Upcoming Dates

- Please submit your peer feedback by Monday at 8 PM.
- Please submit midterm regrade requests by Friday, November 10.
  - If you are unsure whether you should request a regrade, talk to us in office hours first.
- No class next Tuesday. If you are a US citizen, go out and vote!
  - Daphne will still hold office hours.
- Next Thursday: Industry lecture from Deep Ganguli at Anthropic
- Please schedule a project midpoint discussion with your project's assigned mentor some time this or next week.

Eberly Center Focus Group

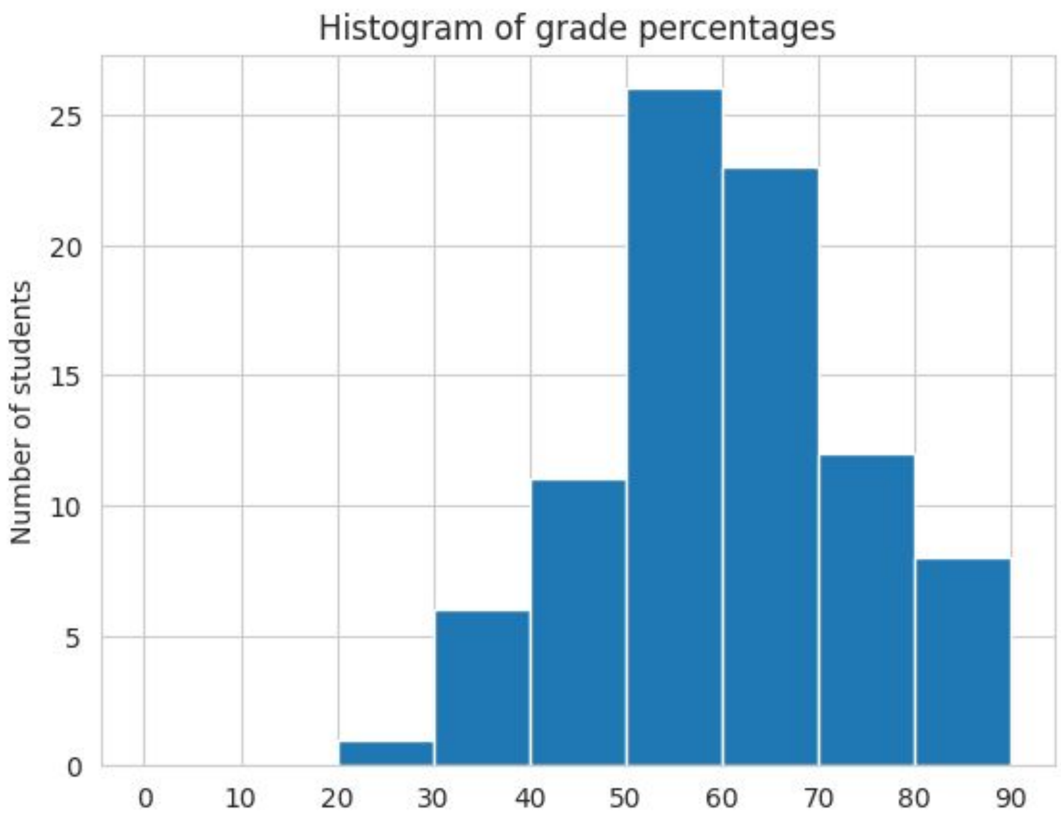
Thanks so much to those who participated in this! And thanks for being guinea pigs!

Your suggestions:

- Give a broader overview about how concepts interconnect.
  - Noted!
- Have more lectures on multimodal applications.
  - Noted!
- Provide practice questions before the midterm
  - This is a graduate level class. We want you to learn the material we are covering because we think it is important, not for the goal of studying to a particular exam format.
- Set more clear expectations for the project
  - Noted!
- Better balance homework difficulty.
  - Noted!

Midterm Results

Score Distributions (Raw score/63)

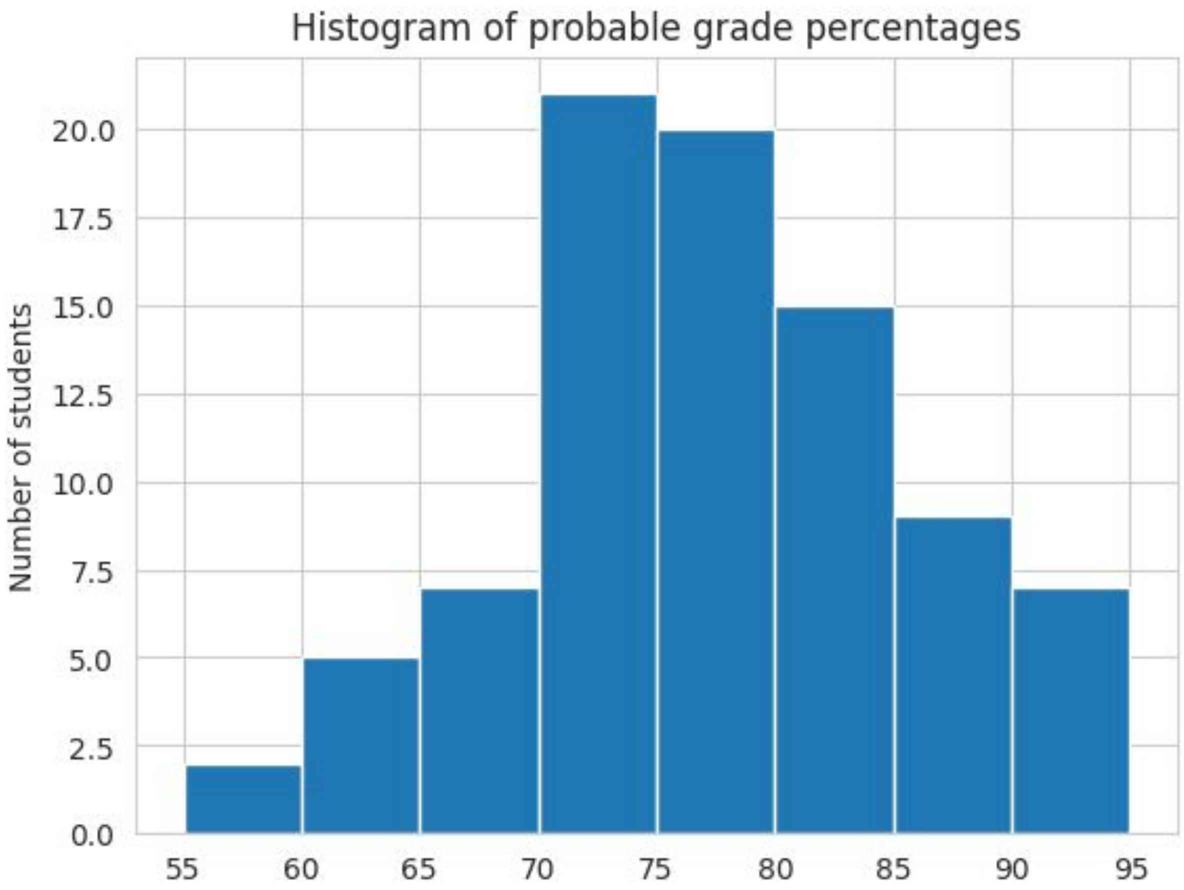



```
Minimum: 13.0
Median: 37.75
Maximum: 55.0
Mean: 37.68
Std Dev: 8.52
```

Midterm is only 20% of your total grade

Midterm Results

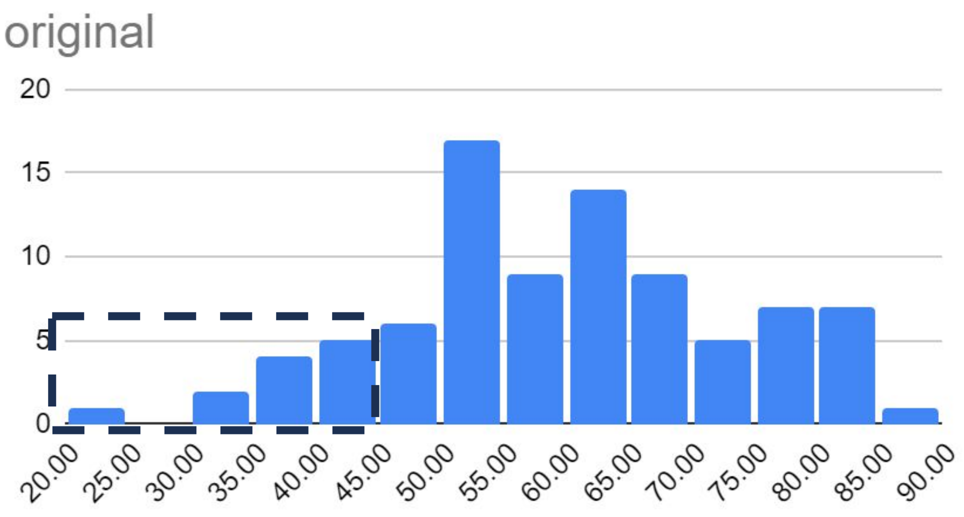
Potential curve : (your_grade = $\sqrt{\text{actual\_grade\_percentage}}$) / 10)

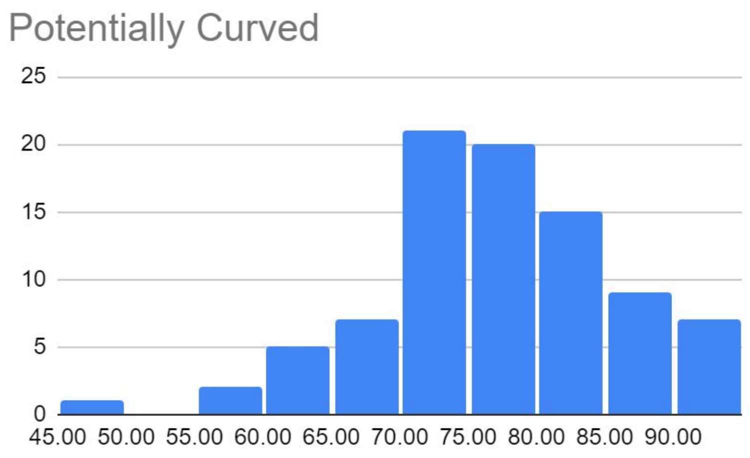


Midterm is only 20% of your total grade

Midterm Results

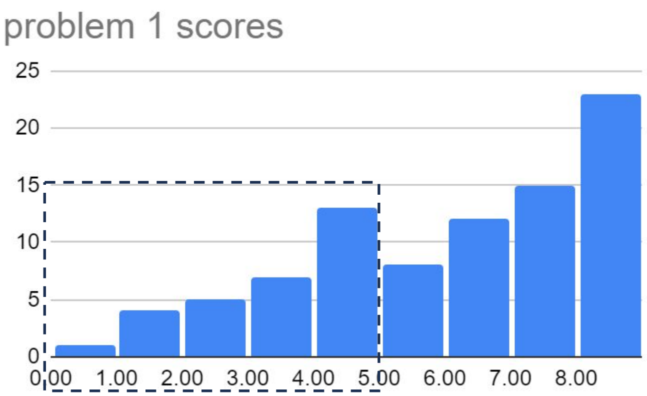
Score Distributions (Raw score/63)

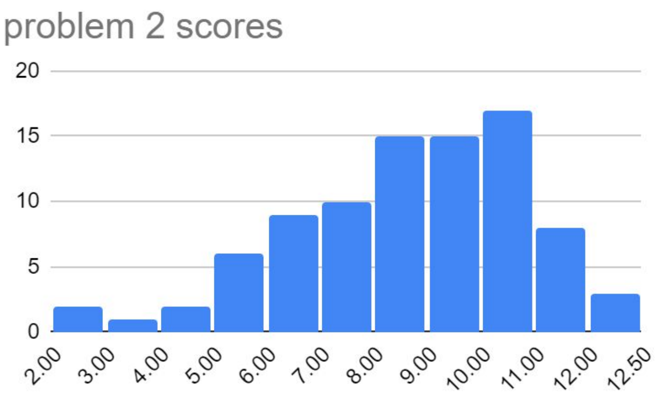


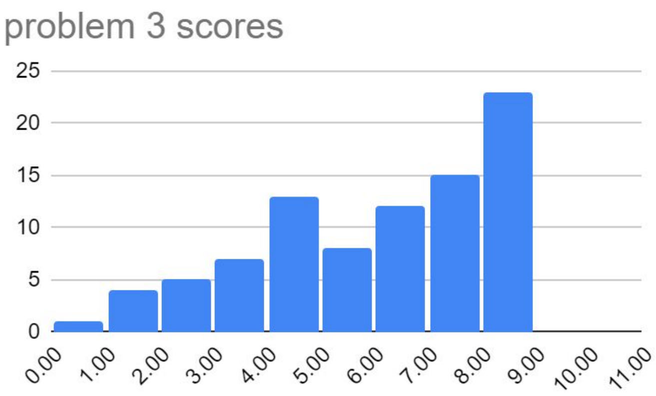


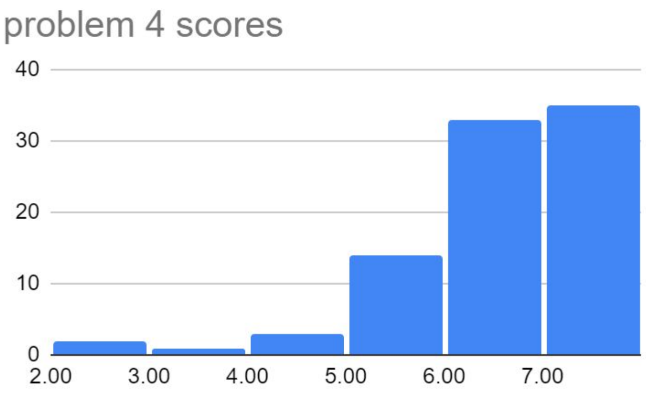
Midterm is only 20% of your total grade

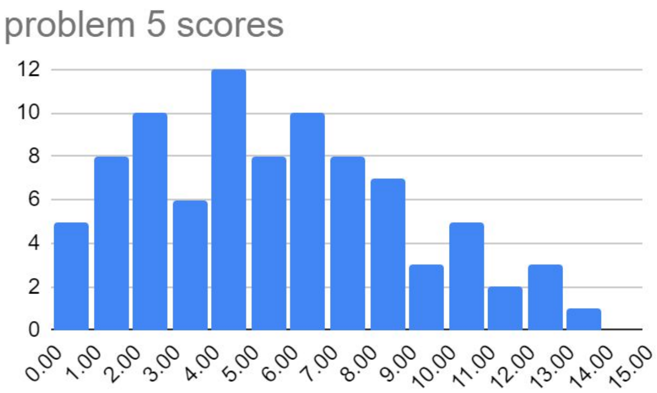
Midterm Results

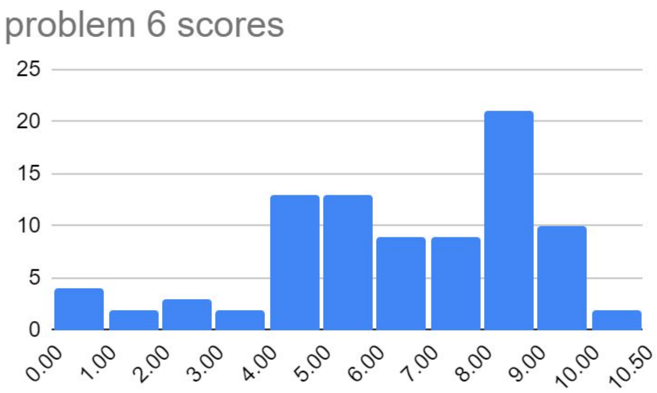












Challenging question: **2a**

Problem 2 [12 pts]. Training and training eata.

a) [**1 pt**] Why do we train to minimize the negative **log** likelihood rather than training to minimize the negative likelihood.

With 10s of thousands of tokens in the vocabulary, the **likelihood** of many of these tokens being the next token will be very, very small. Computers have numeric instability when trying to do mathematical operations on very small models.

$$P(y_t | y_{1:t-1}) = \prod_{i=1}^{t} P(y_i | y_{1:i-1})$$

Challenging question: **2a**

Problem 2 [12 pts]. Training and training eata.

a) [**1 pt**] Why do we train to minimize the negative **log** likelihood rather than training to minimize the negative likelihood.

With 10s of thousands of tokens in the vocabulary, the **likelihood** of many of these tokens being the next token will be very, very small. Computers have numeric instability when trying to do mathematical operations on very small models.

$$\log P(y_t|y_{1:t-1}) = \sum_{i=1}^{t} \log P(y_i|y_{1:i-1})$$

Challenging question: **3b.4**

4. Describe an experiment you could use to disprove the hypothesis that continuous prompts can be made interpretable via natural language.

Hint: consider two prefix-tuned prompts which both perform well on a task, but one maps to a helpful, natural language instruction, and the other maps to gibberish.

Possible experiment: do regular prompt tuning but with an extra loss encouraging the resulting prompt to be in close in embedding space to some arbitrary string of your choice.

If you get high-performing prompts no matter what string you choose, this suggests there is little correspondence between learned prompts and their discrete interpretations.

Challenging question: **4d**

d) [2 pts] In 1-2 sentences, describe an *extrinsic* human evaluation experiment you could do to show how MovieBot performs relative to a baseline model.

Extrinsic evaluation involves doing evaluation of an end-to-end system rather than just evaluating individual components of the system.

We were looking for answers that mentioned having humans interact with MovieBot directly (compared to just assessing pre-computed MovieBot generations).

Challenging question: **6d**

d) [6 pts] Suppose you are building an application which reads in structured metadata (e.g. names of teams, location of match etc.) and gameplay logs (e.g. spatio-temporal info on passes, shots, fouls, etc.) of a soccer match and generates a natural language summary of the game.

1) Describe the steps you would take to implement this application using a classical NLG pipeline.

We were expecting an answer following similar steps to the WeatherReporter case study from *Building NLG Systems* textbook, chapter 3.

Should at minimum have mentioned document planning followed by surface realization.

Classical NLG pipeline means **no language model**.

Scaling Up LLM Pretraining: Scaling Law

Chenyan Xiong

11-667

Outline

- Why Scaling Up
- Which Language Model to Scale Up
- What Factors Matter in Scaling
- What Configurations to Scale Up
- Capabilities Emerged from Scaling Up

Why Scaling Up

Almost guaranteed gains in downstream tasks

Figure 1: Performance of Turing-NLR V5 on MNLI at different model sizes and pretraining steps [1]

- Larger models, better fine-tuning accuracy
- More pretraining steps, better downstream performances

[1] Bajaj et al. "METRO: Efficient denoising pretraining of large scale autoencoding language models with model generated signals". arXiv 2022.

Why Scaling Up

More than just better leaderboard entries

- Significant gains in many real production scenarios
    - Name any existing AI product, likely it benefited from bigger LLMs
- Non-trivial gains from scaled up LLMs
    - Very hard to achieve with more sophisticated, but smaller models
- Distillable gains for efficient serving
    - Scaled up → Distill to smaller models better than pretraining smaller ones
- Deterministic gains
    - Research is risky.

    → Investment is for long term return of the world & human-beings.
    - Scaling up gains are deterministic.

    → Investment leads to predictable gains for "my" business.

[2] Kharya et al. "Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, the World's Largest and Most Powerful Generative Language Model". NVIDIA Blog 2022.

Why Scaling Up

More than just better leaderboard entries

- Significant gains in many real production scenarios
  - Name any existing AI product, likely it benefited from bigger LLMs
- Non-trivial gains from scaled up LLMs
  - Very hard to achieve with more sophisticated, but smaller models
- Distillable gains for efficient serving
  - Scaled up → Distill to smaller models better than pretraining smaller ones
- Deterministic gains
  - Research is risky.

  → Investment is for long term return of the world & human-beings.
  - Scaling up gains are deterministic.

  → Investment leads to predictable gains for “my” business.

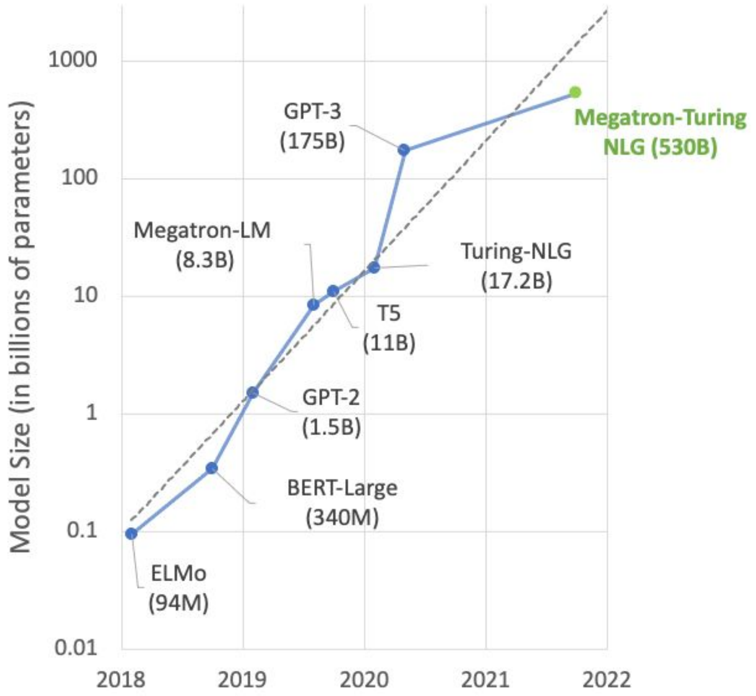
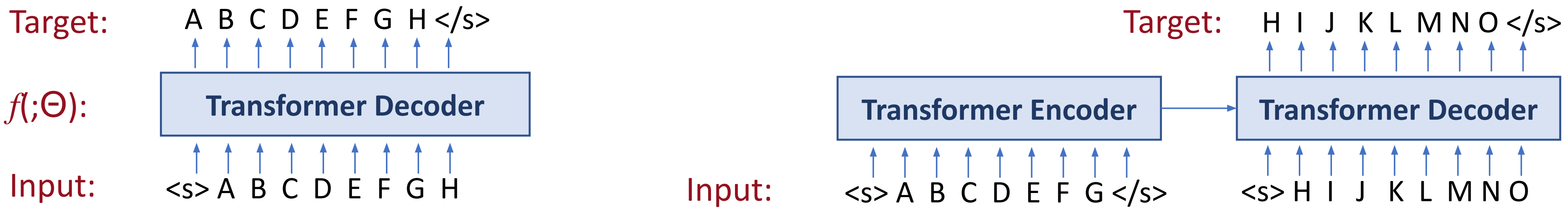


Figure 2: Growth of LLM parameter size as of 2022 [2]

[2] Kharya et al. “Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, the World’s Largest and Most Powerful Generative Language Model”. NVIDIA Blog 2022.

Which Language Model: Architecture

Decoder or Encode-decoder?



Encoder is out of consideration because

1. Encoder-decoder covers the functionality of encoder
2. Hard to do generation with encoder-only

Which Language Model: Pretraining Tasks

Auto-regressive (Causal) LM, Pre-fix (Non-Causal) LM or Denoising Masked-LM?

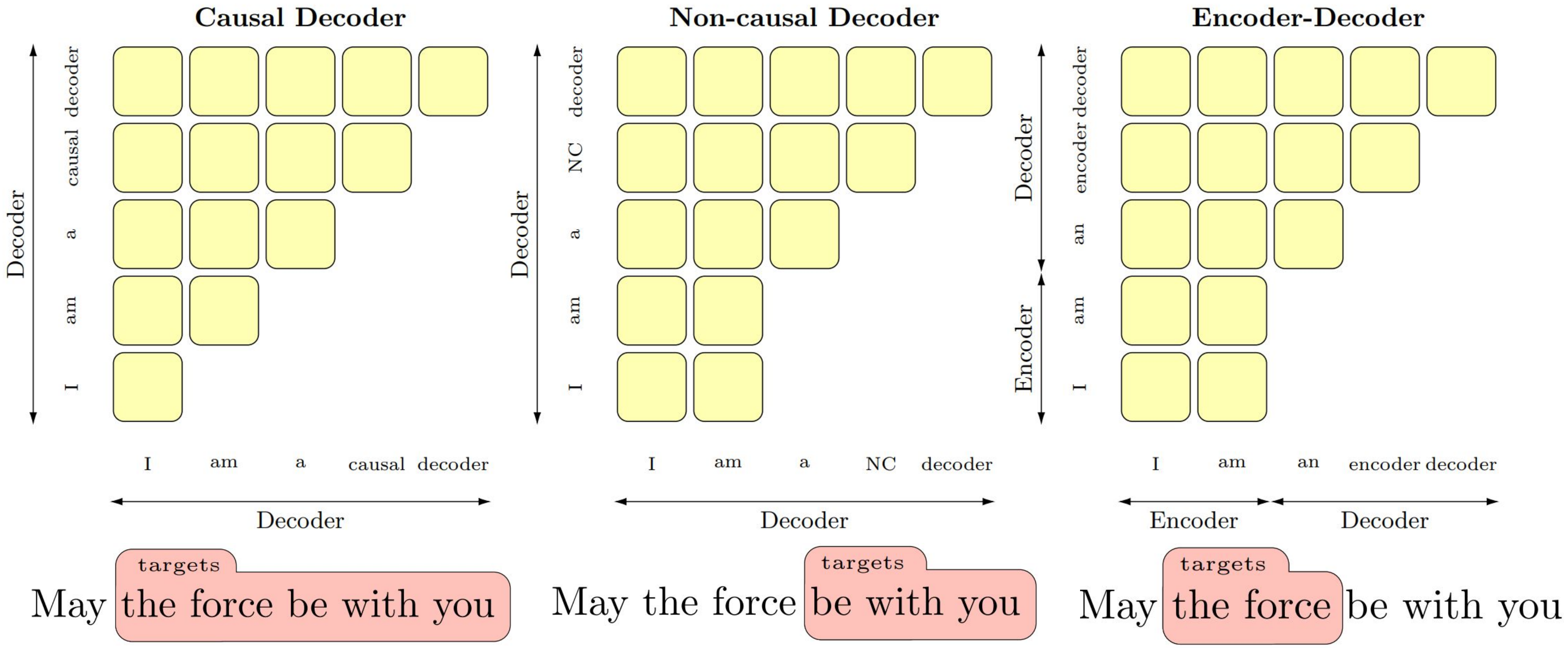


Figure 3: Attention Masks and Pretraining Tasks of Different LLM Architectures [3].

[3] Wang et al. "What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?". ICML 2022.

Which Language Model: Empirical Studies

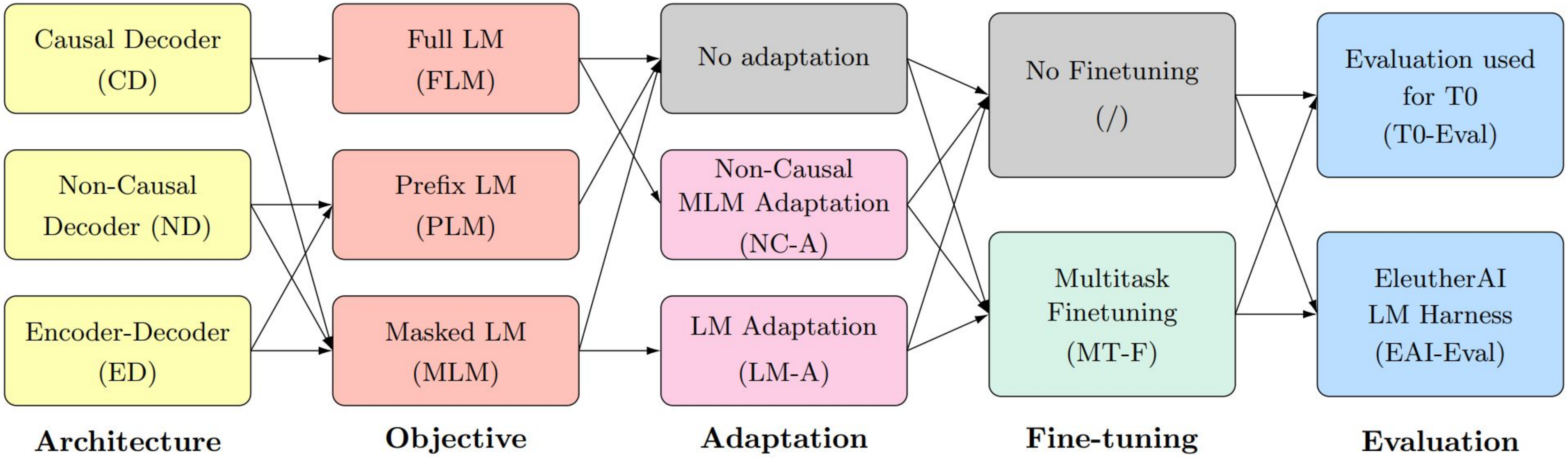


Figure 4: Empirical Study Pipeline on Different Language Model Configurations [3].

[3] Wang et al. "What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?". ICML 2022.

Which Language Model: Empirical Studies

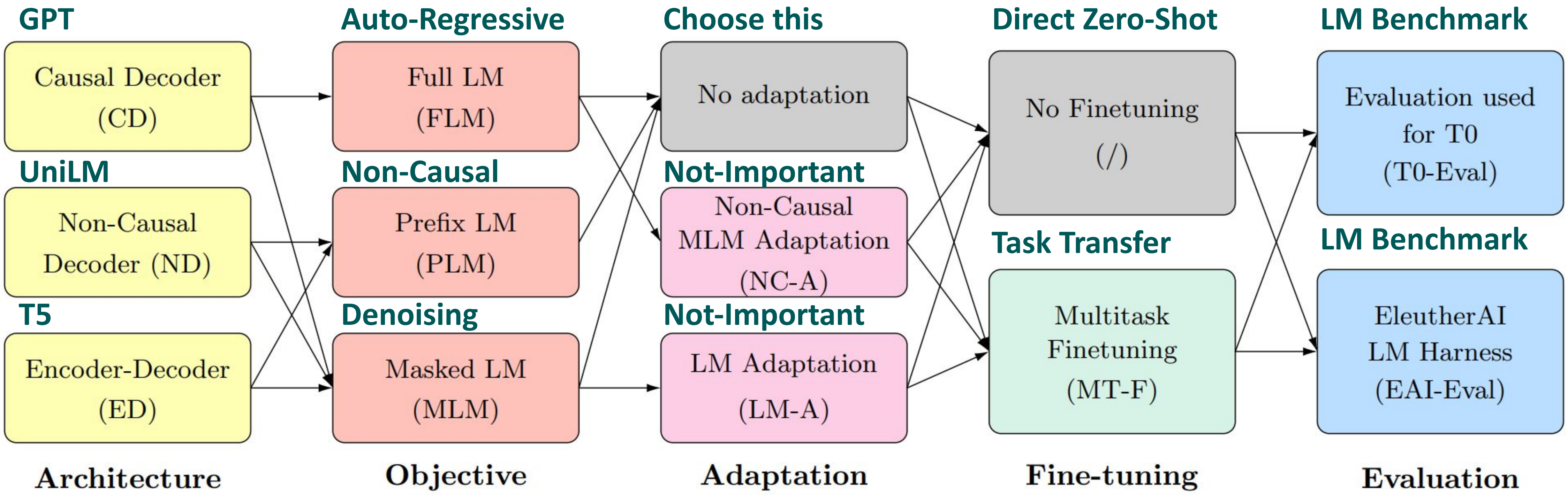


Figure 4: Empirical Study Pipeline on Different Language Model Configurations [3].

[3] Wang et al. “What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?”. ICML 2022.

Which Language Model: Empirical Studies

Experimental Settings

	Models architecture	
	Decoder-only	Encoder-decoder
Parameters	4.8B	11.0B
Vocabulary	32,128	
Positional embed.	T5 relative	
Embedding dim.	4,096	
Attention heads	64	
Feedforward dim.	10,240	
Activation	GEGLU [Shazeer, 2020]	
Layers	24	48
Tied embeddings	True	
Precision	`bfloat16`	

	Pretraining	Multitask finetuning
Dataset	C4	T0-Train
Steps	131,072	10,000
Batch size in tokens	1,282,048	1,310,720
Optimizer	Adafactor(decay_rate=0.8)	
LR schedule	$\frac{1}{\sqrt{\max(n, 10^4)}}$	fixed, 0.001
Dropout	0.0	0.1
z loss	0.0001	
Precision	`bfloat16`	

Table 1: Experimental Settings following T5 pretraining and T0 finetuning configurations [3].

[3] Wang et al. "What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?". ICML 2022.

Which Language Model: Empirical Results

Performances in direct zero-shot, evaluated immediately after self-supervised pretraining, no finetuning.

	EAI-EVAL	T0-EVAL
Causal decoder	**44.2**	**42.4**
Non-causal decoder	43.5	41.8
Encoder-decoder	39.9	41.7
Random baseline	32.9	41.7

Table 1: Experimental Settings following T5 configurations [3].

Decoder only models pretrained with auto-regressive language modeling tasks performances significantly better under the same pretraining configurations.

[3] Wang et al. "What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?". ICML 2022.

Which Language Model: Empirical Results

Performances after multi-task finetuning

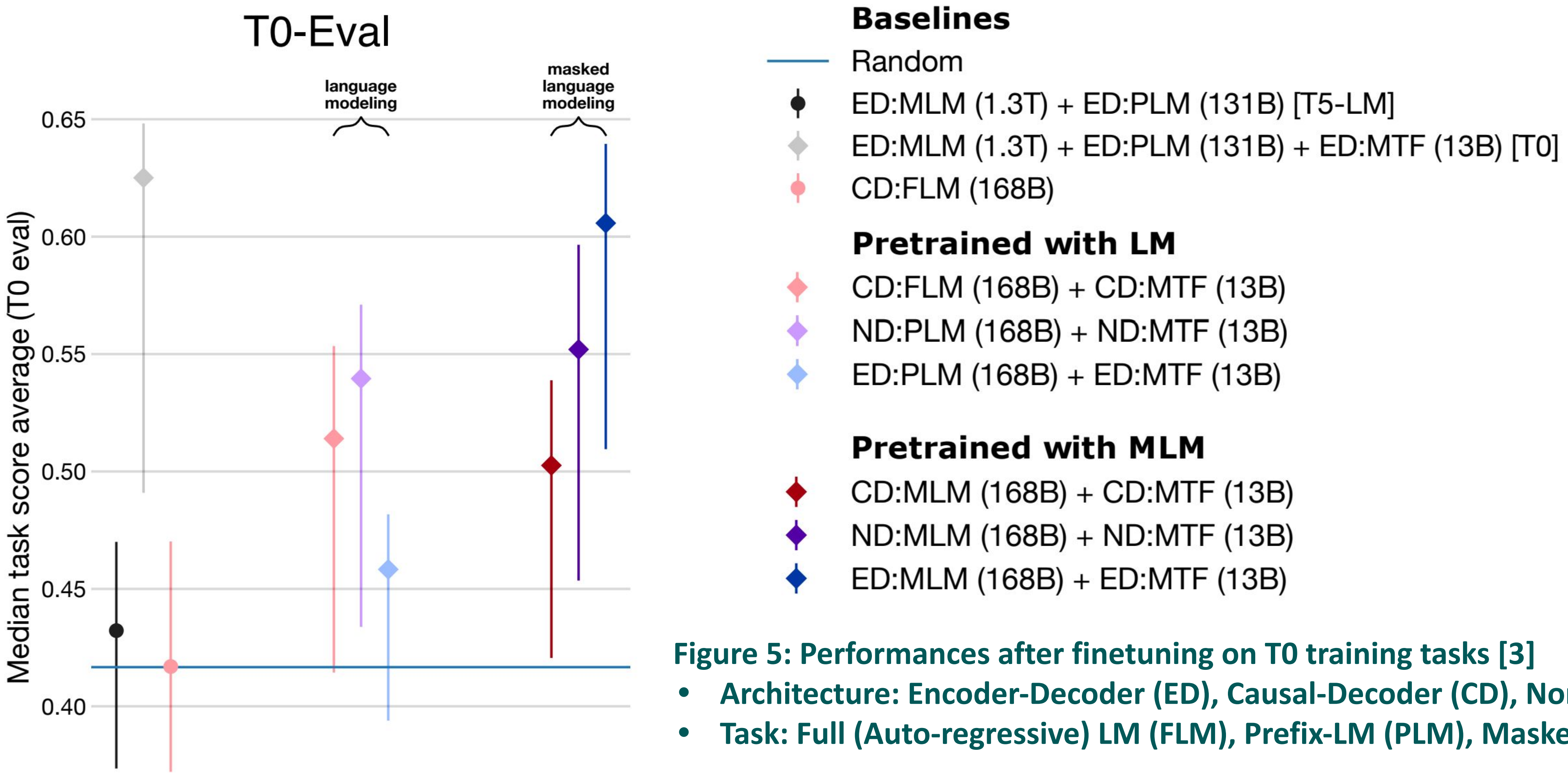


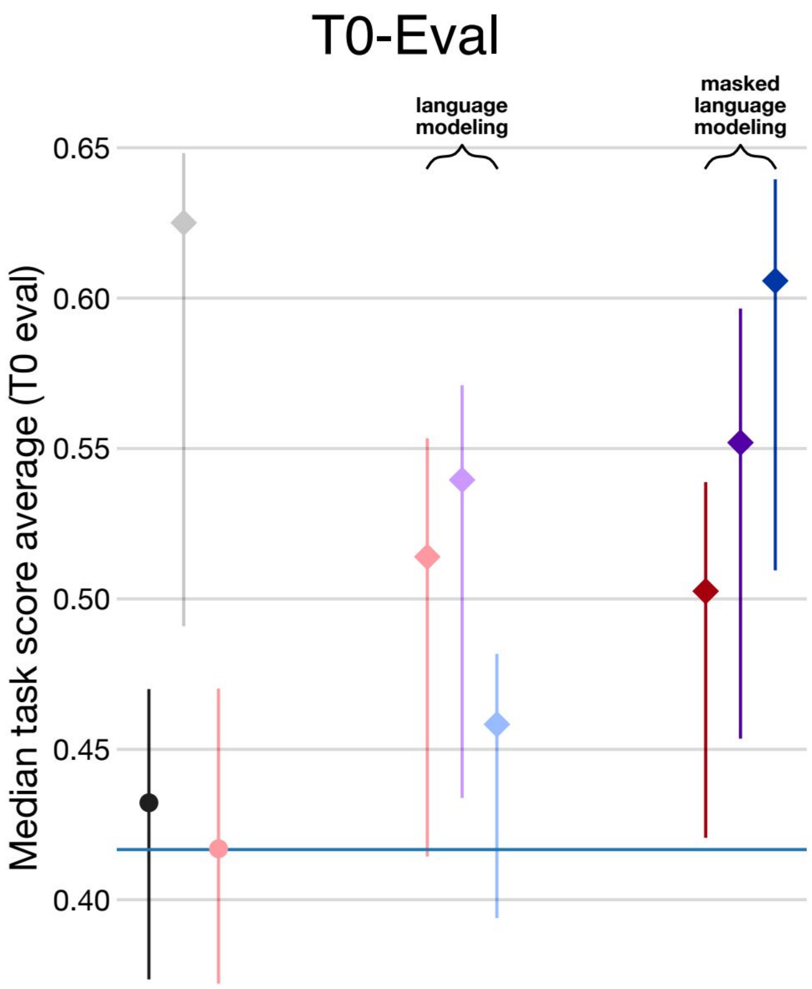
Figure 5: Performances after finetuning on T0 training tasks [3]

- **Architecture: Encoder-Decoder (ED), Causal-Decoder (CD), Non-Casual-Decoder (ND)**
- **Task: Full (Auto-regressive) LM (FLM), Prefix-LM (PLM), Masked-LM (MLM)**

[3] Wang et al. “What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?”. ICML 2022.

Which Language Model: Empirical Results

Performances after multi-task finetuning



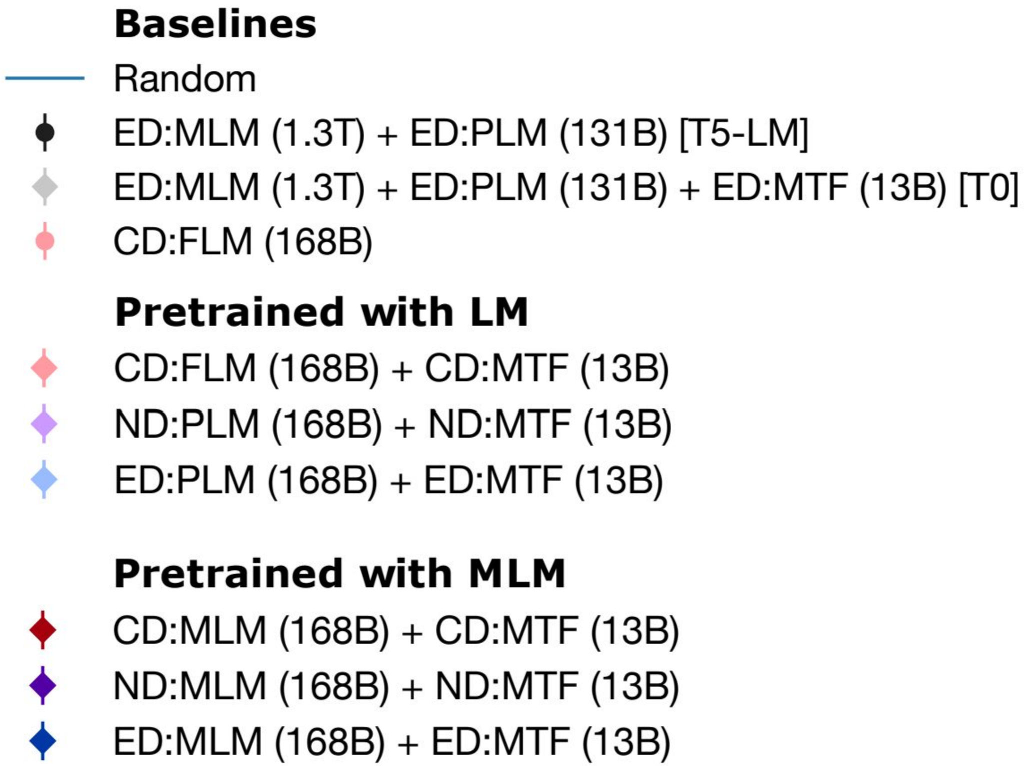


Figure 5: Performances after finetuning on T0 training tasks [3]

- **Architecture: Encoder-Decoder (ED), Causal-Decoder (CD), Non-Casual-Decoder (ND)**
- **Task: Full (Auto-regressive) LM (FLM), Prefix-LM (PLM), Masked-LM (MLM)**

Encoder-decoder and MLM performs best after multi-task fine-tuning

[3] Wang et al. “What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?”. ICML 2022.

Which Language Model: Conclusion

Popular choice: Decoder-only models + Auto-regressive language models

- Empirical results: better generalization right after pretraining, no multi-task supervised learning needed

Which Language Model: Conclusion

Popular choice: Decoder-only models + Auto-regressive language models

- Empirical results: better generalization right after pretraining, no multi-task supervised learning needed

Easy to scale up

- More training signals per sequence: 100% versus 15%

- Converges faster [empirical observations]
- More stable [hands-on observations]

Which Language Model: Conclusion

Popular choice: Decoder-only models + Auto-regressive language models

- Empirical results: better generalization right after pretraining, no multi-task supervised learning needed

Easy to scale up

- More training signals per sequence: 100% versus 15%

- Converges faster [empirical observations]
- More stable [hands-on observations]

OpenAI’s choice

- There is perhaps only one seat for the largest LLM.
- GPT-3 won at that certain point, and took that niche
- Everyone else followed, no evidence to gamble with $$$$$$$

Outline

- Why Scaling Up
- Which Language Model to Scale Up
- **What Factors Matter in Scaling**
- What Configurations to Scale Up
- Capabilities Emerged from Scaling Up

Scaling Factors

Many factors in configuring a scaled up pretraining run for Transformer Decoder + Autoregressive LM

- Model size (parameter counts)
- Pretraining dataset size
- Pretraining compute (FLOPs or TPU/GPU hours)
- Network shape (Parameters allocations)
- Effective batch size
- Learning rate & learning rate schedular
- Context length

Scaling Factors

Many factors in configuring a scaled up pretraining run for Transformer Decoder + Autoregressive LM

- Model size (parameter counts)
- Pretraining dataset size
- Pretraining compute (FLOPs or TPU/GPU hours)
- Network shape (Parameters allocations)

Main factors to study

- Effective batch size
- Learning rate & learning rate schedular

Hyper-parameters with rule of thumb

1. Batch size determined by GPU memory
2. Try biggest LR before blowing up

- Context length

Balance complexity and downstream needs

Scaling Law Study: Setup

Empirically study the relationship between various factors to language model performances [4]

- Model: GPT-style, auto-regressive loss, maximum 1.5 billion non-embedding parameters
- Pretraining data: WebText2, harvest from Reddit out links, at max 23 billion tokens
- Metric: language modeling loss on testing data

[4] Kaplan et al. “Scaling Laws for Neural Language Models”. arXiv 2020.

Scaling Law Study: Observations

Network shape (allocation of parameters at different parts) does not matter as much

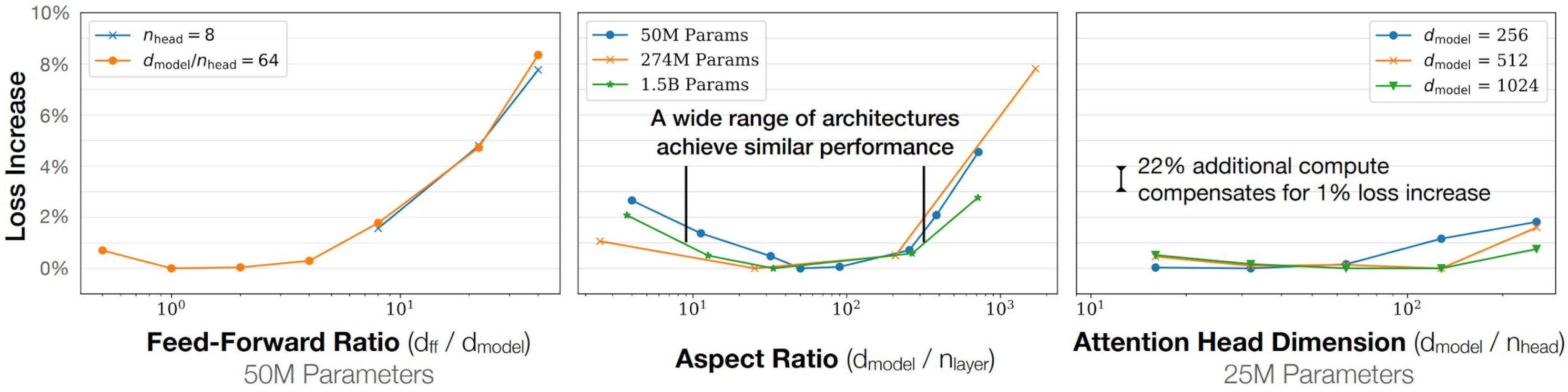


Figure 6: Language model loss changes with different network shape configurations [4].

- As long as the network shape is in a general sweet range, it does not impact performance much

[4] Kaplan et al. "Scaling Laws for Neural Language Models". arXiv 2020.

Scaling Law Study: Observations

A clear mapping from compute, data size, and parameter counts to testing loss

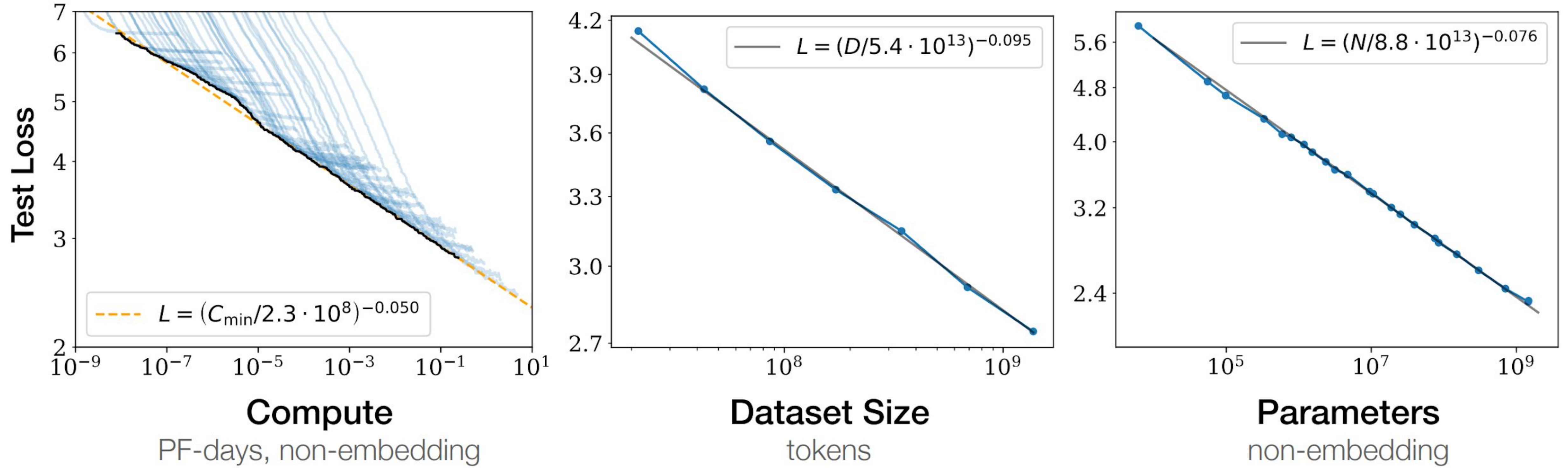


Figure 7: Mapping from compute (Peta-Flops days), data size, and model parameters to language modeling loss on testing data [4].

Scaling Law Study: Observations

A clear mapping from compute, data size, and parameter counts to testing loss

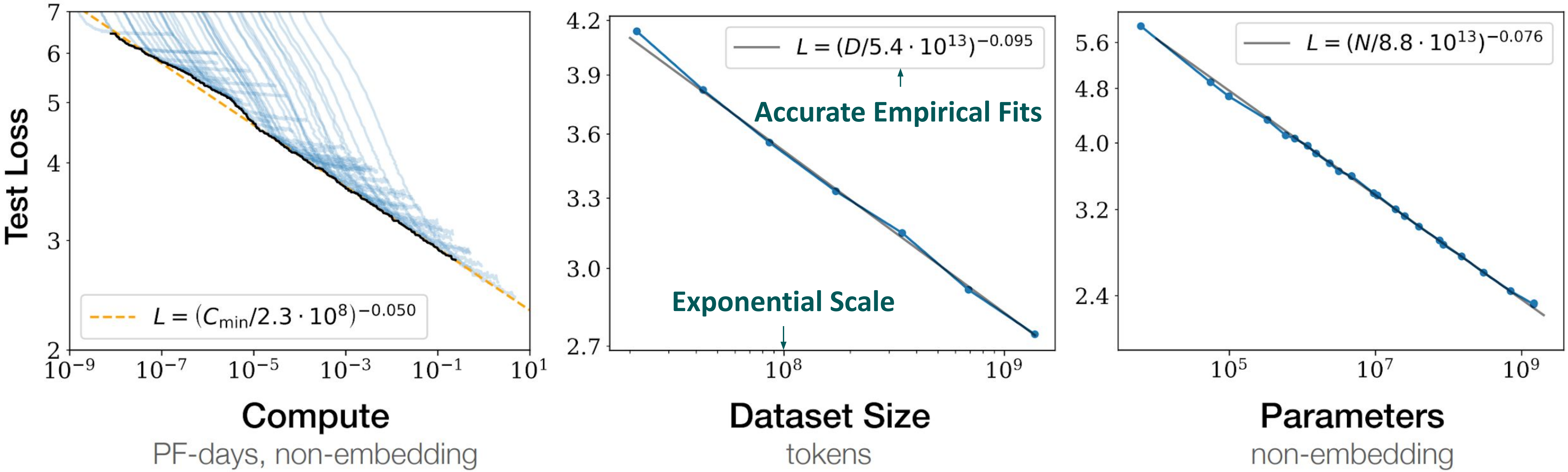


Figure 7: Mapping from compute (Peta-Flops days), data size, and model parameters to language modeling loss on testing data [4].

- Linear increasement of language modeling accuracy requires exponential scaling
- Three factors need to scale jointly to reach target model performance improvements

Scaling Law Study: Observations

Network parameters matter more than embedding parameters

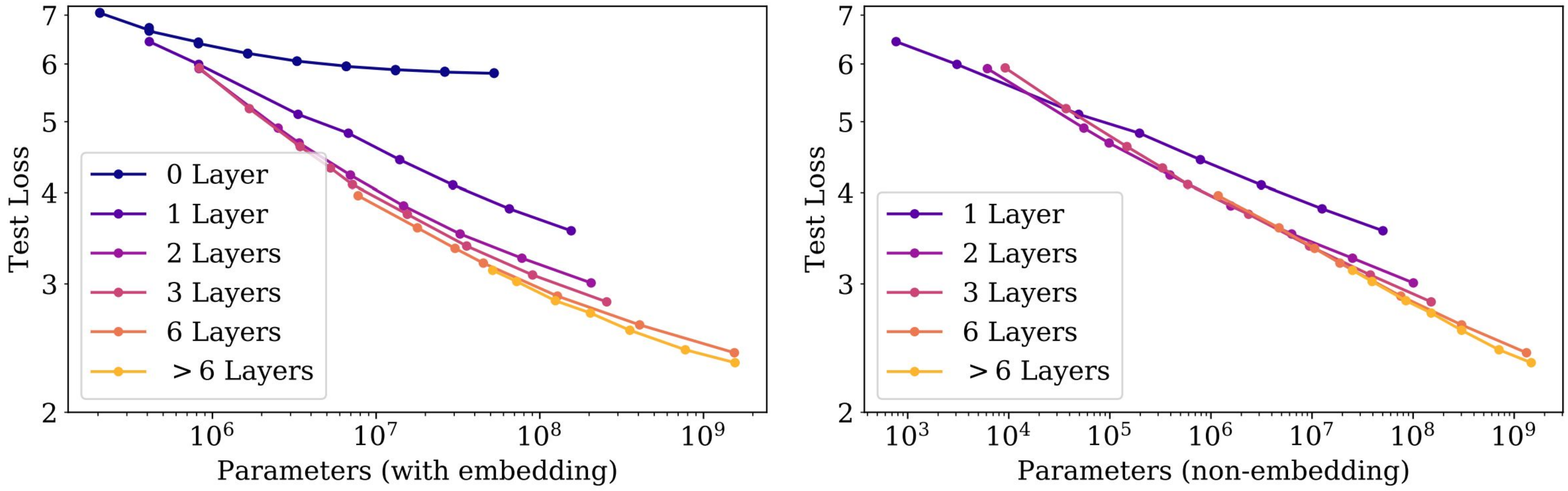


Figure 8: Scaling law with network parameter counts include (left) and exclude (right) embeddings [4].

[4] Kaplan et al. “Scaling Laws for Neural Language Models”. arXiv 2020.

Scaling Law Study: Observations

How large the pretraining corpus should be given target pretraining steps in tokens?

- Large corpus leads to fewer repetitions (epochs)

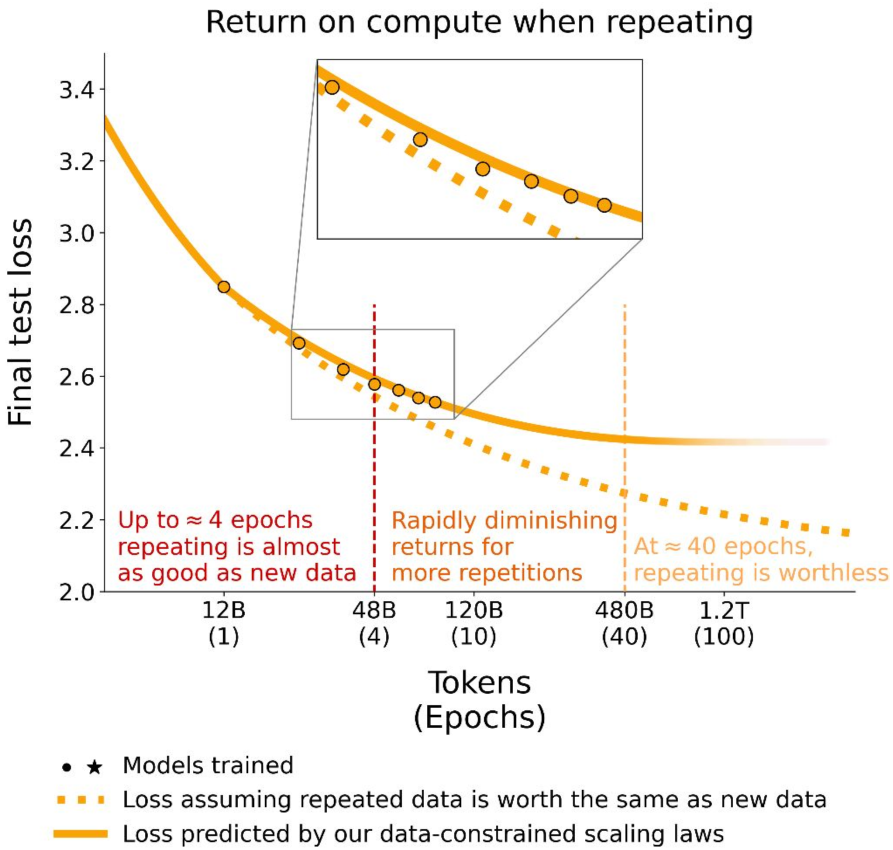


Figure 9: Scaling law with data repetitions [5].

Better to collect more data:

- Fewer than four repetitions is fine.
- More leads to diminishing returns.

Scaling Law Study: Observations

Language modeling loss correlates well with downstream performances

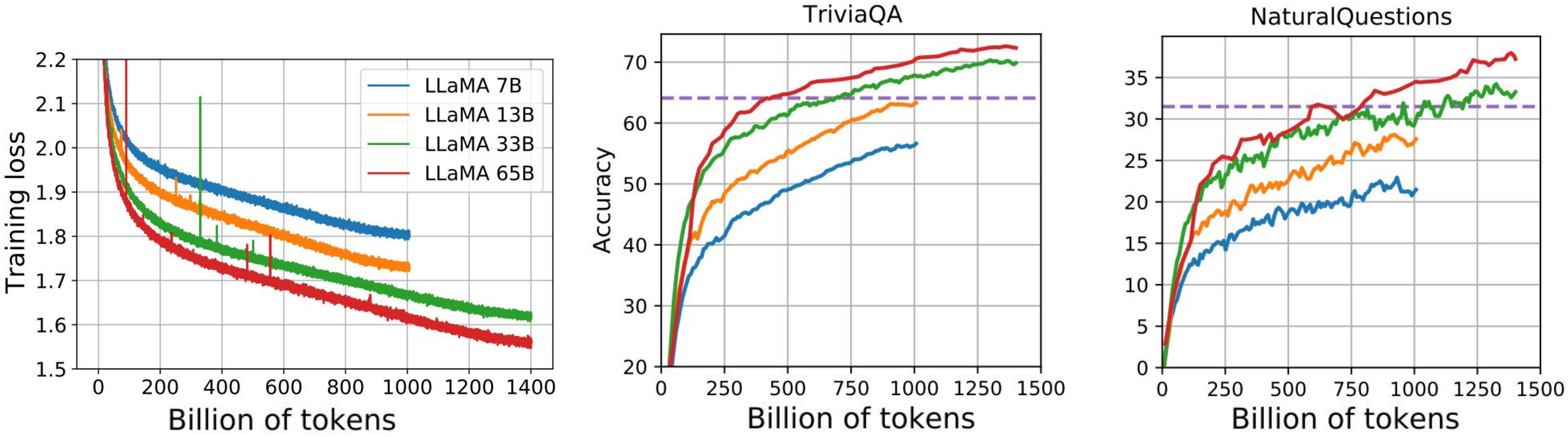


Figure 10: Pretraining loss and downstream zero-shot accuracy during LLaMA pretraining steps [6]

[6] Touvron et al. "LLaMA: Open and Efficient Foundation Language Models". arXiv 2023.

Scaling Law: Recap

Scaling law: A clear mapping from scaling factors to language modeling accuracy

- Given the same model family, data distribution, techniques, etc.
- Exponential scaling law between data size, model size, and computing FLOPs

Scaling Law: Recap

Scaling law: A clear mapping from scaling factors to language modeling accuracy

- Given the same model family, data distribution, techniques, etc.
- Exponential scaling law between data size, model size, and computing FLOPs

What does this mean?

- More predictable bet on scaling up?

→Using observations at smaller scale to determine

- Deterministic but diminishing return?

→ Exponential cost, linear accuracy gains

Outline

- Why Scaling Up
- Which Language Model to Scale Up
- What Factors Matter in Scaling
- **What Configurations to Scale Up**
- Capabilities Emerged from Scaling Up

What Configurations to Scale Up

Goal: Given a computing budget and a candidate language model, select the optimal scaling up configurations

- E.g., One million H100 hours, pretrain the best LLaMA style LLM
- Configurations to choose: Model size (# of parameters) and pretraining data size (# of tokens)

[7] Hoffmann et al. "Training Compute-Optimal Large Language Models". arXiv 2022.

What Configurations to Scale Up

Goal: Given a computing budget and a candidate language model, select the optimal scaling up configurations

- E.g., One million H100 hours, pretrain the best LLaMA style LLM
- Configurations to choose: Model size (# of parameters) and pretraining data size (# of tokens)

A common question when scaling up

- Computing budget is the biggest constraint and is often pre-given and limited
- No room for exploration at target scale
- Only one scaled up pretraining run allowed, both budget-wise and time-wise.

[7] Hoffmann et al. "Training Compute-Optimal Large Language Models". arXiv 2022.

What Configurations to Scale Up

Goal: Given a computing budget and a candidate language model, select the optimal scaling up configurations

- E.g., One million H100 hours, pretrain the best LLaMA style LLM
- Configurations to choose: Model size (# of parameters) and pretraining data size (# of tokens)

A common question when scaling up

- Computing budget is the biggest constraint and is often pre-given and limited
- No room for exploration at target scale
- Only one scaled up pretraining run allowed, both budget-wise and time-wise.

Solution: Scaling law

- Use many experiments at small scale to establish the scaling law
- Use scaling Law to predict best configuration at target compute

Scaling Configuration: Empirical Scaling Law

Empirical Approach #1: Fix model size and varying pretraining tokens [7]

1. Pretrain different sized models to near converge and track loss
2. Record best (model size, data size) at each FLOP.
3. Estimate the scaling law

[7] Hoffmann et al. "Training Compute-Optimal Large Language Models". arXiv 2022.

Scaling Configuration: Empirical Scaling Law

Empirical Approach #1: Fix model size and varying pretraining tokens [7]

1. Pretrain different sized models to near converge and track loss
2. Record best (model size, data size) at each FLOP.
3. Estimate the scaling law

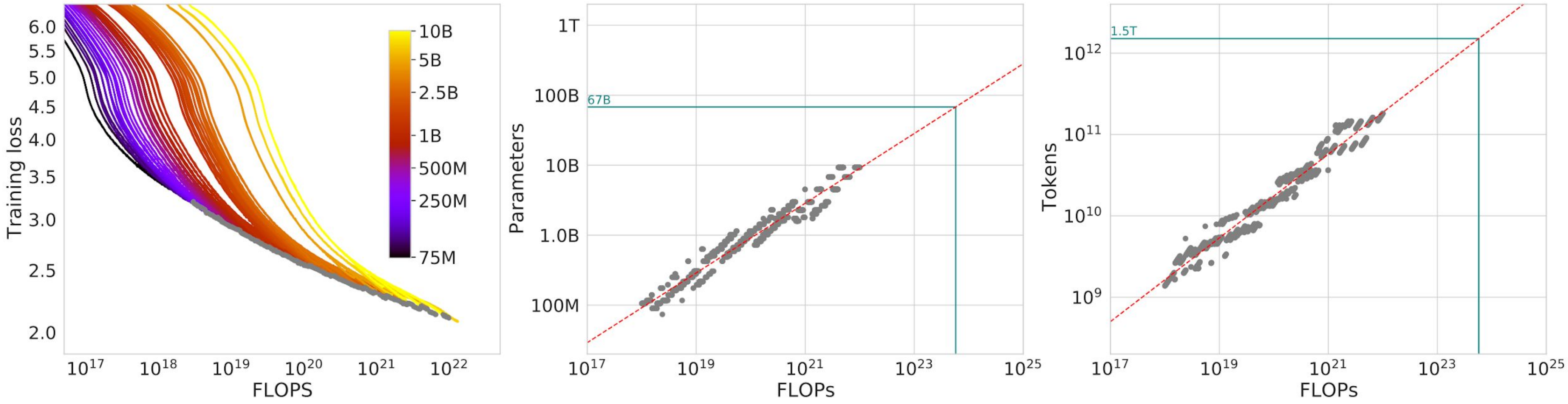


Figure 11: Pretraining loss of varying model (left), and the identified optimal parameters (mid) and tokens (right) at different FLOPS [6]

[7] Hoffmann et al. "Training Compute-Optimal Large Language Models". arXiv 2022.

Scaling Configuration: Empirical Scaling Law

Empirical Approach #1: Fix model size and varying pretraining tokens [7]

1. Pretrain different sized models to near converge and track loss
2. Record best (model size, data size) at each FLOP.
3. Estimate the scaling law

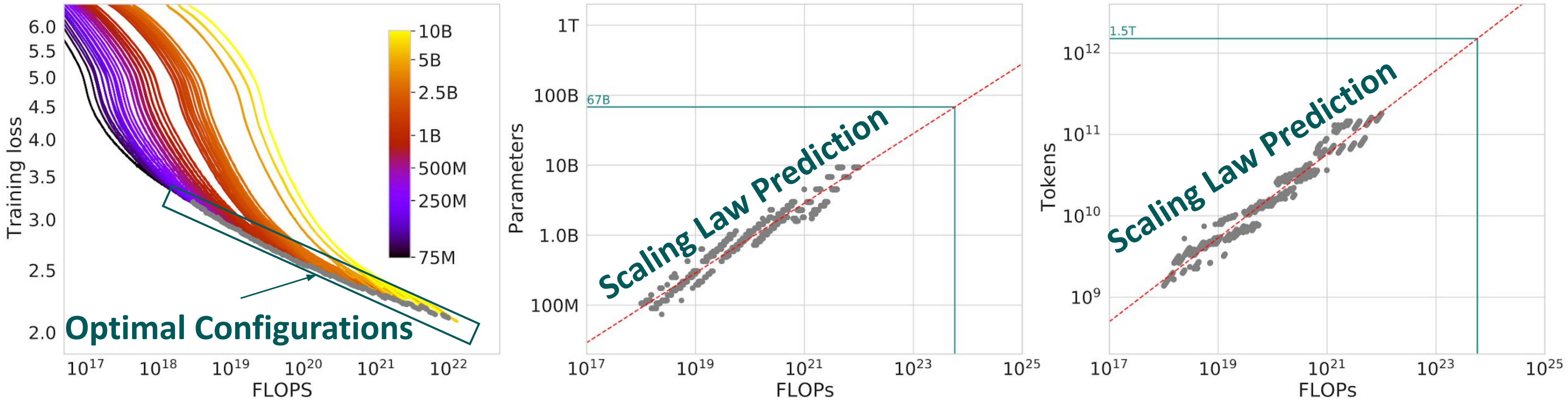


Figure 11: Pretraining loss of varying model (left), and the identified optimal parameters (mid) and tokens (right) at different FLOPS [6]

Scaling Configuration: Empirical Scaling Law

Empirical Approach #2: Fix total FLOPs, pretrain different sized models [7]

1. Pretrain to the # of tokens using total FLOPs and track final loss
2. Track best configurations and vary the total FLOPs and rerun #1
3. Estimate the scaling law

[7] Hoffmann et al. “Training Compute-Optimal Large Language Models”. arXiv 2022.

Scaling Configuration: Empirical Scaling Law

Empirical Approach #2: Fix total FLOPs, pretrain different sized models [7]

1. Pretrain to the # of tokens using total FLOPs and track final loss
2. Track best configurations and vary the total FLOPs and rerun #1
3. Estimate the scaling law

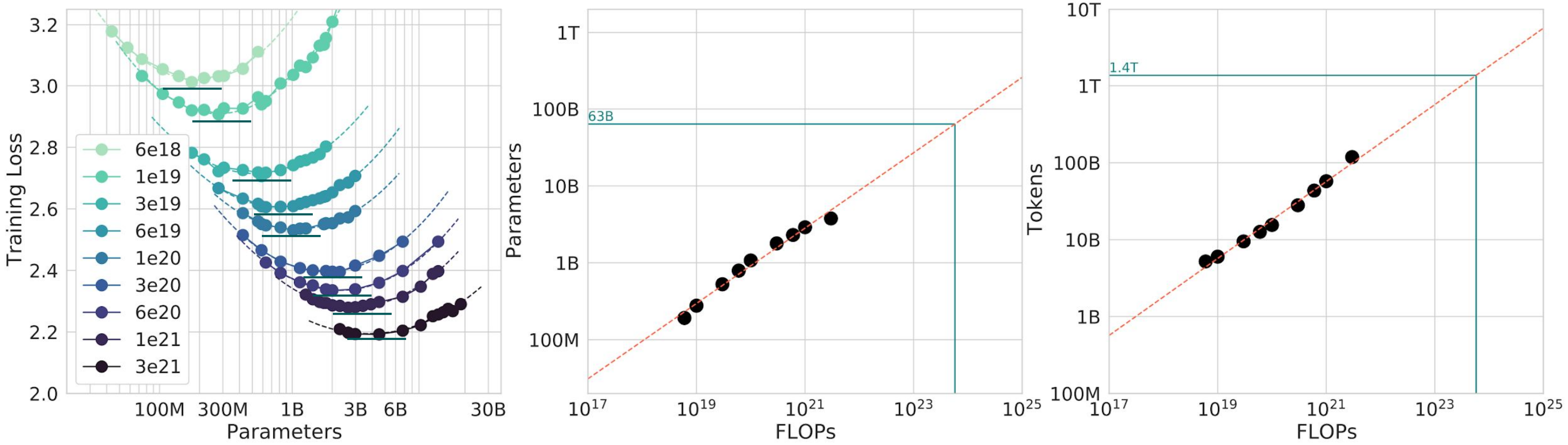


Figure 12: Pretraining loss of varying model sizes at varying FLOPs (left), and the identified optimal parameters (mid) and tokens (right) [6]

[7] Hoffmann et al. "Training Compute-Optimal Large Language Models". arXiv 2022.

Scaling Configuration: Empirical Scaling Law

Empirical Approach #2: Fix total FLOPs, pretrain different sized models [7]

1. Pretrain to the # of tokens using total FLOPs and track final loss
2. Track best configurations and vary the total FLOPs and rerun #1
3. Estimate the scaling law

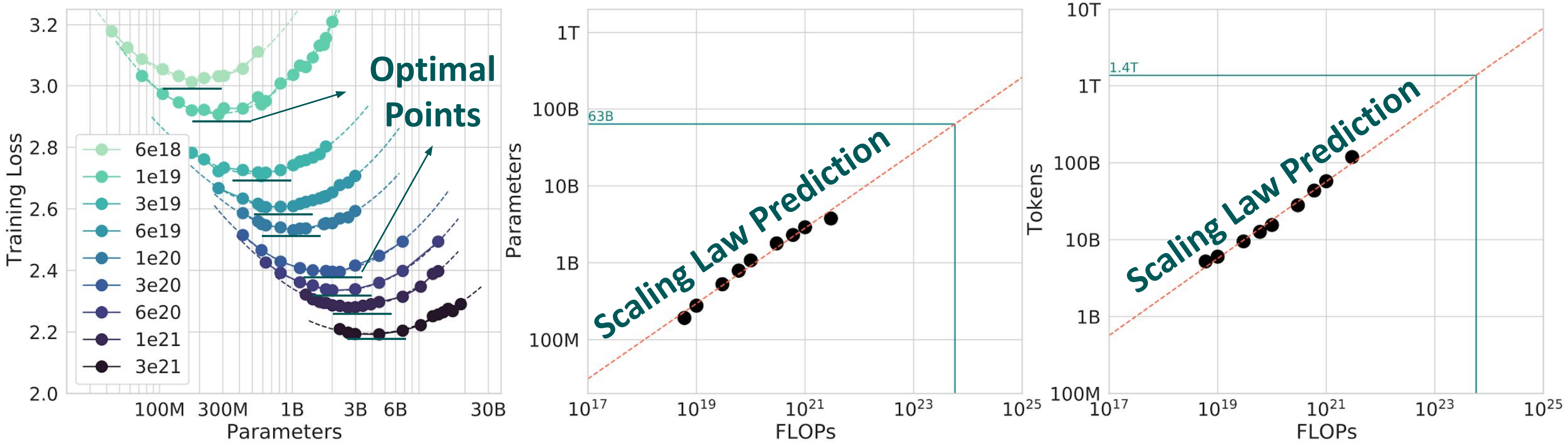


Figure 12: Pretraining loss of varying model sizes at varying FLOPs (left), and the identified optimal parameters (mid) and tokens (right) [6]

[7] Hoffmann et al. “Training Compute-Optimal Large Language Models”. arXiv 2022.

Scaling Configuration: Empirical Scaling Law

Empirical Approach #3: Using data points collected from previous two approaches and fix a parametric functions

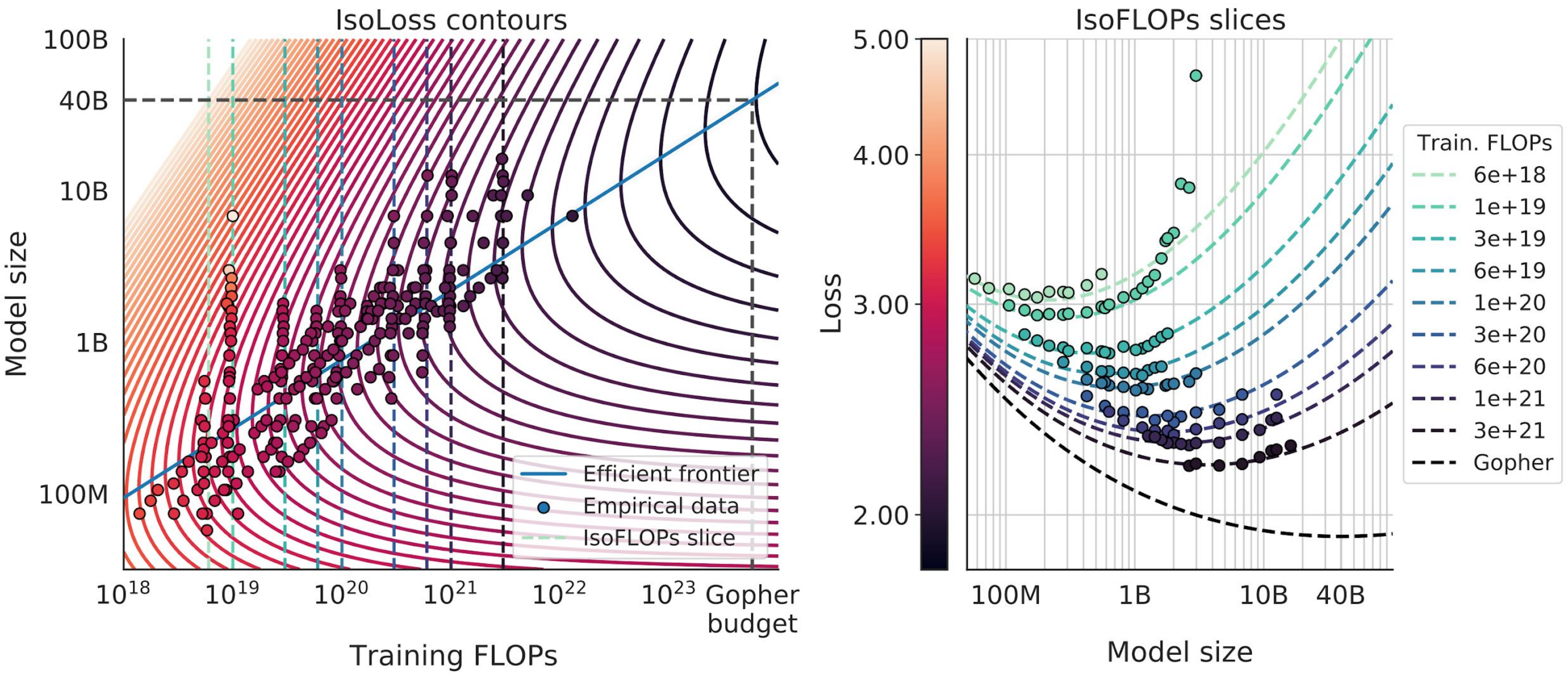


Figure 13: Fitted parametric function of (model size, FLOPs)→Loss using data from approach one (left) and two (right) [6]

Scaling Configuration: Estimated Optimal Configurations

Applying Empirical Approach #1 to common parameter settings

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

Table 2: Examples of estimated scaling configurations at different model sizes [3].

Back-of-envelop calculation: 1e+24 FLOPs ≈ 1 Million A100 Hours/40K A100 Days.

- The one used to pretrain LLaMA-65B
- 512 A100 for 3 months

[7] Hoffmann et al. "Training Compute-Optimal Large Language Models". arXiv 2022.

Scaling Configuration: Performances

Chinchilla: Use scaling law predicted configurations at the same FLOPs of Gopher

- Chinchilla (predicted optimal): 70B parameters and 1.4T (4X) Tokens
- Gopher (guessed setup): 280B (4X) parameters and 300B Tokens

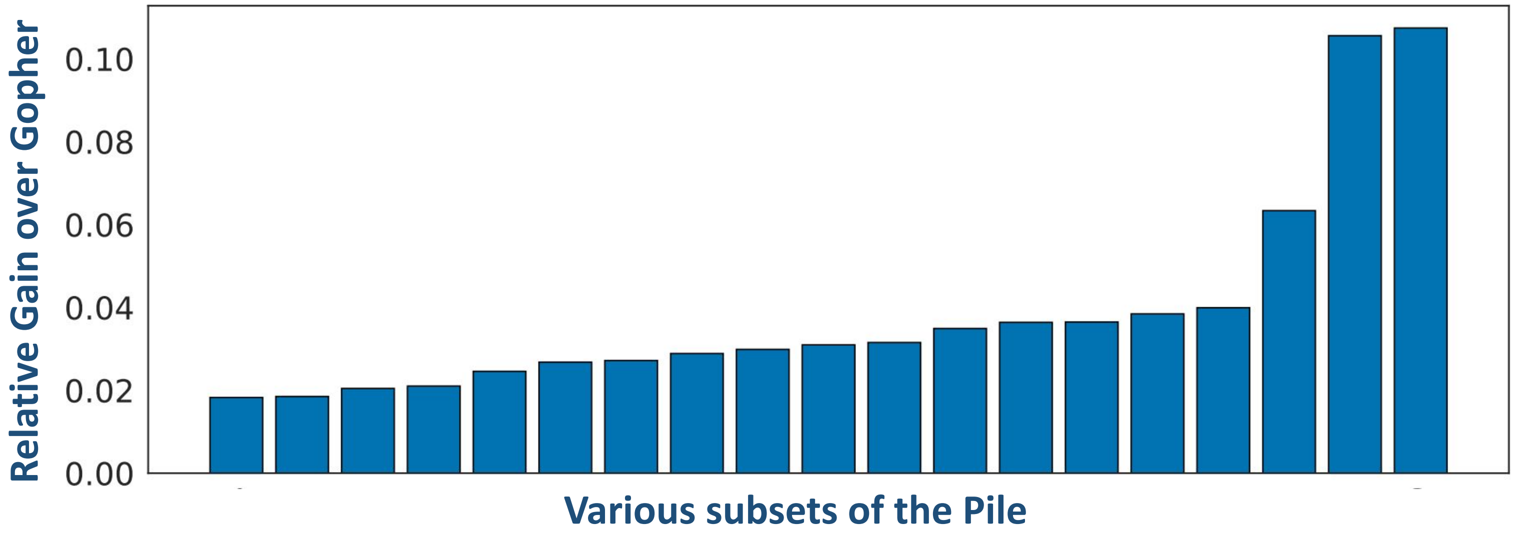


Figure 14: Chinchilla's Language model accuracy gains on different corpora from the Pile [6]

[7] Hoffmann et al. "Training Compute-Optimal Large Language Models". arXiv 2022.

Scaling Configuration: Performances

Universal improvements on various downstream scenarios

- MMLU, BigBench, Close book QA, etc.

	Method	*Chinchilla*	*Gopher*	GPT-3
Natural Questions (dev)	0-shot	16.6%	10.1%	14.6%
	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%	-
	5-shot	64.1%	57.0%	-
	64-shot	64.6%	57.2%	-

Table 2: Close book QA results [3].

[7] Hoffmann et al. "Training Compute-Optimal Large Language Models". arXiv 2022.

Scaling Configuration: Remarks

Scaling law universally exists, but the specific functions differ

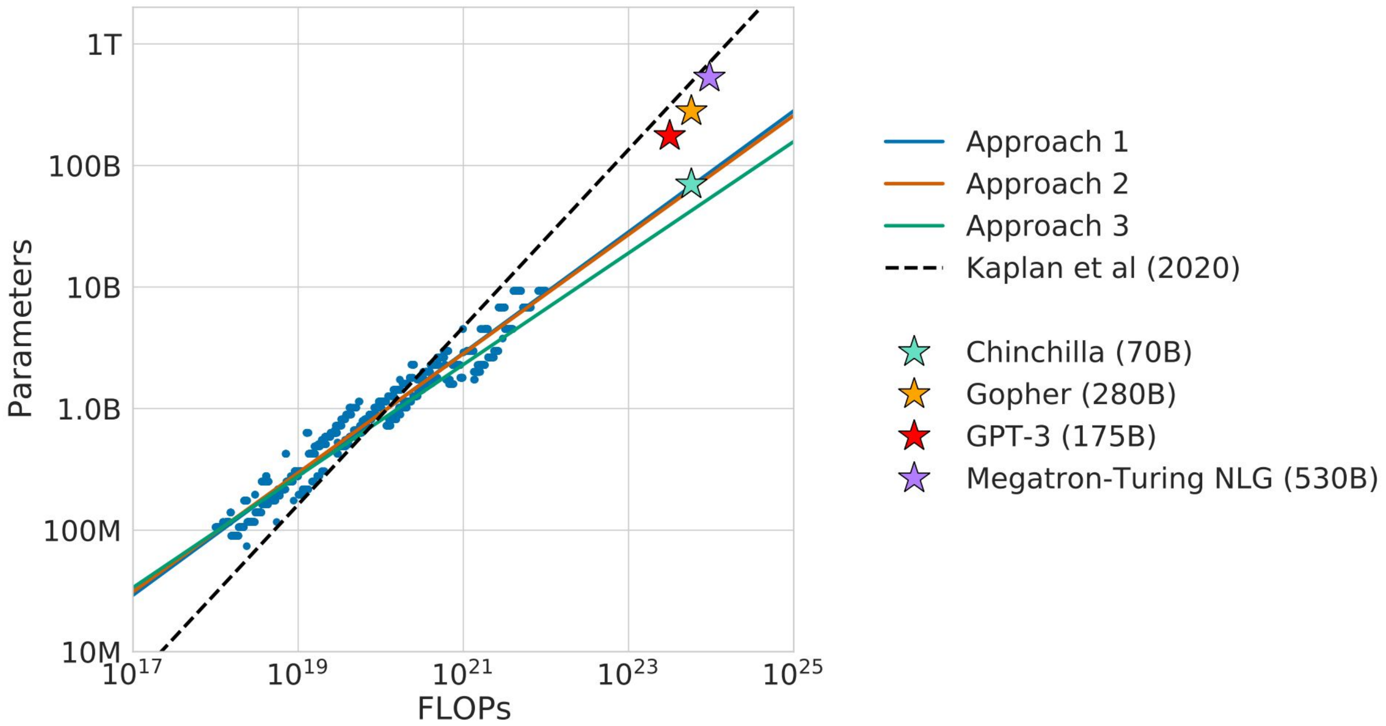


Figure 15: Scaling law predictions in different settings [7]

Many factors can impact the scaling function:

- **Data Properties/Distributions**
- **Transformer Architectures**
- **Pretraining Tasks**
- **Preprocessing Details**

There is no universal scaling function

Scaling Up Pipeline

The current development pipeline of scaling up LLM pretraining, e.g., used by GPT-4, PaLM-2, and many more

Commodity Scale
(Widely Available)

Enormous Scale
(Privileged Access)

Modeling Research (Develop the Best Pretraining Method) → **Scaling Trials** (Verify Robustness with Bigger Scales) → **Scaling Law** (Estimate Scaling Function with Runs) → **Scaling Up** (Million/Billion Dollar Bets)

Scaling Up Pipeline

The current development pipeline of scaling up LLM pretraining, e.g., used by GPT-4, PaLM-2, and many more

Commodity Scale
(Widely Available)

Modeling Research (Develop the Best Pretraining Method) → **Scaling Trials** (Verify Robustness with Bigger Scales) →

Enormous Scale
(Privileged Access)

Scaling Law (Estimate Scaling Function with Runs) → **Scaling Up** (Million/Billion Dollar Bets)

Research
- Fast and many iterations
- Build new knowledge & understanding
- Can/Should take risks
- No guaranteed outcome
- Talent-heavy

Production?
- One scaling up run
- Build a better product: LLM
- Better not screw up
- Clear and short-term deliverables
- Resource-heavy

Outline

- Why Scaling Up
- Which Language Model to Scale Up
- What Factors Matter in Scaling
- What Configurations to Scale Up
- **Capabilities Emerged from Scaling Up**

Emergent Abilities: Observations

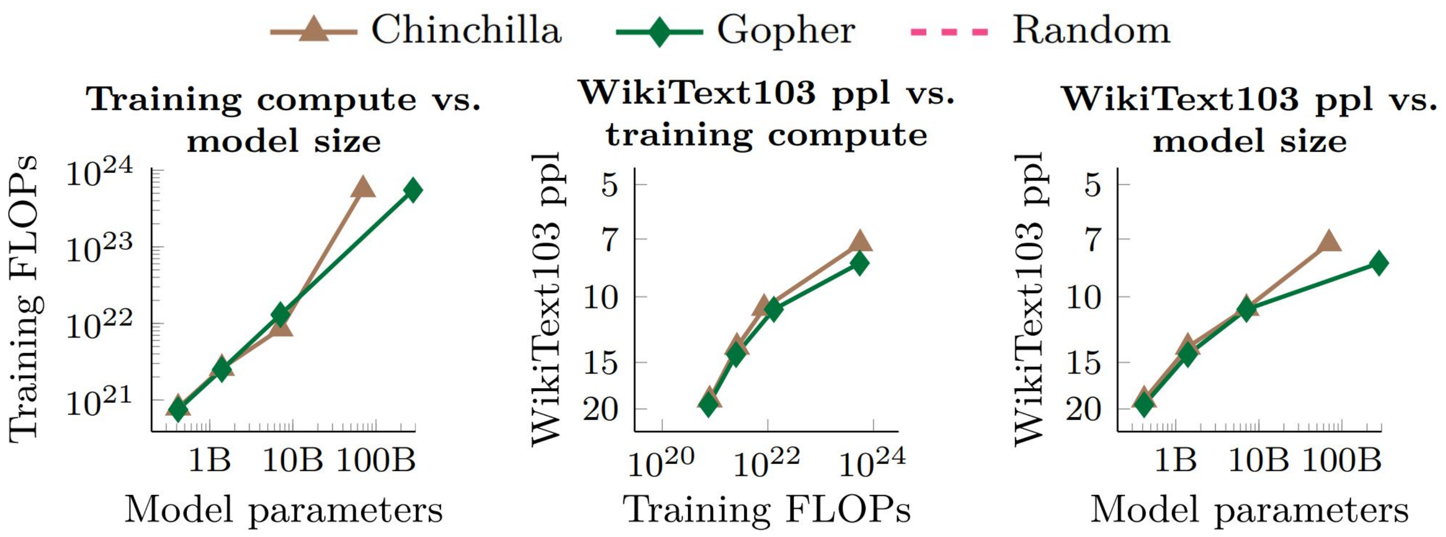


Figure 16: Scaling Law of FLOPs, Model Sizes, and Language Model Accuracy [8]

Emergent Abilities: Observations

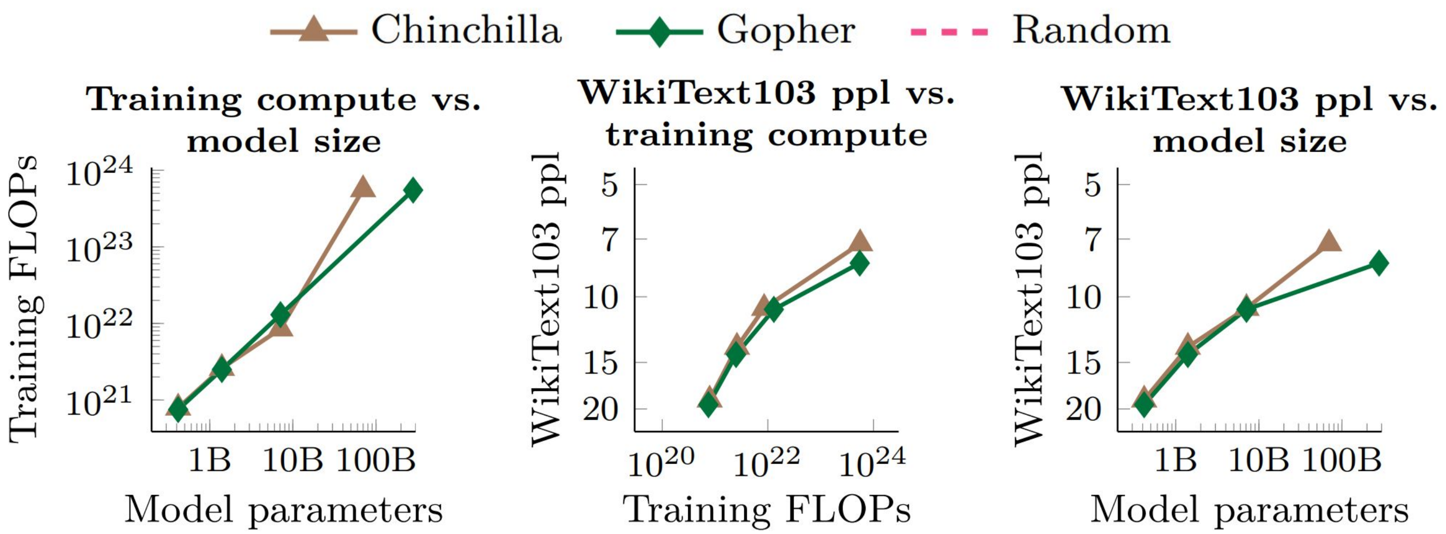


Figure 16: Scaling Law of FLOPs, Model Sizes, and Language Model Accuracy [8]

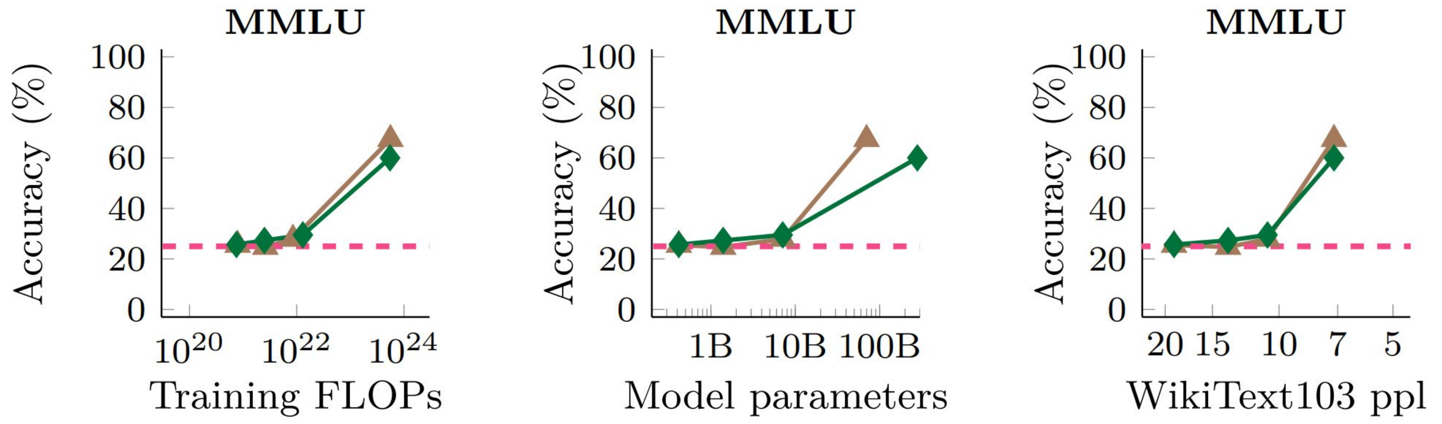


Figure 17: Zero-shot ability on MMLU suddenly emerges at a certain scale [8]

[8] Wei et al. "Emergent Abilities of Large Language Models". TMLR 2022.

Emergent Abilities: Observations

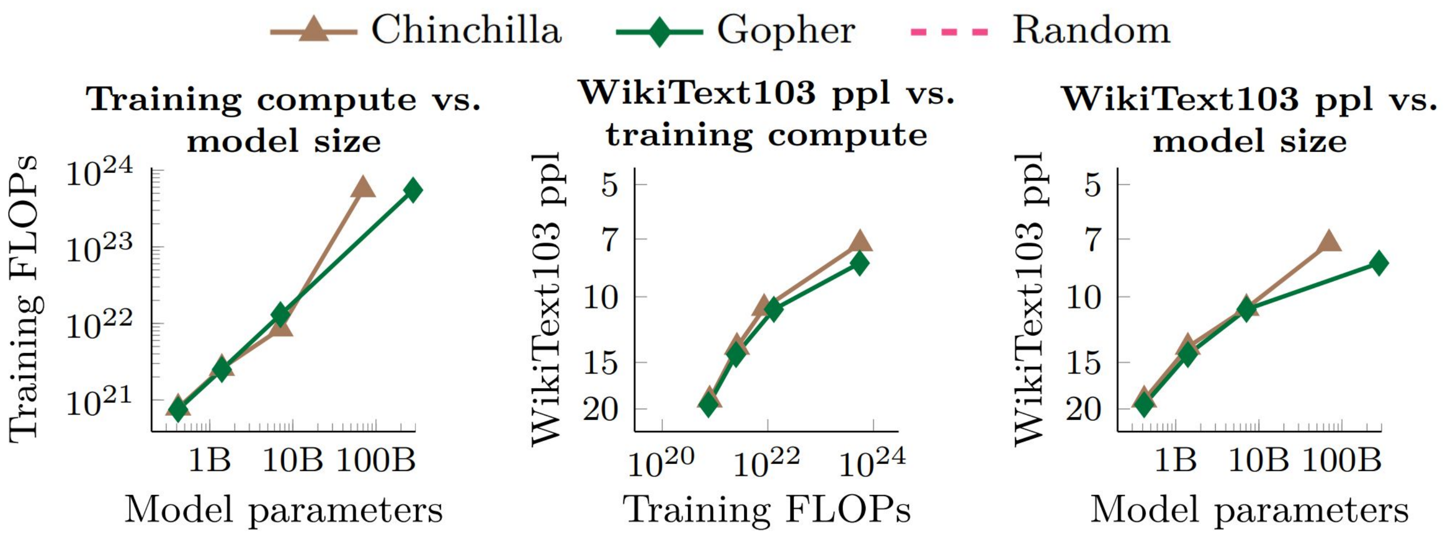


Figure 16: Scaling Law of FLOPs, Model Sizes, and Language Model Accuracy [8]

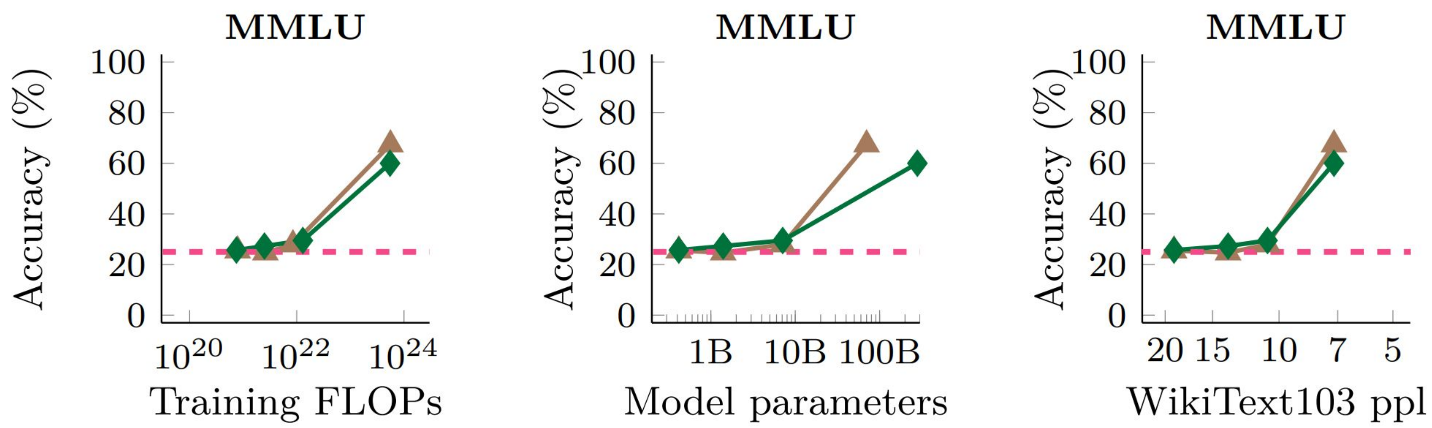


Figure 17: Zero-shot ability on MMLU suddenly emerges at a certain scale [8]

Emergent Ability: an ability not acquired at small scales (i.e., random performance) but suddenly processed at larger scales [8].

- Sharpness: from random to reasonable performance right at a certain scale
- Unpredictability: unclear mapping between model abilities at small and large scales

[8] Wei et al. "Emergent Abilities of Large Language Models". TMLR 2022.

Emergent Abilities: Observations

	Emergent scale		
	Train. FLOPs	Params.	Model
Few-shot prompting abilities			
• Addition/subtraction (3 digit)	2.3E+22	13B	GPT-3
• Addition/subtraction (4-5 digit)	3.1E+23	175B	
• MMLU Benchmark (57 topic avg.)	3.1E+23	175B	GPT-3
• Toxicity classification (CivilComments)	1.3E+22	7.1B	Gopher
• Truthfulness (Truthful QA)	5.0E+23	280B	
• MMLU Benchmark (26 topics)	5.0E+23	280B	
• Grounded conceptual mappings	3.1E+23	175B	GPT-3
• MMLU Benchmark (30 topics)	5.0E+23	70B	Chinchilla
• Word in Context (WiC) benchmark	2.5E+24	540B	PaLM
• Many BIG-Bench tasks (see Appendix E)	Many	Many	Many
Augmented prompting abilities			
• Instruction following (finetuning)	1.3E+23	68B	FLAN
• Scratchpad: 8-digit addition (finetuning)	8.9E+19	40M	LaMDA
• Using open-book knowledge for fact checking	1.3E+22	7.1B	Gopher
• Chain of thought: Math word problems	1.3E+23	68B	LaMDA
• Chain of thought: StrategyQA	2.9E+23	62B	PaLM
• Differentiable search index	3.3E+22	11B	T5
• Self-consistency decoding	1.3E+23	68B	LaMDA
• Leveraging explanations in prompting	5.0E+23	280B	Gopher
• Least-to-most prompting	3.1E+23	175B	GPT-3
• Zero-shot chain of thought reasoning	3.1E+23	175B	GPT-3
• Calibration via P(True)	2.6E+23	52B	Anthropic

Table 3: Abilities and the scale when models acquired them [8].

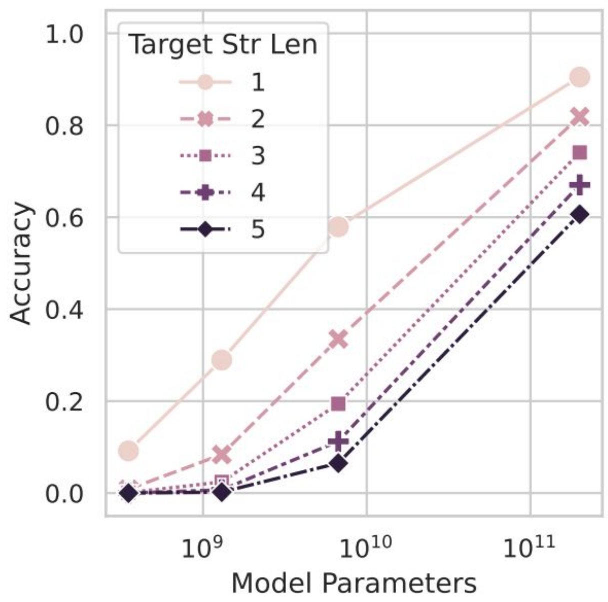
Abilities emerge at different scales

- Hard to map their complexity with emergent scale
- Should be determined by various factors
  - Not clear which factors and their influences

Emergent Abilities: Counter Arguments

"Emergentness" an artifact of exponential metric?

- E.g.: Answer Exact Match: all tokens must be correct to be 1



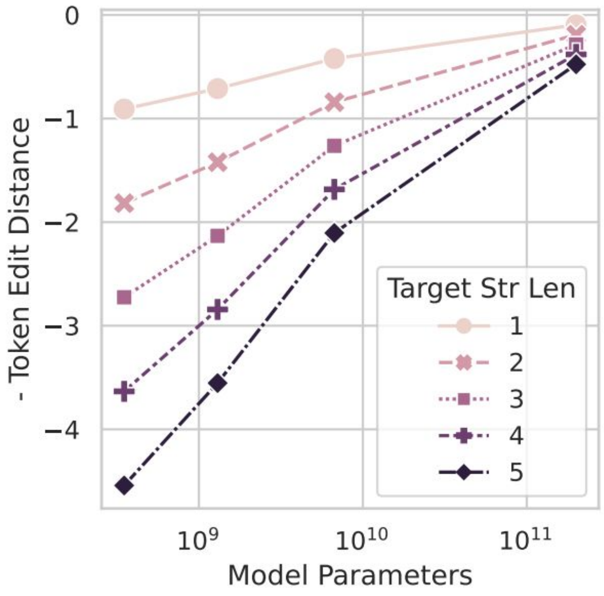


Figure 18: Performance of GPT-3 when evaluated with Exponential (Left) and Continuous (Right) Metrics [9]

Emergent Abilities: Counter Arguments

"Emergentness" an artifact of exponential metric?

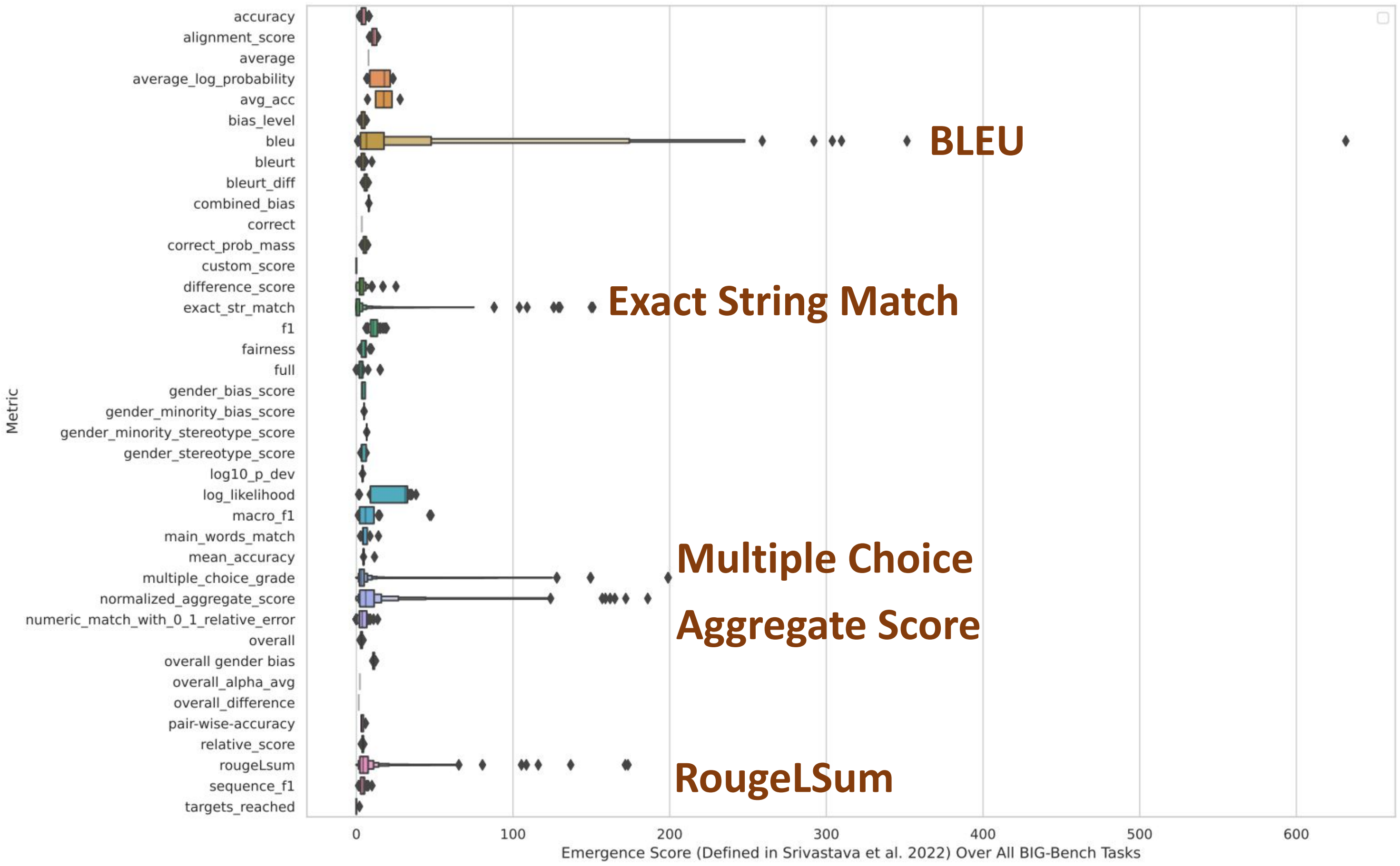


Figure 19: Emergence score for tasks using different metrics in BIG-Bench [9]

Emergent Abilities: Counter Arguments

"Emergentness" an artifact of exponential metric?

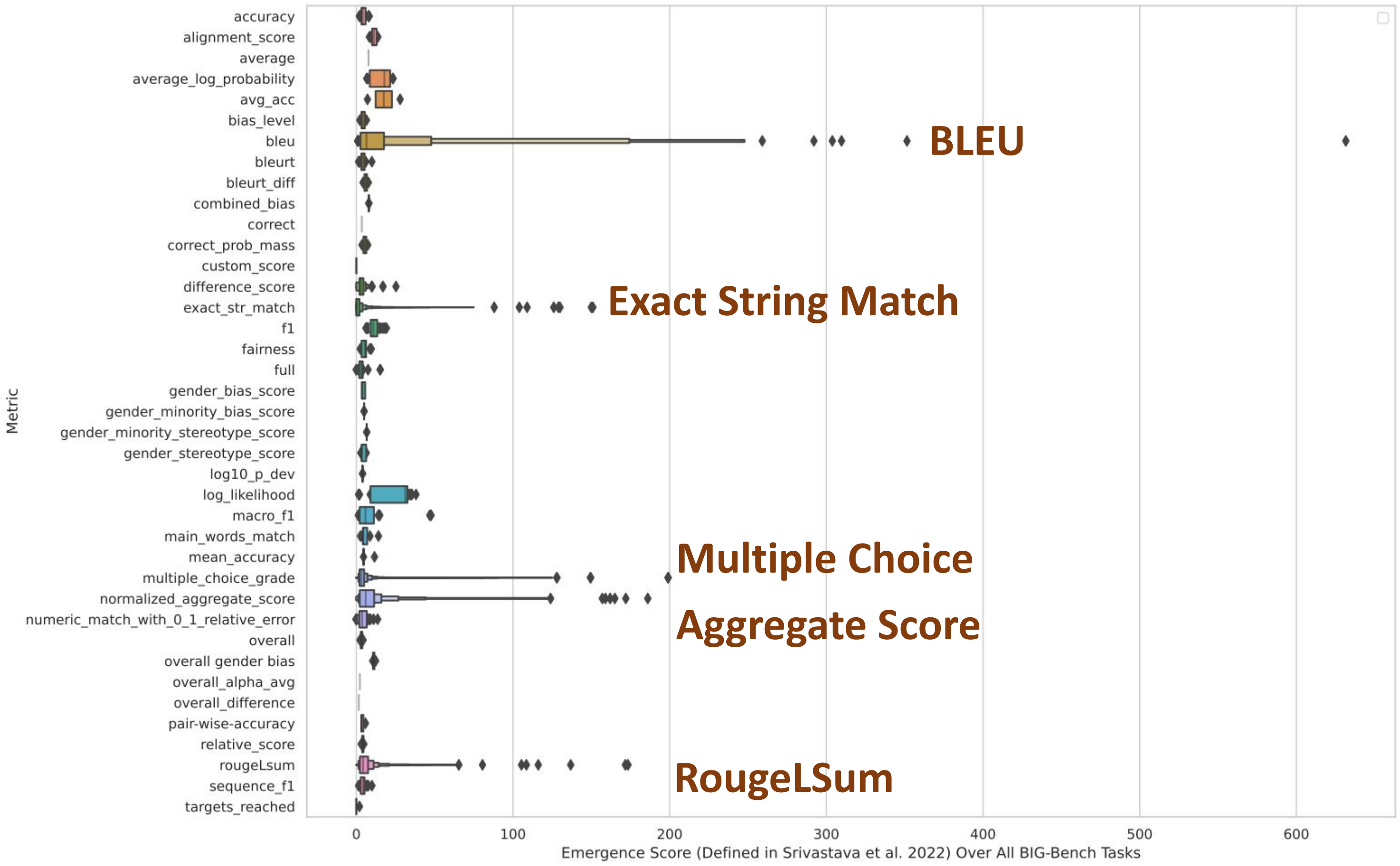


Figure 19: Emergence score for tasks using different metrics in BIG-Bench [9]

Observations:
- Emergentness observed on 5/39 metrics
- These metrics are exponential factors of a more fine-grained prediction

However, user experiences are non-linear
- Pick the right choice to score
- Generate a code that can execute
- Generate a coherent paragraph

Incremental increasement of real-world systems leads to emergent usage too

Emergent Abilities: Remarks

Many of these abilities are what make LLMs great and full of potential

- Zero-shot task solving, Instruction Following, Tool utilization

Open World

Modeling Research (Develop the best Pretraining Method) → **Scaling Trials** (Verify Robustness with bigger scales)

A Few Companies

Scaling Law (Estimate Scaling Function with Runs) → **Scaling Up** (Million/Billion Dollar Bets)

Emergent Abilities: Remarks

Many of these abilities are what make LLMs great and full of potential

- Zero-shot task solving, Instruction Following, Tool utilization

Open World

Modeling Research (Develop the best Pretraining Method) → **Scaling Trials** (Verify Robustness with bigger scales)

A Few Companies

Scaling Law (Estimate Scaling Function with Runs) → **Scaling Up** (Million/Billion Dollar Bets)

Yet they are often acquired at scales not accessible to majority of the community

- Monopoly of technology/knowledge: Only a few places can do it
- Huge burden for scientific approaches: Infeasible to conduct scientific experiments at large scale

Scaling Law: Summary

- Why Scaling Up
  - Predictable benefits in nearly all scenarios
- Which Language Model to Scale Up
  - Benefits of decoder models
- What Factors Matter in Scaling
  - Strong mapping from compute, model size, and pretraining data size to language model performances
- What Configurations to Scale Up
  - Establish scaling law with small scale explorations, scaling up based on scaling law predictions
- Capabilities Emerged from Scaling Up
  - Lots of unknowns and challenges!

Quiz: Why linear improvements of LLM accuracy requires exponentially more compute, model parameters, and pretraining data?