Large Language Models

A brief history of LLMs

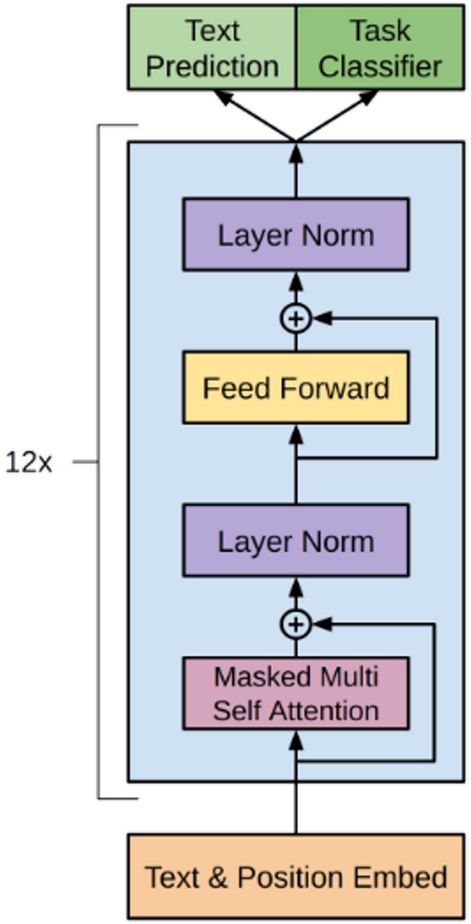
Flavio Giobergia

The GPT Family

- Evolution of GPT models
  - GPT-1 (2018)
    - First in a (long series) of models
    - Decoder-only transformer architecture
    - Pretraining on a large corpus, fine-tuning on various tasks
  - GPT-2 (2019)
    - Larger training set
    - Initial controversies around the (lack of) release for potential misuse
  - GPT-3 (2020)
    - Scaling up: 175B parameters
    - & more data!
  - GPT-4 (2023)
    - OpenAI no longer providing information on the model :(

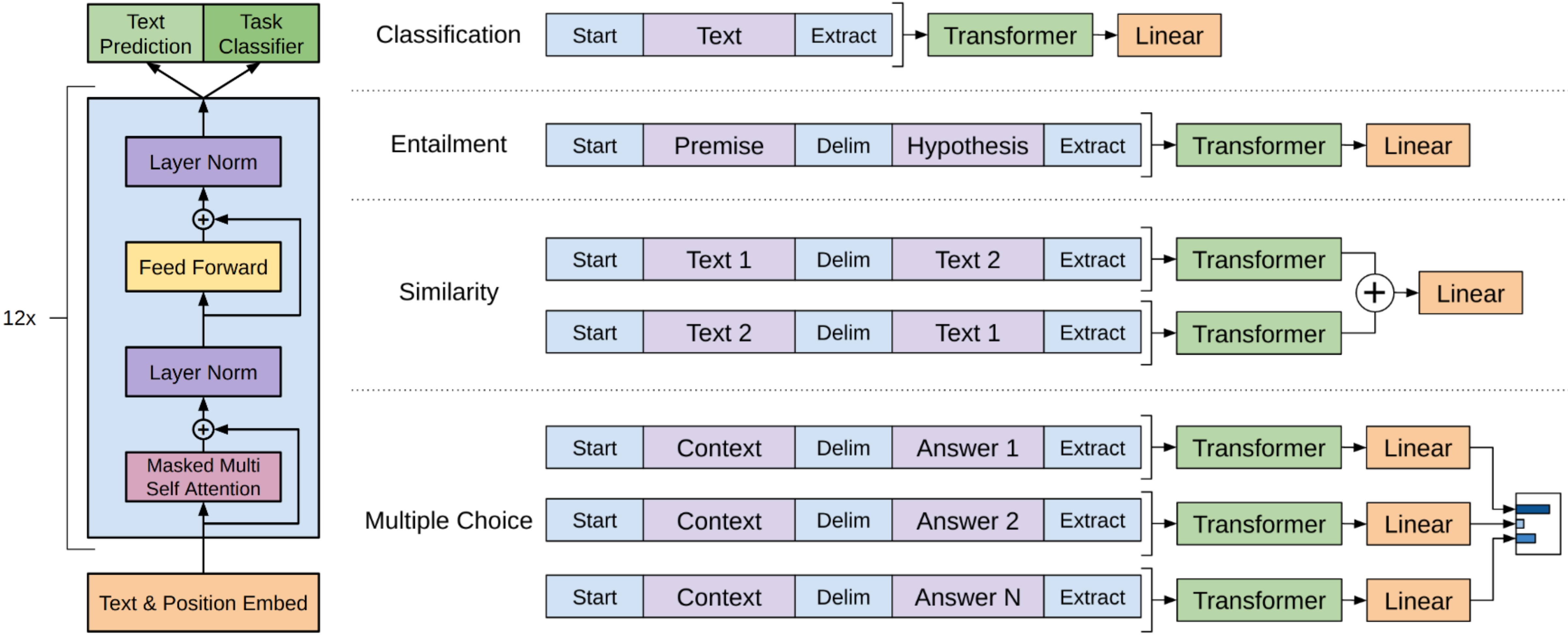
GPT-1 (Generative Pre-trainined Transformer)

- Decoder-only transformer, pretrained on the text generation task
- Training done with:
  - *Unsupervised pretraining*
    - i.e., next token prediction
    - can be done on large datasets (data collection is cheap!)
  - And *supervised fine-tuning*
    - e.g., on supervised datasets (sentiment analysis, …)
    - Typically done on smaller datasets (more expensive!)



Radford, Alec et al. "Improving language understanding by generative pre-training." (2018). https://hayate-lab.com/wp-content/uploads/2023/05/43372bfa750340059ad87ac8e538c53b.pdf

GPT-1 Architecture & tasks



GPT-1 setup

- Unsupervised pretraining
  - BooksCorpus
  - 7,000 books of various genres (e.g., adventure, fantasy, romance)
  - Approx 5 GB of text
- Fine-tuning tasks
  - *Natural language inference*
  - *Question answering*
  - *Semantic similarity* (paraphrase detection)
  - *Classification* (sentiment analysis, grammatical correctness, …)

- Architecture
  - 12 layers decoder-only, 12 heads each
  - 768 dimensional states
  - BPE with 40,000 merges
  - Learned positional embeddings
  - Context size: 512 tokens
  - <u>117M</u> parameters

GPT-2

- Released in 2019
- Based on GPT-1 recipe, with
  - Larger training corpus (~10x)
  - Larger model (~10x)
- Unsupervised pretraining, no fine-tuning
- GPT-2 shows emergent capabilities
  - It could solve problems it was not explicitly trained on!
- "Increasing [model capacity] improves performance in a log-linear fashion across tasks"
- *"Due to concerns about large language models being used to generate deceptive, biased, or abusive language at scale, we are only releasing a much smaller version of GPT-2 along with sampling code"*
  - Concerns starting to emerge… plus marketing

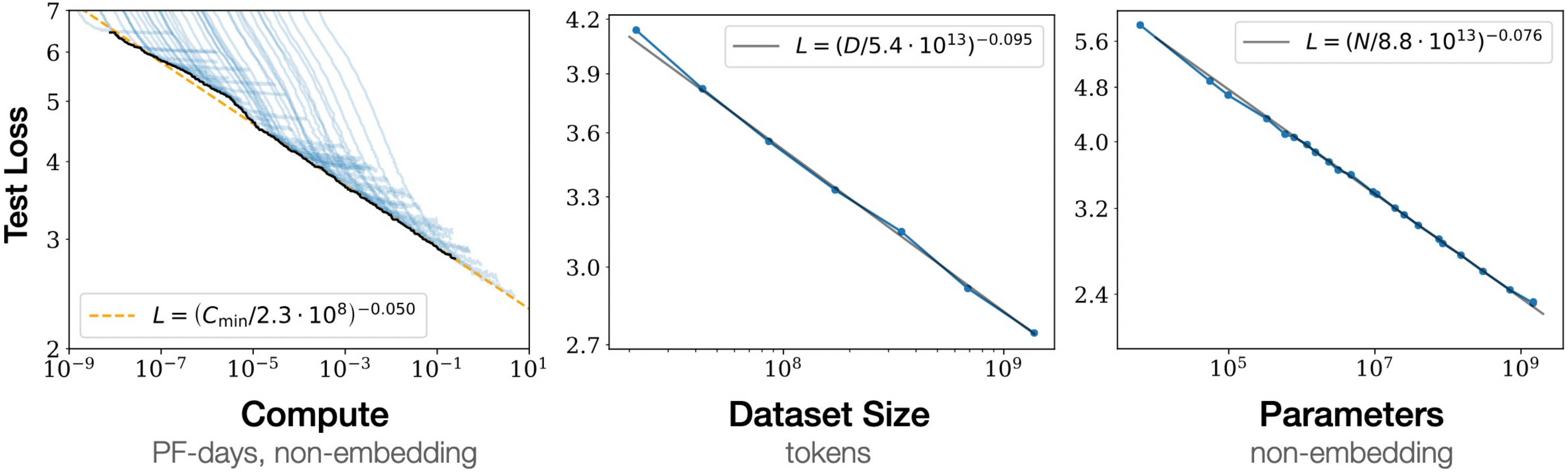
GPT-2 setup

- Unsupervised pretraining
  - WebText
  - Text from 45M links (40GB of text) [GPT-1: 7,000 books, 5GB of text]
    - semi-curated results – links from Reddit posts with at least 3 upvotes
- Fine-tuning
  - *No longer used!*
- Architecture
  - Same as GPT-1
  - BPE with 50,257 tokens [GPT-1 : 40,000 merges]
  - Context size: 1024 [GPT-1: 512]
  - Up to 1.5B parameters [GPT-1: 117M]

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. "Language models are unsupervised multitask learners." *OpenAI blog* 1, no. 8 (2019): 9. https://insightcivic.s3.us-east-1.amazonaws.com/language-models.pdf

Scaling Laws for Neural Language Models

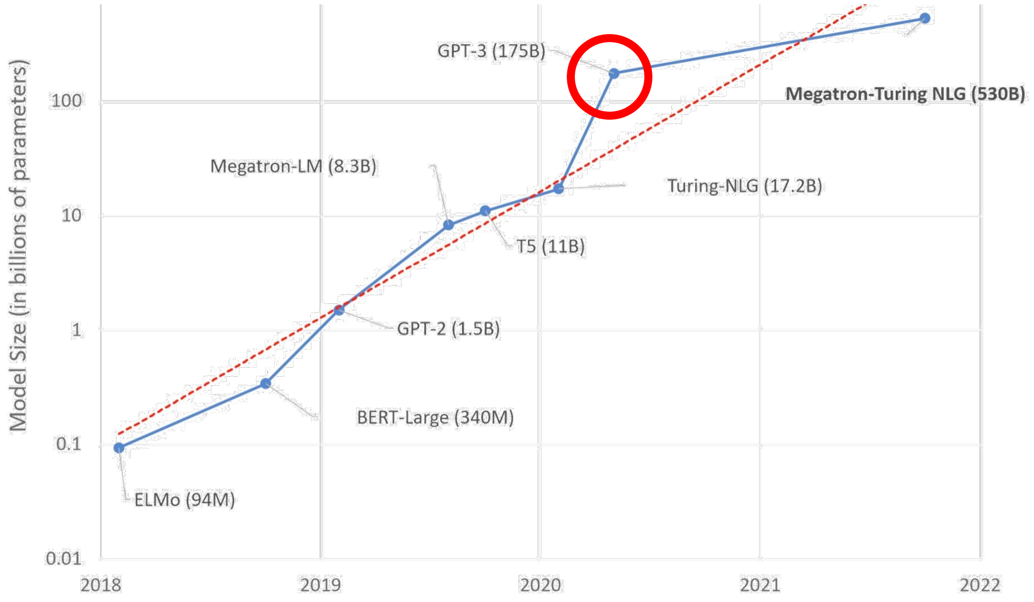
- Published in January 2020, by OpenAI
- https://arxiv.org/pdf/2001.08361
- Shows various *empirical* takeaways
  - The loss scales as a power-law with *model size*, *dataset size*, and *amount of compute*



Other takeaways

- Performance depends very weakly on other architectural hyperparameters such as depth vs. width (number of layers vs embedding size)
  - (for a fixed overall number of parameters)
- The following empirically optimal results emerge:
  - $N \propto C^{0.73}, B \propto C^{0.24}, S \propto C^{0.03}$
  - Where N = model size, B = batch size, S = number of steps, C = computing budget
  - "*As the computational budget C increases, <u>it should be spent primarily on larger models</u>, without dramatic increases in training time or dataset size*"
  - 10x more computing budget ➔ 5.4x model size, 1.7x batch size, 1.07x training steps

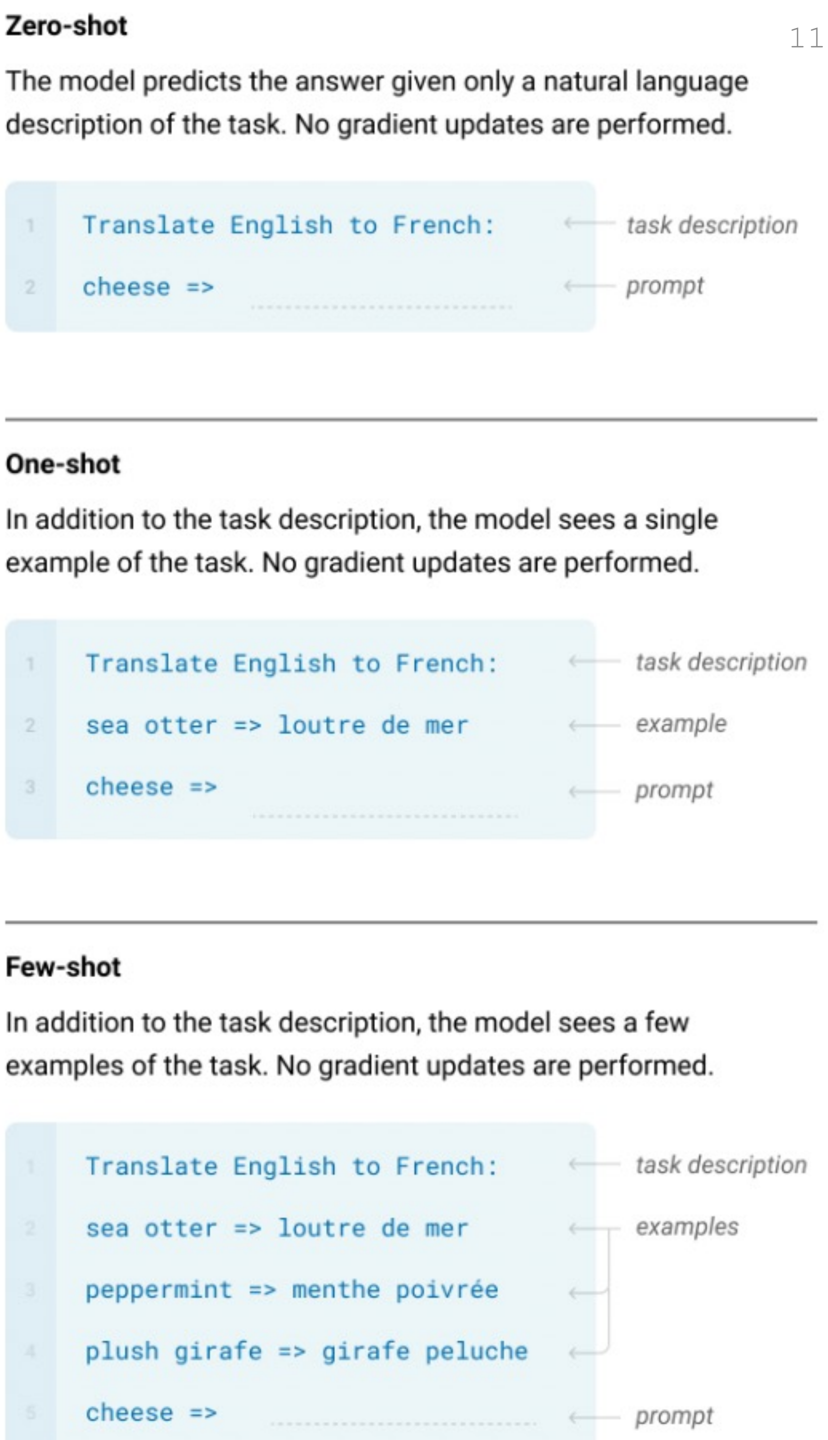
GPT-3



- June 2020, by OpenAI
  - “Language Models are Few-Shot Learners”
    - https://arxiv.org/pdf/2005.14165
- 175B parameters
  - 10x previous models!
  - No other meaningful architectural changes w.r.t. GPT-2
- Multiple datasets
  - (Filtered) CommonCrawl (https://commoncrawl.org/)
    - Web scraped, much of it is low quality (45TB pre-filter, 570GB post-filter)
  - WebText2, Books1, Books2, Wikipedia
    - Sampled with different rates based on their quality (as determined by OpenAI)
  - (notice how defining the precise dataset used gets harder!)

Fine-tuning vs in-context

- *Fine-tuning*
  - Update model weights on a task-specific dataset
  - Previously considered the go-to approach
  - No fine-tuning done on GPT-3 – in-context learning only!
- *In-context learning* 🡐 main focus of GPT-3’s paper
  - Model weights no longer updated
  - Task described as a part of the prompt in natural language
  - *Zero-shot*
    - No other information provided to the model
  - *One-shot*
    - One input/output example of provided in addition to task description
  - *Few-shot*
    - Multiple examples of input/output pairs provided
    - Limited by maximum context size available



Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
Translate English to French:
cheese =>
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
Translate English to French:
sea otter => loutre de mer
cheese =>
```

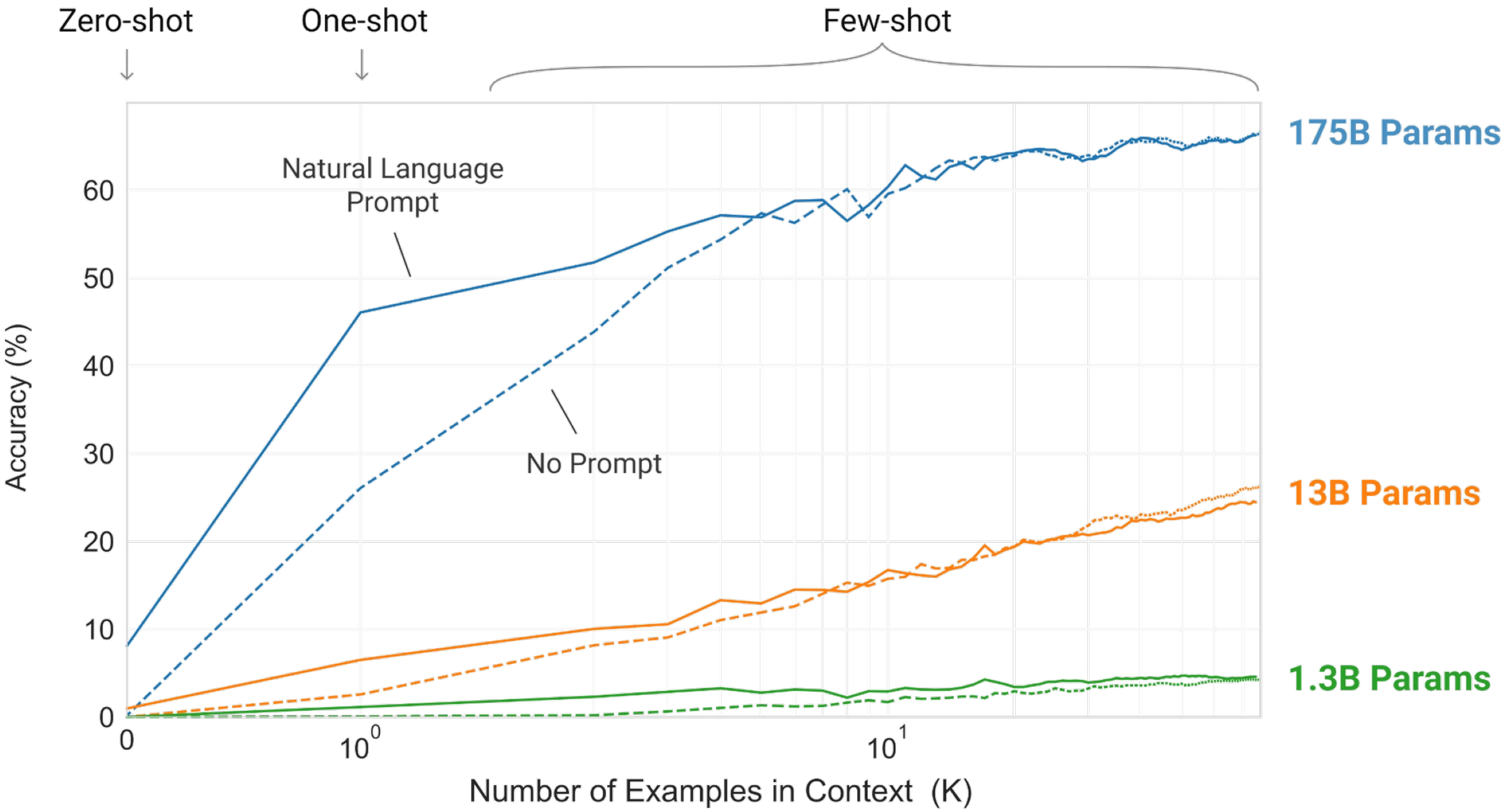
Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

In-context performance

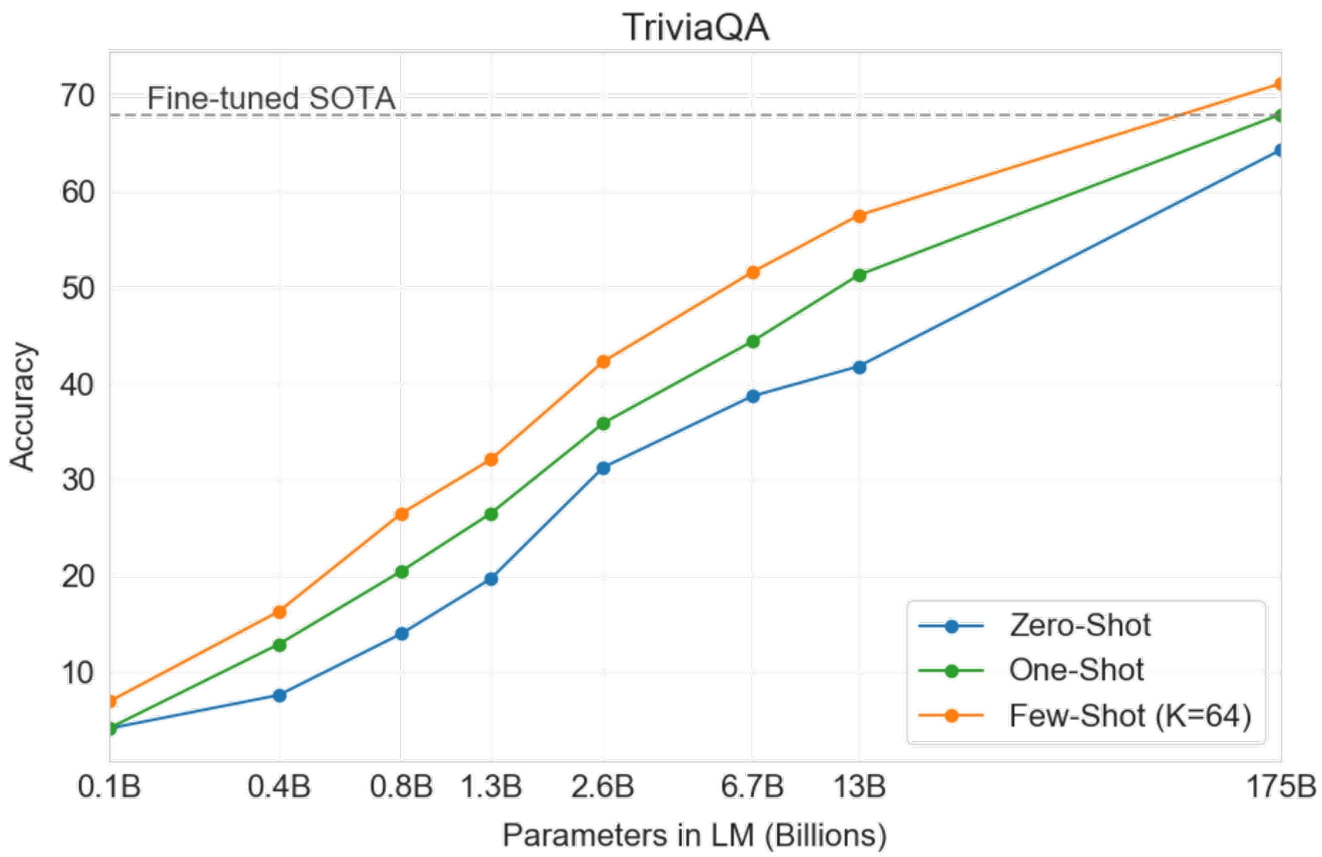
- Larger models show remarkable 1- and few-shot performance w.r.t. smaller ones
- The gap between 0- and 1-shot performance is quite remarkable
  - (It's typically a good idea to pass shots!)



(Task of removing some symbols from an input text)

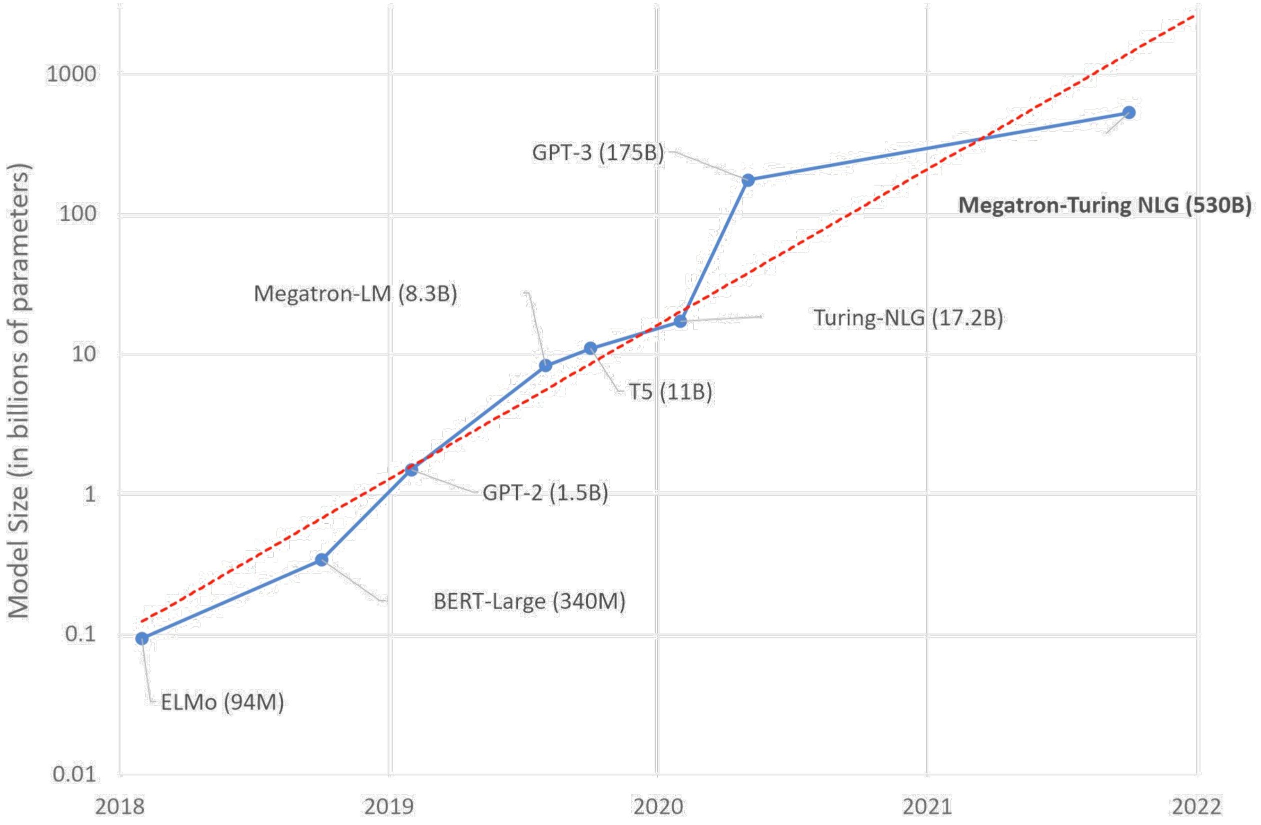
Comparison with fine-tuned models

- The 175B few-shot model is comparable (or outperforms) fine-tuned SOTA models
- In the plot: TriviaQA, but similar results on other problems
- This is a remarkable result:
  - Scaling the model allows not fine-tuning on task-specific datasets and still get competitive results!



The race to bigger models

- These scaling laws resulted in a race toward building larger models
- GPT-3 was the first 100B+ parameters model
- Other larger models have been developed, following this trend



Big models got bigger

- *Jurassic-1*
  - 178B parameters, *AI21labs* (2021)
- *Gopher*
  - 280B parameters, *DeepMind* (2021)
  - (120 pages paper: 1-7 “methodology”, 8-120: results & examples)
    - This has become the current format!
    - https://arxiv.org/pdf/2112.11446
- *Megatron-Turing NLG*
  - 530B parameters, *NVIDIA* + *Microsoft* (2021)
- *PaLM* (*Pathways Language Model*)
  - 540B parameters, *Google* (2022)

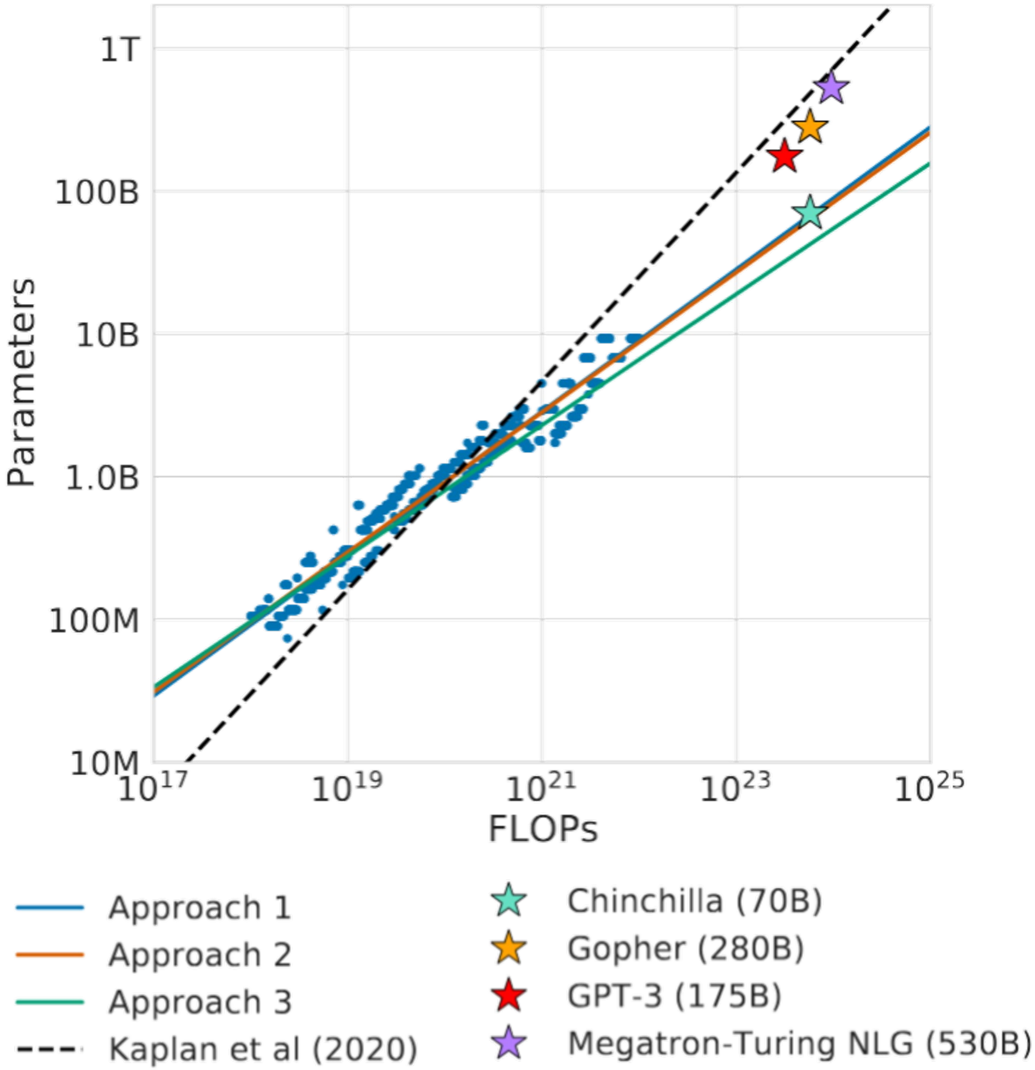
Oversized and undertrained!

- DeepMind publishes “Training Compute-Optimal Large Language Models” in March 2022
- https://arxiv.org/pdf/2203.15556

- Main claims:
  - *“current large language models are significantly under-trained, a consequence of the recent focus on scaling language models whilst keeping […] data constant”*
  - *“for every doubling of model size the number of training tokens should also be doubled”*
  - *Chinchilla*, a new correctly-sized model, outperforms larger ones!

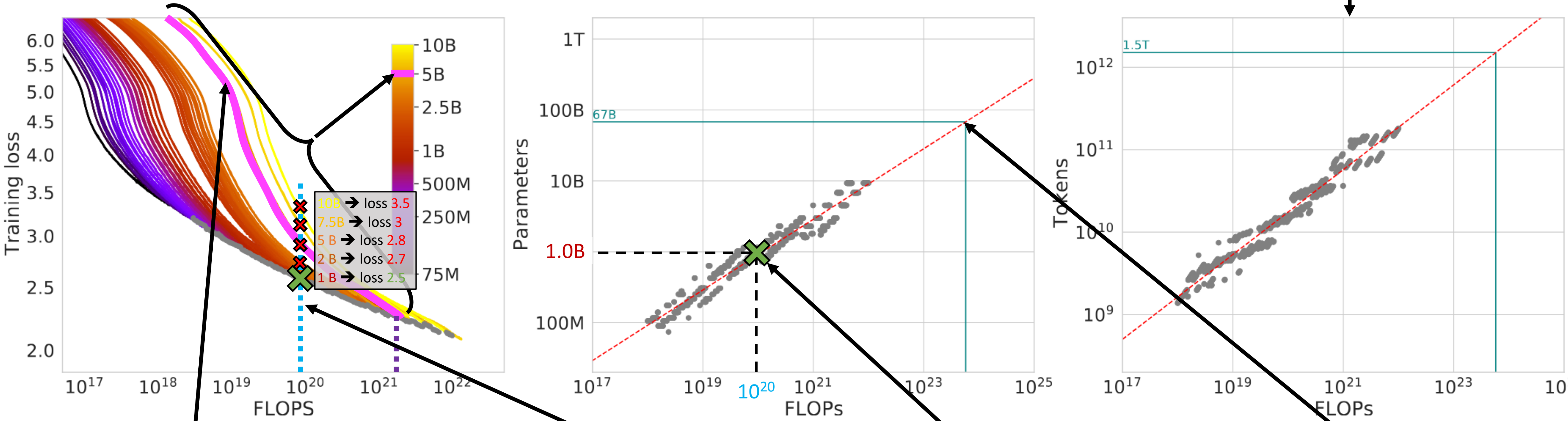
Oversized and undertrained – explained

- DeepMind runs additional experiments with 3 approaches
- All results indicate that the conclusions of Kaplan et al (2020) give too much importance to model size
- For instance, the resources used for Gopher (280B) should have been used to train a 40-70B parameters model



Approach 1 (fixed-sized mode)

Similar considerations have been made for the number of tokens used from training.



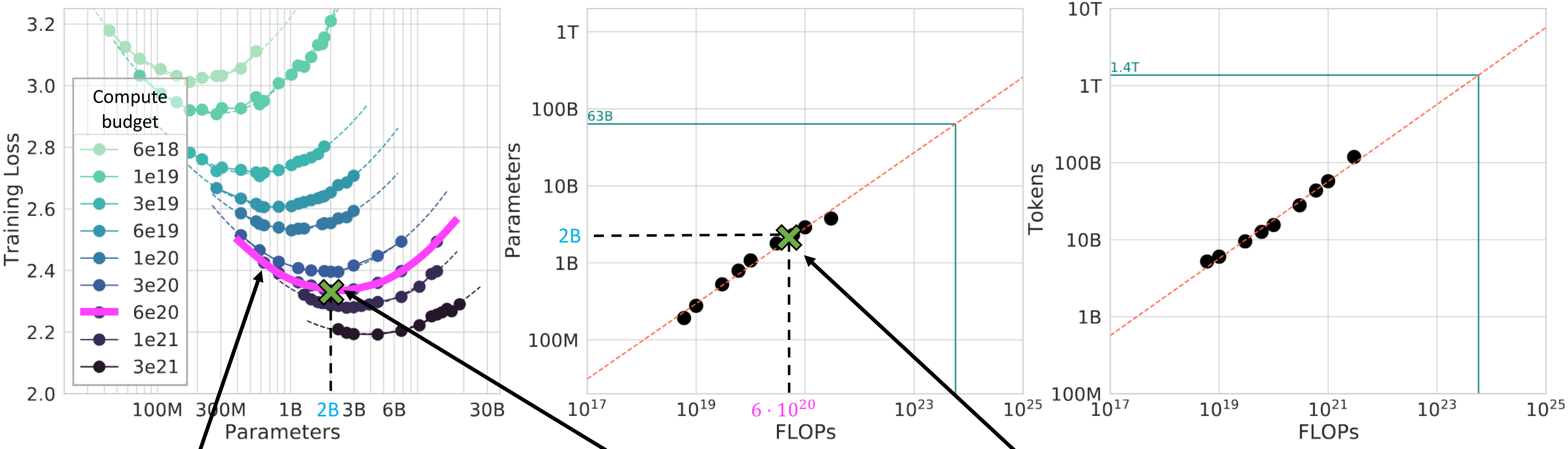
The authors trained, with a specific compute budget (measured in FLOPs), models of various sizes.
For instance, a 5B parameters model was trained using up to 10^{21} FLOPS

For each compute budget, we can now find the model with the lowest loss we can reach. The number of parameters of that model is the “best” we can reach, for the given compute budget. e.g., for 10^{20} FLOPS, the lowest loss is achieved by the 1B parameters model

The optimal model size is reported for each budget size. A linear model is fit (red dashed line ---)

The compute budget used for Gopher ($5.8 \cdot 10^{23}$ FLOPS) shows that the best model size should be 67B parameters (not 280B!)

Approach 2 (IsoFLOPs)

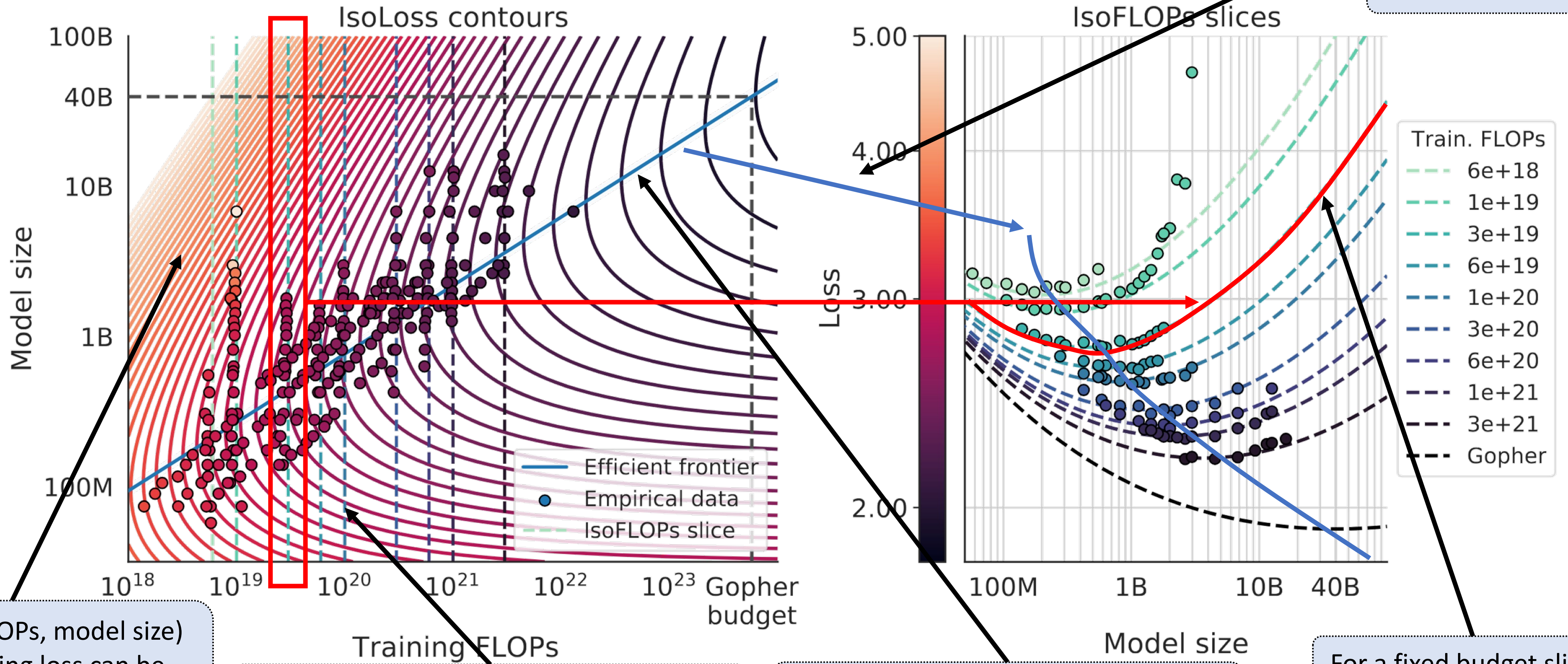


For a fixed compute budget, the authors trained models of various sizes.
Empirically, the behavior of the loss follows a parabola.

The minimum of the parabola corresponds to the model size that, for a fixed compute budget, gets the lowest loss (best model).
For instance, for a budget of $6 \cdot 10^{20}$ FLOPS, the model with the lowest loss (~2.3) has 2B parameters

For each compute budget, the best model is reported. The red dashed (---) model interpolates for other model sizes. Gopher should now be a 63B model!

Approach 3



The efficient frontier passes through the minimums of each curve.

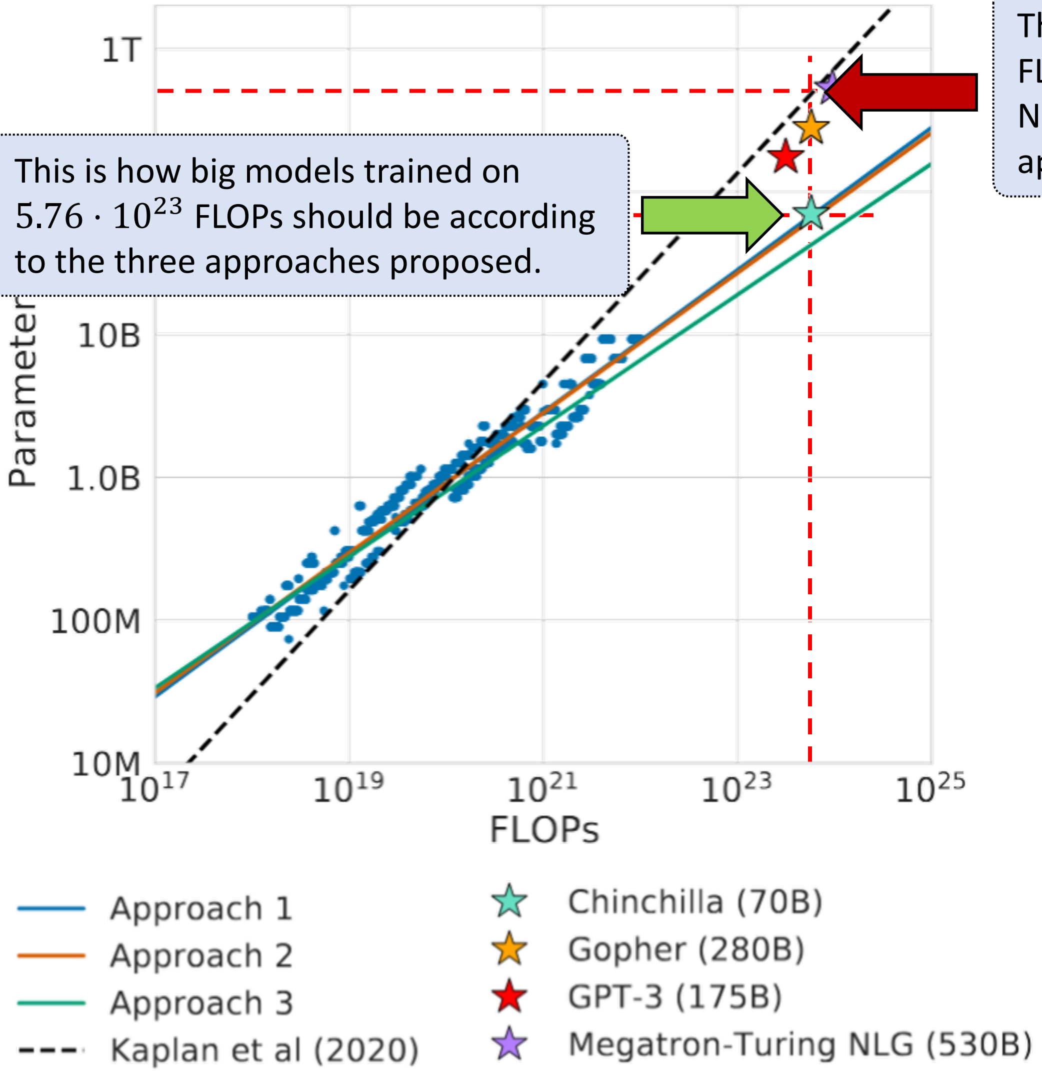
For a given (FLOPs, model size) pair, the resulting loss can be computed. A parametric model is fit based on the available data, as shown here (contours represent levels with the same loss value)

Vertical lines represent all points (model sizes). The color represents the loss obtained for the each point having that fixed compute budget (e.g., all models trained with 10^{20} FLOPs

Along each "isoFLOPs" line, we can identify the "best" model (i.e., the model with the minimum loss). The efficient frontier represents this point as the compute budget varies.

For a fixed budget slice, we can the plot of model size vs loss.

Models are undertrained!



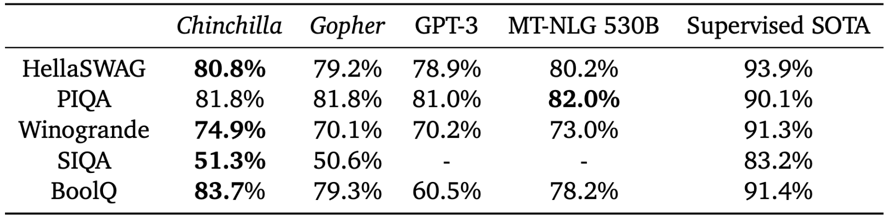
This is how big models trained on $5.76 \cdot 10^{23}$ FLOPs should be according to the three approaches proposed.

This is how big models trained on $5.76 \cdot 10^{23}$ FLOPs should be according to Kaplan et al, 2020. Notice that many previous models approximately follow that relationship.

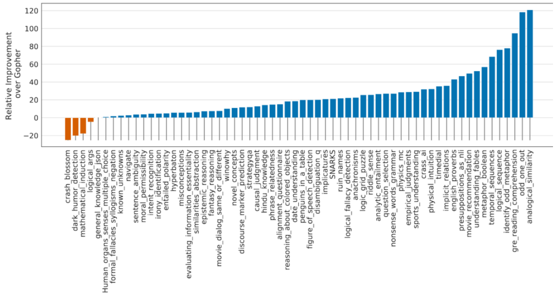
Chinchilla

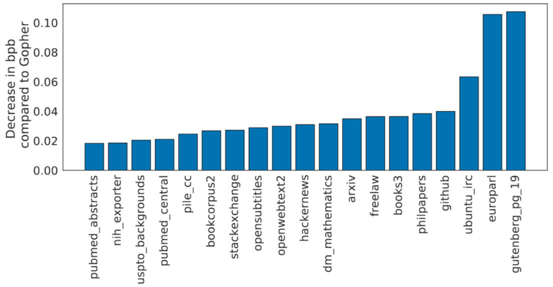
Generated with CogView3-Plus

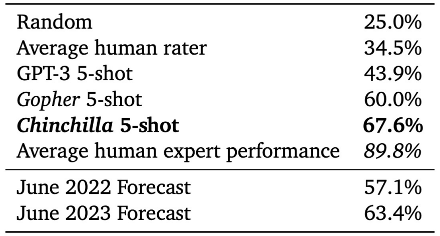
- In the same paper, DeepMind also presented *Chinchilla*
- Chinchilla is a *70B parameters* model
- Trained on the same compute budget as Gopher (280B)
  - $5.76 \cdot 10^{23}$ FLOPs
- Chinchilla is ¼ of Gopher’s size, and is trained on more tokens
  - 1.4T (Chinchilla) vs 300B (Gopher) → 4x more!
- The paper shows how Chinchilla generally *outperforms* Gopher, but also GPT-3, on various tasks



	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%





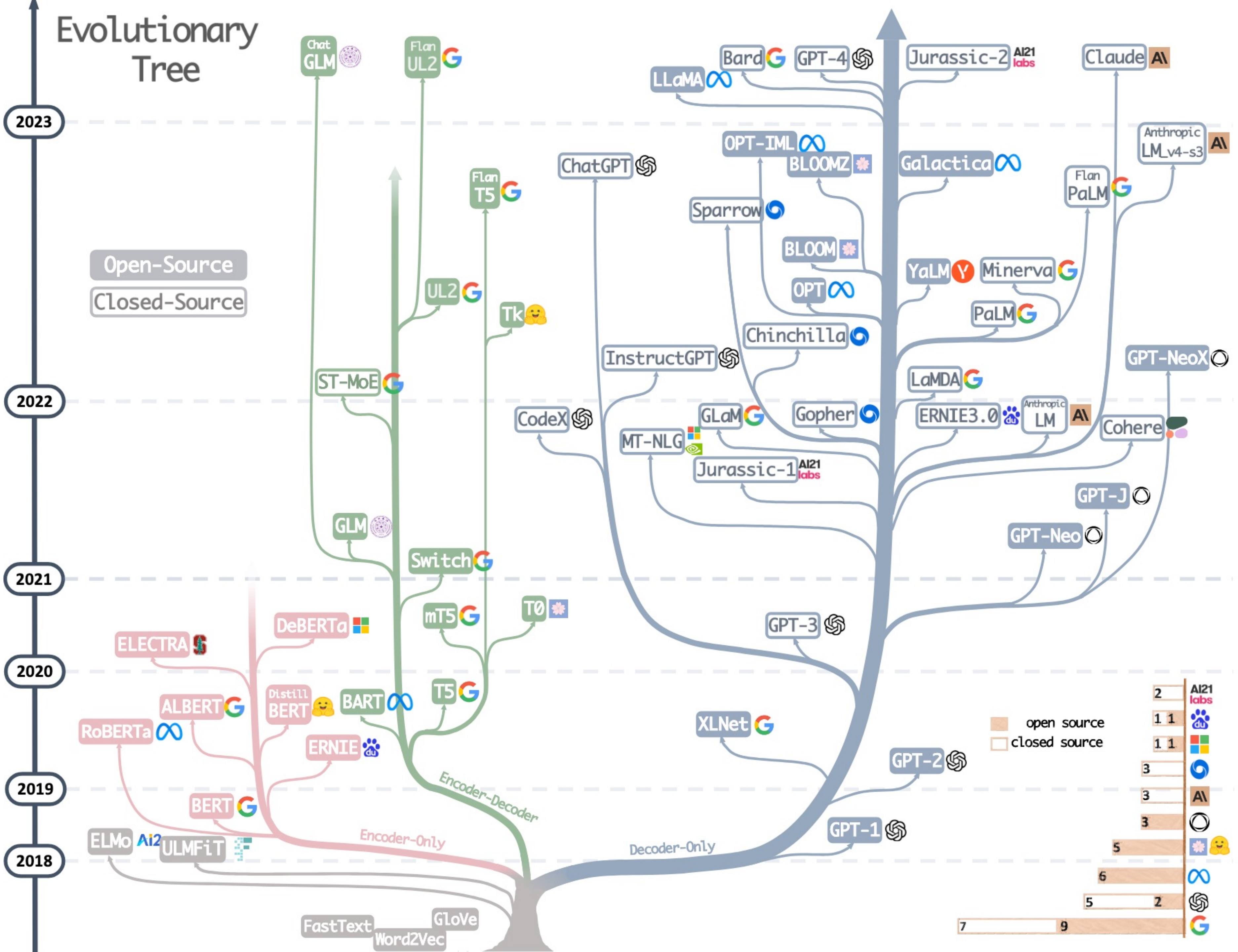


Random	25.0%
Average human rater	34.5%
GPT-3 5-shot	43.9%
Gopher 5-shot	60.0%
***Chinchilla* 5-shot**	**67.6%**
Average human expert performance	*89.8%*
June 2022 Forecast	57.1%
June 2023 Forecast	63.4%

A new trend

- The previous trend of “always larger” models started fading
- There has since been a return to smaller models, trained for longer
- Smaller models can achieve better performance!
  - LLMs become more accessible
  - This led to a large ecosystem of (open) models

Evolutionary Tree



https://arxiv.org/pdf/2304.13712

Llama family

Generated with CogView3-Plus

- Llama (Large Language Model Meta AI) is a family of models introduced by Meta AI, starting in 2023
  - https://huggingface.co/meta-llama
- All *autoregressive*, *decoder-only* architectures, trained on *open datasets*
- All models are all openly available
  - LLaMA (Feb ‘23) → 7B, 13B, 32B, 65B
  - Llama 2 (Jul ‘23) → 7B, 13B, 70B
  - Llama 3 (Apr ‘24) → 8B, 70B
    - 3.1 (Jul ‘24) → 8B, 70B, 405B
    - 3.2 (Sep ‘24) → 1B, 3B, 11B, 90B (with multimodal version)
  - Plus other versions (e.g. Code Llama – based on Llama 2, or instruction-tuned models)

Other families of *open* models

- *GPT-Neo/GPT-J* (EleutherAI, 🇺🇸) – open source alternatives to the GPT family

- *Mistral* (MistralAI, 🇫🇷) – wide variety of model sizes, code-tuned versions (for 80+ languages), multimodal versions (Pixtral)

- *GLM* (Zhipu AI, 🇨🇳) – General Language Model, more oriented toward the Chinese language, but also works well on other languages, including English

- *Falcon* (Technology Innovation Institute, 🇦🇪) – different sized models, they also released a Mamba-based model (State Space Language Models!)