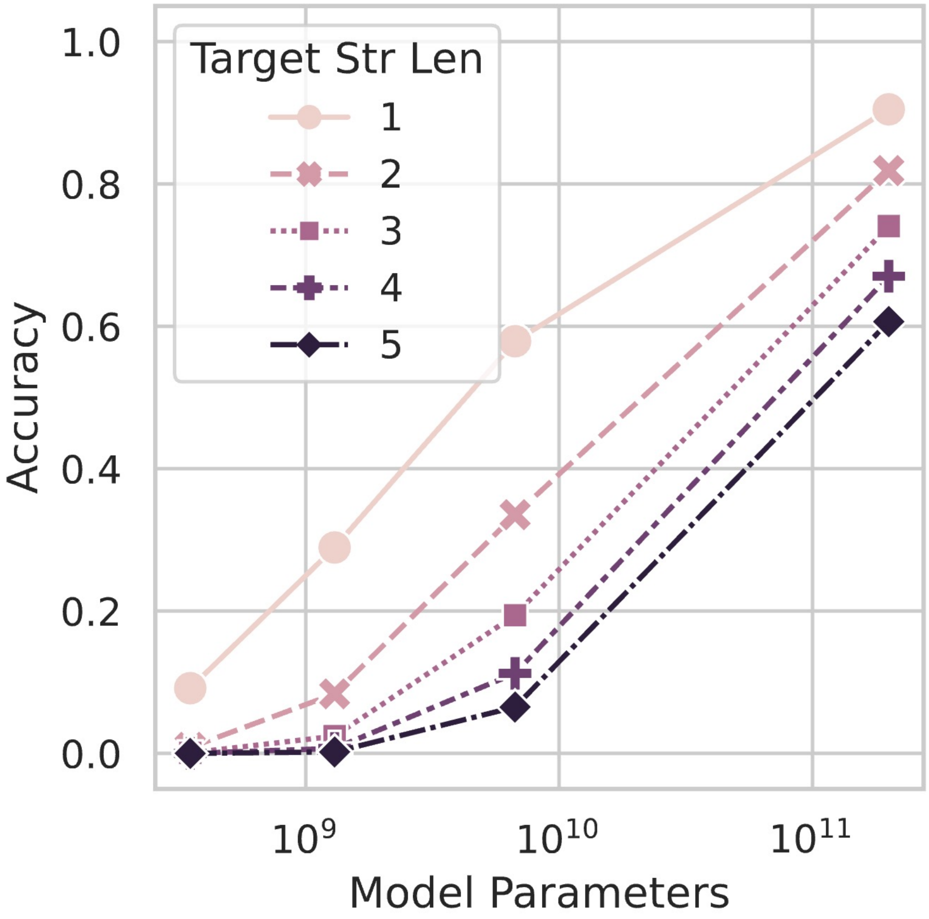


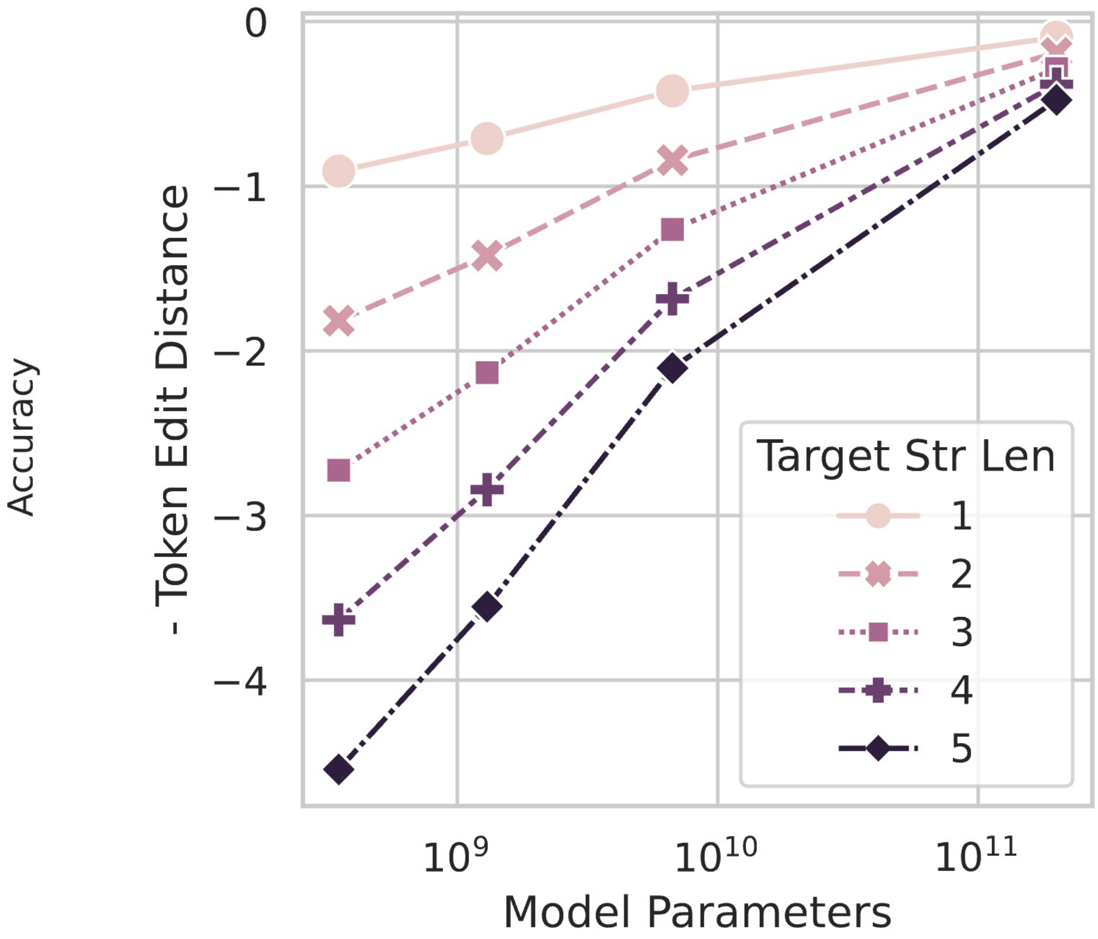


Evaluation metrics

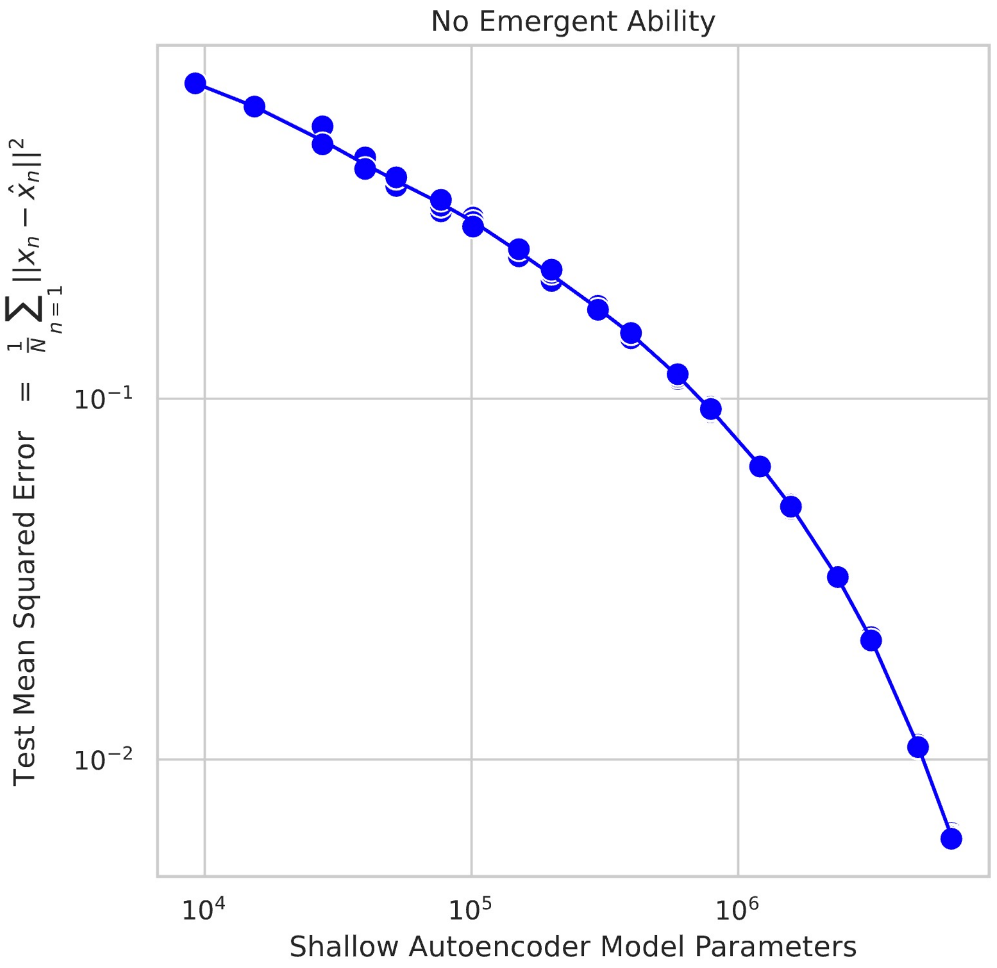


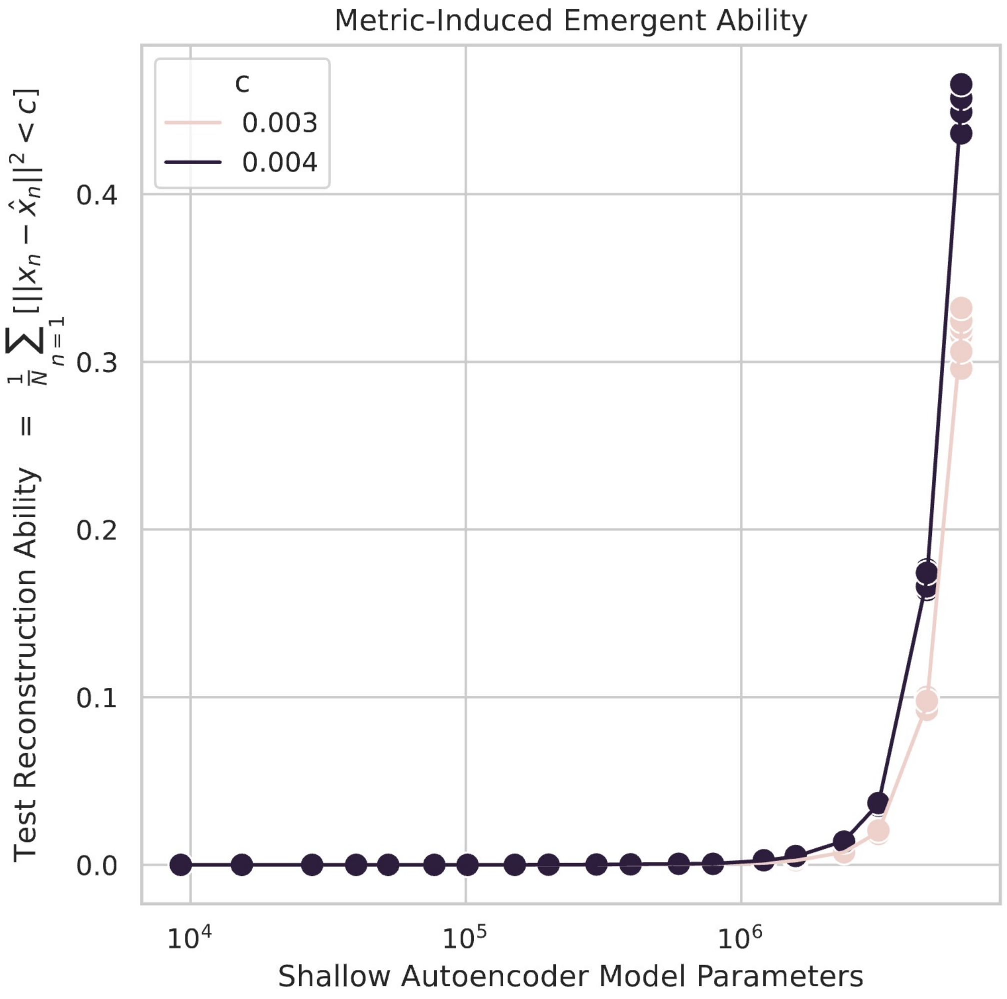
Claimed emergent abilities evaporate upon changing the metric





Induced emergent reconstruction ability in shallow nonlinear autoencoders (for CIFAR-100)

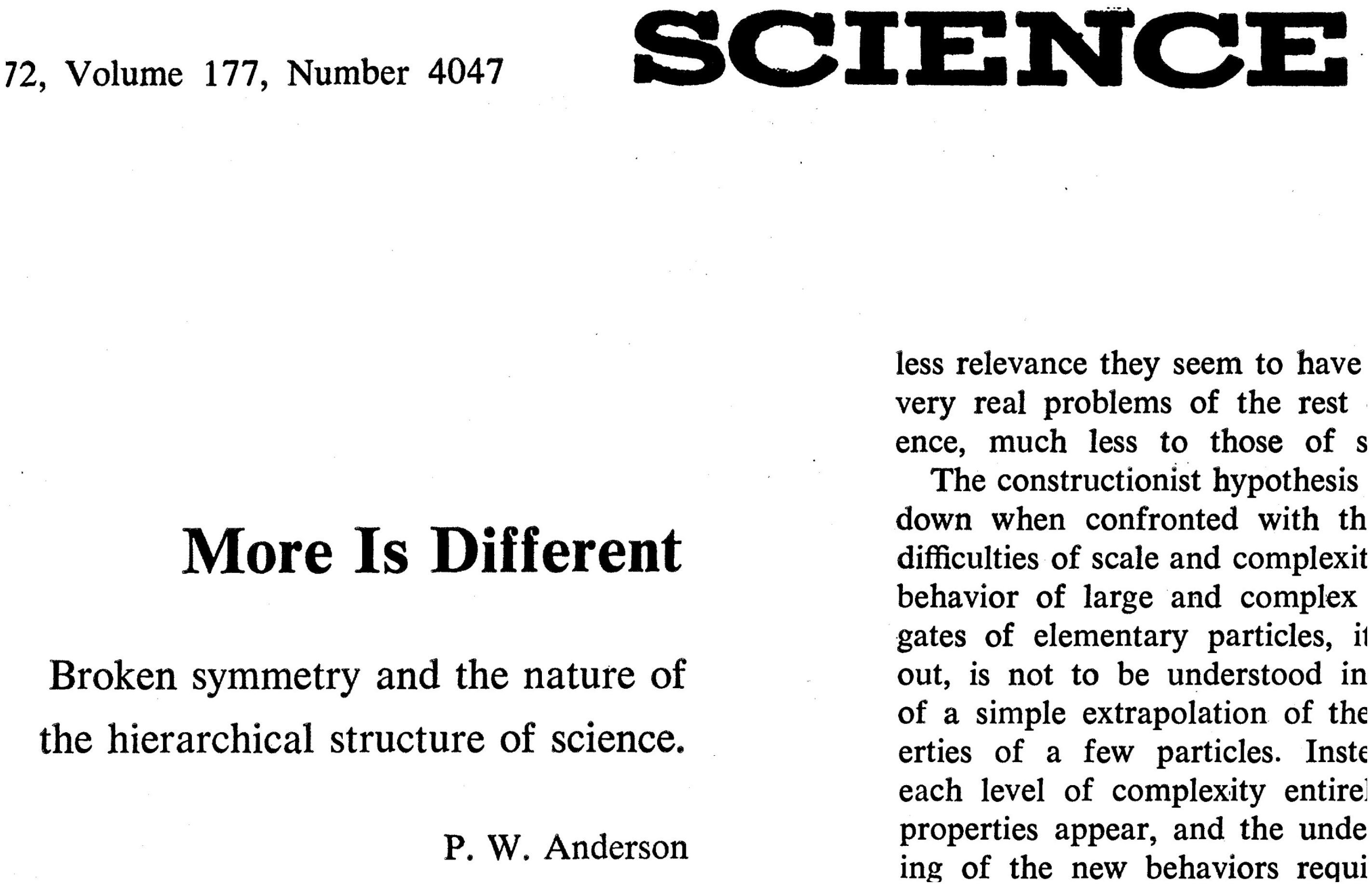

No Emergent Ability


Metric-Induced Emergent Ability

Scaling law vs. Emergence

- Scaling law says that LLM is predictable.
- Emergence emphasizes that LLM ability is unpredictable.
- Any contradictions? No!
  - Scaling law is about the pretraining loss/PPL.
  - Emergence is about the downstream performance.

More is different?



72, Volume 177, Number 4047 **SCIENCE**

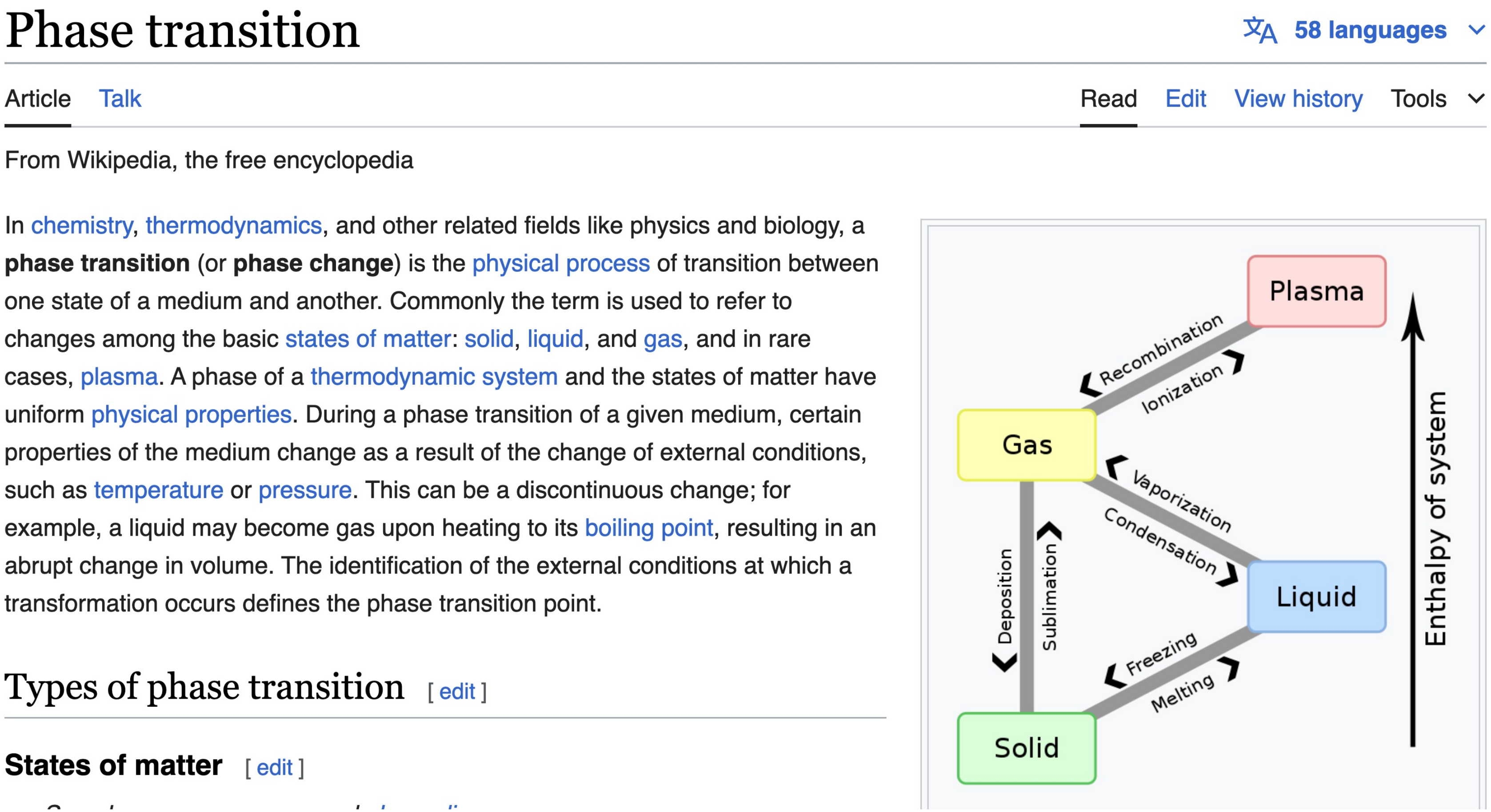
More Is Different

Broken symmetry and the nature of
the hierarchical structure of science.

P. W. Anderson

less relevance they seem to have
very real problems of the rest
ence, much less to those of s
The constructionist hypothesis
down when confronted with th
difficulties of scale and complexit
behavior of large and complex
gates of elementary particles, it
out, is not to be understood in
of a simple extrapolation of the
erties of a few particles. Inste
each level of complexity entirel
properties appear, and the unde
ing of the new behaviors requi

Phase transition



Phase transition

From Wikipedia, the free encyclopedia

In chemistry, thermodynamics, and other related fields like physics and biology, a **phase transition** (or **phase change**) is the physical process of transition between one state of a medium and another. Commonly the term is used to refer to changes among the basic states of matter: solid, liquid, and gas, and in rare cases, plasma. A phase of a thermodynamic system and the states of matter have uniform physical properties. During a phase transition of a given medium, certain properties of the medium change as a result of the change of external conditions, such as temperature or pressure. This can be a discontinuous change; for example, a liquid may become gas upon heating to its boiling point, resulting in an abrupt change in volume. The identification of the external conditions at which a transformation occurs defines the phase transition point.

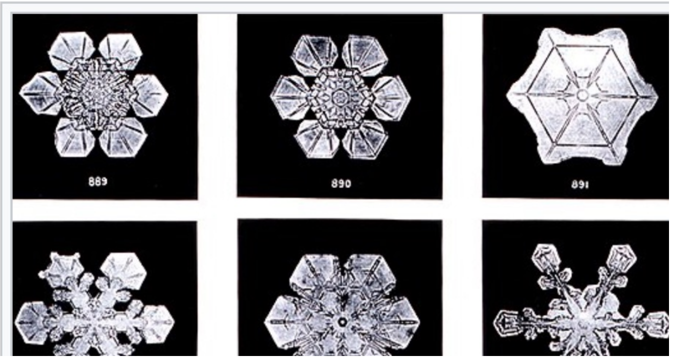
Types of phase transition [edit]

States of matter [edit]

Emergence vs. Phase transition

In philosophy, systems theory, science, and art, **emergence** occurs when a complex entity has properties or behaviors that its parts do not have on their own, and emerge only when they interact in a wider whole.

Emergence plays a central role in theories of integrative levels and of complex systems. For instance, the phenomenon of life as studied in biology is an emergent property of chemistry and quantum physics.



Summary

- Understanding emergence is crucial for
  - **Accelerate the Emergence of Desirable Abilities**: In training large language models (LLMs), emergent abilities like in-context learning (ICL) often appear very late in the process. By understanding emergence, we can potentially speed up the development of these beneficial capabilities.
  - **Prevent the Emergence of Undesirable Abilities**: Some unknown or harmful abilities might also emerge during training. By understanding emergence, we can take steps to prevent these from developing.
- To fully understand ICL, it is important to investigate why ICL abilities appear so late in the training process.

Reading

- Wei et al., Emergent Abilities of Large Language Models, TMLR2023.
- Schaeffe et al., Are Emergent Abilities of Large Language Models a Mirage?, NeurIPS 2023.