NVIDIA | DEEP LEARNING INSTITUTE

SELF-SUPERVISION, BERT, AND BEYOND

Building Transformer-Based Natural Language Processing Applications (Part 2)

FULL COURSE AGENDA

Part 1: Machine Learning in NLP

Lecture: NLP background and the role of DNNs leading to the Transformer architecture

Lab: Tutorial-style exploration of a ***translation task*** using the Transformer architecture

Part 2: Self-Supervision, BERT, and Beyond

Lecture: Discussion of how language models with self-supervision have moved beyond the basic Transformer to BERT and ever larger models

Lab: Practical hands-on guide to the NVIDIA NeMo API and exercises to build a ***text classification task*** and a ***named entity recognition task*** using BERT-based language models

Part 3: Production Deployment

Lecture: Discussion of production deployment considerations and NVIDIA Triton Inference Server

Lab: Hands-on deployment of an example ***question answering task*** to NVIDIA Triton

Part 2: Self-Supervision, BERT and Beyond

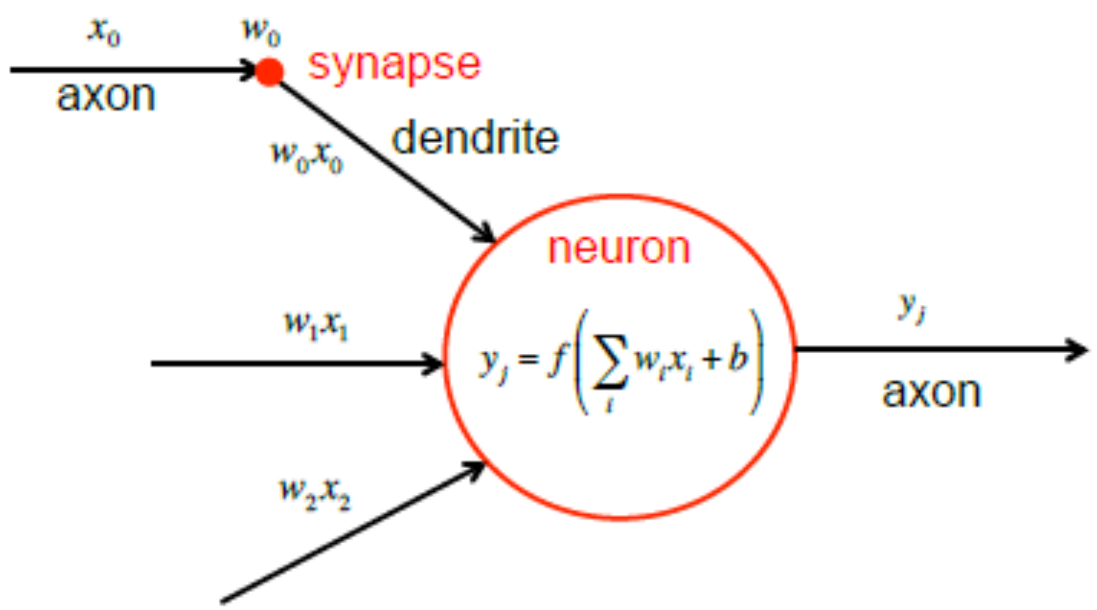
- Lecture
  - Why Do DNNs Work Well?
  - Self-Supervised Learning
  - BERT
- Lab
  - Explore the Data
  - Explore NeMo
  - Text Classifier Project
- Lecture (cont’d)
  - Bigger is Better
  - Can and should we go even bigger?
- Lab (cont’d)
  - Named Entity Recognizer

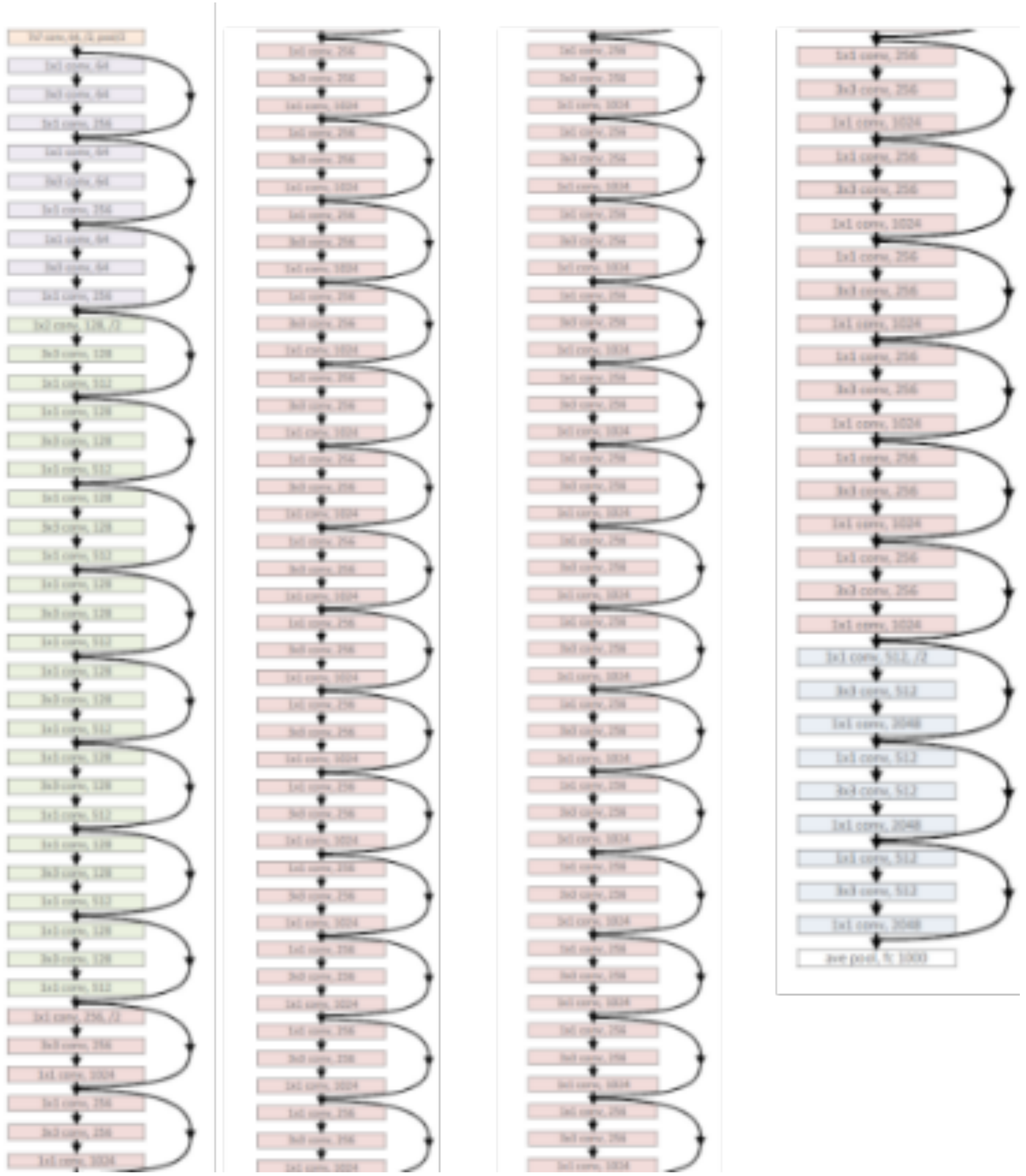
Part 2: Self-Supervision, BERT and Beyond

- Lecture
  - Why Do DNNs Work Well?
  - Self-Supervised Learning
  - BERT
- Lab
  - Explore the Data
  - Explore NeMo
  - Text Classifier Project
- Lecture (cont’d)
  - Bigger is Better
  - Can and should we go even bigger?
- Lab (cont’d)
  - Named Entity Recognizer

NEURAL NETWORKS ARE NOT NEW

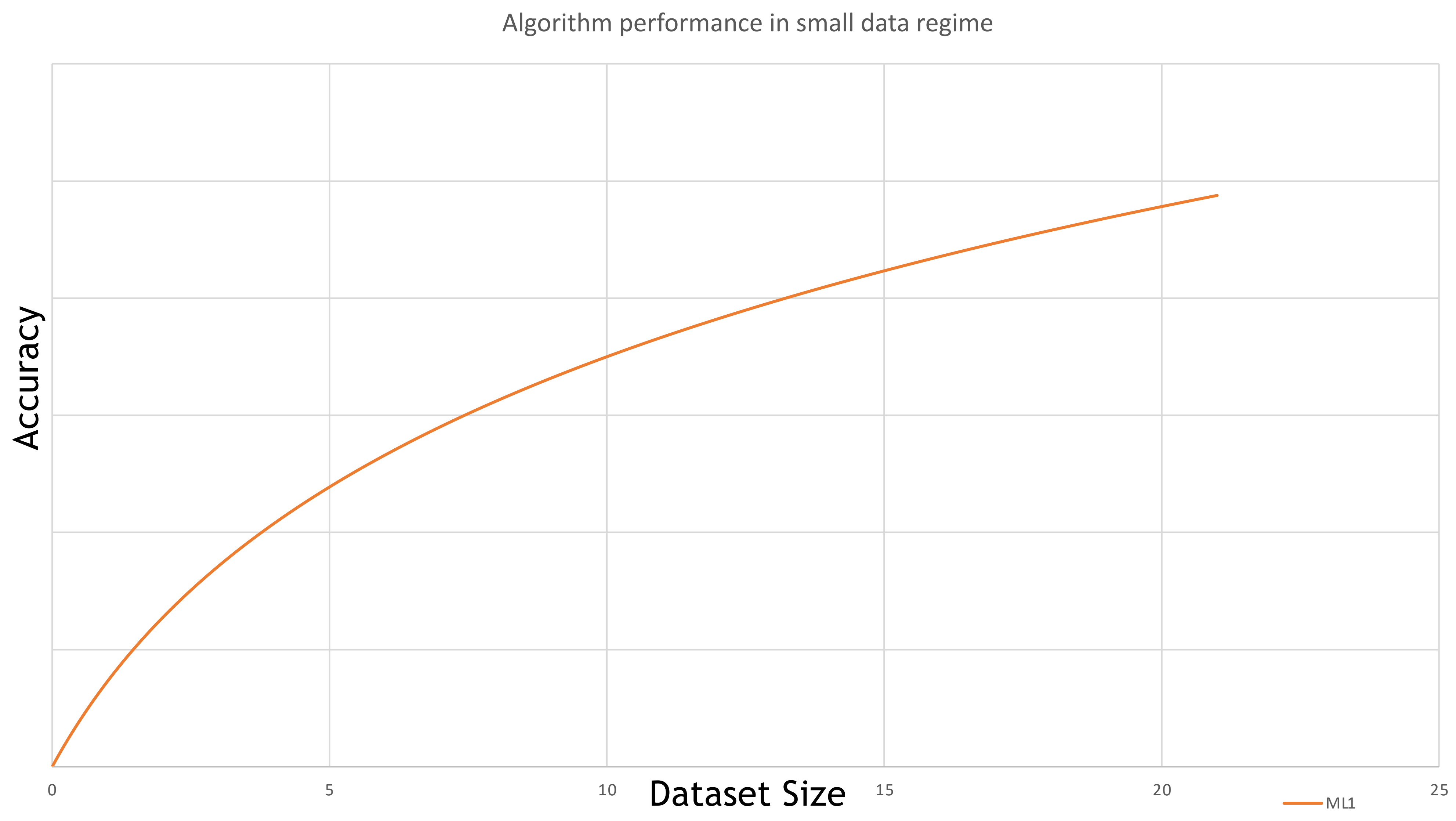
They are surprisingly simple as an algorithm





NEURAL NETWORKS ARE NOT NEW

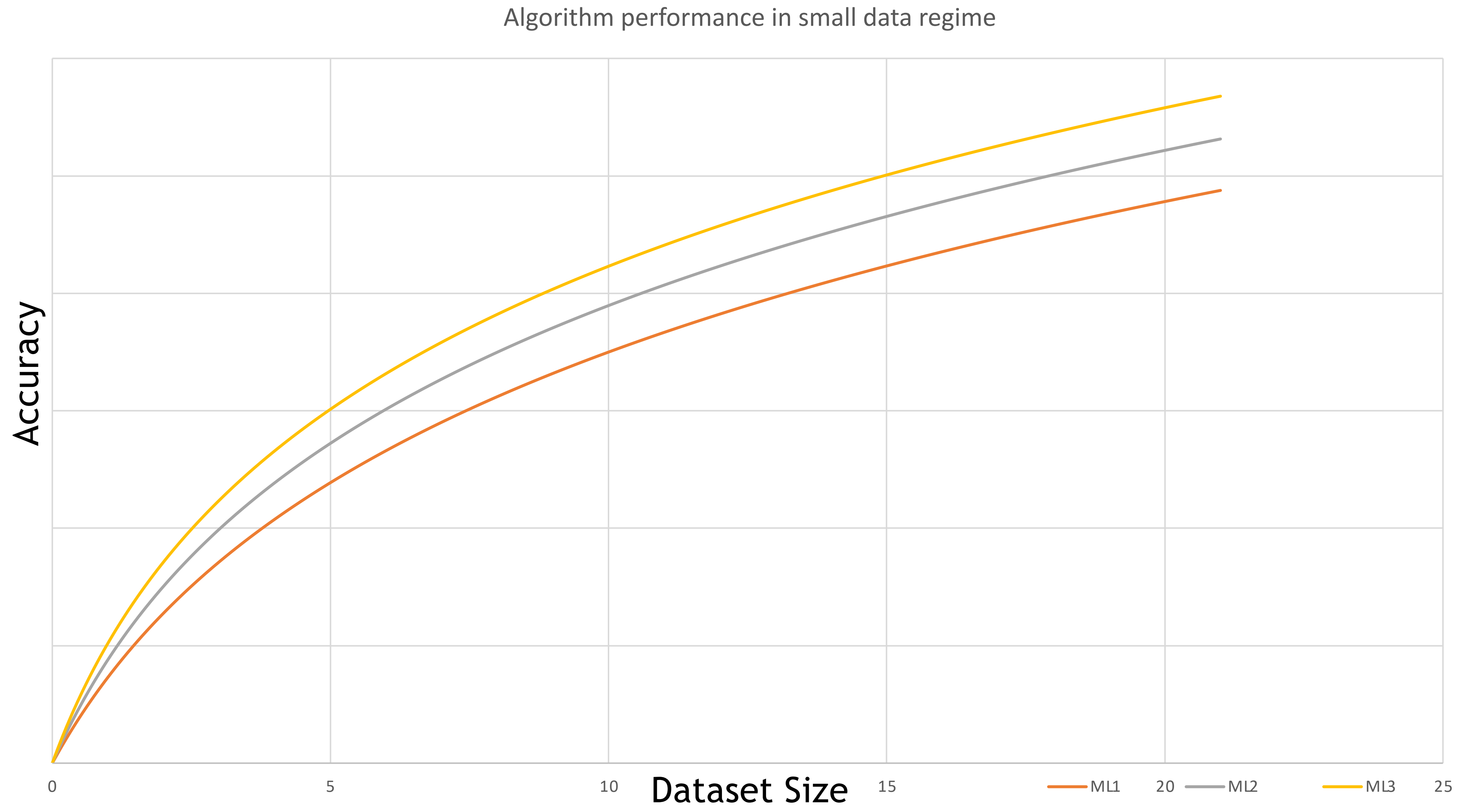
They just historically never worked well



Andrew Ng, “Nuts and Bolts of Applying Deep Learning”, https://www.youtube.com/watch?v=F1ka6a13S9I

NEURAL NETWORKS ARE NOT NEW

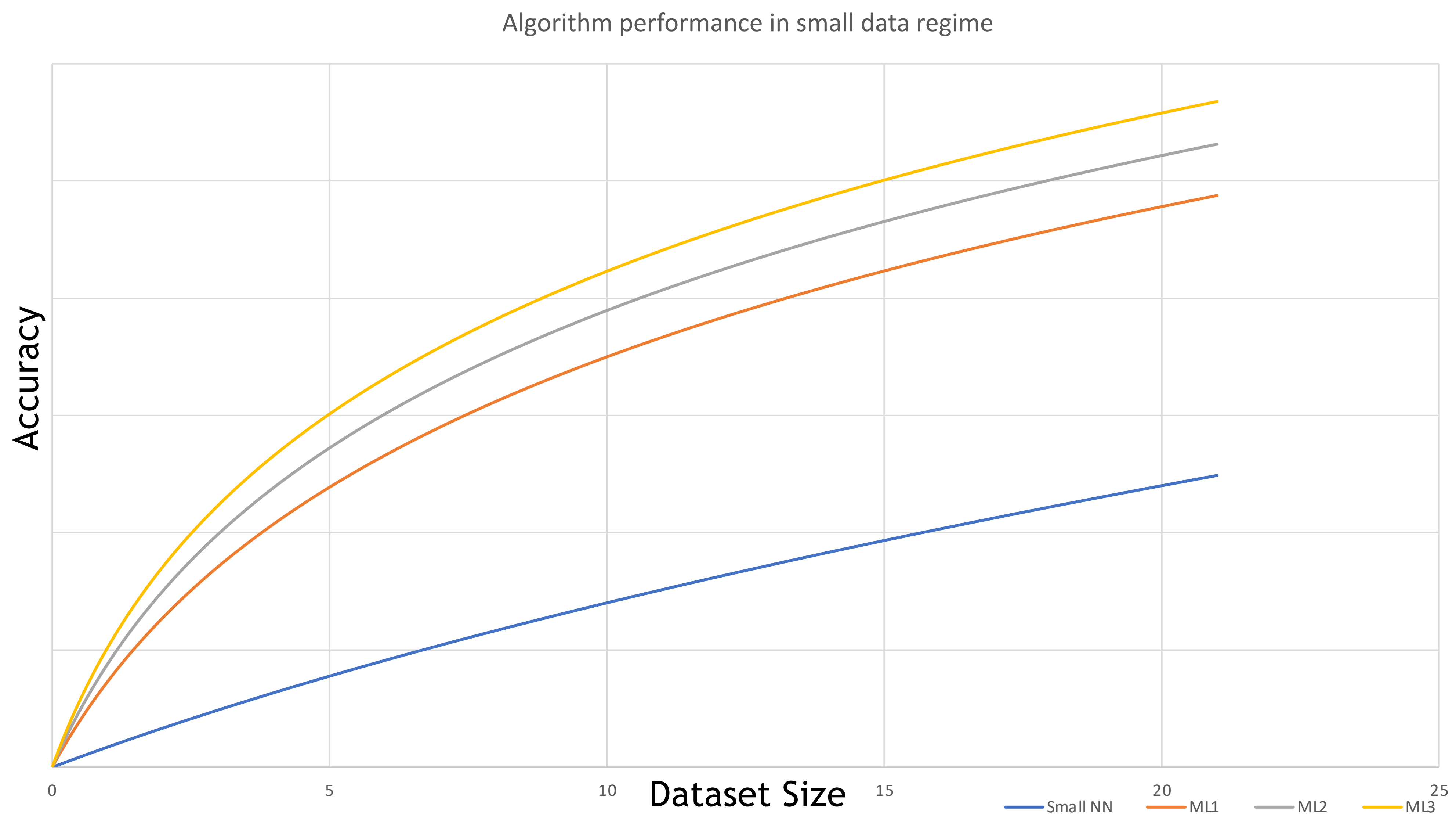
They just historically never worked well



Andrew Ng, “Nuts and Bolts of Applying Deep Learning”, https://www.youtube.com/watch?v=F1ka6a13S9I

NEURAL NETWORKS ARE NOT NEW

They just historically never worked well

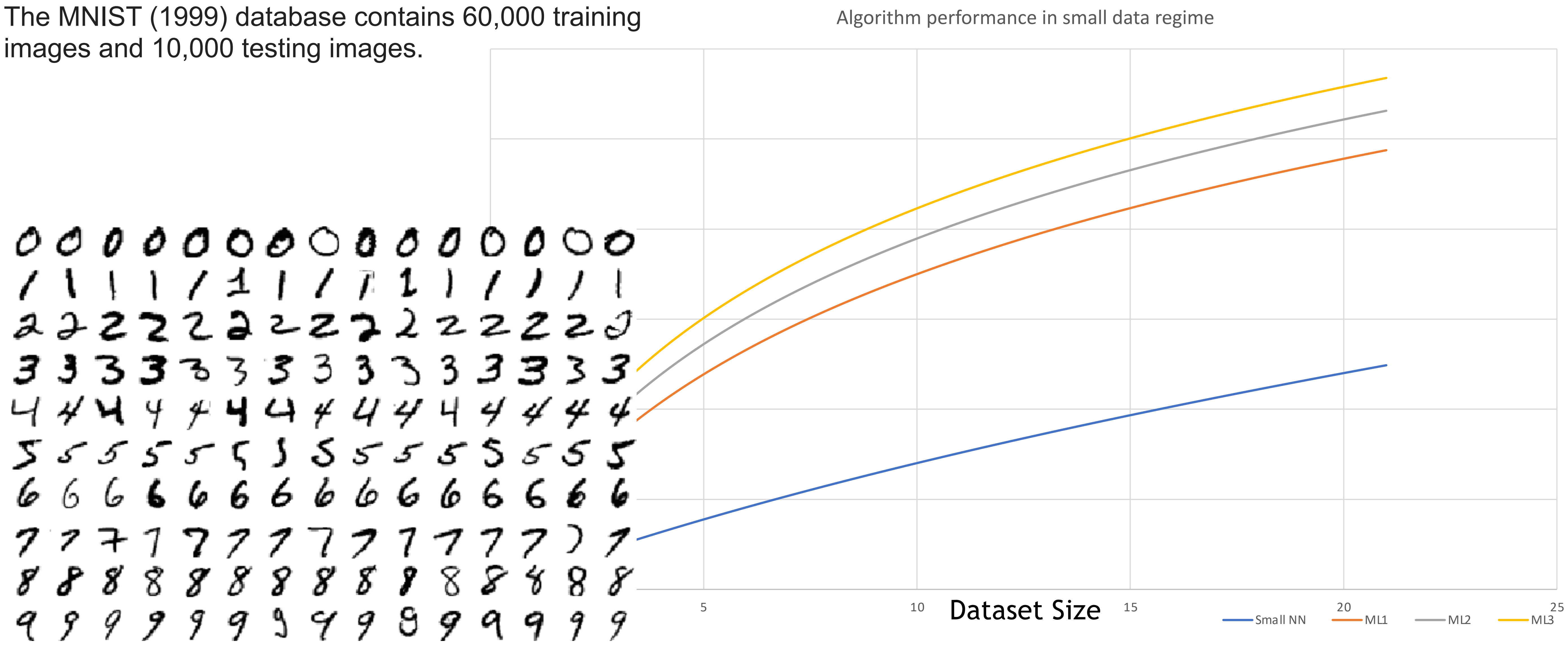


Andrew Ng, “Nuts and Bolts of Applying Deep Learning”, https://www.youtube.com/watch?v=F1ka6a13S9I

NEURAL NETWORKS ARE NOT NEW

Historically, we never had large datasets or computers

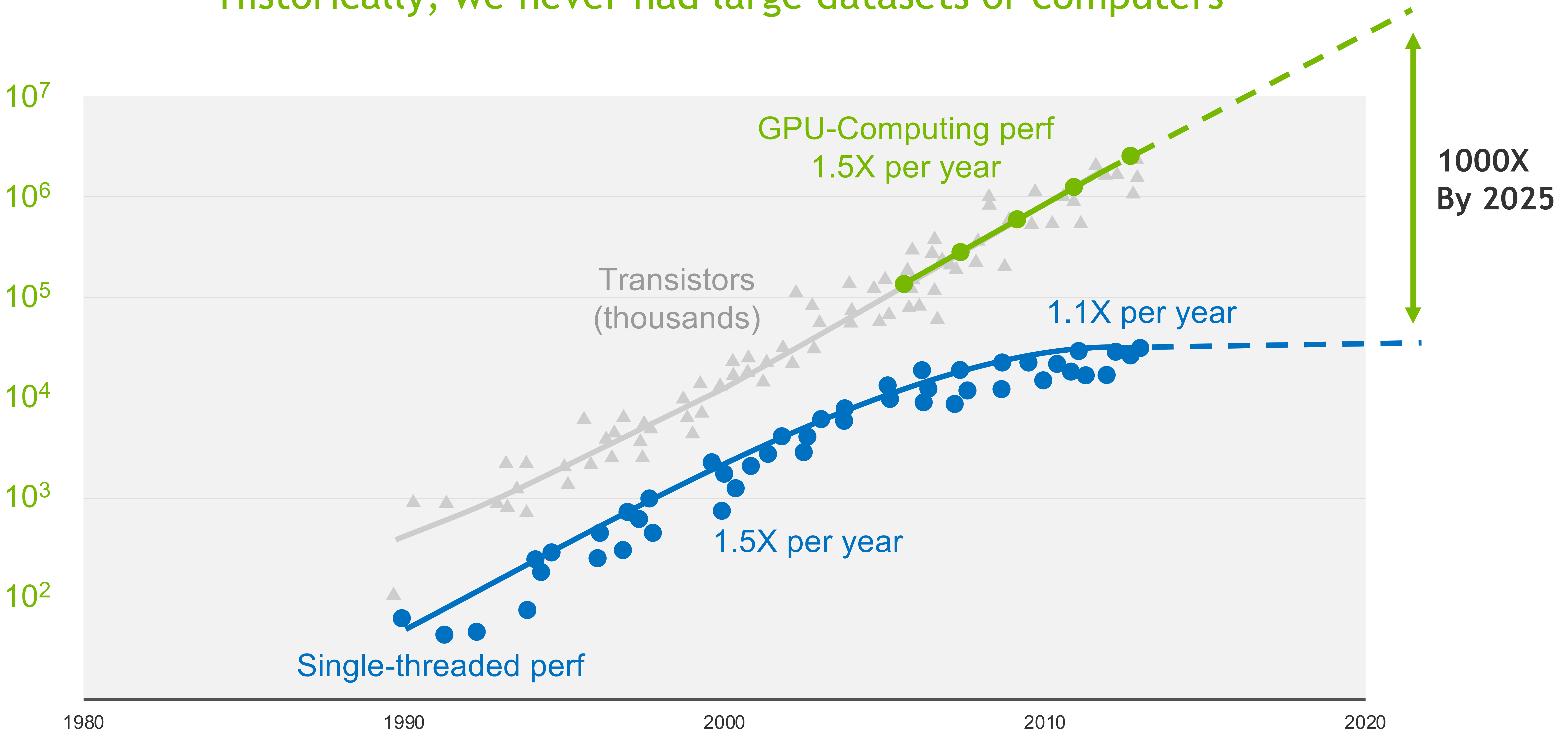
The MNIST (1999) database contains 60,000 training images and 10,000 testing images.



Andrew Ng, “Nuts and Bolts of Applying Deep Learning”, https://www.youtube.com/watch?v=F1ka6a13S9I

COMPUTE

Historically, we never had large datasets or computers



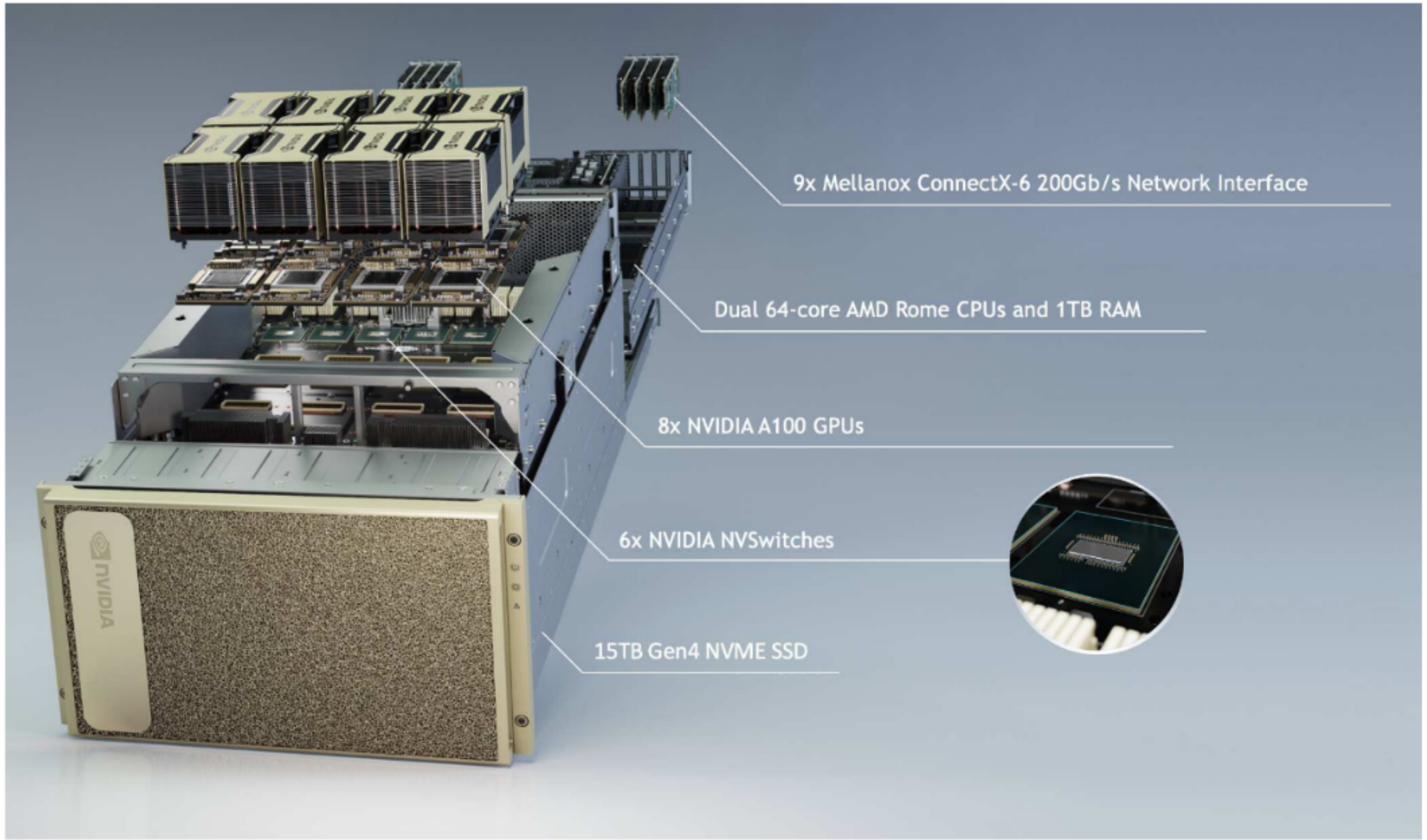
CONTEXT

CONTEXT

8 petaFLOPs in June 2011 (K Computer)

CONTEXT

5 petaFLOPs for AI - today

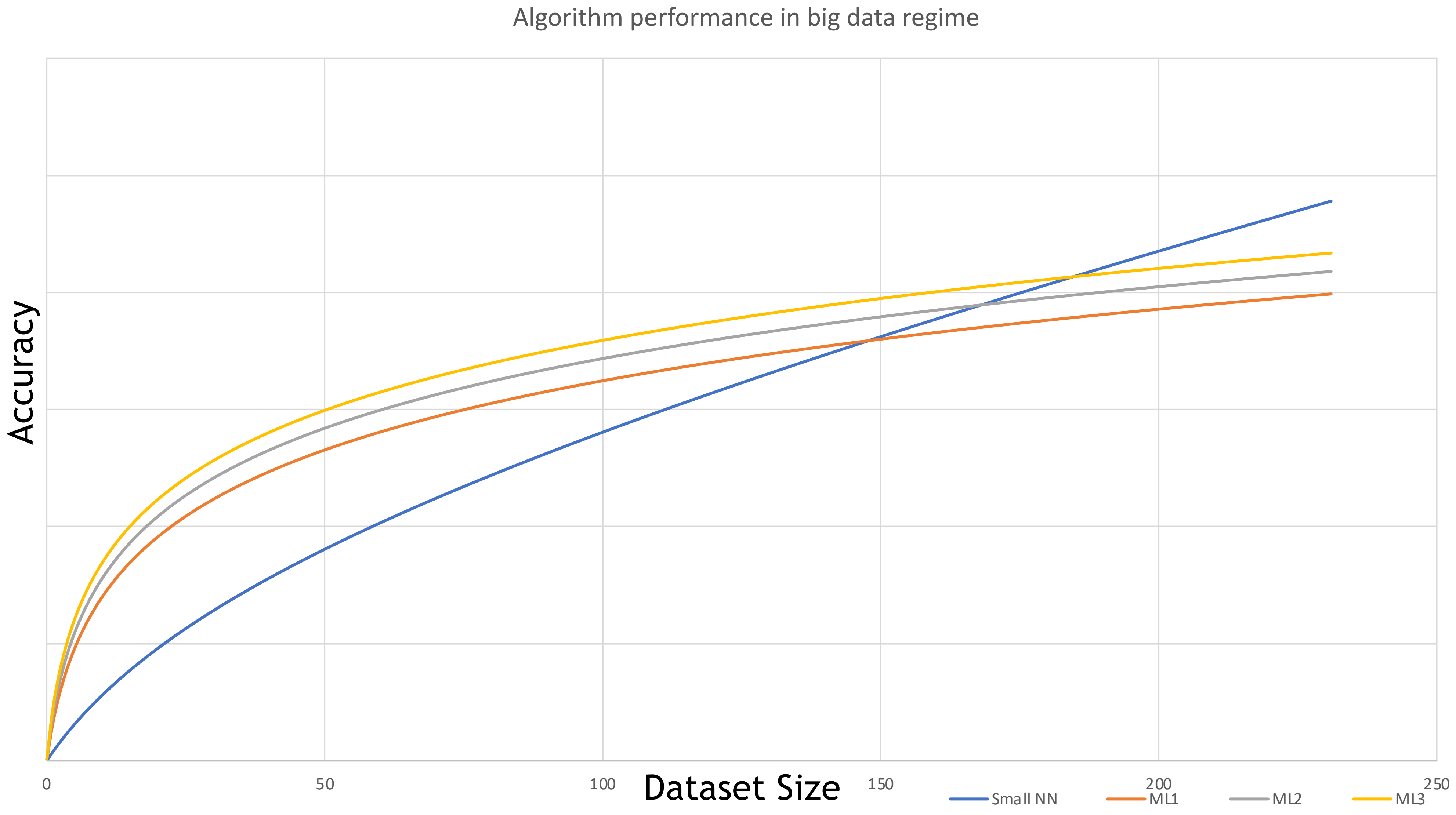


CONTEXT

~100 PFLOPS (FP16) or 48 PFLOPS (TF32) for AI - today

NEURAL NETWORKS ARE NOT NEW

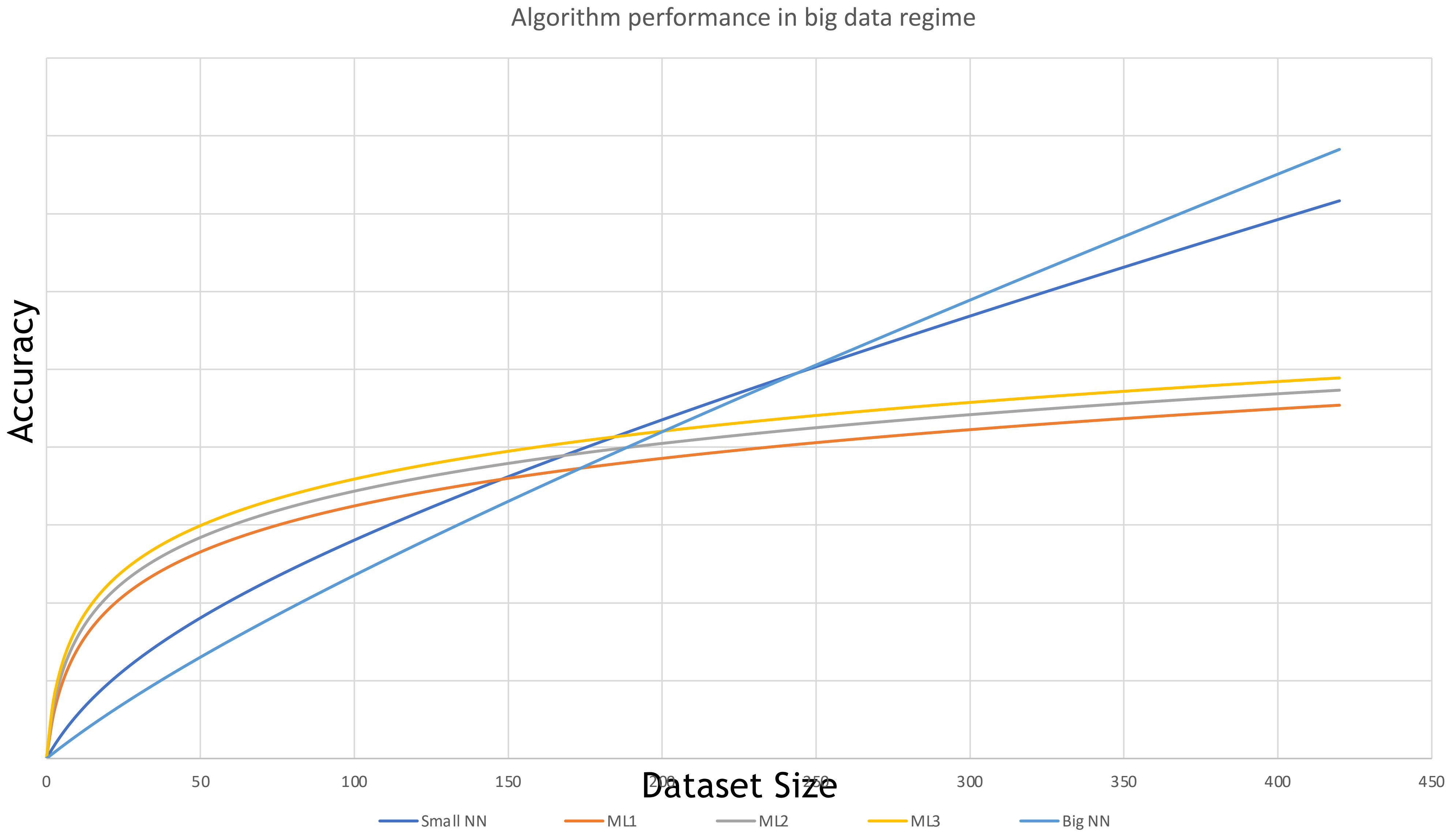
Large datasets and faster compute transformed the way we do machine learning



Andrew Ng, “Nuts and Bolts of Applying Deep Learning”, https://www.youtube.com/watch?v=F1ka6a13S9I

NEURAL NETWORKS ARE NOT NEW

Data and model size the key to accuracy



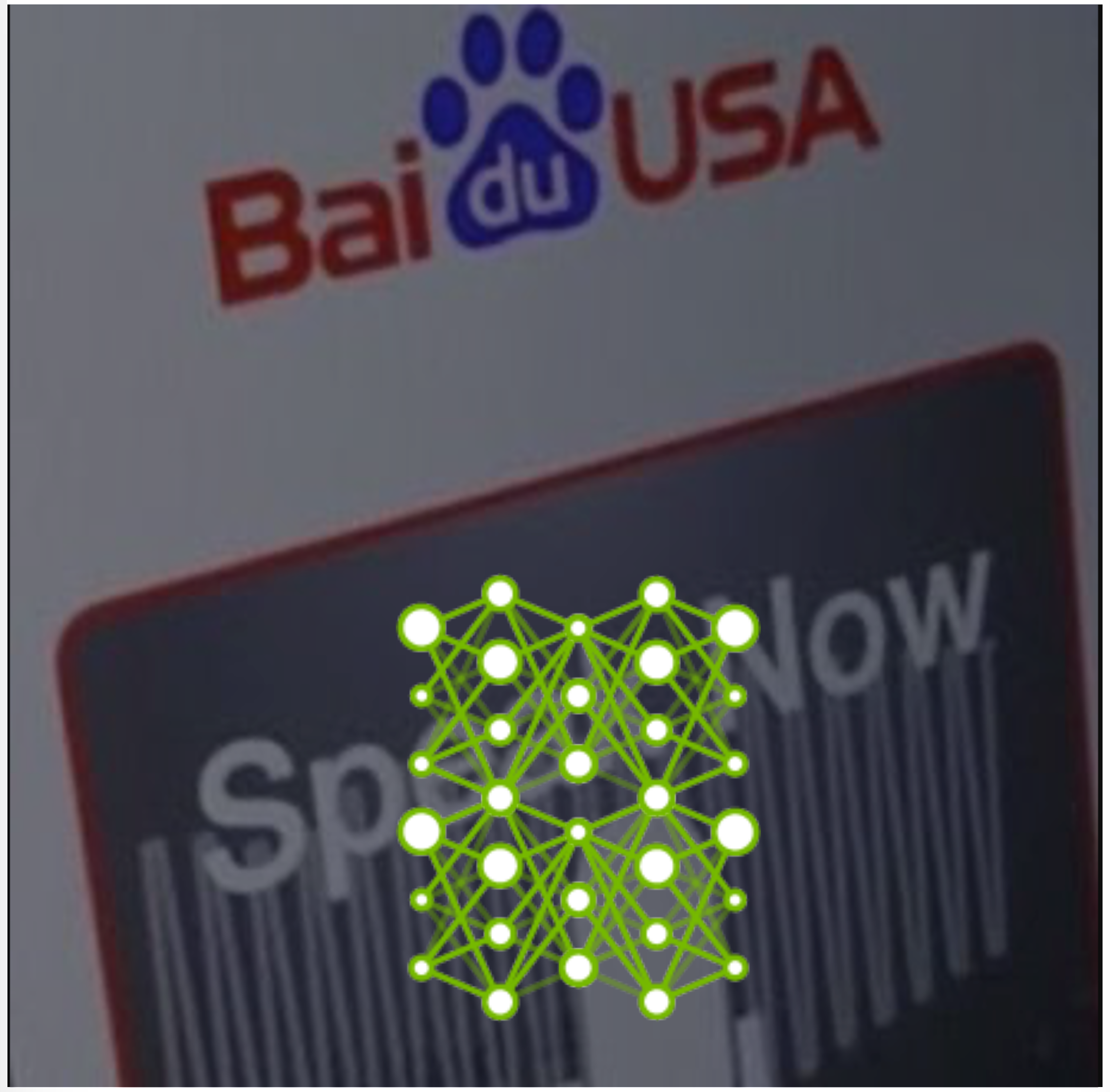
NEURAL NETWORK COMPLEXITY IS EXPLODING

To Tackle Increasingly Complex Challenges

7 ExaFLOPS
60 Million Parameters

2015 - Microsoft ResNet
Superhuman Image Recognition

20 ExaFLOPS
300 Million Parameters



2016 - Baidu Deep Speech 2
Superhuman Voice Recognition

100 ExaFLOPS
8700 Million Parameters

2017 - Google Neural Machine Translation
Near Human Language Translation

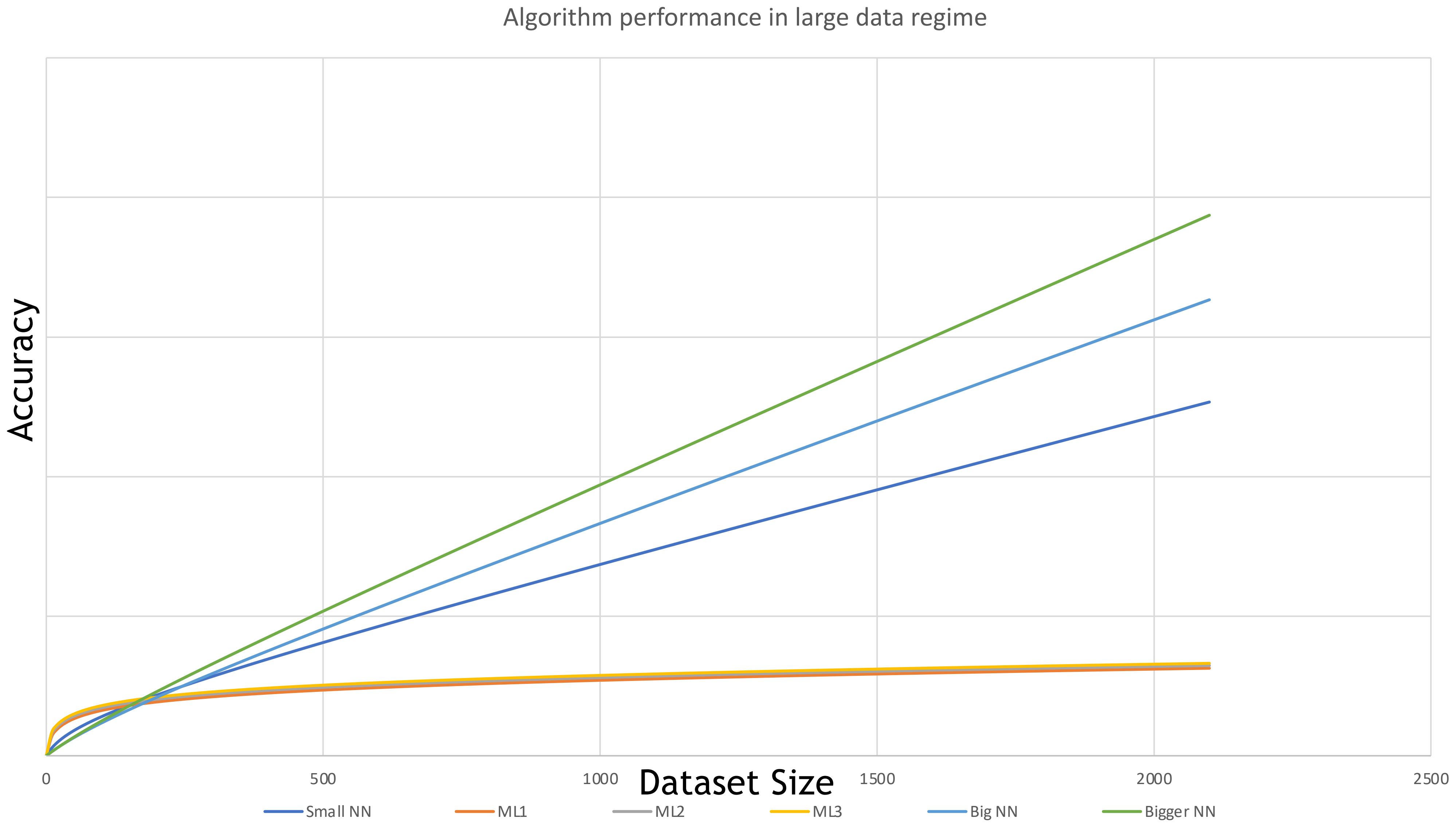
100 EXAFLOPS

~=

2 YEARS ON A DUAL CPU SERVER

NEURAL NETWORKS ARE NOT NEW

Exceeding human level performance



Andrew Ng, “Nuts and Bolts of Applying Deep Learning”, https://www.youtube.com/watch?v=F1ka6a13S9I

EMPIRICAL EVIDENCE

EXPLODING DATASETS

Logarithmic relationship between the dataset size and accuracy

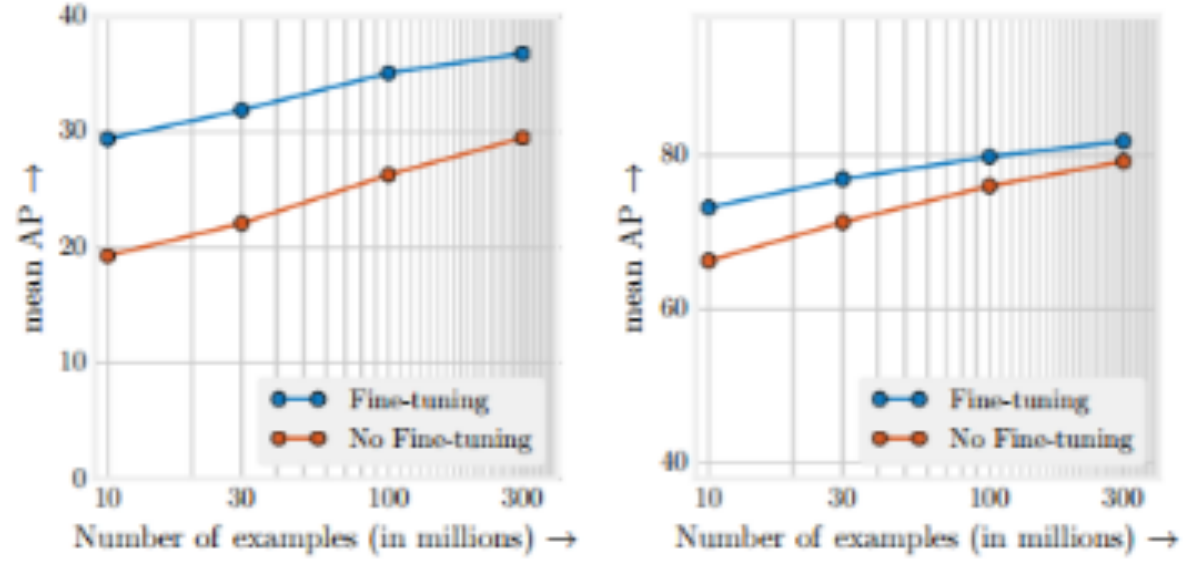


Figure 4. Object detection performance when initial checkpoints are pre-trained on different subsets of JFT-300M from scratch. x-axis is the data size in log-scale, y-axis is the detection performance in mAP@[.5,.95] on COCO minival* (left), and in mAP@.5 on PASCAL VOC 2007 test (right).

Initialization	mIOU
ImageNet	73.6
300M	75.3
ImageNet+300M	**76.5**

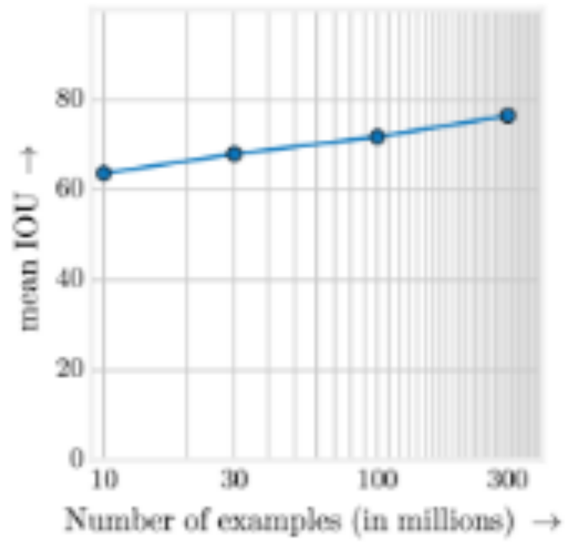
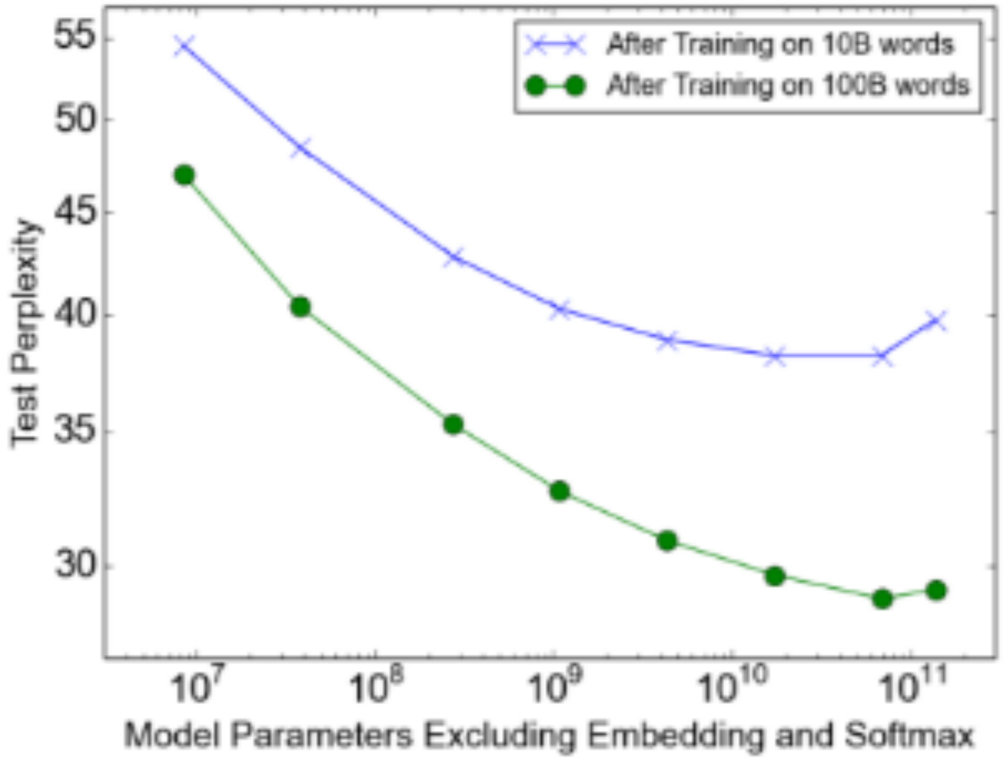


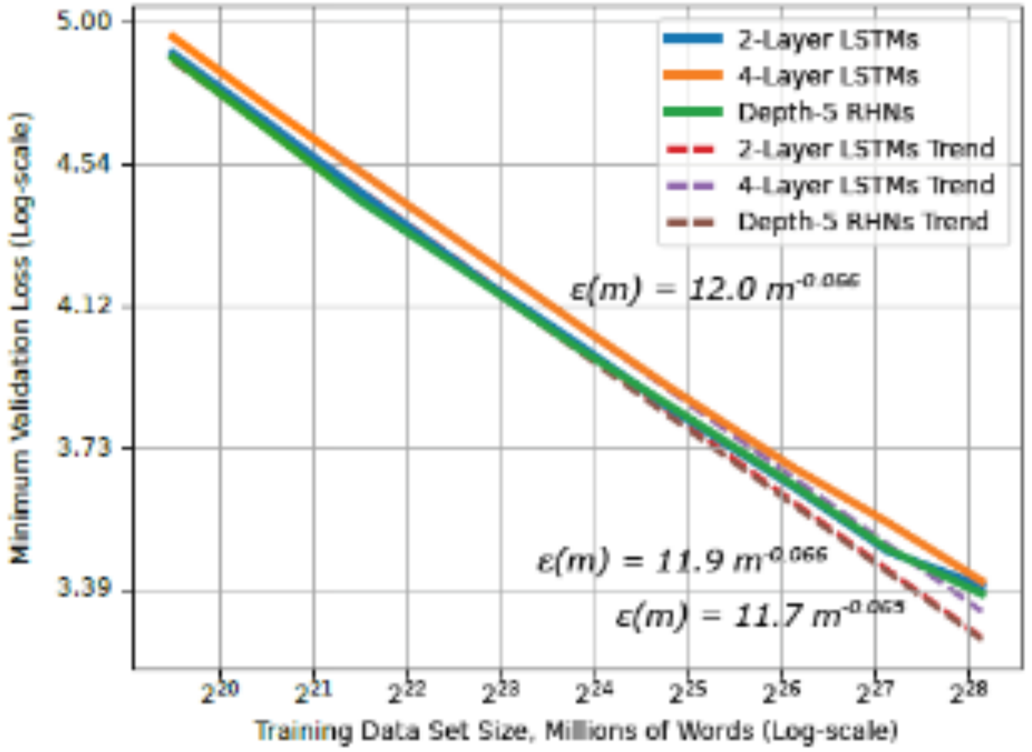
Figure 6. Semantic segmentation performance on Pascal VOC 2012 val set. (left) Quantitative performance of different initializations; (right) Impact of data size on performance.



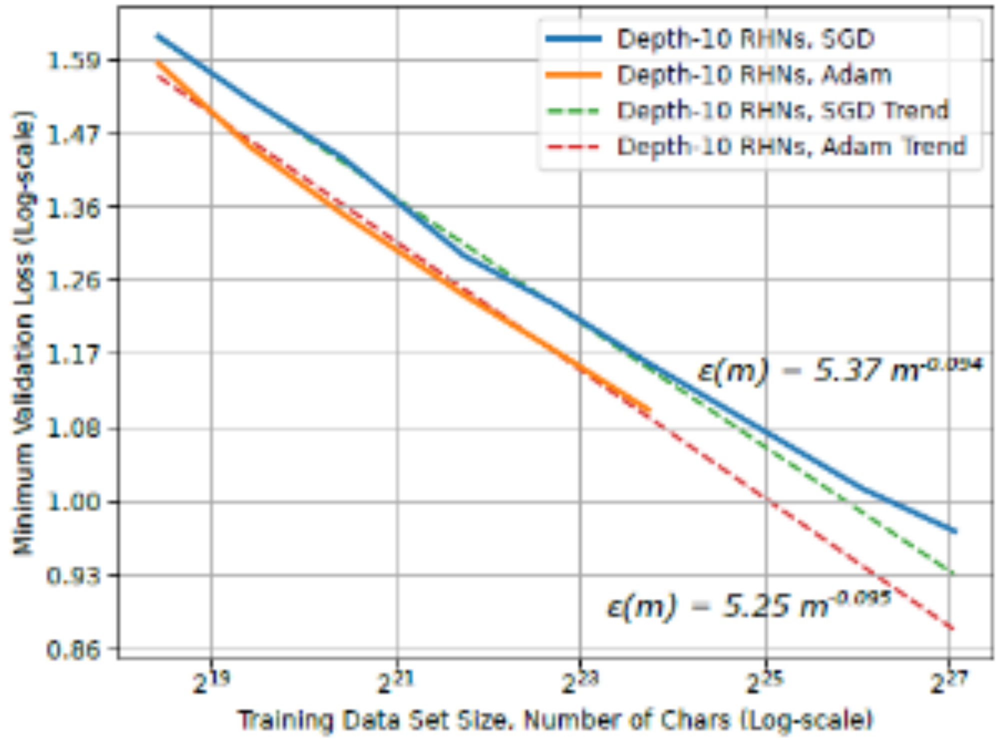
Sun, Chen, et al. "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era." *arXiv preprint arXiv:1707.02968* (2017).
Shazeer, Noam, et al. "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer." arXiv preprint arXiv:1701.06538 (2017).

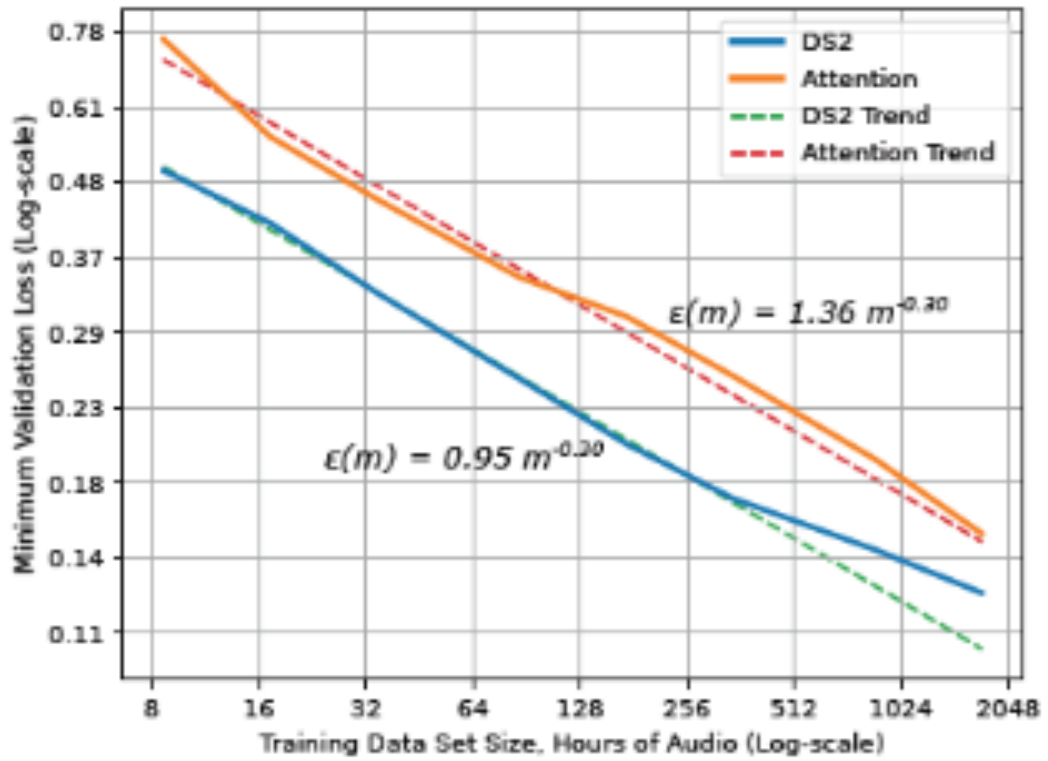
EXPLODING DATASETS

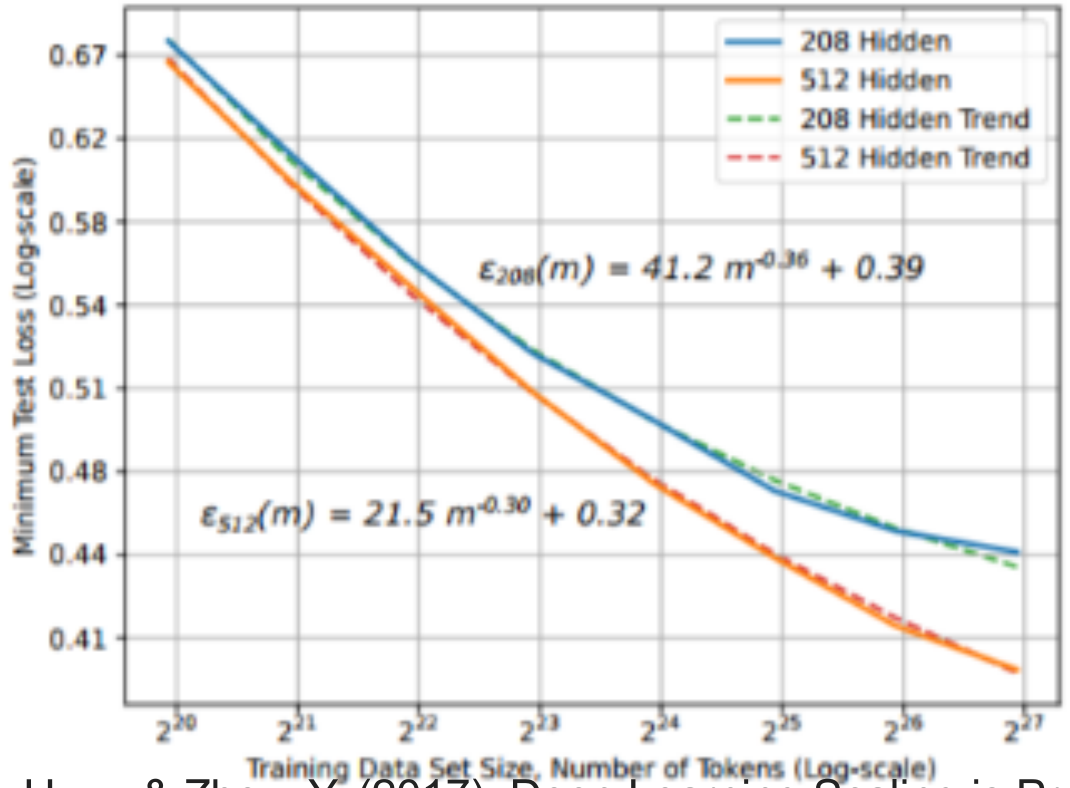
Logarithmic relationship between the dataset size and accuracy

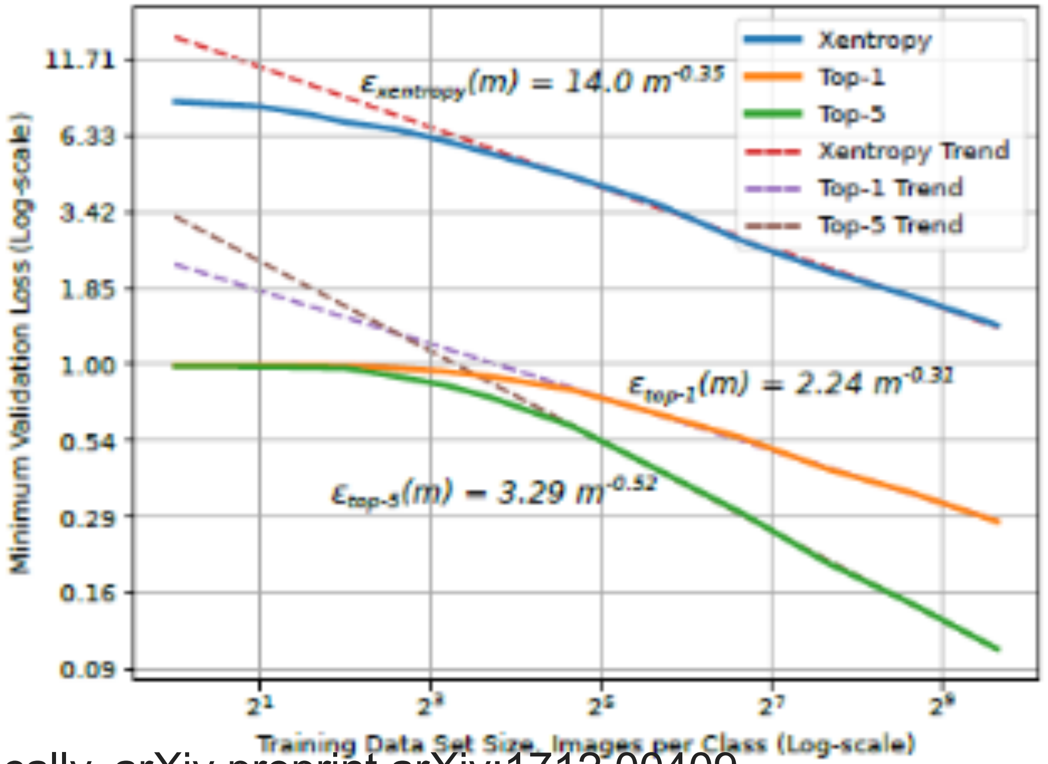


- Translation
- Language Models
- Character Language Models
- Image Classification
- Attention Speech Models





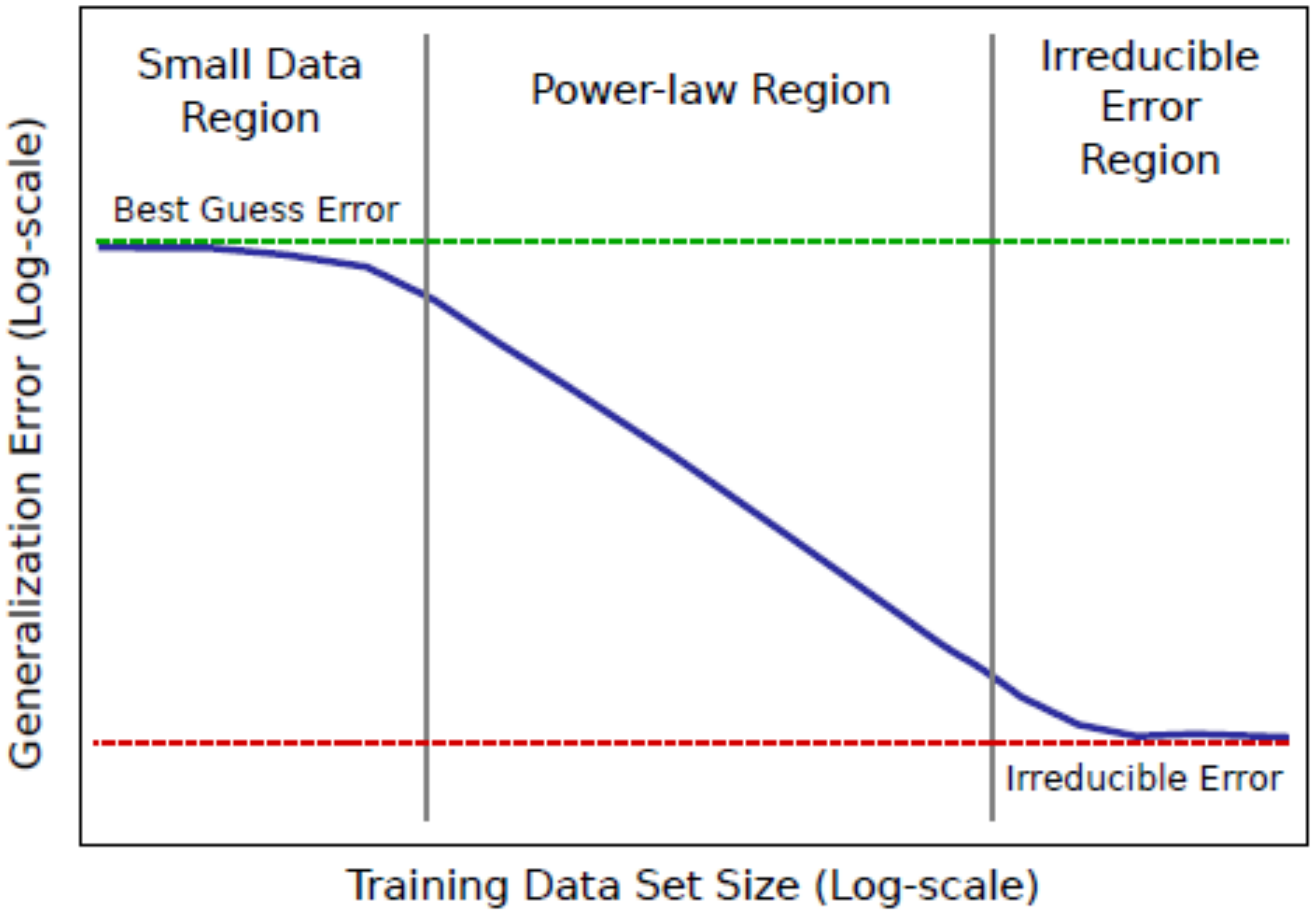




Hestness, J., Narang, S., Ardalani, N., Diamos, G., Jun, H., Kianinejad, H., ... & Zhou, Y. (2017). Deep Learning Scaling is Predictable, Empirically. arXiv preprint arXiv:1712.00409.

EXPLODING DATASETS

Logarithmic relationship between the dataset size and accuracy

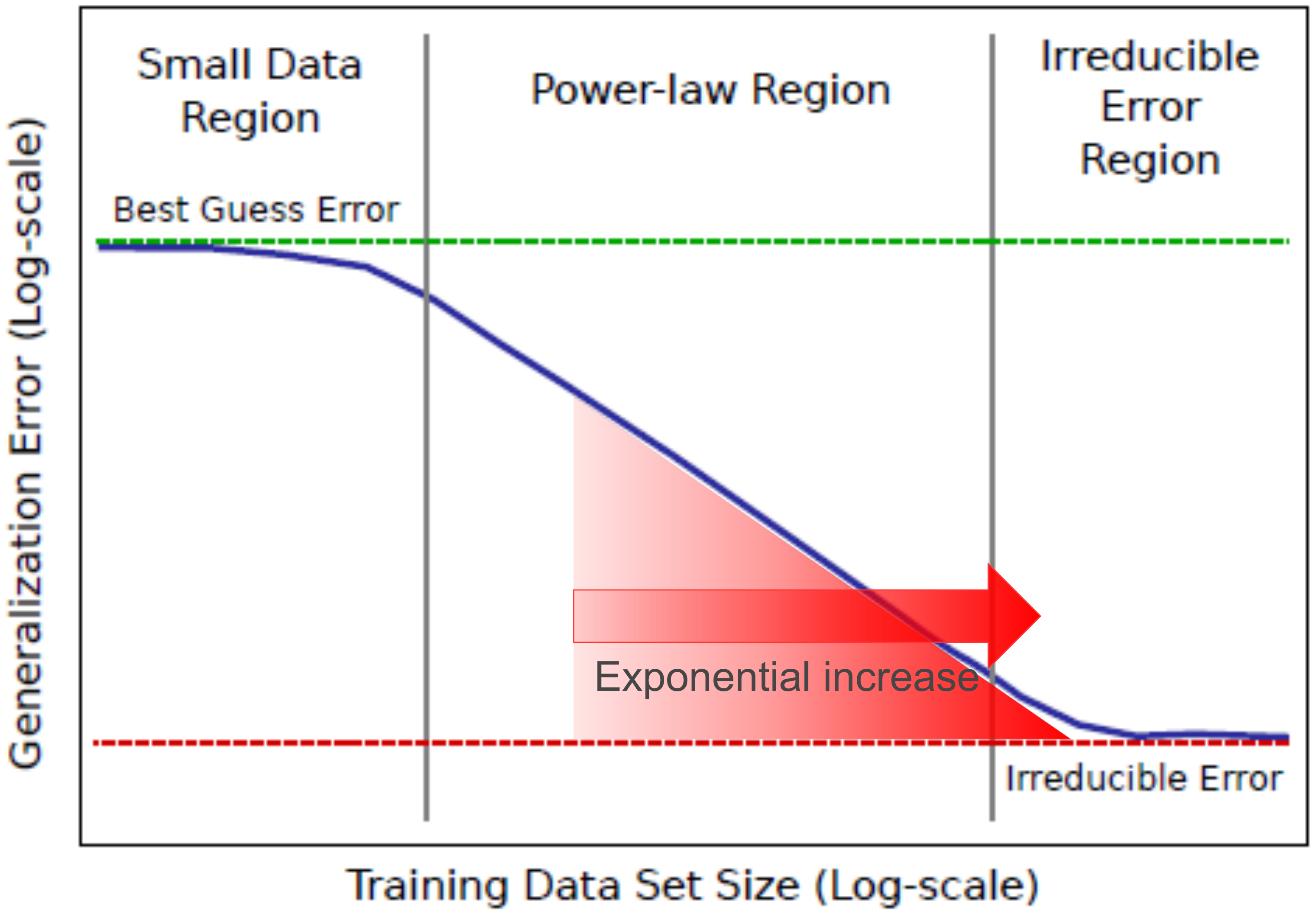


Hestness, J., Narang, S., Ardalani, N., Diamos, G., Jun, H., Kianinejad, H., ... & Zhou, Y. (2017). Deep Learning Scaling is Predictable, Empirically. arXiv preprint arXiv:1712.00409

THE COST

THE COST OF LABELING

Limits the utility of deep learning models



Hestness, J., Narang, S., Ardalani, N., Diamos, G., Jun, H., Kianinejad, H., ... & Zhou, Y. (2017). Deep Learning Scaling is Predictable, Empirically. arXiv preprint arXiv:1712.00409

Part 2: Self-Supervision, BERT and Beyond

- Lecture
  - Why Do DNNs Work Well?
  - Self-Supervised Learning
  - BERT
- Lab
  - Explore the Data
  - Explore NeMo
  - Text Classifier Project
- Lecture (cont'd)
  - Bigger is Better
  - Can and should we go even bigger?
- Lab (cont'd)
  - Named Entity Recognizer

SELF-SUPERVISED LEARNING

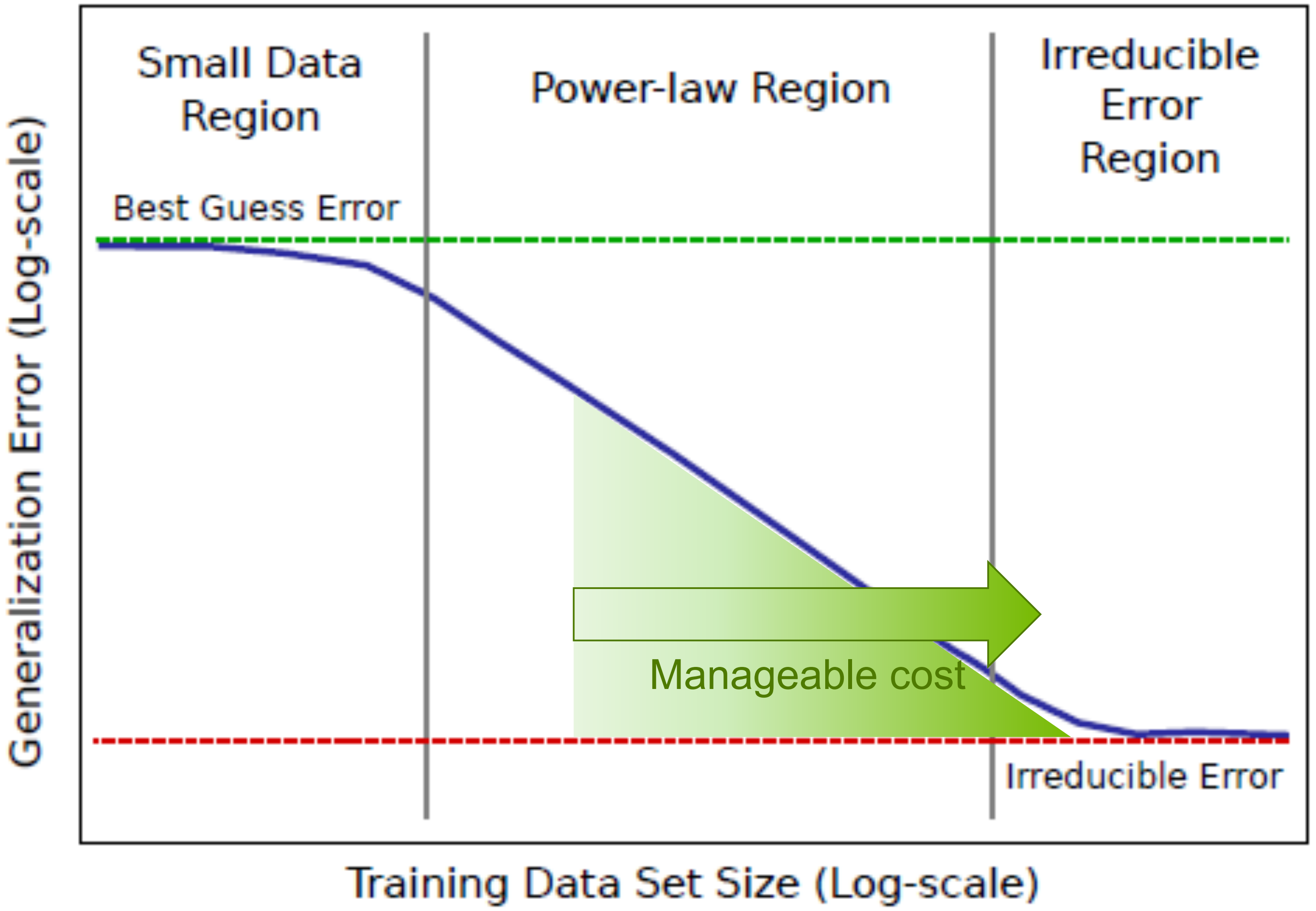
Example training tasks

- Natural Language Processing:
  - Masked Language Model: We mask a percentage of the input tokens at random (say 15%) and ask the neural network to predict the entire sentence
  - Next Sentence Prediction: We choose either two consecutive sentences from text, or two random sentences from the text. We ask the neural network to establish whether the two sentences occur one after another.
  - We use another simpler neural network to replace random words in the sequence and ask the primary neural network to detect which words were replaced (using a GAN like configuration).
- Computer Vision:
  - Contrastive Learning: Randomly modify (crop and resize, flip, distort color, rotate, cut-out, noise, blur, etc.) and either feed the same image, or two randomly selected images, into the neural network, asking it to say whether it is the same image or not
  - Noisy labels/Self Training: Use labels generated by a weak algorithm (potentially older generation of the target model) to train a target-robust feature extractor

Dai, A. M., & Le, Q. V. (2015). Semi-supervised sequence learning. In Advances in neural information processing systems (pp. 3079-3087).
Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. arXiv preprint arXiv:2002.05709.
Xie, Q., Hovy, E., Luong, M. T., & Le, Q. V. (2019). Self-training with Noisy Student improves ImageNet classification. arXiv preprint arXiv:1911.04252.

THE COST OF LABELING

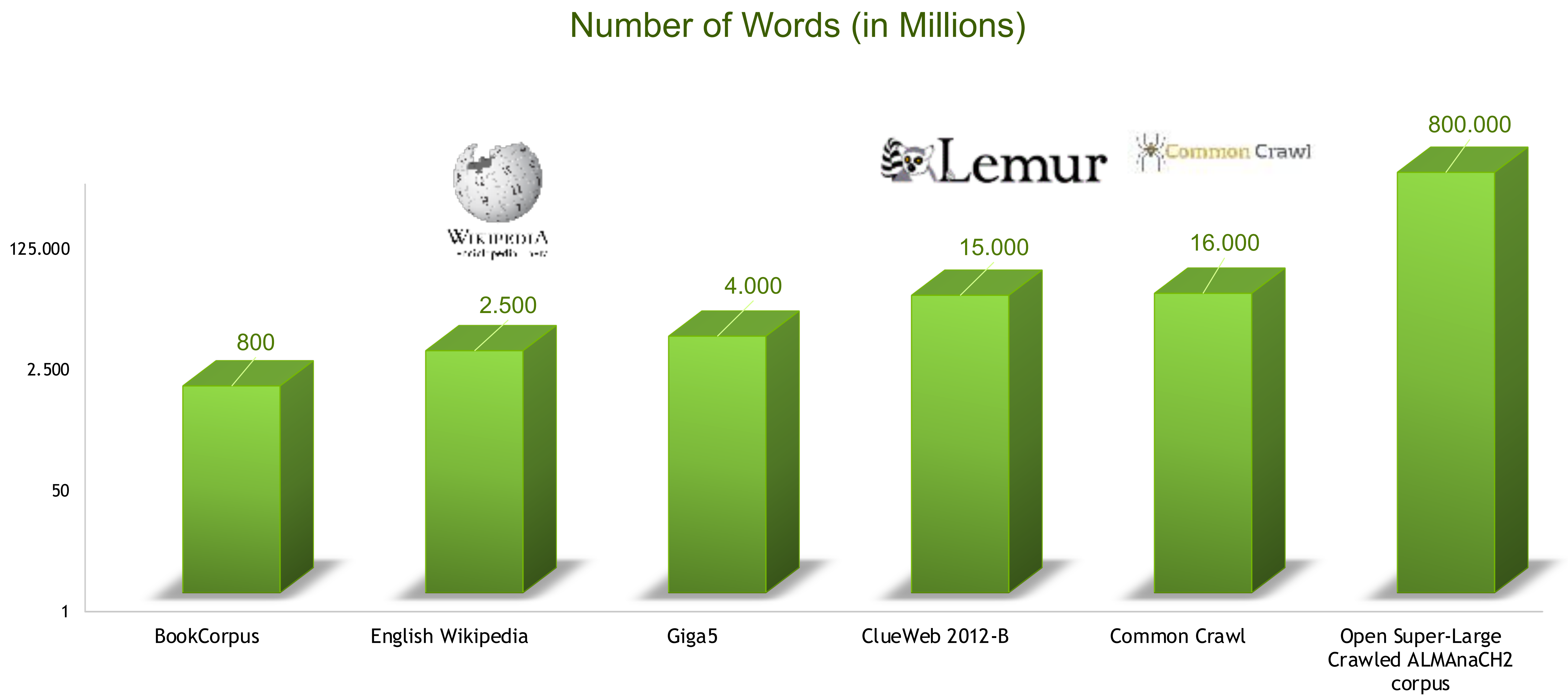
Semi-supervised models



Hestness, J., Narang, S., Ardalani, N., Diamos, G., Jun, H., Kianinejad, H., ... & Zhou, Y. (2017). Deep Learning Scaling is Predictable, Empirically. arXiv preprint arXiv:1712.00409

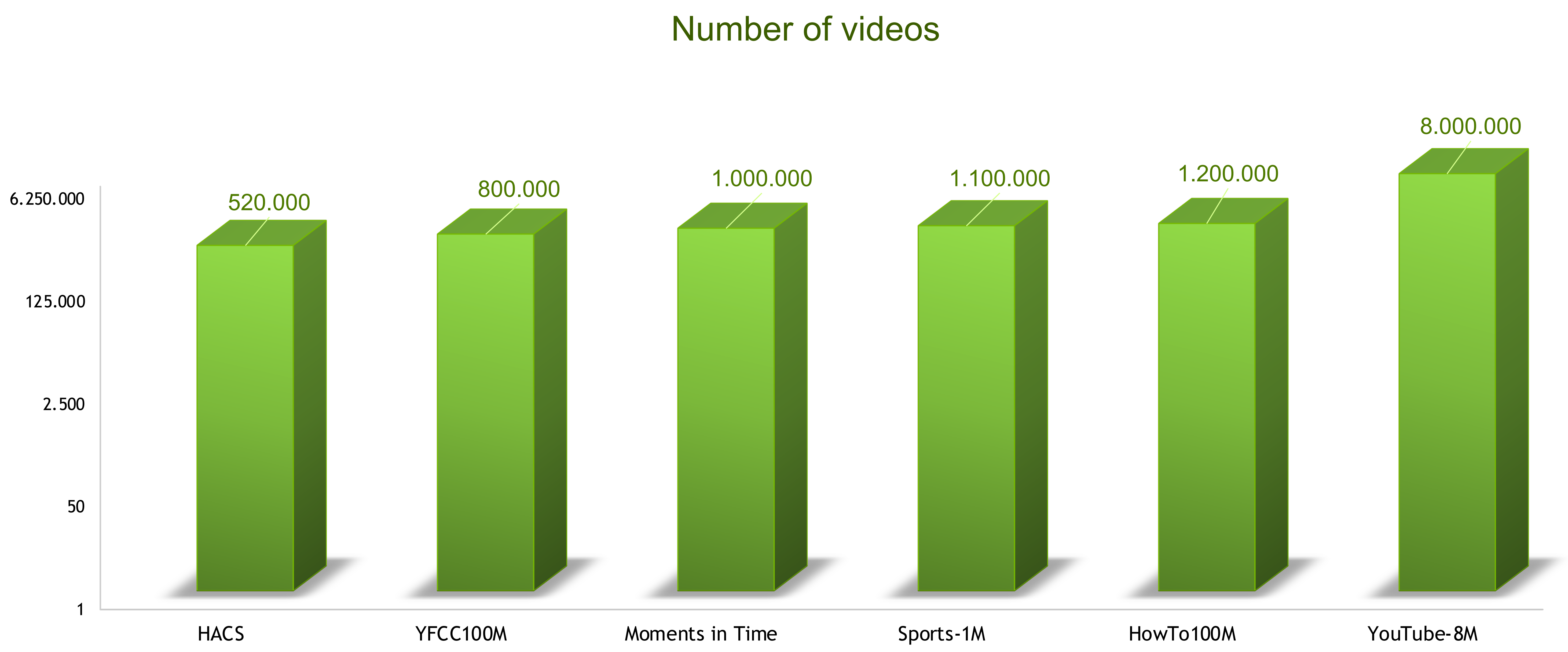
SELF-SUPERVISED LEARNING

Abundance of unlabeled data



SELF-SUPERVISED LEARNING

Abundance of unlabeled data



OLD IDEAS

SELF-SUPERVISED LEARNING

What was missing?

Semi-supervised Sequence Learning

Andrew M. Dai
Google Inc.
adai@google.com

Quoc V. Le
Google Inc.
qvl@google.com

Abstract

We present two approaches that use unlabeled data to improve sequence learning with recurrent networks. The first approach is to predict what comes next in a sequence, which is a conventional language model in natural language processing. The second approach is to use a sequence autoencoder, which reads the input sequence into a vector and predicts the input sequence again. These two algorithms can be used as a "pretraining" step for a later supervised sequence learning algorithm. In other words, the parameters obtained from the unsupervised step can be used as a starting point for other supervised training models. In our experiments, we find that long short term memory recurrent networks after being pretrained with the two approaches are more stable and generalize better. With pretraining, we are able to train long short term memory recurrent networks up to a few hundred timesteps, thereby achieving strong performance in many text classification tasks, such as IMDB, DBpedia and 20 Newsgroups.

THE SCALE

GENERATIVE PRETRAINING (GPT)

The scale

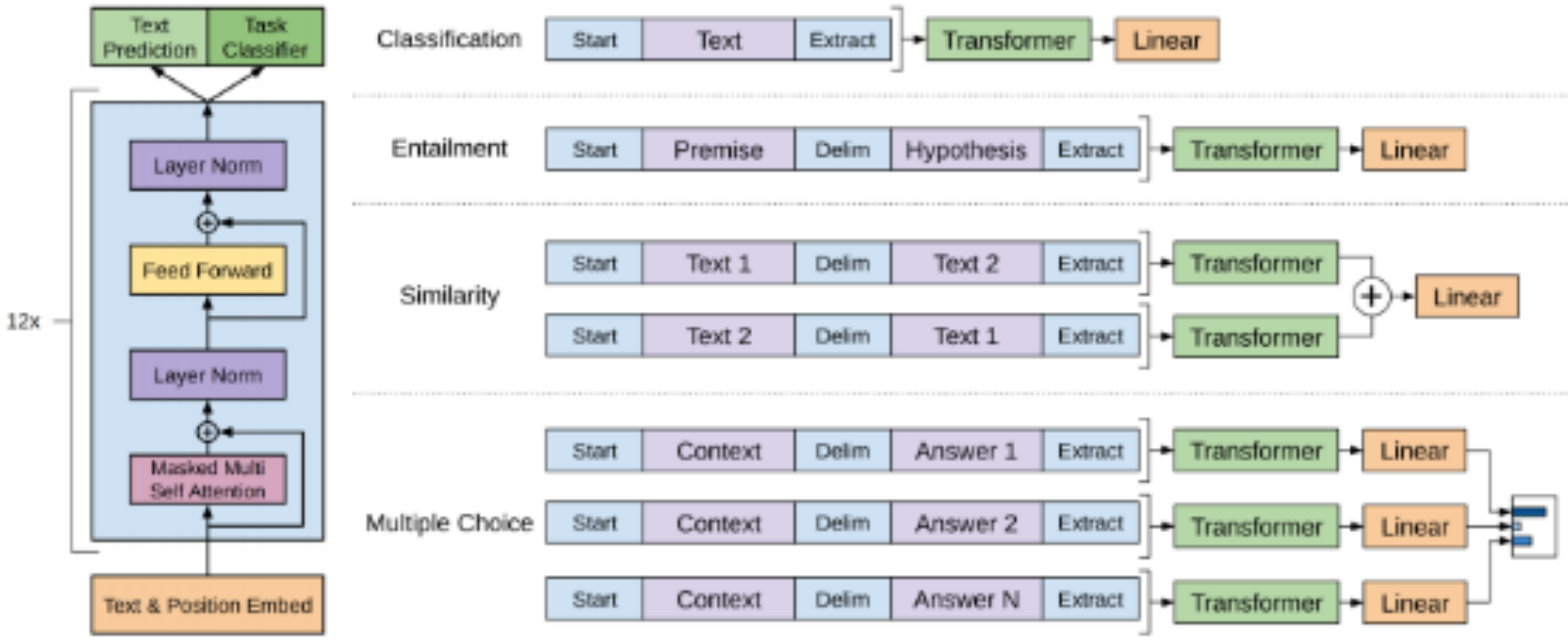
"Many previous approaches to NLP tasks train relatively small models on a single GPU from scratch. Our approach requires an expensive pre-training step - 1 month on 8 GPUs. Luckily, this only has to be done once and we're releasing our model so others can avoid it. It is also a large model (in comparison to prior work) and consequently uses more compute and memory — we used a 37-layer (12 block) Transformer architecture, and we train on sequences of up to 512 tokens. Most experiments were conducted on 4 and 8 GPU systems. The model does fine-tune to new tasks very quickly which helps mitigate the additional resource requirements."

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf.*

GENERATIVE PRETRAINING (GPT)

The design

Transformer Decoder



Self-Supervised Training

Figure 1: **(left)** Transformer architecture and training objectives used in this work. **(right)** Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear+softmax layer.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf.*

GENERATIVE PRETRAINING (GPT)

The approach

Pre-training our model on a large corpus of text significantly improves its performance on challenging natural language processing tasks like Winograd Schema Resolution.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf.*

GENERATIVE PRETRAINING (GPT)

The implications

Pre-training our model on a large corpus of text significantly improves its performance on challenging natural language processing tasks like Winograd Schema Resolution.

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf.*

GENERATIVE PRETRAINING (GPT)
The implications

DATASET	TASK	SOTA	OURS
SNLI	Textual Entailment	89.3	**89.9**
MNLI Matched	Textual Entailment	80.6	**82.1**
MNLI Mismatched	Textual Entailment	80.1	**81.4**
SciTail	Textual Entailment	83.3	**88.3**
QNLI	Textual Entailment	82.3	**88.1**
RTE	Textual Entailment	**61.7**	56.0
STS-B	Semantic Similarity	81.0	**82.0**
QQP	Semantic Similarity	66.1	**70.3**
MRPC	Semantic Similarity	**86.0**	82.3
RACE	Reading Comprehension	53.3	**59.0**
ROCStories	Commonsense Reasoning	77.6	**86.5**
COPA	Commonsense Reasoning	71.2	**78.6**
SST-2	Sentiment Analysis	**93.2**	91.3
CoLA	Linguistic Acceptability	35.0	**45.4**
GLUE	Multi Task Benchmark	68.9	**72.8**

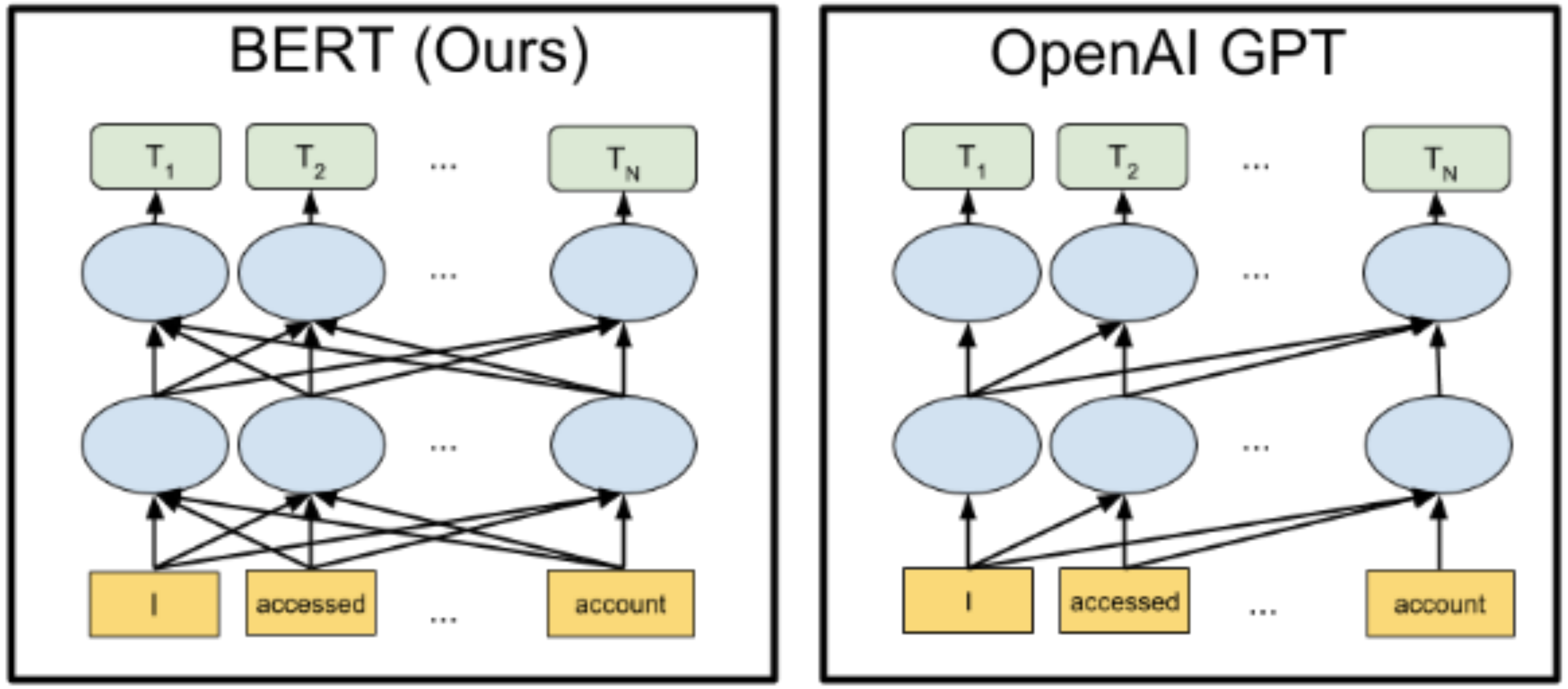
Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf.*

Part 2: Self-Supervision, BERT and Beyond

- Lecture
  - Why Do DNNs Work Well?
  - Self-Supervised Learning
  - BERT
- Lab
  - Explore the Data
  - Explore NeMo
  - Text Classifier Project
- Lecture (cont'd)
  - Bigger is Better
  - Can and should we go even bigger?
- Lab (cont'd)
  - Named Entity Recognizer

BIDIRECTIONAL TRANSFORMERS (BERT)

Building on the shoulders of giants



Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

BIDIRECTIONAL TRANSFORMERS (BERT)

The “pre” and “post” OpenAI ages

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

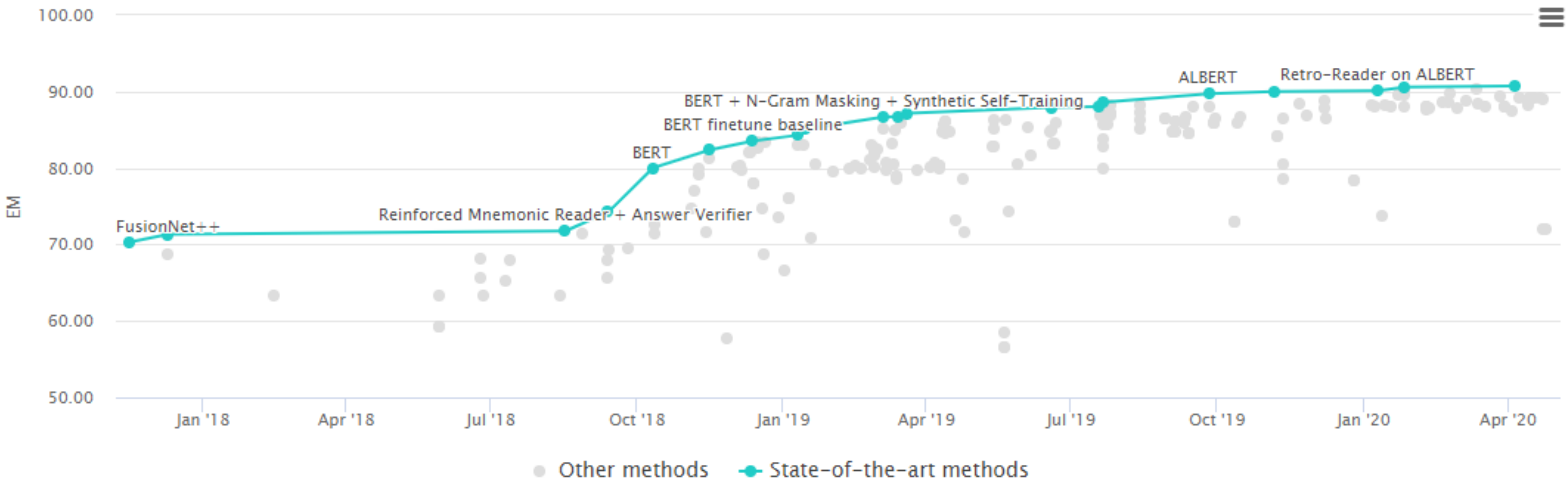
Table 1: GLUE Test results, scored by the evaluation server (`https://gluebenchmark.com/leaderboard`). The number below each task denotes the number of training examples. The “Average” column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

SQUAD 2.0

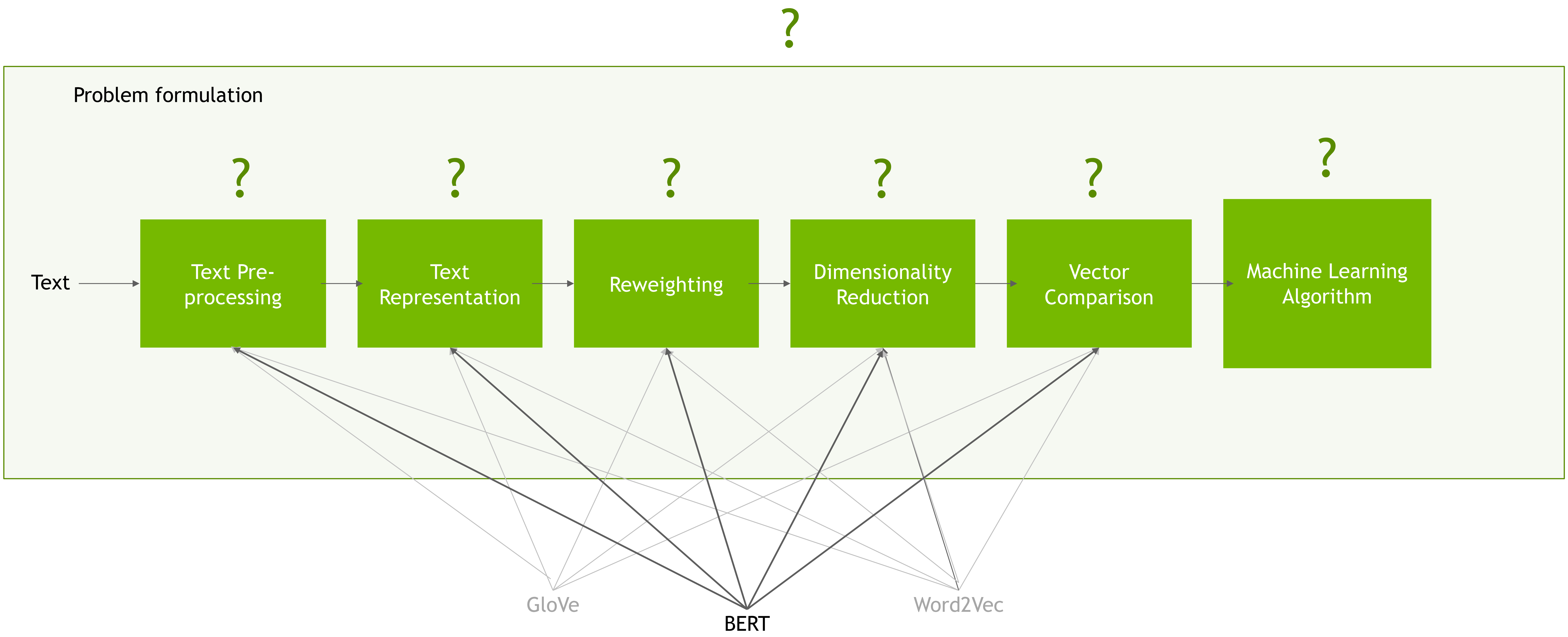
Human performance 91.2

Question Answering on SQuAD2.0



USING BERT

Feature extractor

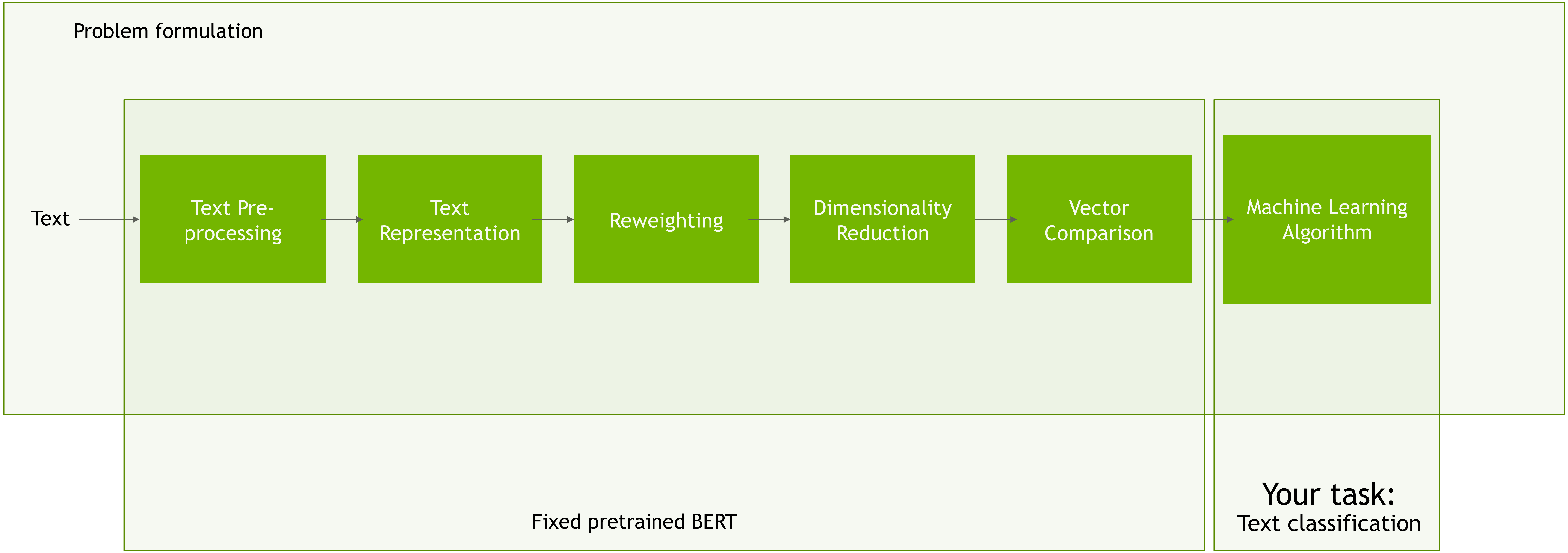


THE LAB

LAB OVERVIEW

Notebooks 1, 2, 3

Text classification



Part 2: Self-Supervision, BERT and Beyond

- Lecture
  - Why Do DNNs Work Well?
  - Self-Supervised Learning
  - BERT
- Lab
  - Explore the Data
  - Explore NeMo
  - Text Classifier Project
- Lecture (cont'd)
  - Bigger is Better
  - Can and should we go even bigger?
- Lab (cont'd)
  - Named Entity Recognizer

Part 2: Self-Supervision, BERT and Beyond

- Lecture
  - Why Do DNNs Work Well?
  - Self-Supervised Learning
  - BERT
- Lab
  - Explore the Data
  - Explore NeMo
  - Text Classifier Project
- Lecture (cont’d)
  - Bigger is Better
  - Can and should we go even bigger?
- Lab (cont’d)
  - Named Entity Recognizer

BIDIRECTIONAL TRANSFORMERS (BERT)

Base vs Large

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table 1: GLUE Test results, scored by the evaluation server (`https://gluebenchmark.com/leaderboard`). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

GPT-2

GPT vs GPT-2

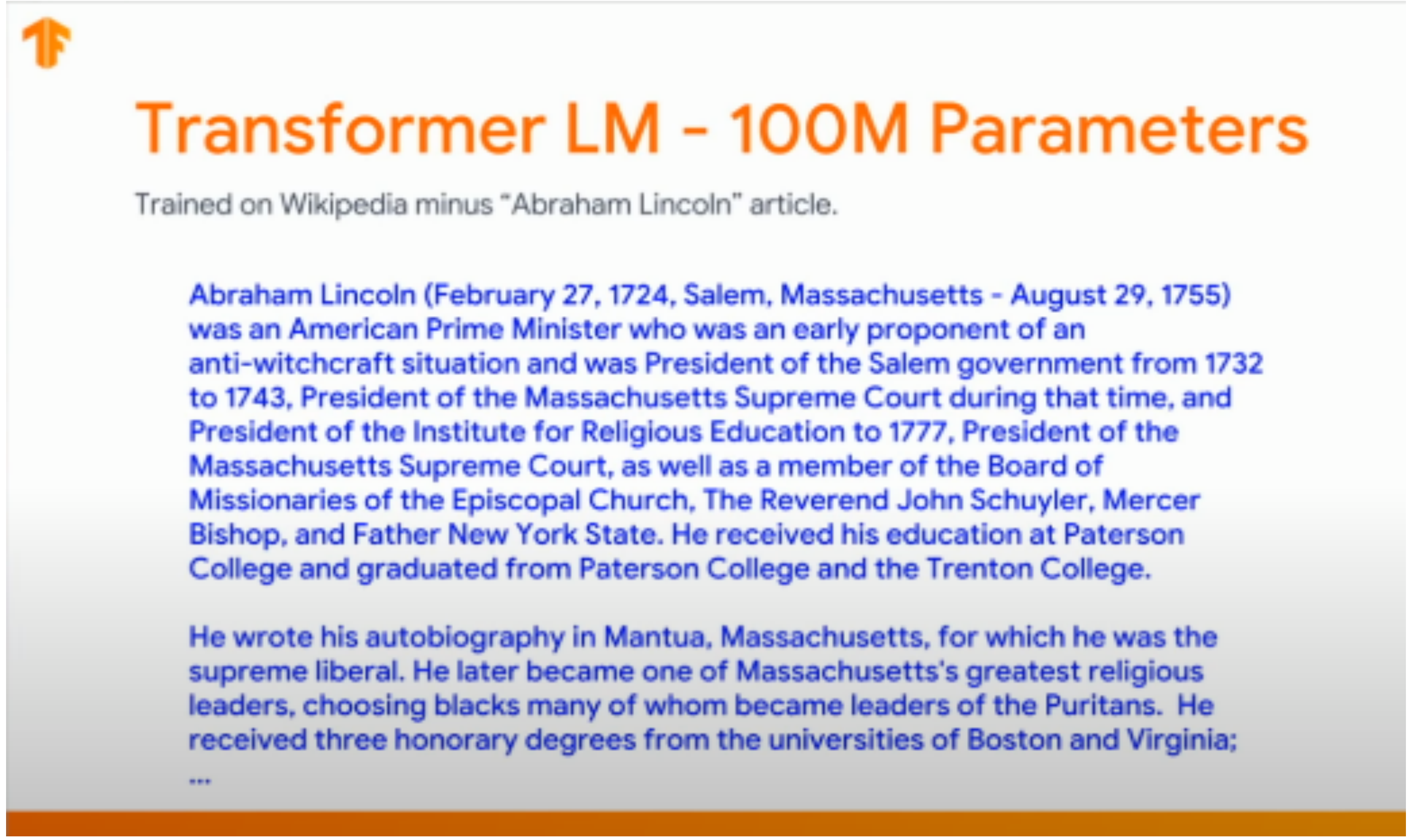
- Largely the same but:
  - Larger in every way:
    - More decoder layers: 12->48
    - Larger vocabulary: 50,257
    - Larger context: 512 -> 1024
    - Larger batch size
  - Changes to layer normalization
  - Different initialization scheme

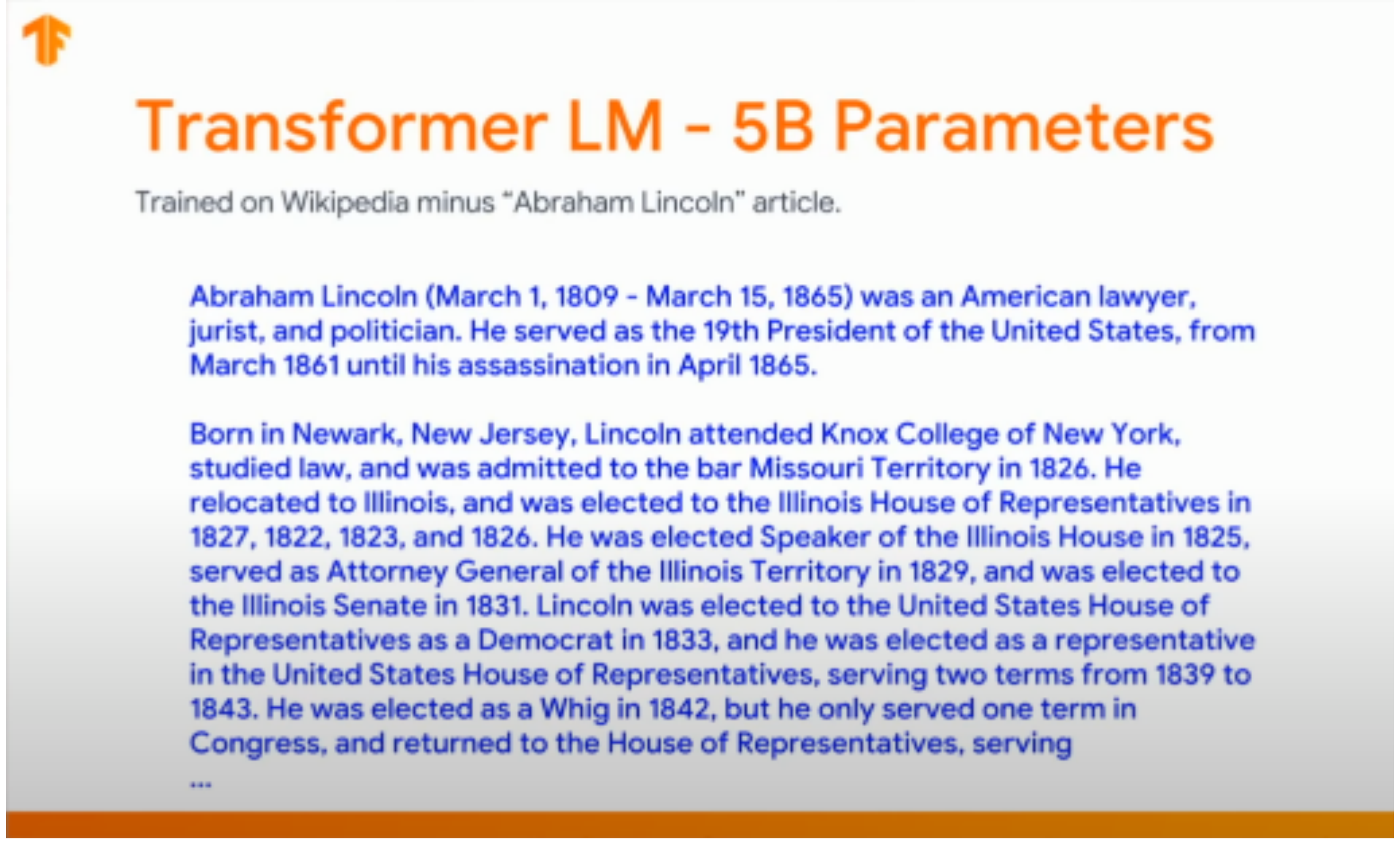
Parameters	Layers	d_{model}
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600

Table 2. Architecture hyperparameters for the 4 model sizes.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI Blog, 1(8), 9.

GPT-2
The Impact





Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI Blog, 1(8), 9.

BUT BIGGER IS BETTER

ROBERTA

Robustly Optimized BERT Pretraining Approach

Simplification of the core idea:

- training the model longer, with bigger batches, over more data
- removing the next sentence prediction objective
- training on longer sequences
- dynamically changing the masking pattern applied to the training data

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

ROBERTA

Increasing the dataset size

16GB -> 160GB

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

ROBERTA

Results

Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
Our reimplementation (with NSP loss):				
SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0
Our reimplementation (without NSP loss):				
FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6
$\text{BERT}_{\text{BASE}}$	88.5/76.3	84.3	92.8	64.3
$\text{XLNet}_{\text{BASE}}$ (K = 7)	–/81.3	85.8	92.7	66.1
$\text{XLNet}_{\text{BASE}}$ (K = 6)	–/81.0	85.6	93.4	66.7

Table 2: Development set results for base models pretrained over BOOKCORPUS and WIKIPEDIA. All models are trained for 1M steps with a batch size of 256 sequences. We report F1 for SQuAD and accuracy for MNLI-m, SST-2 and RACE. Reported results are medians over five random initializations (seeds). Results for $\text{BERT}_{\text{BASE}}$ and $\text{XLNet}_{\text{BASE}}$ are from Yang et al. (2019).

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7
$\text{XLNet}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	94.0/87.8	88.4	94.4
+ additional data	126GB	2K	500K	94.5/88.8	89.8	95.6

ble 4: Development set results for RoBERTa as we pretrain over more data (16GB → 160GB of text) and pretrain longer (100K → 300K → 500K steps). Each row accumulates improvements from the rows above. RoBERTa tches the architecture and training objective of $\text{BERT}_{\text{LARGE}}$. Results for $\text{BERT}_{\text{LARGE}}$ and $\text{XLNet}_{\text{LARGE}}$ are from vlin et al. (2019) and Yang et al. (2019), respectively. Complete results on all GLUE tasks can be found in the pendix.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

ROBERTA

Results

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
$BERT_{LARGE}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$XLNet_{LARGE}$	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	**90.2/90.2**	**94.7**	**92.2**	**86.6**	**96.4**	**90.9**	**68.0**	**92.4**	**91.3**	-
Ensembles on test (from leaderboard as of July 25, 2019)										
ALICE	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	**96.8**	**93.0**	67.8	91.6	**90.4**	88.4
RoBERTa	**90.8/90.2**	**98.9**	90.2	**88.2**	96.7	92.3	67.8	**92.2**	89.0	**88.5**

Table 5: Results on GLUE. All results are based on a 24-layer architecture. $BERT_{LARGE}$ and $XLNet_{LARGE}$ results are from Devlin et al. (2019) and Yang et al. (2019), respectively. RoBERTa results on the development set are a median over five runs. RoBERTa results on the test set are ensembles of *single-task* models. For RTE, STS and MRPC we finetune starting from the MNLI model instead of the baseline pretrained model. Averages are obtained from the GLUE leaderboard.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

ROBERTA

Additional observations

“We note that even our longest-trained model does not appear to overfit our data and would likely benefit from additional training.“

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

WE NEED EVEN LARGER MODELS!

TRANSFORMER EXTRA LONG (XL)

Challenges with the Transformer architecture

- The challenge:
  - Fixed-length contexts not respecting semantic boundaries
  - Inability to learn longer dependencie
  - Relatively slow to execute
- The solution (Transformer XL):
  - Segment-level recurrence mechanism
  - Positional encoding scheme
- The results:
  - Learns 80% longer dependencies than RNNs and 450% longer than Transformer
  - Up to 1800 times faster than vanilla Transformer

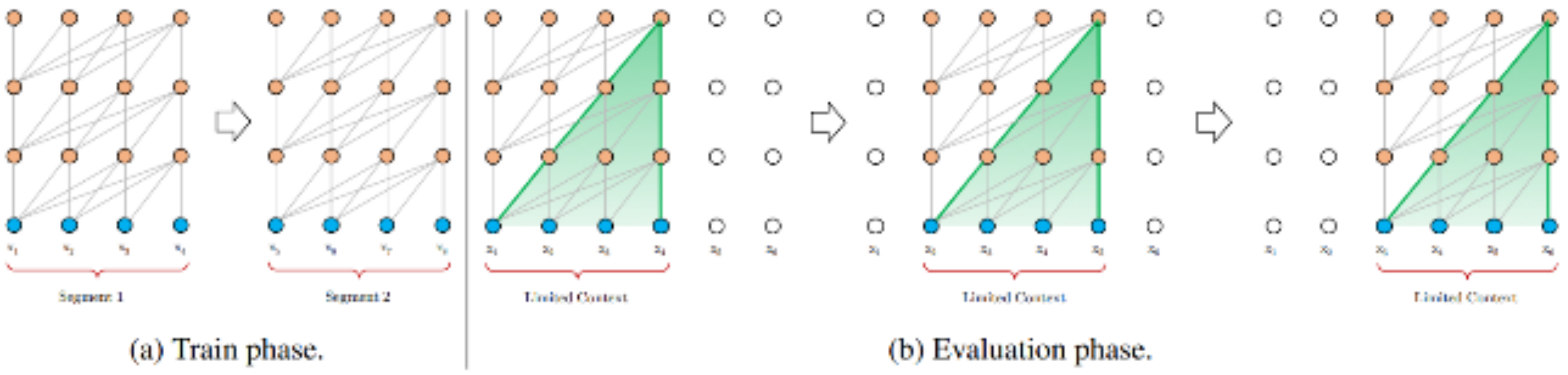


Figure 1: Illustration of the vanilla model with a segment length 4.

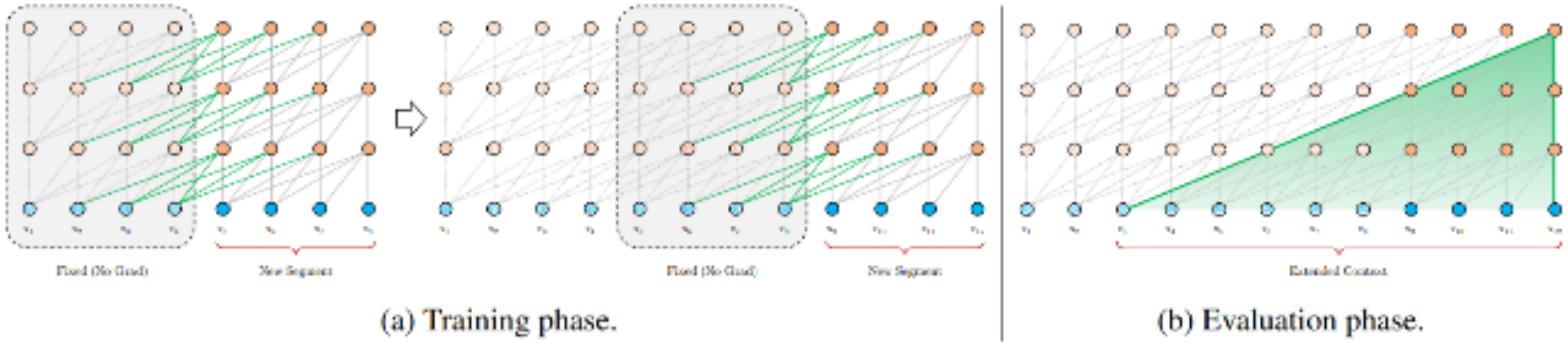


Figure 2: Illustration of the Transformer-XL model with a segment length 4.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. arXiv preprint arXiv:1901.02860.

CHALLENGES WITH BERT

Masking and independent predictions

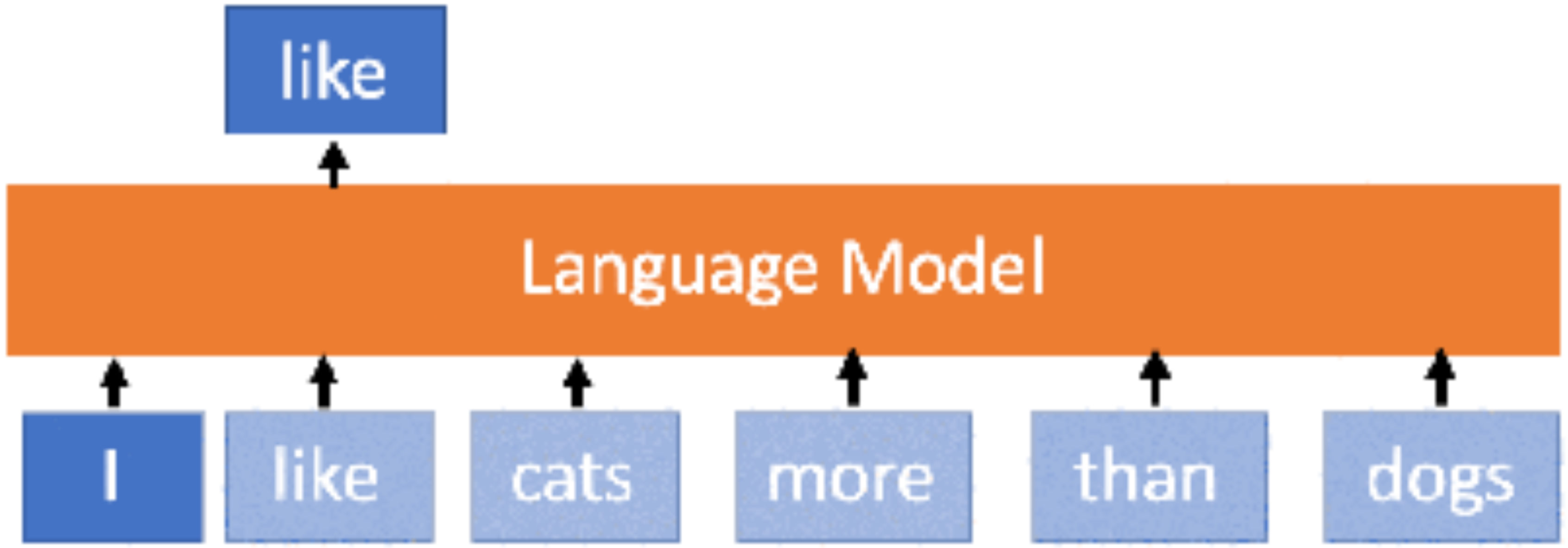
- The [MASK] token used during pretraining is not used during fine-tuning
- BERT generates predictions for individual [MASK] tokens independently, not forcing the model to learn dependencies

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems (pp. 5754-5764).

XLNET

TransformerXL + Permutational Language Model

1. Transformer -> TransformerXL
2. TransformerXL cannot be applied naively and must be adopted
3. “Maximizes the expected log likelihood of a sequence w.r.t all possible permutations of the factorization order.”
4. Does not rely on data corruption ([MASK])



Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems (pp. 5754-5764).
https://mlexplained.com/2019/06/30/paper-dissected-xlnet-generalized-autoregressive-pretraining-for-language-understanding-explained/

XLNET

And more data

13GB* -> 13GB + 19GB + 110GB = 142GB

* Different pre-processing routine is used hence not 16GB as per ROBERTA

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems (pp. 5754-5764).

XLNET

“Fair” comparison with BERT

Model	SQuAD1.1	SQuAD2.0	RACE	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B
BERT-Large (Best of 3)	86.7/92.8	82.8/85.5	75.1	87.3	93.0	91.4	74.0	94.0	88.7	63.7	90.2
XLNet-Large-wikibooks	88.2/94.0	85.1/87.8	77.4	88.4	93.9	91.8	81.2	94.4	90.0	65.2	91.1

Table 1: Fair comparison with BERT. All models are trained using the same data and hyperparameters as in BERT. We use the best of 3 BERT variants for comparison; i.e., the original BERT, BERT with whole word masking, and BERT without next sentence prediction.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems (pp. 5754-5764).

XLNET

Ablation study

#	Model	RACE	SQuAD2.0 F1	SQuAD2.0 EM	MNLI m/mm	SST-2
1	BERT-Base	64.3	76.30	73.66	84.34/84.65	92.78
2	DAE + Transformer-XL	65.03	79.56	76.80	84.88/84.45	92.60
3	XLNet-Base ($K = 7$)	66.05	**81.33**	**78.46**	**85.84/85.43**	92.66
4	XLNet-Base ($K = 6$)	66.66	80.98	78.18	85.63/85.12	**93.35**
5	- memory	65.55	80.15	77.27	85.32/85.05	92.78
6	- span-based pred	65.95	80.61	77.91	85.49/85.02	93.12
7	- bidirectional data	66.34	80.65	77.87	85.31/84.99	92.66
8	+ next-sent pred	**66.76**	79.83	76.94	85.32/85.09	92.89

Table 6: The results of BERT on RACE are taken from [38]. We run BERT on the other datasets using the official implementation and the same hyperparameter search space as XLNet. K is a hyperparameter to control the optimization difficulty (see Section 2.3).

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for understanding. In Advances in neural information processing systems (pp. 5754-5764). language

XLNET

Scaling up

RACE	Accuracy	Middle	High	Model	NDCG@20	ERR@20
GPT [28]	59.0	62.9	57.4	DRMM [13]	24.3	13.8
BERT [25]	72.0	76.6	70.1	KNRM [8]	26.9	14.9
BERT+DCMN* [38]	74.1	79.5	71.8	Conv [8]	28.7	18.1
RoBERTa [21]	83.2	86.5	81.8	BERT†	30.53	18.67
XLNet	**85.4**	**88.6**	**84.0**	XLNet	**31.10**	**20.28**

Table 2: Comparison with state-of-the-art results on the test set of RACE, a reading comprehension task, and on ClueWeb09-B, a document ranking task. ∗ indicates using ensembles. † indicates our implementations. “Middle” and “High” in RACE are two subsets representing middle and high school difficulty levels. All BERT, RoBERTa, and XLNet results are obtained with a 24-layer architecture with similar model sizes (aka BERT-Large).

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems (pp. 5754-5764).

SCALING UP?

XLNET

Scaling up

"… we scale up the training of XLNet-Large by using all the datasets described above. Specifically, we train on 512 TPU v3 chips for 500K steps with an Adam weight decay optimizer, linear learning rate decay, and a batch size of 8192, which takes about 5.5 days."

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems (pp. 5754-5764).

XLNET

Scaling up

"It was observed that the model still underfits the data at the end of training."

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems (pp. 5754-5764).

SCALING UP?

BERT

5.5 days -> 76 minutes

- Inspired by NVIDIA LARS (Layer-wise Adaptive Rate Scaling) they develop LAMB
- This allows to scale batch size to 32k without degrading performance
- A lot of improvements introduced since. Please use NVLAMB.

Solver	batch size	steps	F1 score on dev set	TPUs	Time
Baseline	512	1000k	90.395	16	81.4h
LAMB	512	1000k	91.752	16	82.8h
LAMB	1k	500k	91.761	32	43.2h
LAMB	2k	250k	91.946	64	21.4h
LAMB	4k	125k	91.137	128	693.6m
LAMB	8k	62500	91.263	256	390.5m
LAMB	16k	31250	91.345	512	200.0m
LAMB	32k	15625	91.475	1024	101.2m
LAMB	64k/32k	8599	90.584	1024	76.19m

https://devblogs.nvidia.com/pretraining-bert-with-layer-wise-adaptive-learning-rates/

You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., ... & Hsieh, C. J. (2019, September). Large batch optimization for deep learning: Training bert in 76 minutes. In International Conference on Learning Representations

BERT

5.5 days -> 76 minutes

- Inspired by NVIDIA LARS (Layer-wise Adaptive Rate Scaling) they develop LAMB
- This allows to scale batch size to 32k without degrading performance
- A lot of improvements introduced since. Please use NVLAMB.

NVLAMB

1. For every training mini-batch x and training step t, compute gradient $g_l^i(t)$ on weights $w_l^i(t)$, for each weight i in layer l.

2. Normalize gradients by L2 norm of gradient of the entire model.

$$\widehat{g}_l^i(t) = g_l^i(t) \;/\; \| \; g(t) \; \|_2$$

3. Update velocity $v(t)$ and momentum $m(t)$ values corresponding to each layer weight $w_l^i(t)$ based on gradients $g(t)$ with hyperparameters β_1 and β_2.

$$m_l^i(t) = \beta_1 m_l^i(t-1) + (1-\beta_1)\widehat{g}_l^i(t) \qquad (1)$$

$$v_l^i(t) = \beta_2 v_l^i(t-1) + (1-\beta_2)(\widehat{g}_l^i(t))^2 \qquad (2)$$

4. Apply beta-correction on velocity and momentum values to obtain unbiased estimates.

$$\widehat{m}_l^i(t) = \frac{m_l^i(t)}{1-\beta_1^t} \qquad (3)$$

$$\widehat{v}_l^i(t) = \frac{v_l^i(t)}{1-\beta_2^t} \qquad (4)$$

5. Compute update $u_l^i(t)$ on weight $w_l^i(t)$ with weight decay parameter γ and ϵ as follows:

$$u_l^i(t) = \frac{\widehat{m}_l^i(t)}{\sqrt{\widehat{v}_l^i(t) + \epsilon}} + \gamma w_l^i(t)$$

6. For each layer l, compute the ratio $r_l(t)$ of norm of weights $w_l(t)$ and norm of update $u_l(t)$ as follows:

$$r_l(t) = \frac{\| \; w_l(t) \; \|_2}{\| \; u_l(t) \; \|_2}$$

7. Update the weights with learning rate λ:

$$w_l^i(t+1) = w_l^i(t) - \lambda * r_l(t) * u_l^i(t)$$

https://devblogs.nvidia.com/pretraining-bert-with-layer-wise-adaptive-learning-rates/
You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., ... & Hsieh, C. J. (2019, September). Large batch optimization for deep learning: Training bert in 76 minutes. In International Conference on Learning Representations

BERT

Fastest training time

BERT-Large Training Times on GPUs

Time	System	Number of Nodes	Number of V100 GPUs
47 min	DGX SuperPOD	92 x DGX-2H	1,472
67 min	DGX SuperPOD	64 x DGX-2H	1,024
236 min	DGX SuperPOD	16 x DGX-2H	256

https://devblogs.nvidia.com/training-bert-with-gpus/

CAN WE USE PARAMETERS MORE EFFICIENTLY?

ALBERT

A Lite BERT for Self-Supervised Learning of Language Representations

- The size of the model is becoming a challenge
- FP16 is addressing the problem to some extent but still the footprint is considerable
- Describes a set of methods for reducing the memory footprint/ improving parameter efficiency

FP32 TF 1.13.1 16GB GPU

System	Seq Length	Max Batch Size
XLNet-Base	64	120
...	128	56
...	256	24
...	512	8
XLNet-Large	64	16
...	128	8
...	256	2
...	512	1

FP32 TF 1.11.0 12GB GPU

System	Seq Length	Max Batch Size
BERT-Base	64	64
...	128	32
...	256	16
...	320	14
...	384	12
...	512	6
BERT-Large	64	12
...	128	6
...	256	2
...	320	1
...	384	0
...	512	0

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942.

ALBERT

Model size is the key to success

Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

Table 6: Ablation over BERT model size. #L = the number of layers; #H = hidden size; #A = number of attention heads. "LM (ppl)" is the masked LM perplexity of held-out training data.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942.

ALBERT

Factorized Embeddings

- "… WordPiece embedding size E is tied with the hidden layer size H, i.e., E ≡ H"
- "… hidden-layer embeddings are meant to learn context-dependent representations." so we want H >> E
- Embedding matrix size is V x E (vocabulary size time embedding size)
- "… natural language processing usually requires the vocabulary size V to be large." (BERT V=30000)
- So we end up with LargeNumber x LargeNumber

- Factorization of the embeddings matrix:

  O(V x H) transformed into O(V x E + E x H)

Model		Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768	False
	large	334M	24	1024	1024	False
ALBERT	base	12M	12	768	128	True
	large	18M	24	1024	128	True
	xlarge	60M	24	2048	128	True
	xxlarge	235M	12	4096	128	True

Table 1: The configurations of the main BERT and ALBERT models analyzed in this paper.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942.

ALBERT

Cross Layer Parameter Sharing and Inter-Sentence Coherence Loss

- Proposes several cross-layer parameter-sharing schemes
- The default Albert configuration shares all parameters across all layers
- SOP Loss (Sentence Order Prediction) rather than NSP Loss (Next Sentence Prediction)

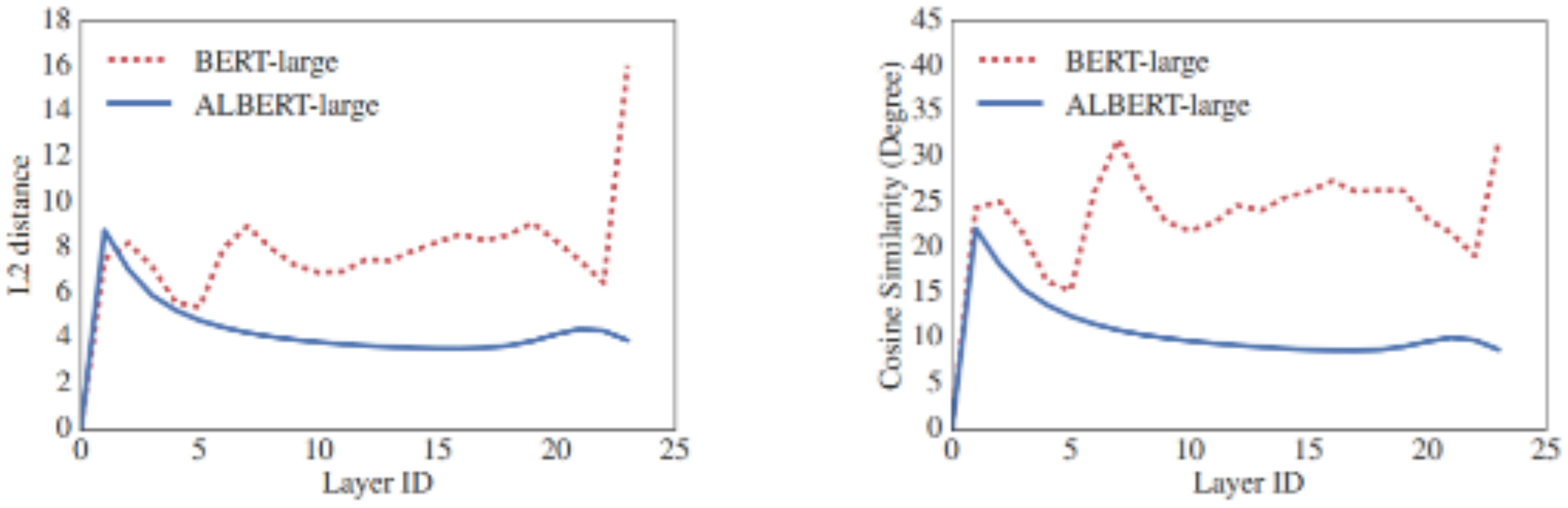


Figure 1: The L2 distances and cosine similarity (in terms of degree) of the input and output embedding of each layer for BERT-large and ALBERT-large.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942.

ALBERT

Results

Model		Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768	False
	large	334M	24	1024	1024	False
ALBERT	base	12M	12	768	128	True
	large	18M	24	1024	128	True
	xlarge	60M	24	2048	128	True
	xxlarge	235M	12	4096	128	True

Table 1: The configurations of the main BERT and ALBERT models analyzed in this paper.

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3	4.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	5.6x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	1.7x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	0.6x
	xxlarge	235M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	0.3x

Table 2: Dev set results for models pretrained over BookCorpus and Wikipedia for 125k steps. Here and everywhere else, the Avg column is computed by averaging the scores of the downstream tasks to its left (the two numbers of F1 and EM for each SQuAD are first averaged).

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942.

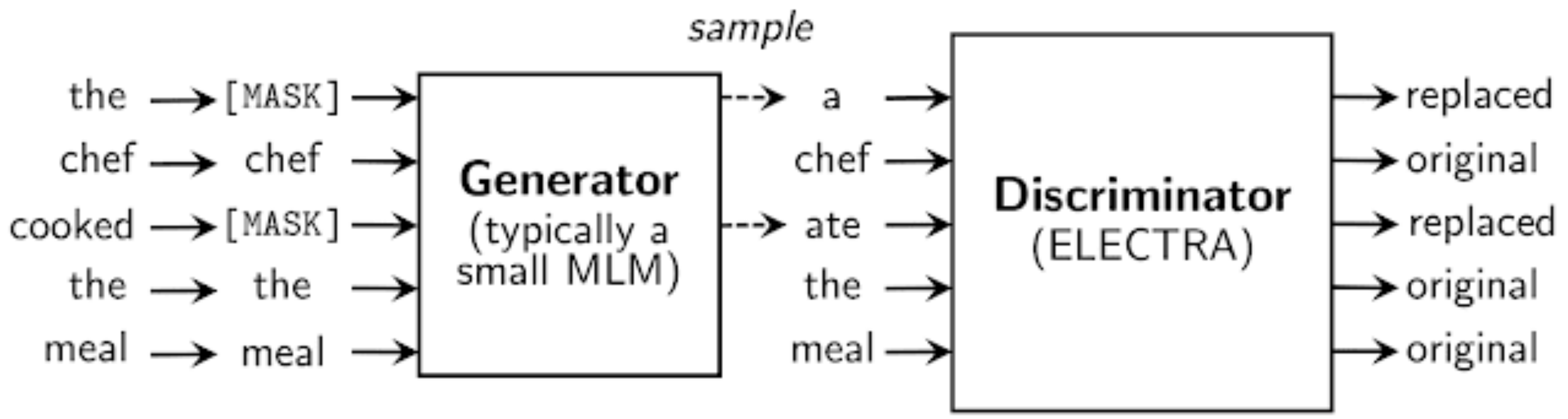
CAN WE IMPROVE THE OBJECTIVE FUNCTION FURTHER?

ELECTRA

Pre-training Text Encoders as Discriminators Rather Than Generators

Replaced Token Detection

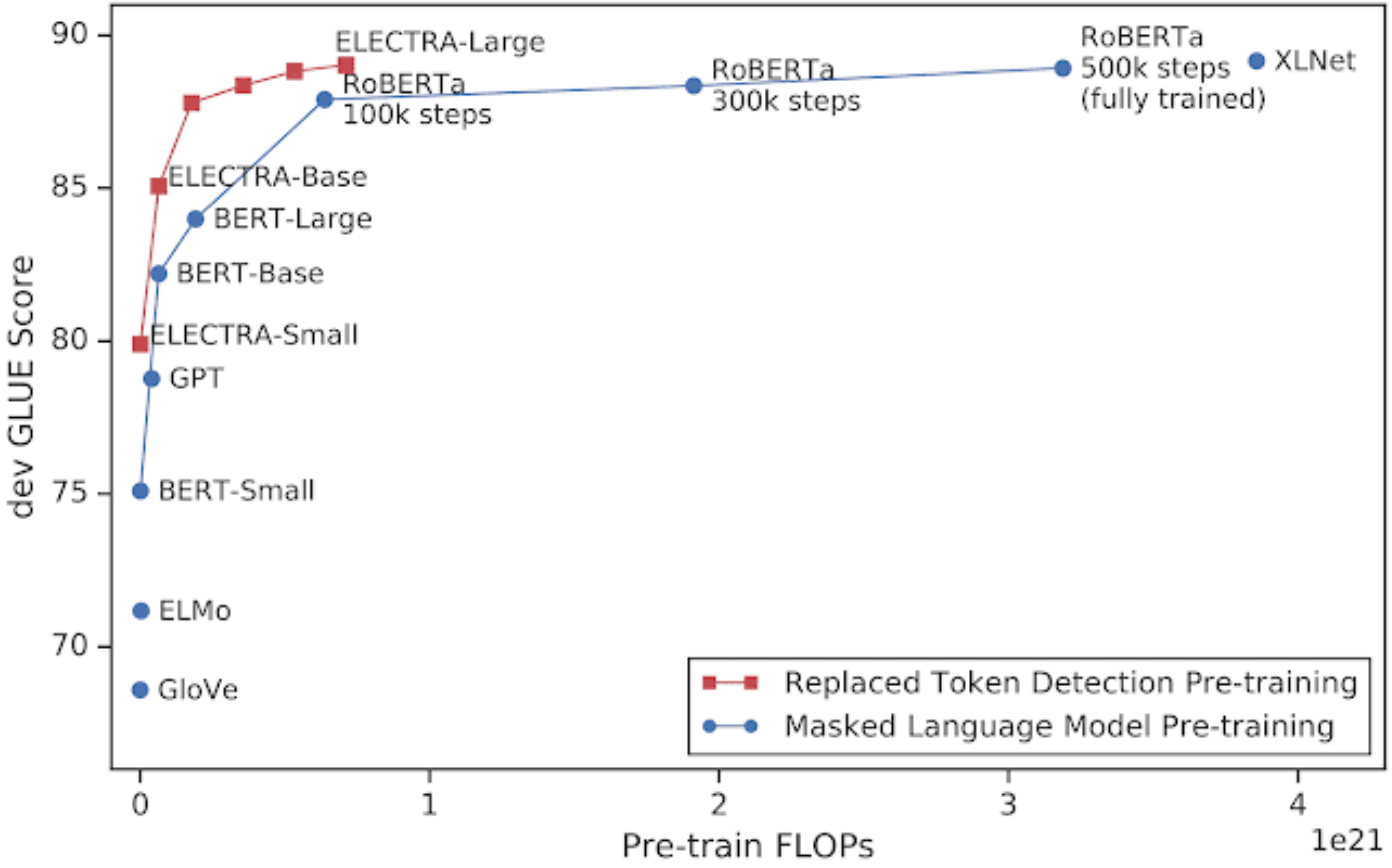
the chef cooked the meal



Clark, K., Luong, M. T., Le, Q. V., & Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. arXiv preprint arXiv:2003.10555.

ELECTRA

Pre-training Text Encoders as Discriminators Rather Than Generators



Clark, K., Luong, M. T., Le, Q. V., & Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. arXiv preprint arXiv:2003.10555.

MULTI-TASK LEARNING

ERNIE 2.0

Why use only a limited number of simple pretraining tasks?

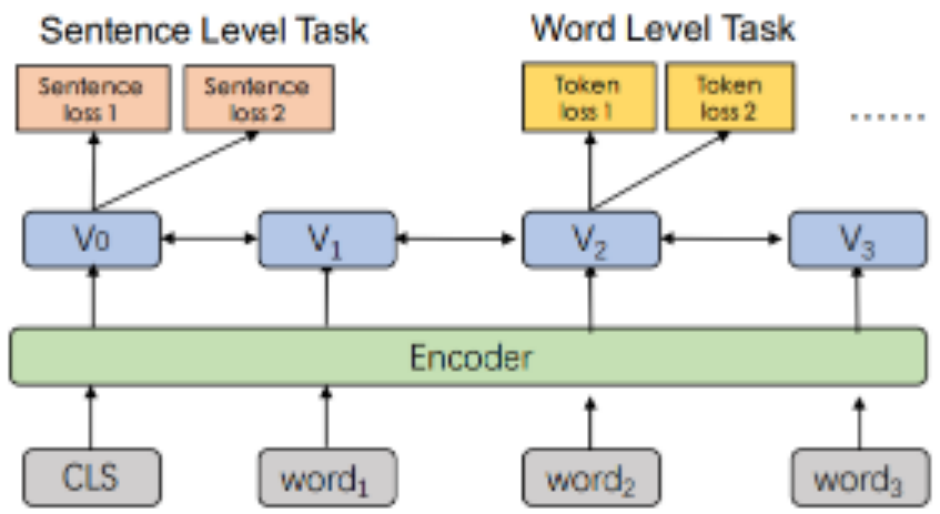


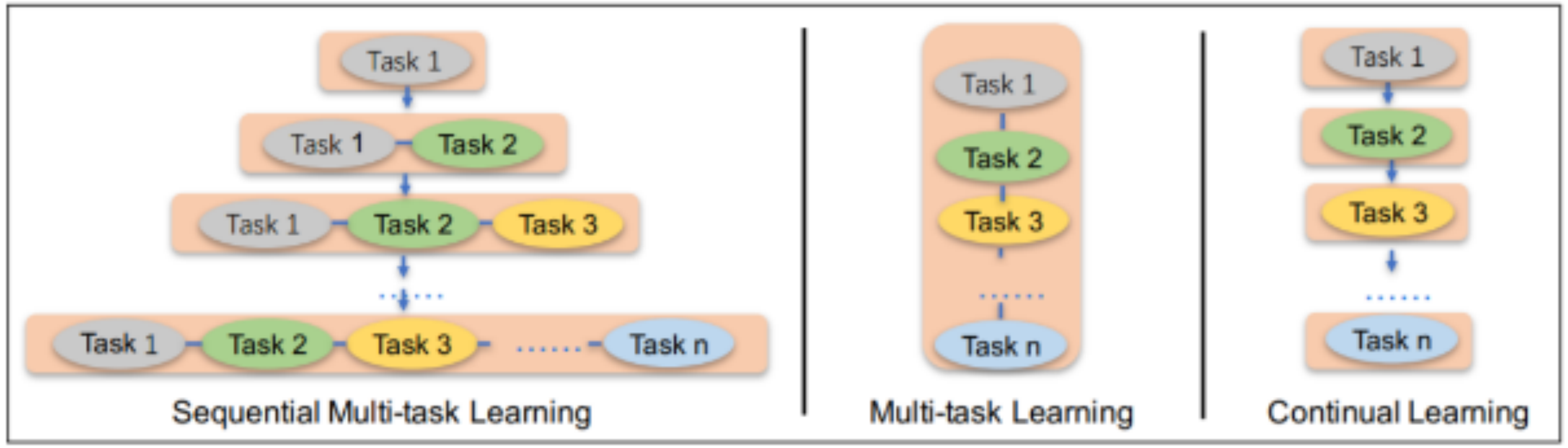
Figure 4: The architecture of multi-task learning in the ERNIE 2.0 framework, in which the encoder can be recurrent neural networks or a deep transformer.

Task / Corpus	Token-Level Loss			Sentence-Level Loss			
	Knowledge Masking	Capital Prediction	Token-Document Relation	Sentence Reordering	Sentence Distance	Discourse Relation	IR Relevance
Encyclopedia	✓	✓	✓	✓	✓	×	×
BookCorpus	✓	✓	✓	✓	✓	×	×
News	✓	✓	✓	✓	✓	×	×
Dialog	✓	✓	✓	✓	✓	×	×
IR Relevance Data	×	×	×	×	×	×	✓
Discourse Relation Data	×	×	×	×	×	✓	×

Table 1: The Relationship between pre-training task and pre-training dataset. We use different pre-training dataset to construct different tasks. A type of pre-trained dataset can correspond to multiple pre-training tasks.

ERNIE 2.0

Why use only a limited number of simple pretraining tasks?



Pre-training method	Pre-training task	Training iterations (steps)				Fine-tuning result		
		Stage 1	Stage 2	Stage 3	Stage 4	MNLI	SST-2	MRPC
Continual Learning	Knowledge Masking	50k	-	-	-	77.3	86.4	82.5
	Capital Prediction	-	50k	-	-			
	Token-Document Relation	-	-	50k	-			
	Sentence Reordering	-	-	-	50k			
Multi-task Learning	Knowledge Masking	50k				78.7	87.5	83.0
	Capital Prediction	50k						
	Token-Document Relation	50k						
	Sentence Reordering	50k						
continual Multi-task Learning	Knowledge Masking	20k	10k	10k	10k	**79.0**	**87.8**	**84.0**
	Capital Prediction	-	30k	10k	10k			
	Token-Document Relation	-	-	40k	10k			
	Sentence Reordering	-	-	-	50k			

DEEP LEARNING INSTITUTE

ERNIE 2.0

Performance

Task(Metrics)	*BASE model*		*LARGE model*				
	Test		Dev			Test	
	BERT	ERNIE 2.0	BERT	XLNet	ERNIE 2.0	BERT	ERNIE 2.0
CoLA (Matthew Corr.)	52.1	**55.2**	60.6	63.6	**65.4**	60.5	**63.5**
SST-2 (Accuracy)	93.5	**95.0**	93.2	95.6	**96.0**	94.9	**95.6**
MRPC (Accurary/F1)	84.8/88.9	**86.1/89.9**	88.0/-	89.2/-	**89.7/-**	85.4/89.3	**87.4/90.2**
STS-B (Pearson Corr./Spearman Corr.)	87.1/85.8	**87.6/86.5**	90.0/-	91.8/-	**92.3/-**	87.6/86.5	**91.2/90.6**
QQP (Accuracy/F1)	89.2/71.2	**89.8/73.2**	91.3/-	91.8/-	**92.5/-**	89.3/72.1	**90.1/73.8**
MNLI-m/mm (Accuracy)	84.6/83.4	**86.1/85.5**	86.6/-	**89.8/-**	89.1/-	86.7/85.9	**88.7/88.8**
QNLI (Accuracy)	90.5	**92.9**	92.3	93.9	**94.3**	92.7	**94.6**
RTE (Accuracy)	66.4	**74.8**	70.4	83.8	**85.2**	70.1	**80.2**
WNLI (Accuracy)	**65.1**	**65.1**	-	-	-	65.1	**67.8**
AX(Matthew Corr.)	34.2	**37.4**	-	-	-	39.6	**48.0**
Score	78.3	**80.6**	-	-	-	80.5	**83.6**

Table 6: The results on GLUE benchmark, where the results on dev set are the median of five experimental results and the results on test set are scored by the GLUE evaluation server (`https://gluebenchmark.com/leaderboard`). The state-of-the-art results are in bold. All of the fine-tuned models of AX is trained by the data of MNLI.

Part 2: Self-Supervision, BERT and Beyond

- Lecture
  - Why Do DNNs Work Well?
  - Self-Supervised Learning
  - BERT
- Lab
  - Explore the Data
  - Explore NeMo
  - Text Classifier Project
- Lecture (cont'd)
  - Bigger is Better
  - Can and should we go even bigger?
- Lab (cont'd)
  - Named Entity Recognizer

GOING BIGGER

The challenge

- If we only consider Parameters, Gradients, and Optimizer states and ignore activations
- If we use FP16 data representation (so two bytes)
- If we use Adam as an optimizer (storing twelve bytes per parameter in mixed precision mode)
- If we consider a model with <u>one billion </u>parameters

10^9 * (2B + 2B + 12B) = 10^9*16B = 14.90GB

1 billion parameters

2 bytes per parameter

2 bytes per gradient

12 bytes per optimizer state

GOING BIGGER

The challenge

- What about activations?
- What about 2 or 3 billion parameter models?

10^9 * (2B + 2B + 12B) = 10^9*16B = 14.90GB

1 billion parameters

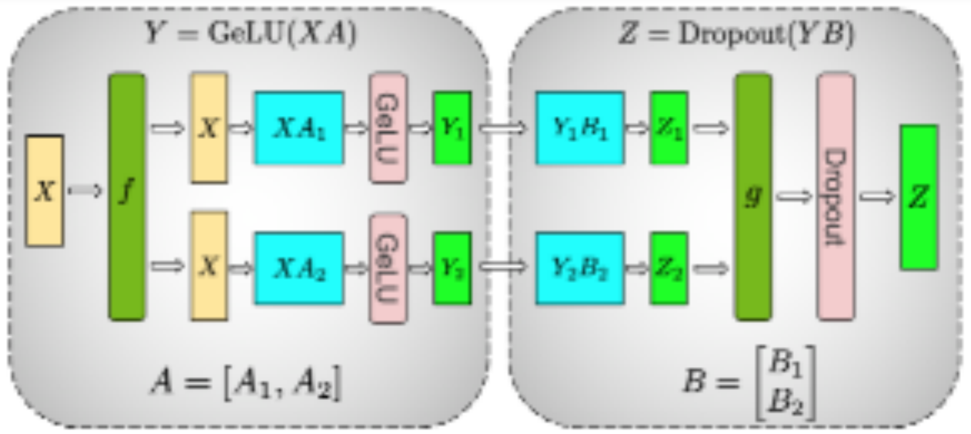
2 bytes per gradient

12 bytes per optimizer state

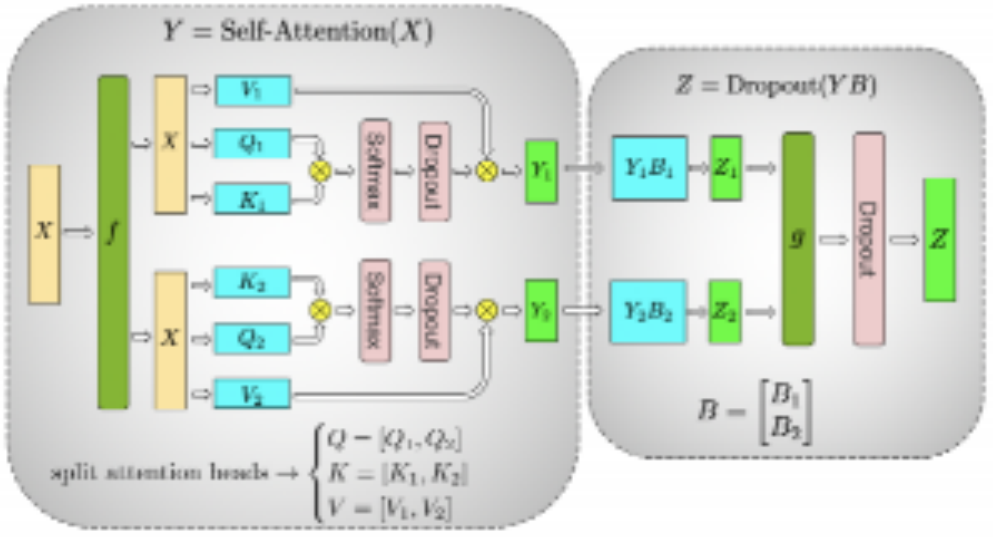
2 bytes per parameter

MEGATRON

Model Parallel Transformer



(a) MLP



(b) Self-Attention

Figure 3. Blocks of Transformer with Model Parallelism. f and g are conjugate. f is an identity operator in the forward pass and all reduce in the backward pass while g is an all reduce in the forward pass and identity in the backward pass.

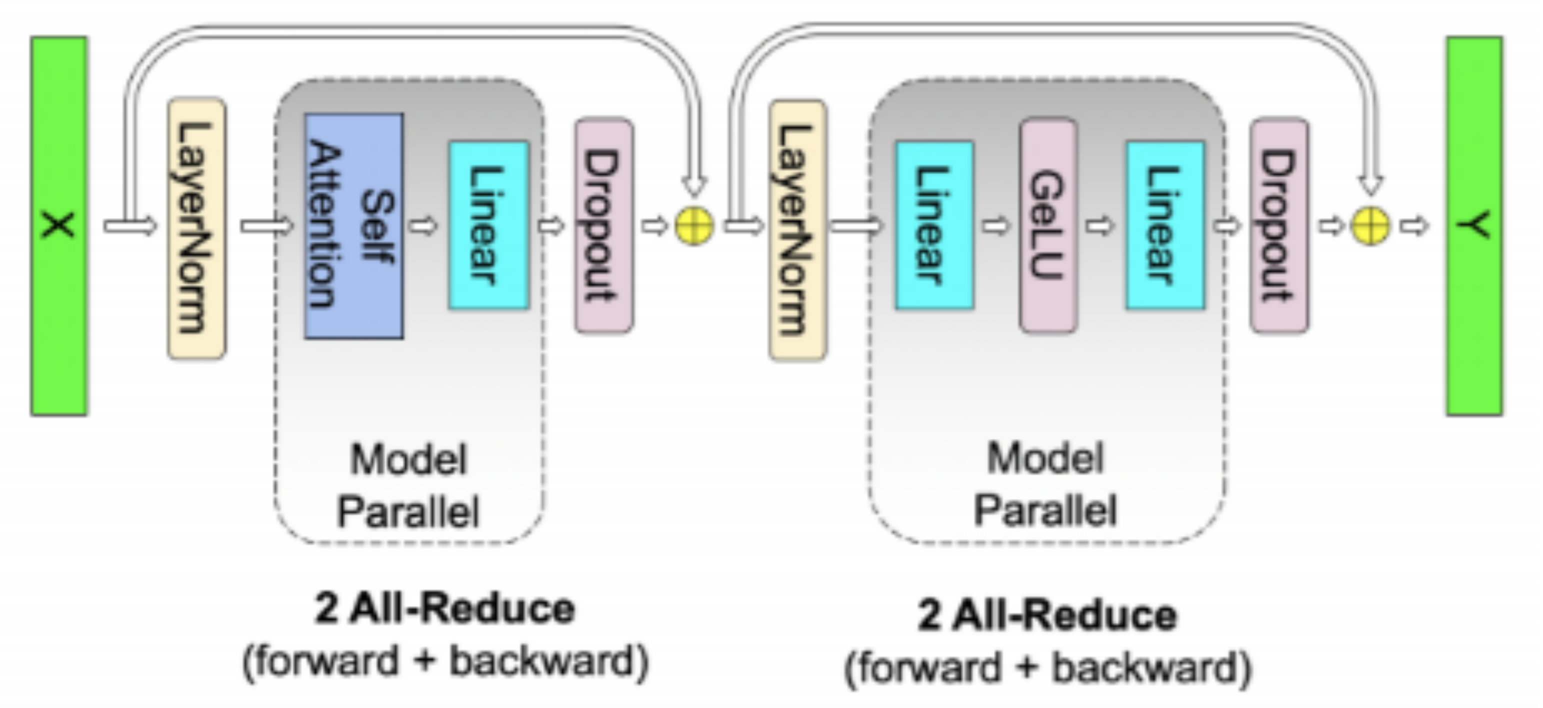


Figure 4. Communication operations in a transformer layer. There are 4 total communication operations in the forward and backward pass of a single model parallel transformer layer.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., & Catanzaro, B. (2019). Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. arXiv preprint arXiv:1909.08053.

MEGATRON

76% scaling efficiency using 512 GPUs

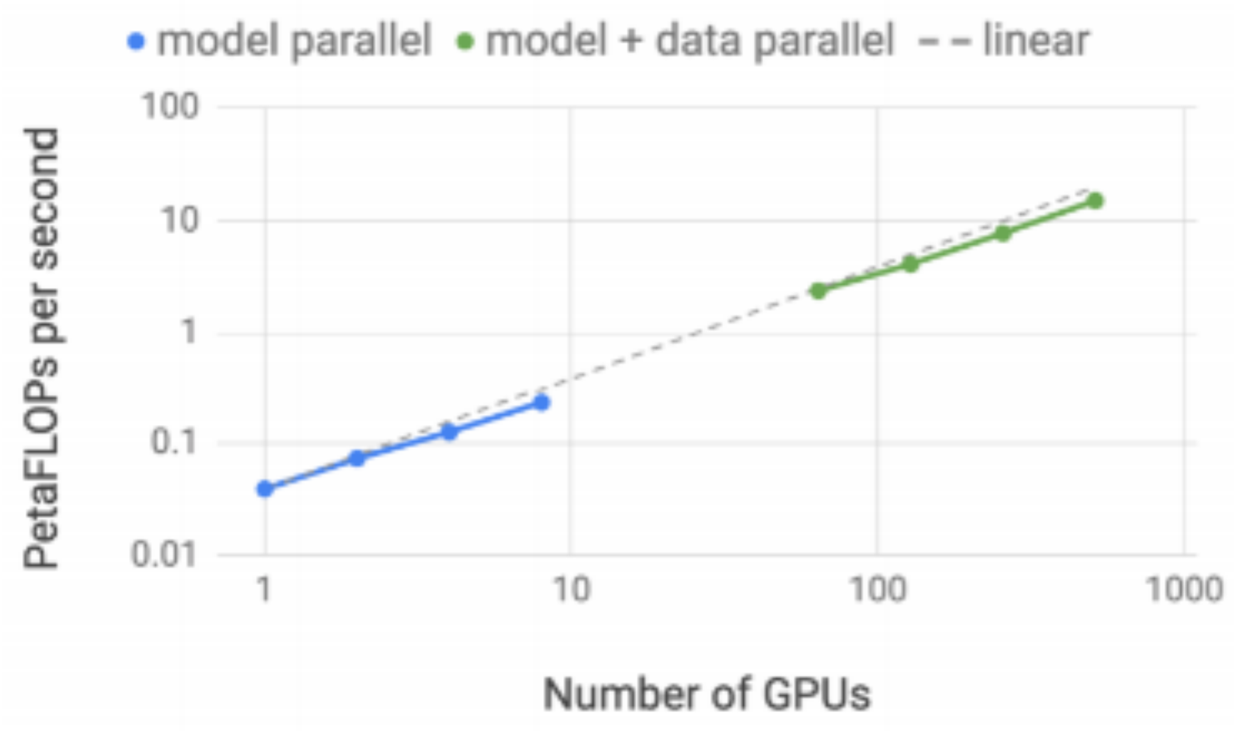


Figure 1. Model (blue) and model+data (green) parallel FLOPS as a function of number of GPUs. Model parallel (blue): up to 8-way model parallel weak scaling with approximately 1 billion parameters per GPU (e.g. 2 billion for 2 GPUs and 4 billion for 4 GPUs). Model+data parallel (green): similar configuration as model parallel combined with 64-way data parallel.

Table 1. Parameters used for scaling studies. Hidden size per attention head is kept constant at 96.

Hidden Size	Attention heads	Number of layers	Number of parameters (billions)	Model parallel GPUs	Model +data parallel GPUs
1536	16	40	1.2	1	64
1920	20	54	2.5	2	128
2304	24	64	4.2	4	256
3072	32	72	8.3	8	512

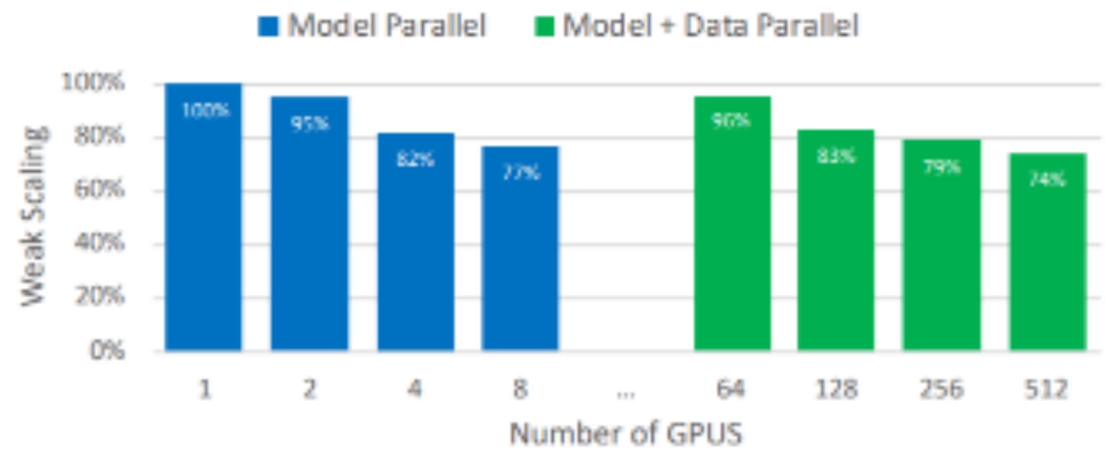


Figure 5. Model and model + data parallel weak scaling efficiency as a function of the number of GPUs.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., & Catanzaro, B. (2019). Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. arXiv preprint arXiv:1909.08053.

MEGATRON

Results

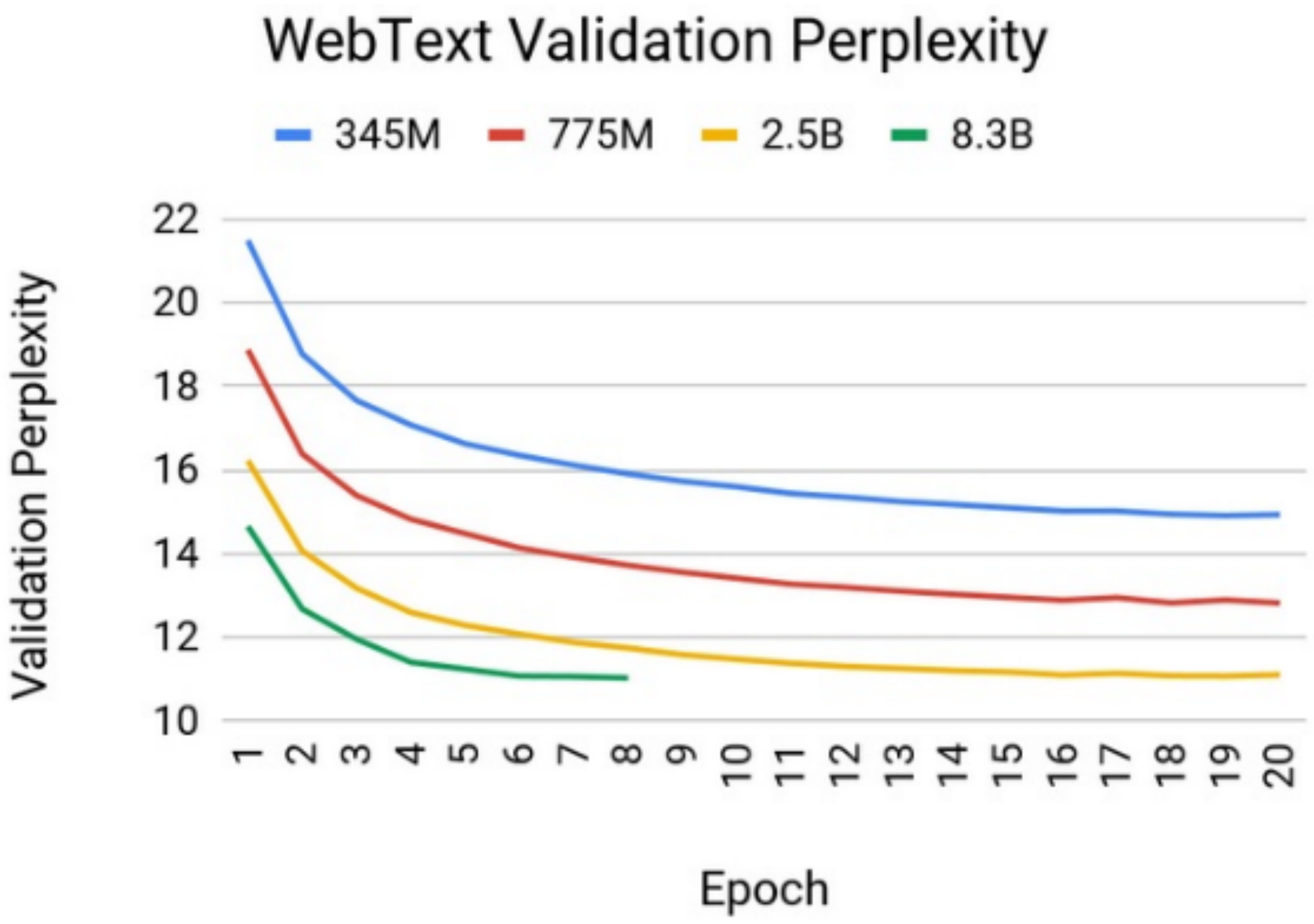
Table 5. Development set results for MNLI, QQP, SQuAD 1.1 and SQuAD 2.0 and test set results for RACE. The trained tokens represents consumed tokens during model pretraining (proportional to batch size times number of iterations) normalized by consumed tokens during model pretraining for our 336M model.

Model	trained tokens ratio	MNLI m/mm accuracy (dev set)	QQP accuracy (dev set)	SQuAD 1.1 F1 / EM (dev set)	SQuAD 2.0 F1 / EM (dev set)	RACE m/h accuracy (test set)
RoBERTa (Liu et al., 2019b)	2	90.2 / 90.2	92.2	94.6 / 88.9	89.4 / 86.5	83.2 (86.5 / 81.8)
ALBERT (Lan et al., 2019)	3	90.8	92.2	94.8 / 89.3	90.2 / 87.4	86.5 (89.0 / 85.5)
XLNet (Yang et al., 2019)	2	90.8 / 90.8	92.3	95.1 / 89.7	90.6 / 87.9	85.4 (88.6 / 84.0)
Megatron-336M	1	89.7 / 90.0	92.3	94.2 / 88.0	88.1 / 84.8	83.0 (86.9 / 81.5)
Megatron-1.3B	1	90.9 / 91.0	92.6	94.9 / 89.1	90.2 / 87.1	87.3 (90.4 / 86.1)
Megatron-3.9B	1	**91.4 / 91.4**	**92.7**	**95.5 / 90.0**	**91.2 / 88.5**	**89.5 (91.8 / 88.6)**
ALBERT ensemble (Lan et al., 2019)				95.5 / 90.1	91.4 / 88.9	89.4 (91.2 / 88.6)
Megatron-3.9B ensemble				**95.8 / 90.5**	**91.7 / 89.0**	**90.9 (93.1 / 90.0)**

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., & Catanzaro, B. (2019). Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. arXiv preprint arXiv:1909.08053.

MEGATRON

More importantly!



Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., & Catanzaro, B. (2019). Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. arXiv preprint arXiv:1909.08053.

THE SCALING LAWS

THE SCALING LAWS

As you increase the dataset size, you must increase the model size

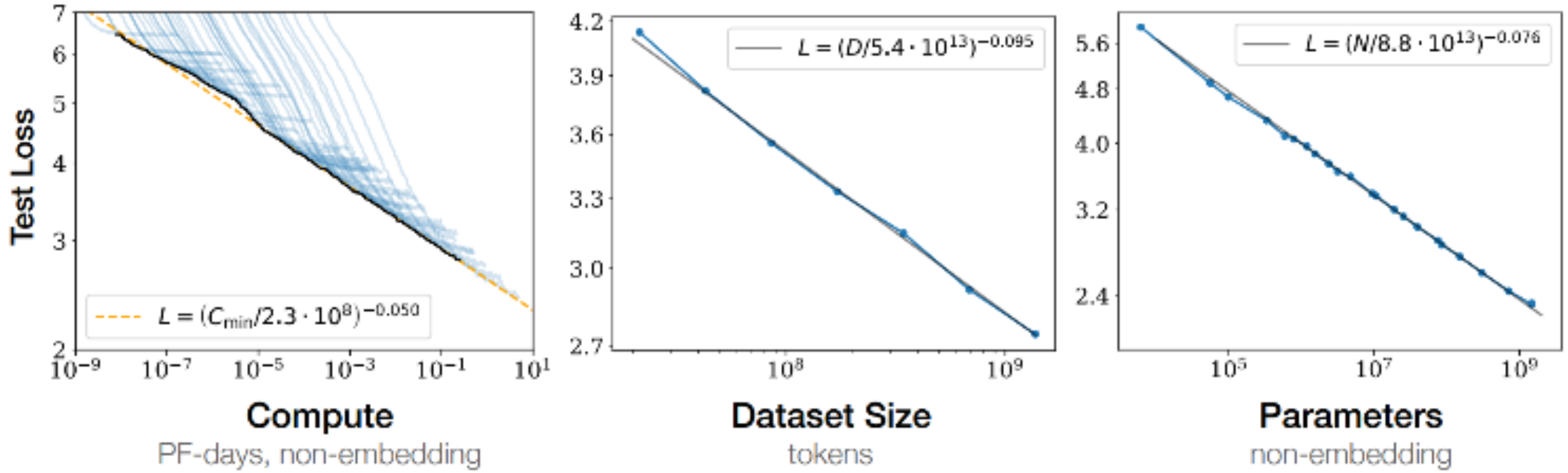


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361*.

THE SCALING LAWS

Larger models are more sample-efficient

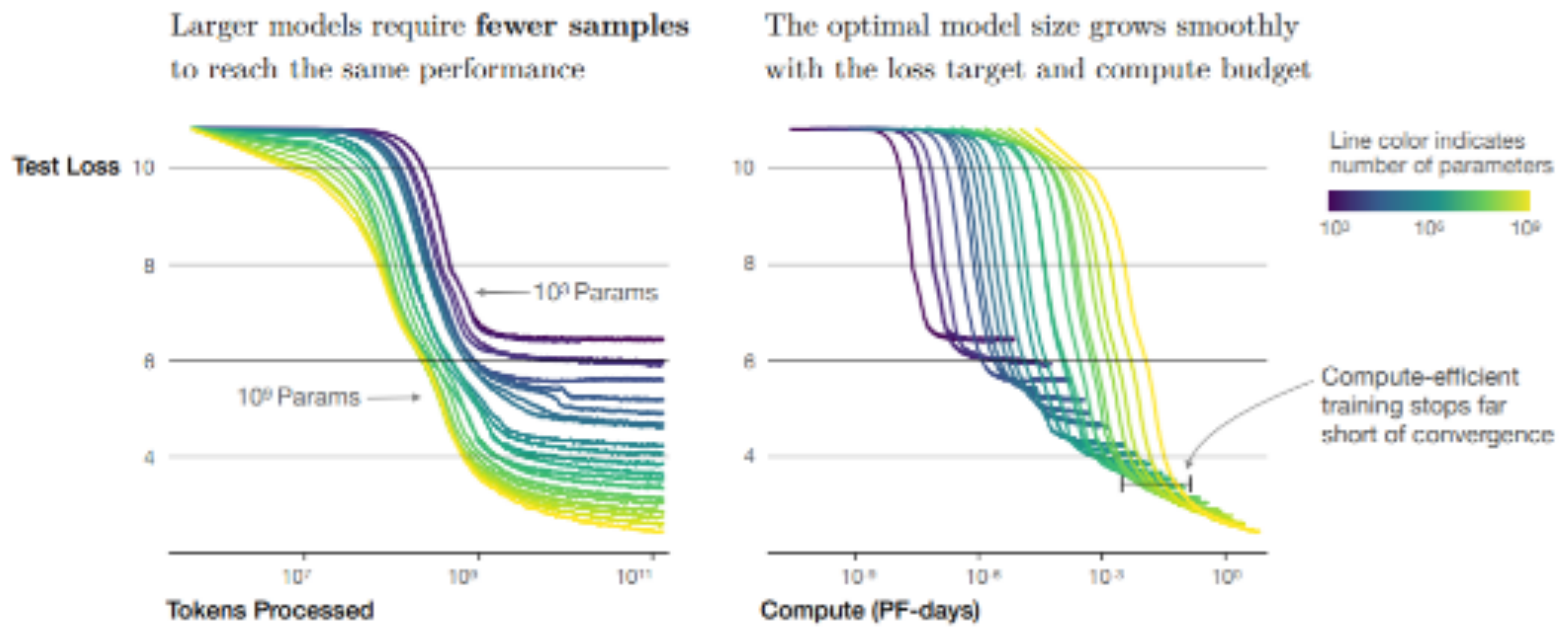


Figure 2 We show a series of language model training runs, with models ranging in size from 10^3 to 10^9 parameters (excluding embeddings).

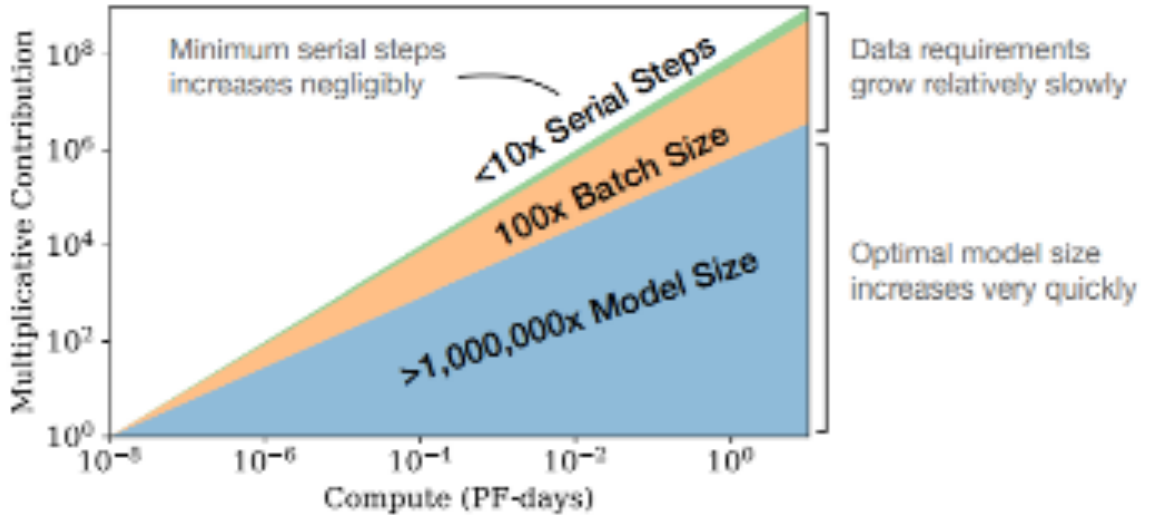


Figure 3 As more compute becomes available, we can choose how much to allocate towards training larger models, using larger batches, and training for more steps. We illustrate this for a billion-fold increase in compute. For optimally compute-efficient training, most of the increase should go towards increased model size. A relatively small increase in data is needed to avoid reuse. Of the increase in data, most can be used to increase parallelism through larger batch sizes, with only a very small increase in serial training time required.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361*.

THE SCALING LAWS

Larger models generalize better

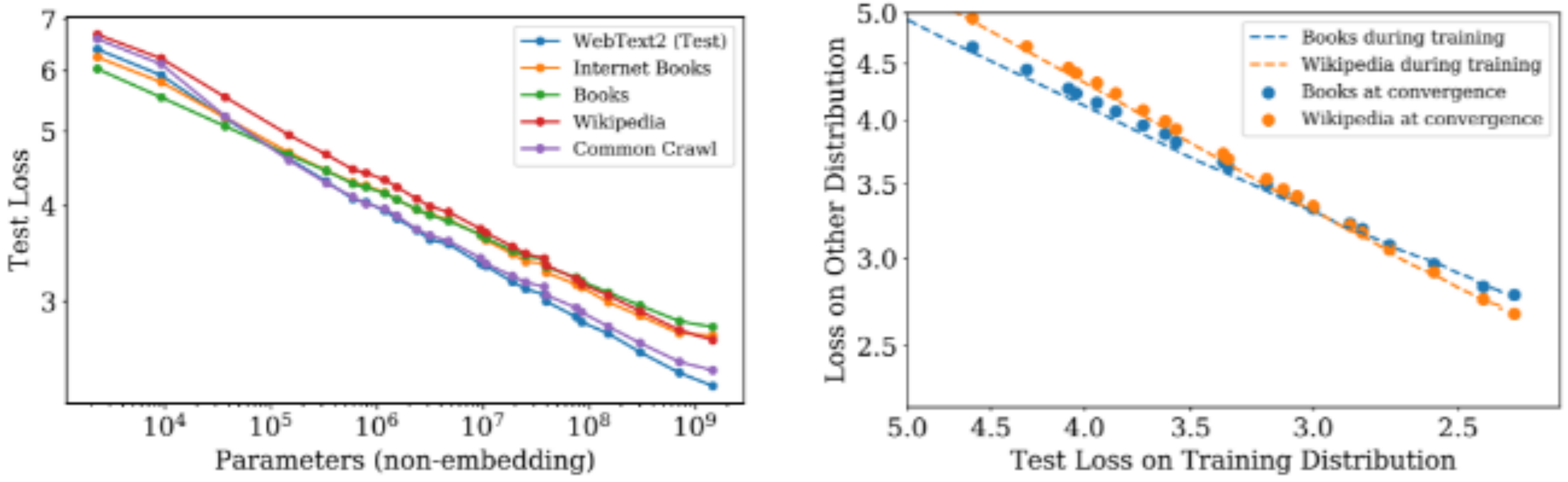


Figure 8 **Left:** Generalization performance to other data distributions improves smoothly with model size, with only a small and very slowly growing offset from the WebText2 training distribution. **Right:** Generalization performance depends only on training distribution performance, and not on the phase of training. We compare generalization of converged models (points) to that of a single large model (dashed curves) as it trains.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361*.

THE SCALING LAWS

Its cheaper to use a larger model

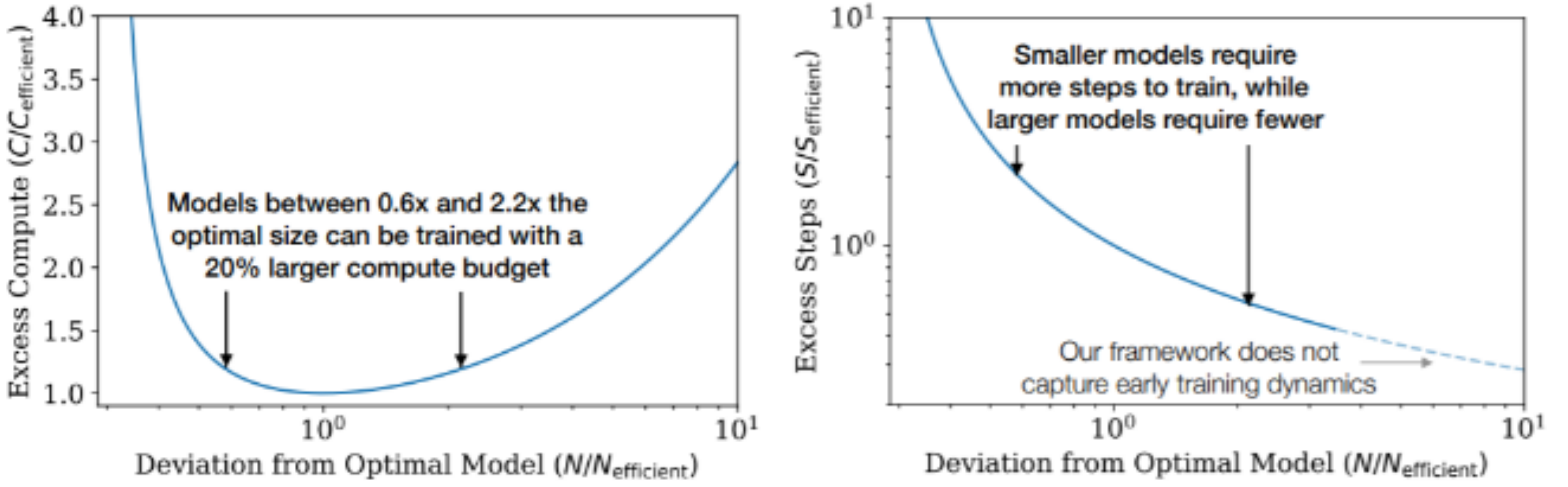
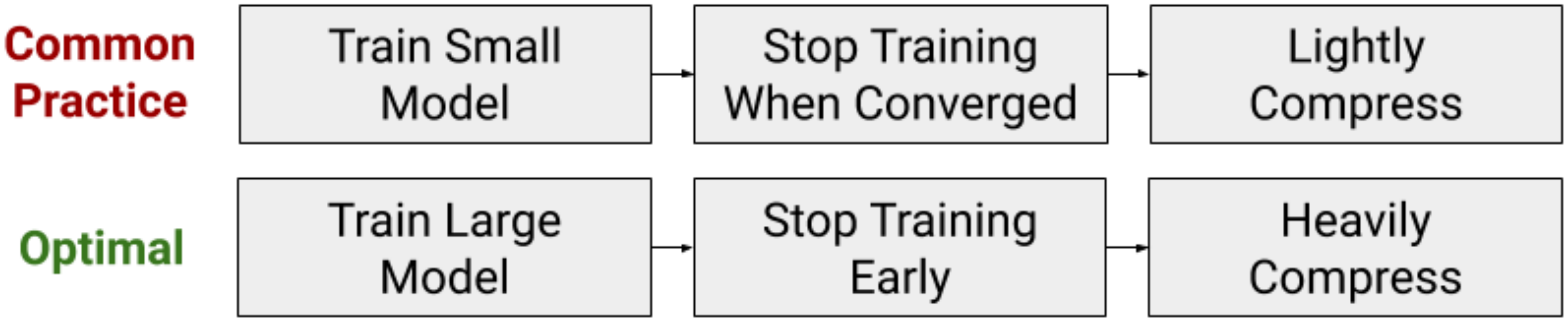


Figure 12 **Left:** Given a fixed compute budget, a particular model size is optimal, though somewhat larger or smaller models can be trained with minimal additional compute. **Right:** Models larger than the compute-efficient size require fewer steps to train, allowing for potentially faster training if sufficient additional parallelism is possible. Note that this equation should not be trusted for very large models, as it is only valid in the power-law region of the learning curve, after initial transient effects.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361*.

THE SCALING LAWS

Larger models train faster



https://bair.berkeley.edu/blog/2020/03/05/compress/

THE SCALING LAWS

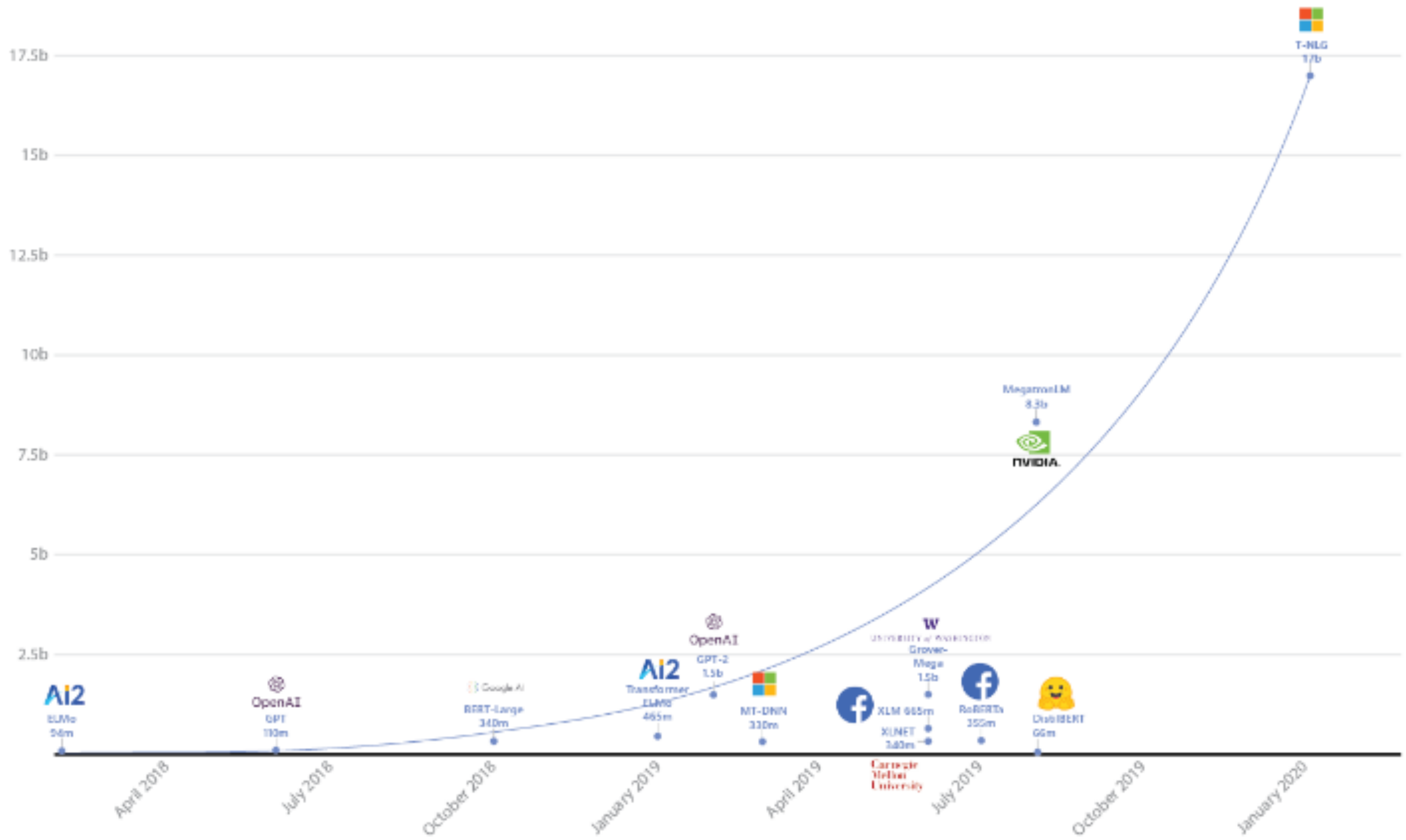
MOST IMPORTANT!!

“… more importantly, we find that the precise architectural hyperparameters are unimportant compared to the overall scale of the language model.”

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., ... & Amodei, D. (2020). Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361*.

THE SCALING LAWS

Next two years will bring much larger models



TOWARDS A TRILLION-PARAMETER MODEL

TURINGNLG

17 billion parameters

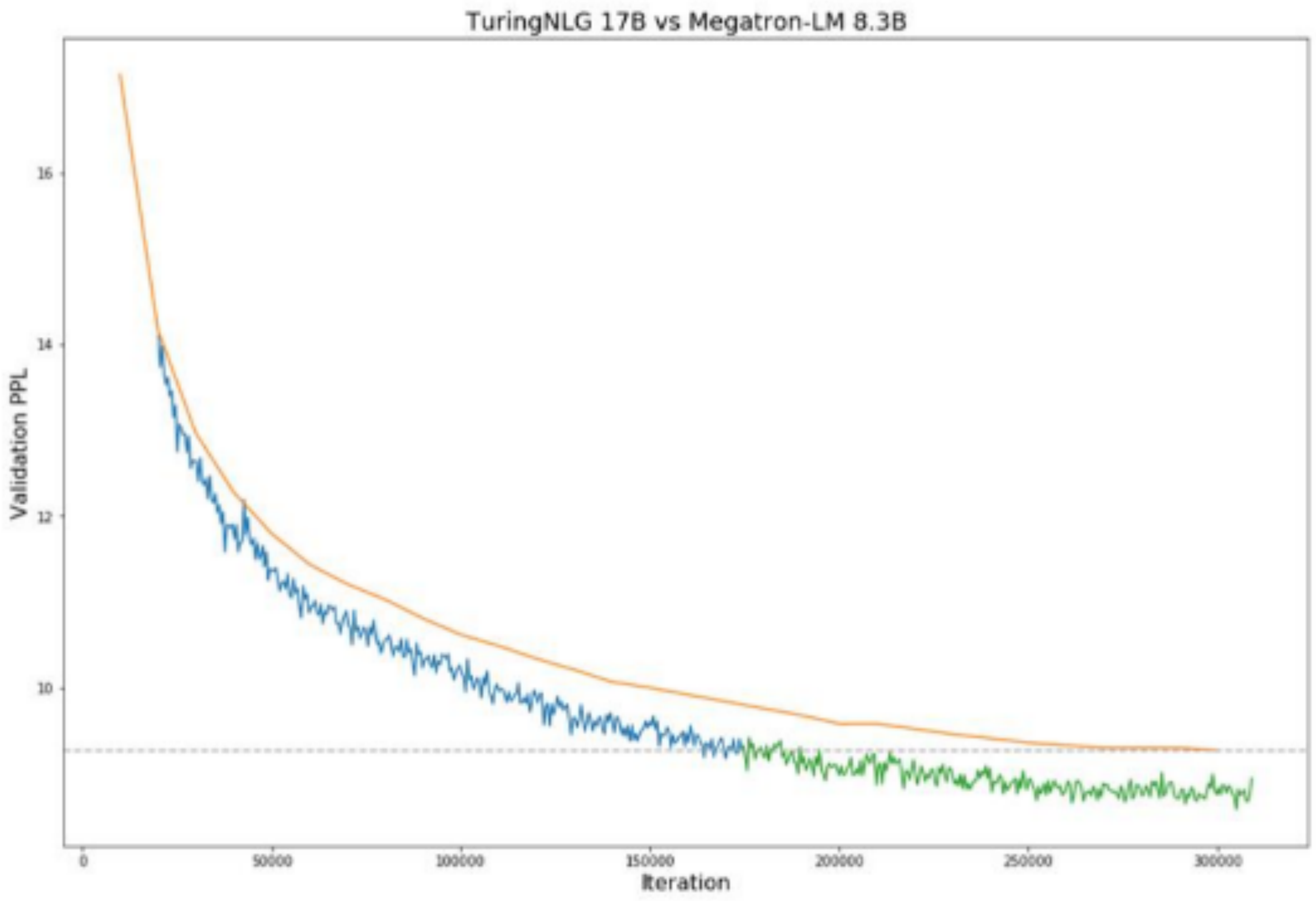
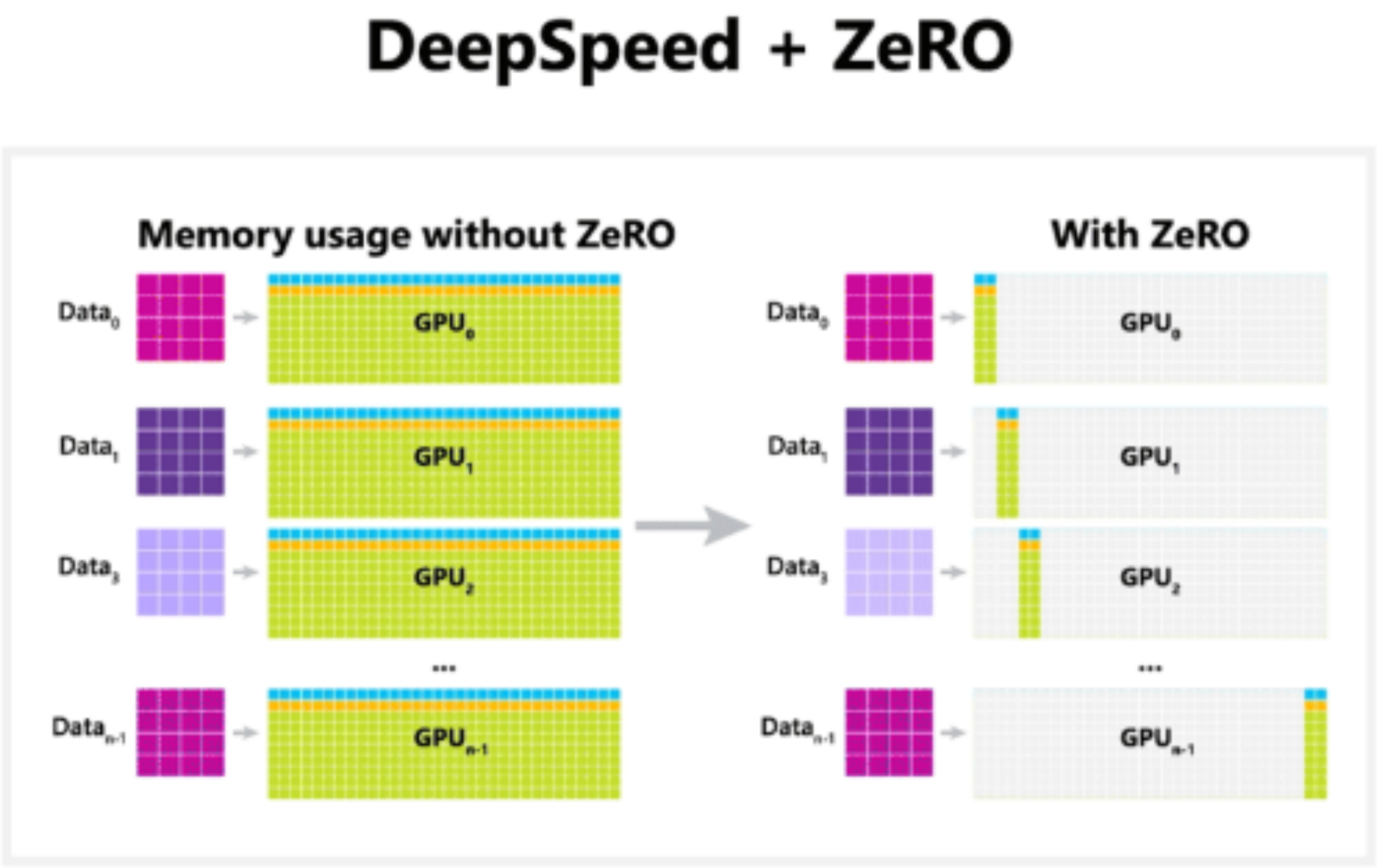


Figure 1: Comparison of the validation perplexity of Megatron-8B parameter model (orange line) vs T-NLG 17B model during training (blue and green lines). The dashed line represents the lowest validation loss achieved by the current public state of the art model. The transition from blue to green in the figure indicates where T-NLG outperforms public state of the art.

THE FUTURE

Towards a trillion-parameter model



Scale
- 100B parameter
- 10X bigger

Speed
- Up to 5X faster

Cost
- Up to 5X cheaper

Usability
- Minimal code change

GPT-3

EVEN MORE IMPORTANTLY

Large neural networks use data more efficiently

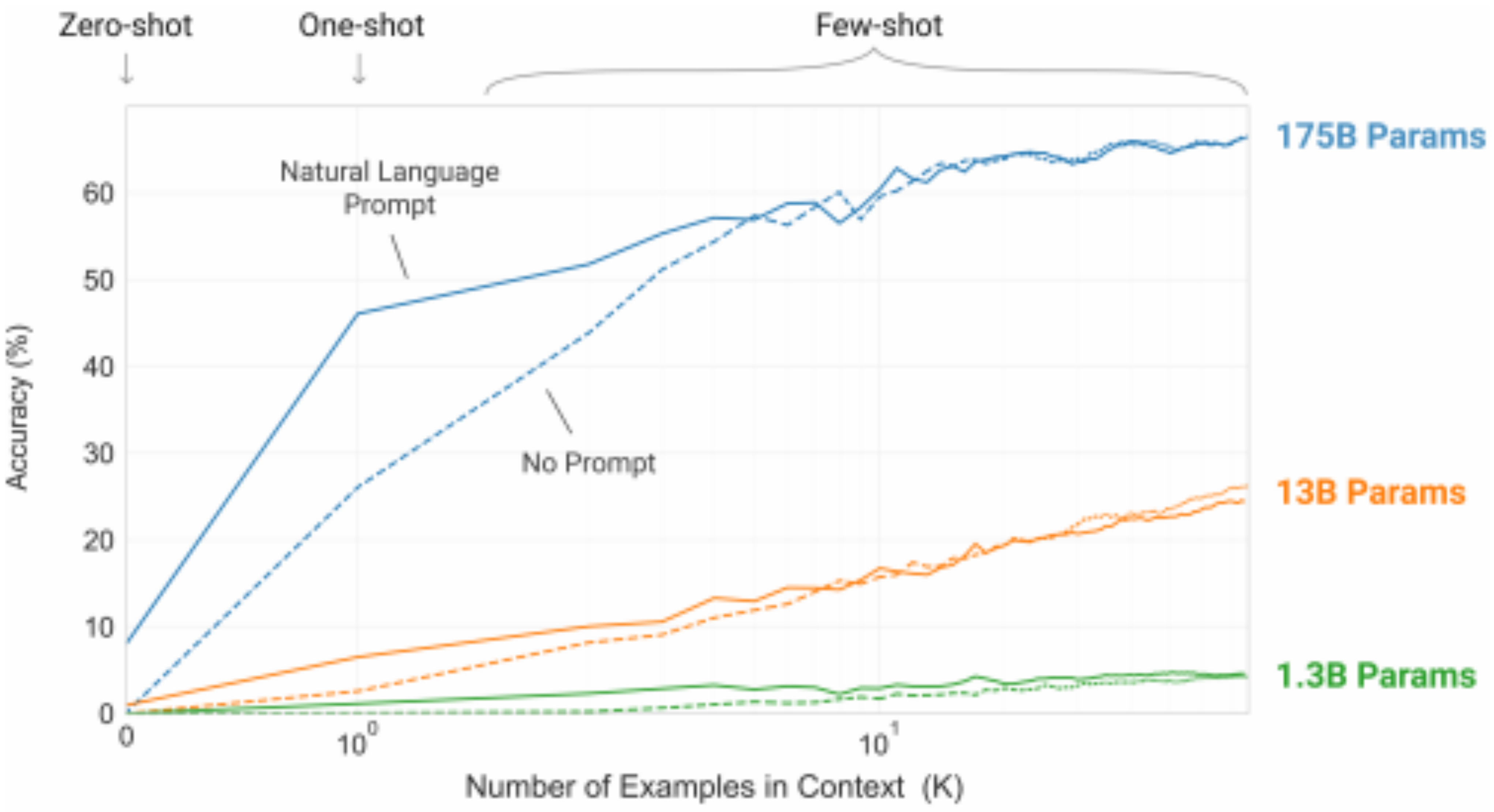


Figure 1.2: Larger models make increasingly efficient use of in-context information. We show in-context learning performance on a simple task requiring the model to remove random symbols from a word, both with and without a natural language task description (see Sec. 3.9.2). The steeper "in-context learning curves" for large models demonstrate improved ability to learn a task from contextual information. We see qualitatively similar behavior across a wide range of tasks.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., & Catanzaro, B. (2019). Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. arXiv preprint arXiv:1909.08053
Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Agarwal, S. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*..

EVEN MORE IMPORTANTLY

Large neural networks use data more efficiently

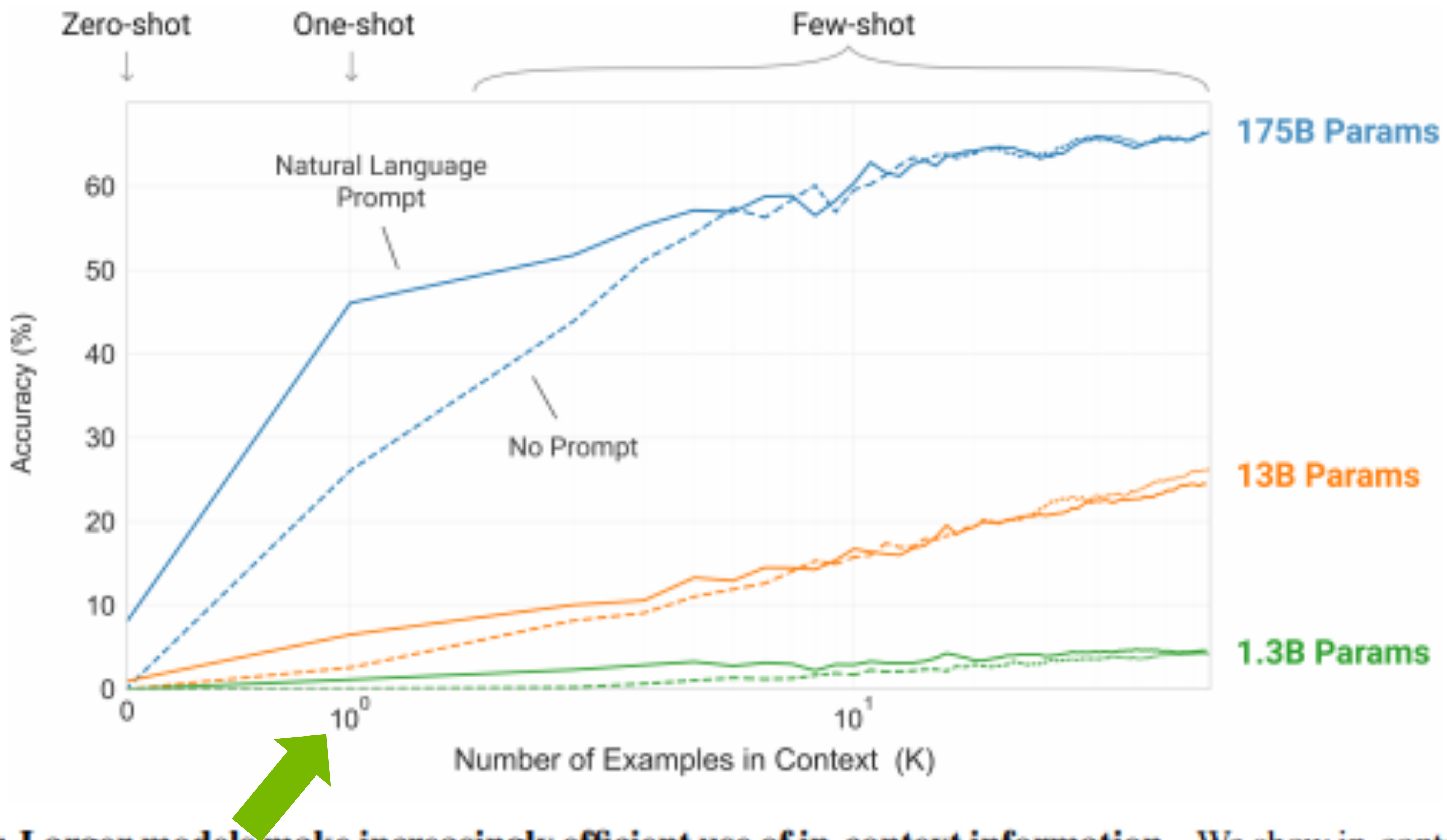


Figure 1.2: Larger models make increasingly efficient use of in-context information. We show in-context learning performance on a simple task requiring the model to remove random symbols from a word, both with and without a natural language task description (see Sec. 3.9.2). The steeper "in-context learning curves" for large models demonstrate improved ability to learn a task from contextual information. We see qualitatively similar behavior across a wide range of tasks.

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., & Catanzaro, B. (2019). Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. arXiv preprint arXiv:1909.08053

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Agarwal, S. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*..

WHAT DO WE MEAN BY BIG?

GPT-3 size comparison

Not a linear scale

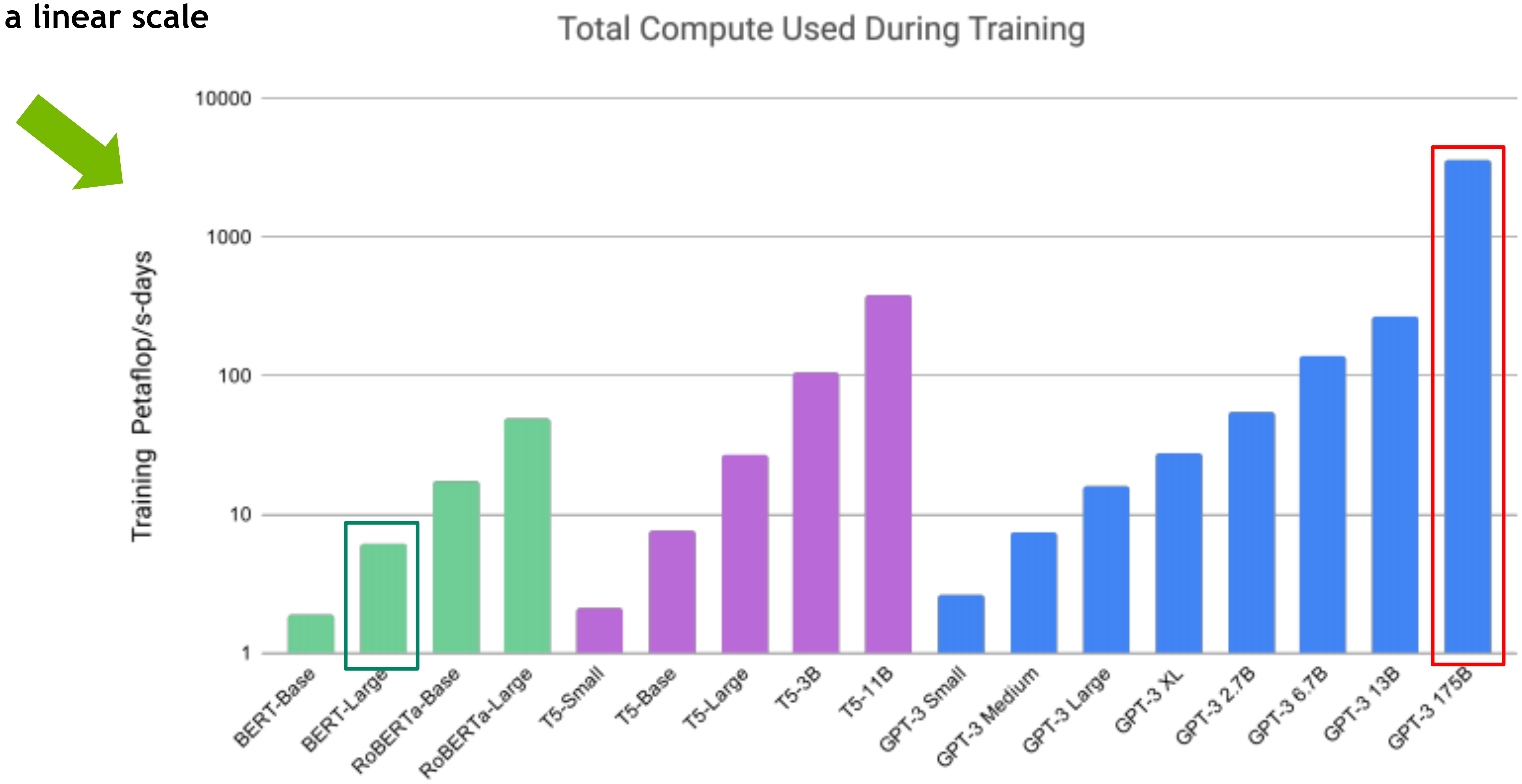


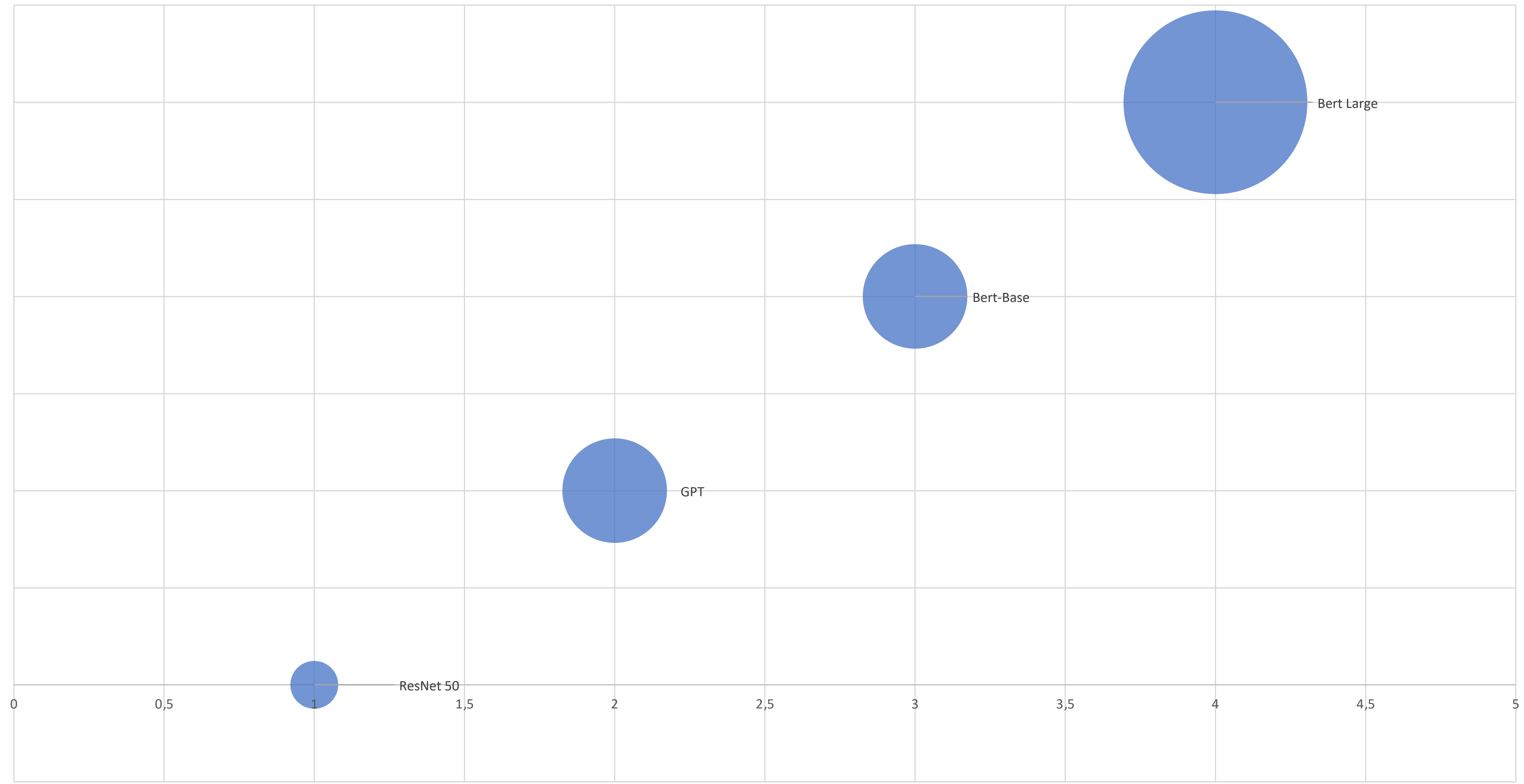
Figure 2.2: Total compute used during training. Based on the analysis in Scaling Laws For Neural Language Models [KMH+20] we train much larger models on many fewer tokens than is typical. As a consequence, although GPT-3 3B is almost 10x larger than RoBERTa-Large (355M params), both models took roughly 50 petaflop/s-days of compute during pre-training. Methodology for these calculations can be found in Appendix D.

PERSPECTIVE

WHAT DO WE MEAN BY BIG?

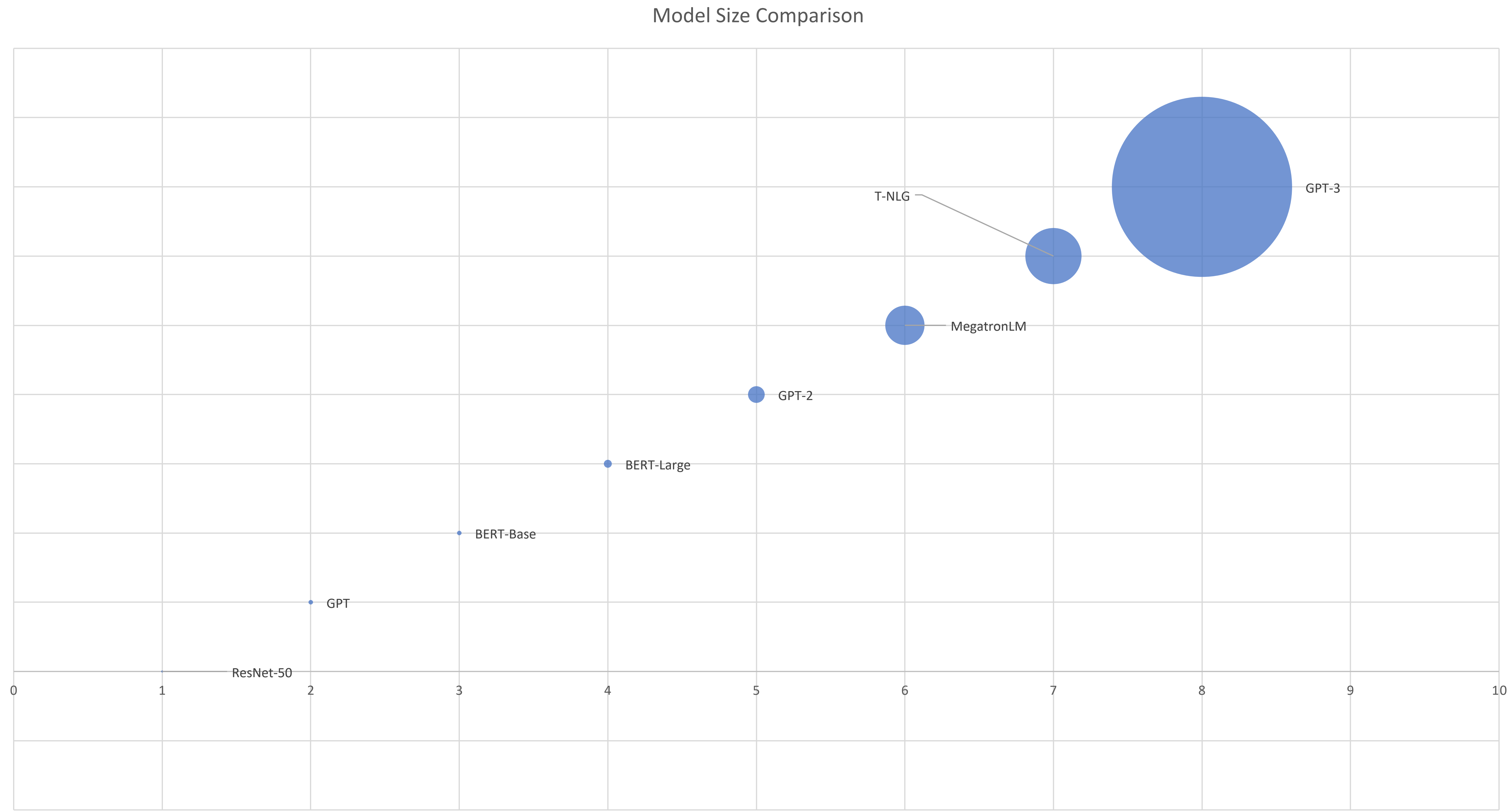
Perspective

Model Size Comparison



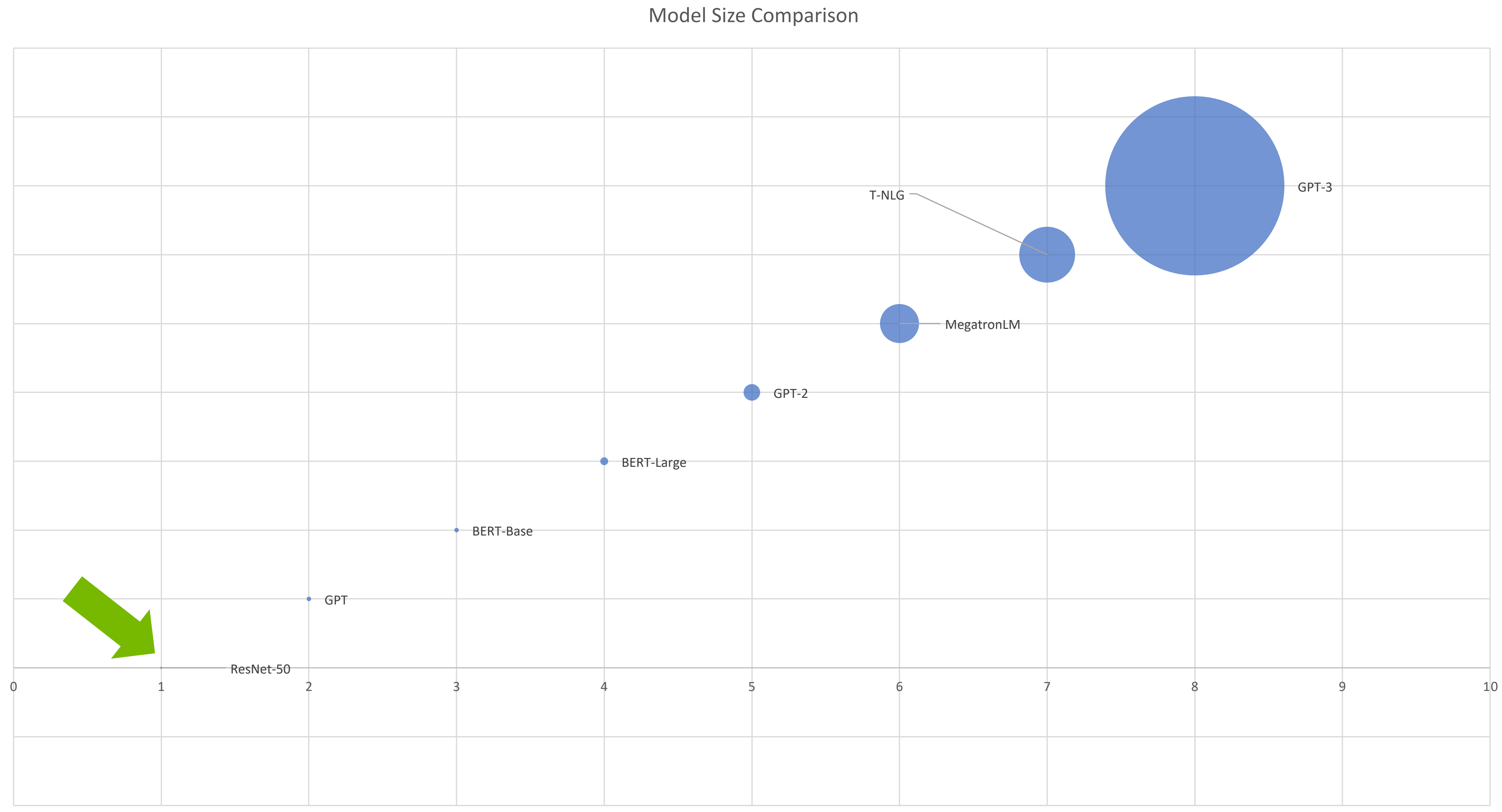
WHAT DO WE MEAN BY BIG?

Perspective



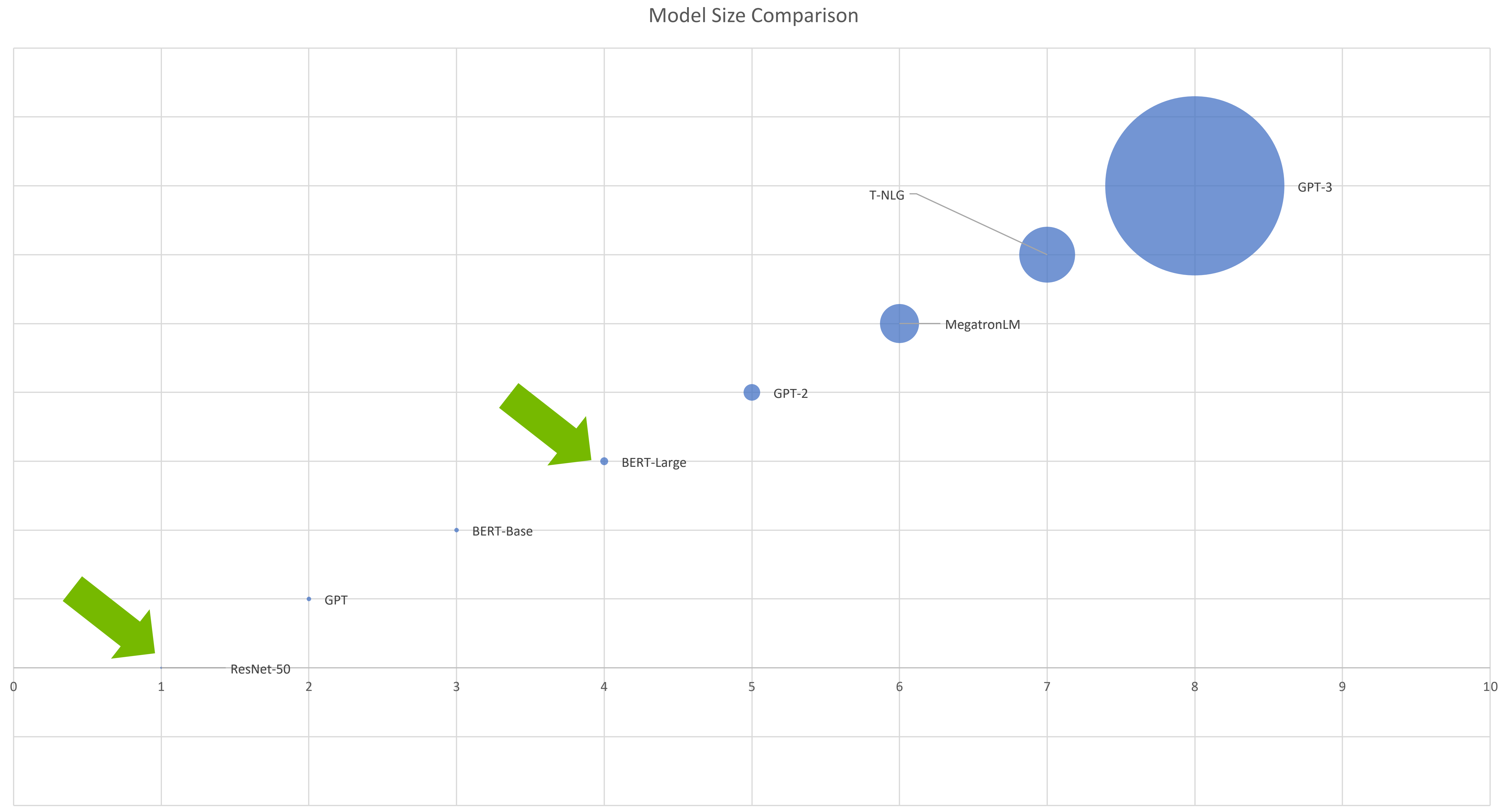
WHAT DO WE MEAN BY BIG?

Perspective



WHAT DO WE MEAN BY BIG?

Perspective



WHAT DO WE MEAN BY BIG?

GPT-3 size comparison: 538x Bigger than BERT-Large

Not a linear scale

355 years on a single V100

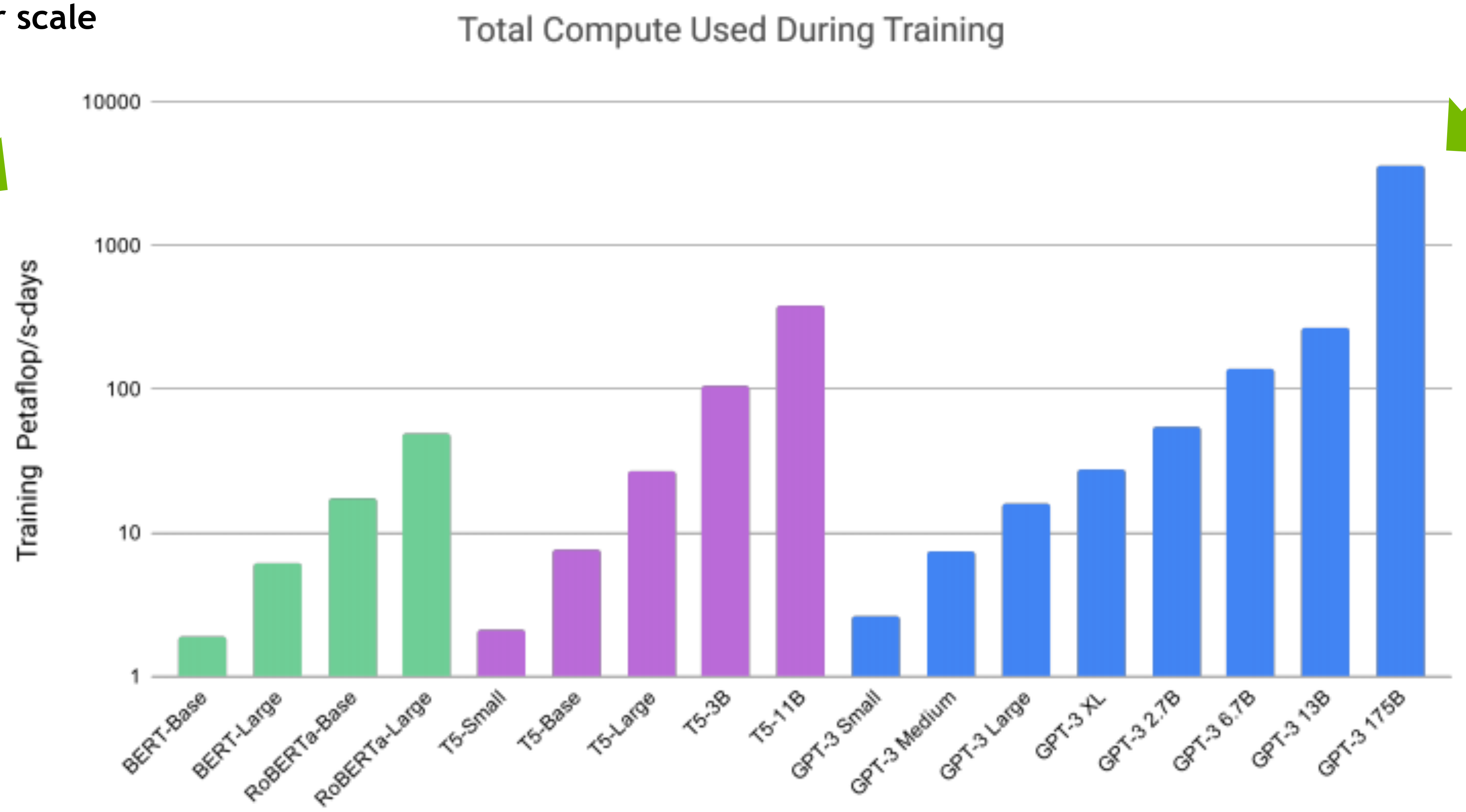


Figure 2.2: Total compute used during training. Based on the analysis in Scaling Laws For Neural Language Models [KMH+20] we train much larger models on many fewer tokens than is typical. As a consequence, although GPT-3 3B is almost 10x larger than RoBERTa-Large (355M params), both models took roughly 50 petaflop/s-days of compute during pre-training. Methodology for these calculations can be found in Appendix D.

THE LAB

Part 2: Self-Supervision, BERT and Beyond

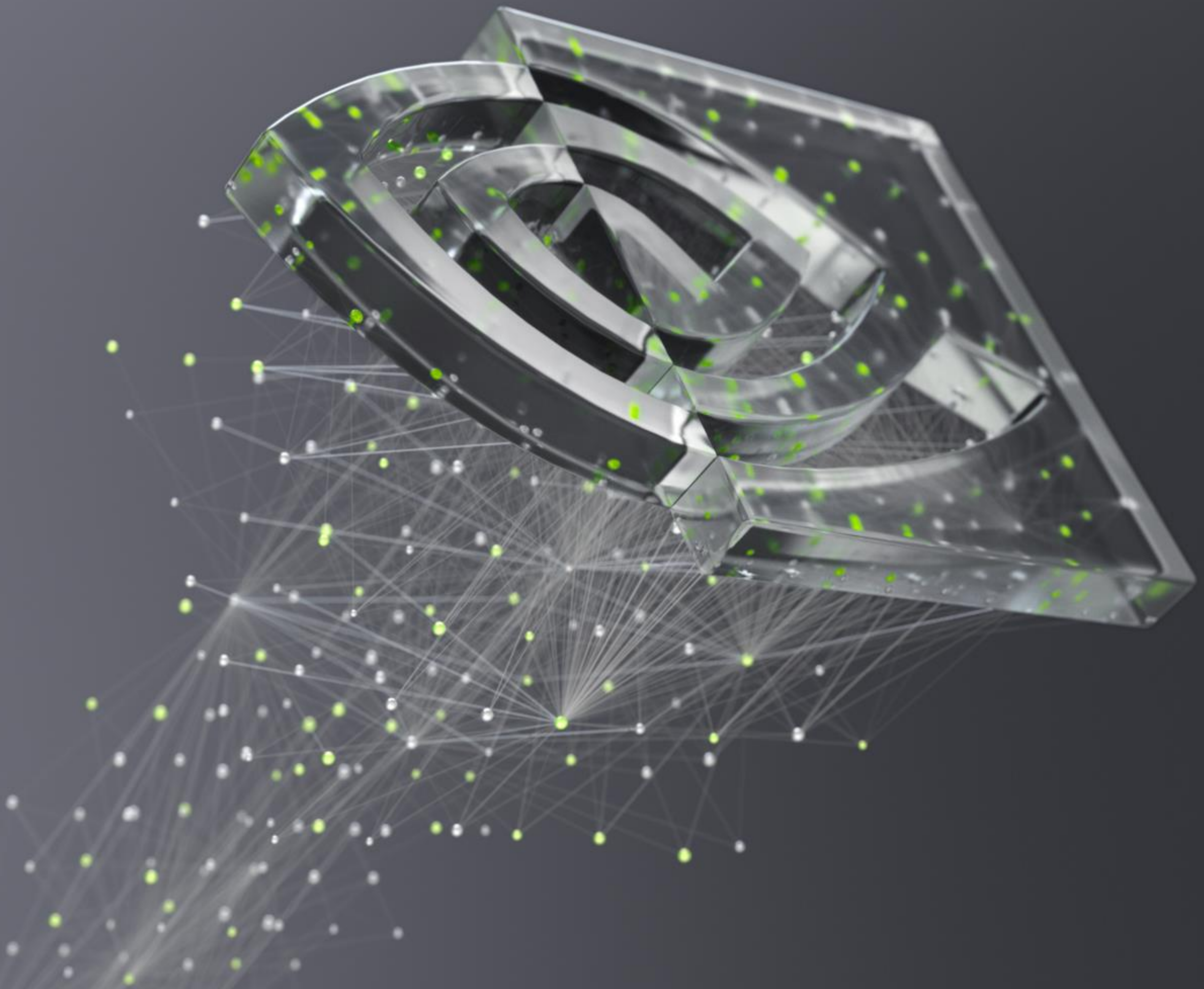
- Lecture
  - Why DNNs?
  - Self-Supervision
  - BERT
- Lab
  - Explore the Data
  - Explore NeMo
  - Text Classifier Project
- Lecture (cont'd)
  - Bigger is Better
  - Can and should we go even bigger?
- Lab (cont'd)
  - Named Entity Recognizer

IN THE NEXT CLASS…

NEXT CLASS

Overview

1. Discuss how to design your model for efficient inference
2. Discuss how to optimise your model for efficient execution
3. Discuss how to efficiently host a largely Conversational AI application



NVIDIA

DEEP LEARNING INSTITUTE