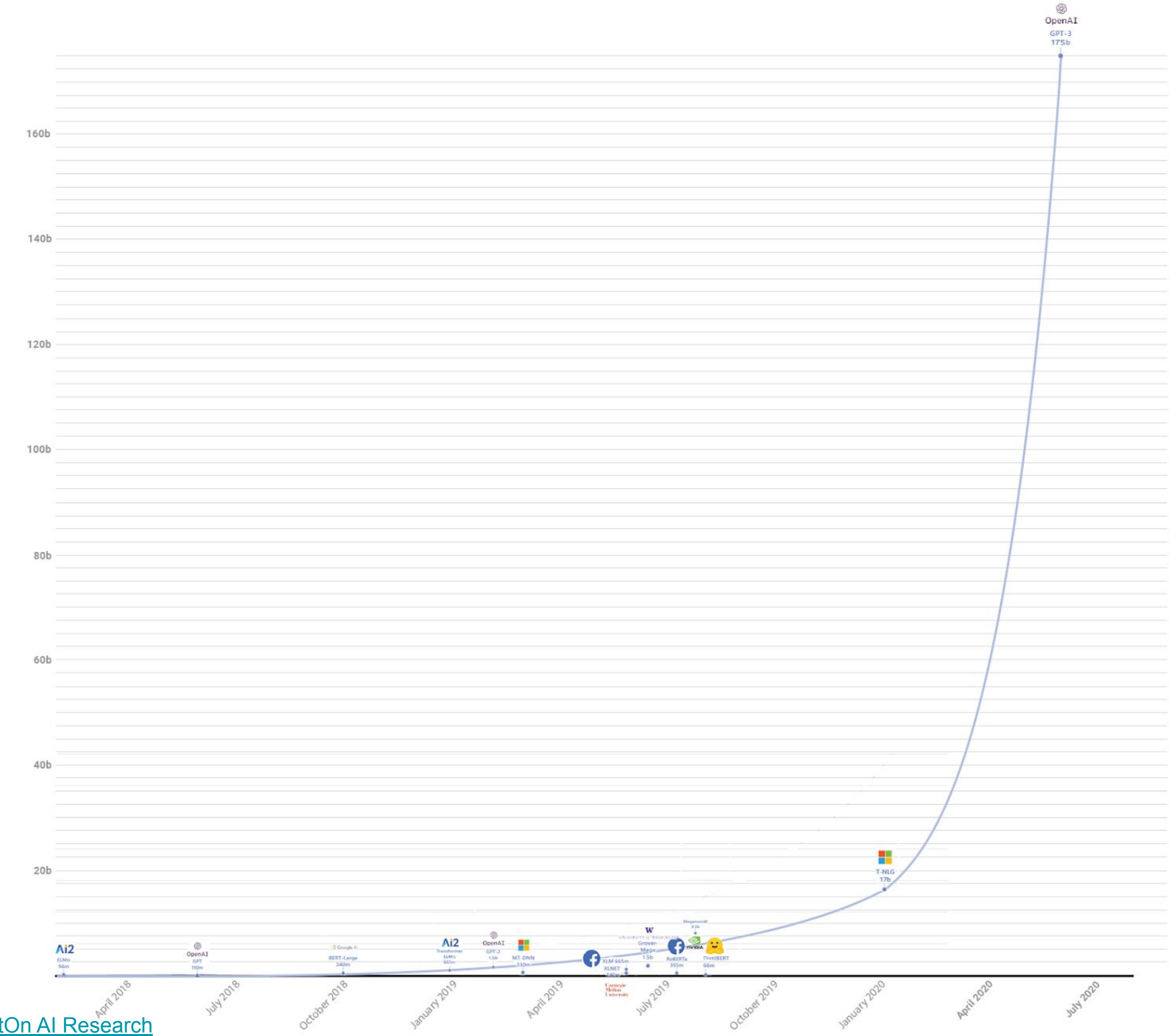
Intro to scaling laws for NN

Michael M. Pieler

Scaling?



Why GPT-3 Matters, most recent updates: The Akronomicon — LightOn AI Research

Scaling laws for language models (1)

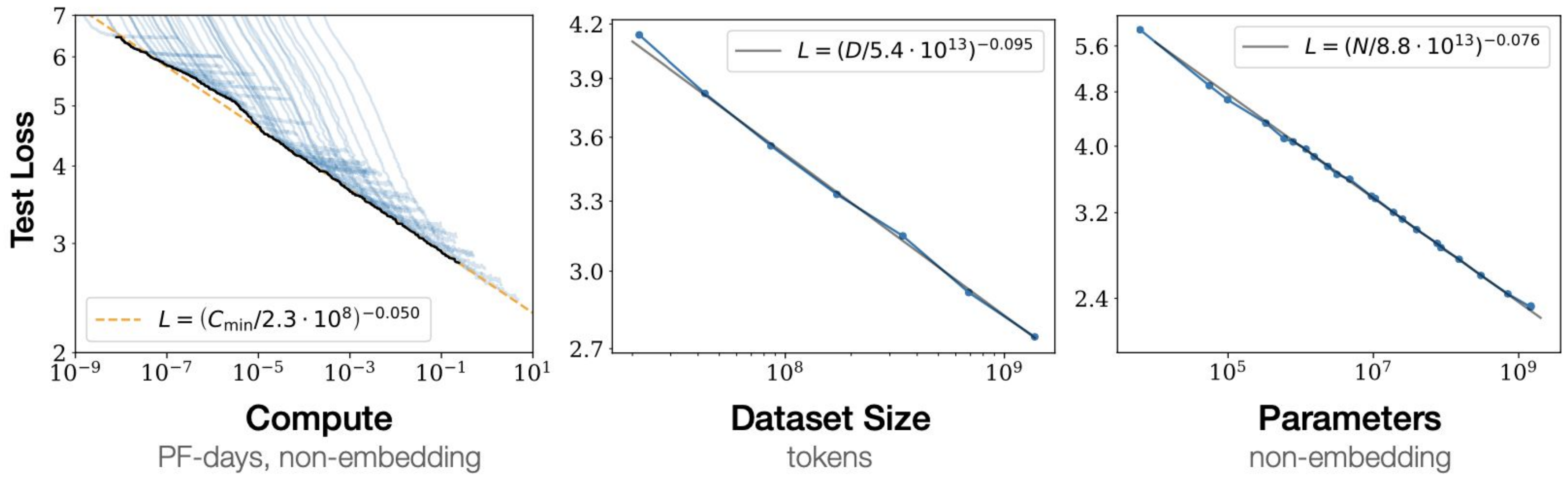
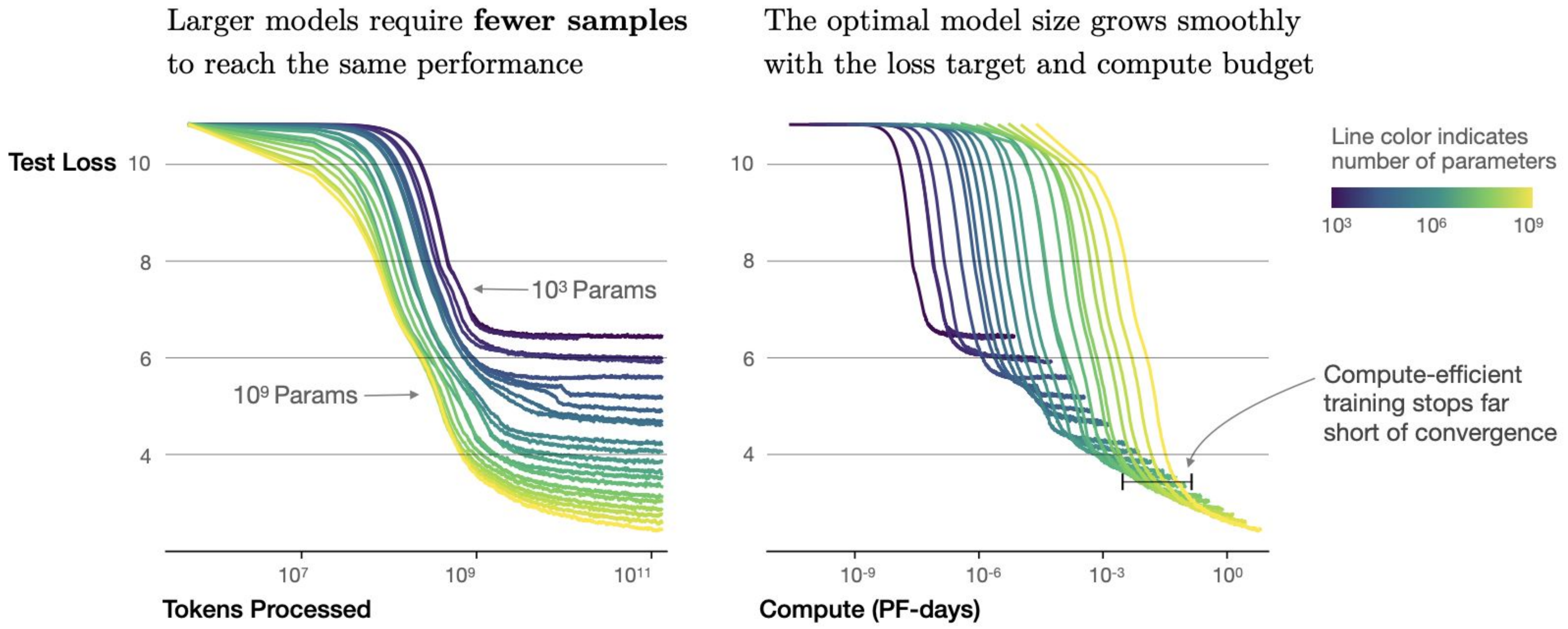


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when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

[2001.08361] Scaling Laws for Neural Language Models

Scaling laws for language models (2)



[2001.08361] Scaling Laws for Neural Language Models

Scaling laws for language models (3)

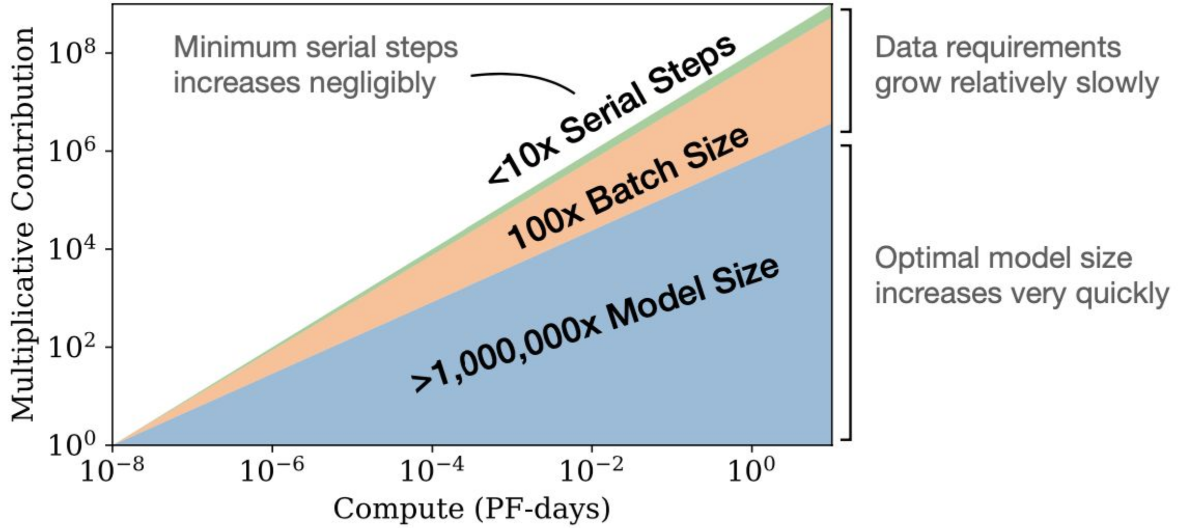
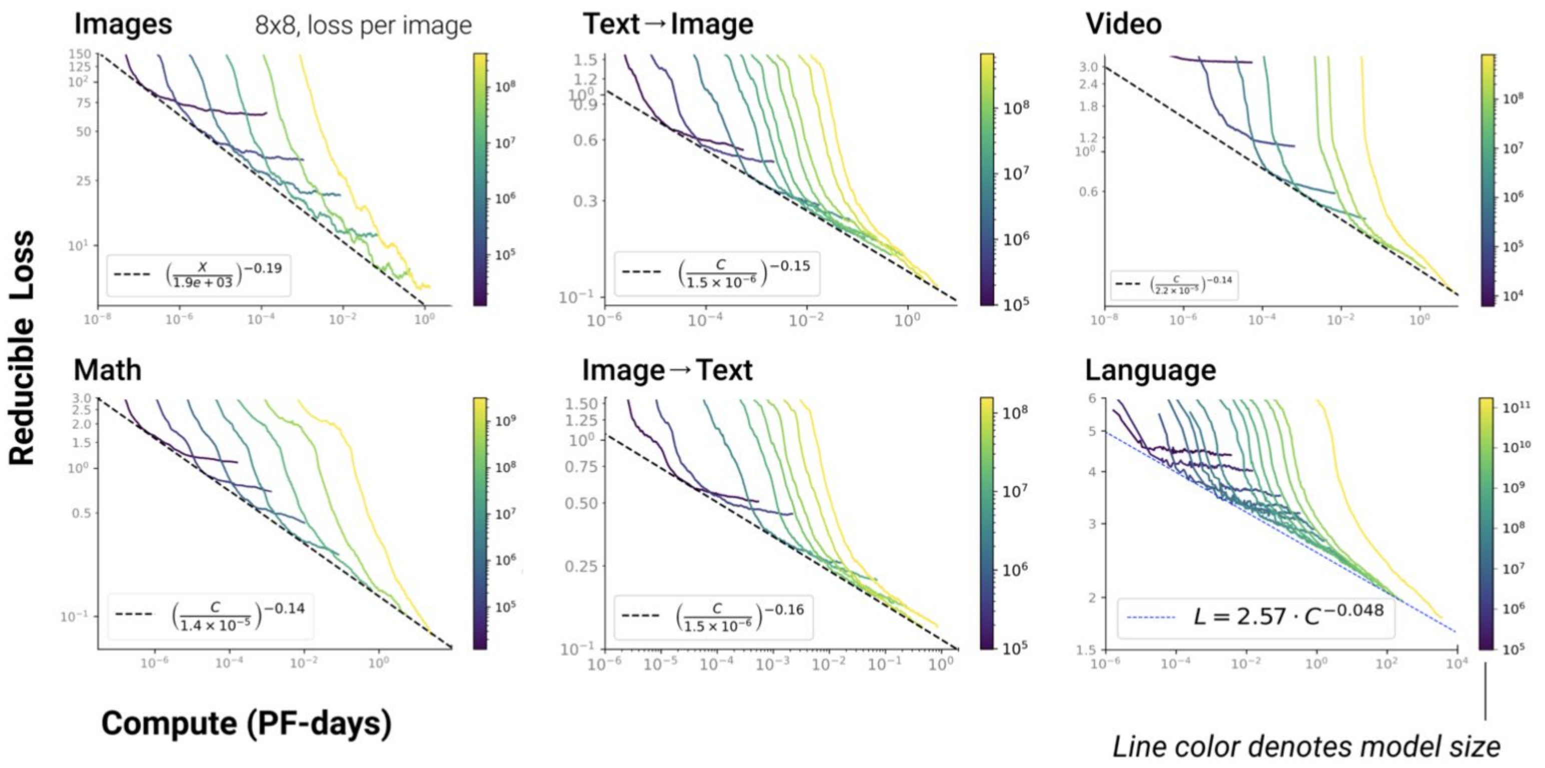


Figure 3 As more compute becomes available, we can choose how much to allocate towards training larger models, using larger batches, and training for more steps. We illustrate this for a billion-fold increase in compute. For optimally compute-efficient training, most of the increase should go towards 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.

[2001.08361] Scaling Laws for Neural Language Models

Scaling laws for other data modalities



[2010.14701] Scaling Laws for Autoregressive Generative Modeling

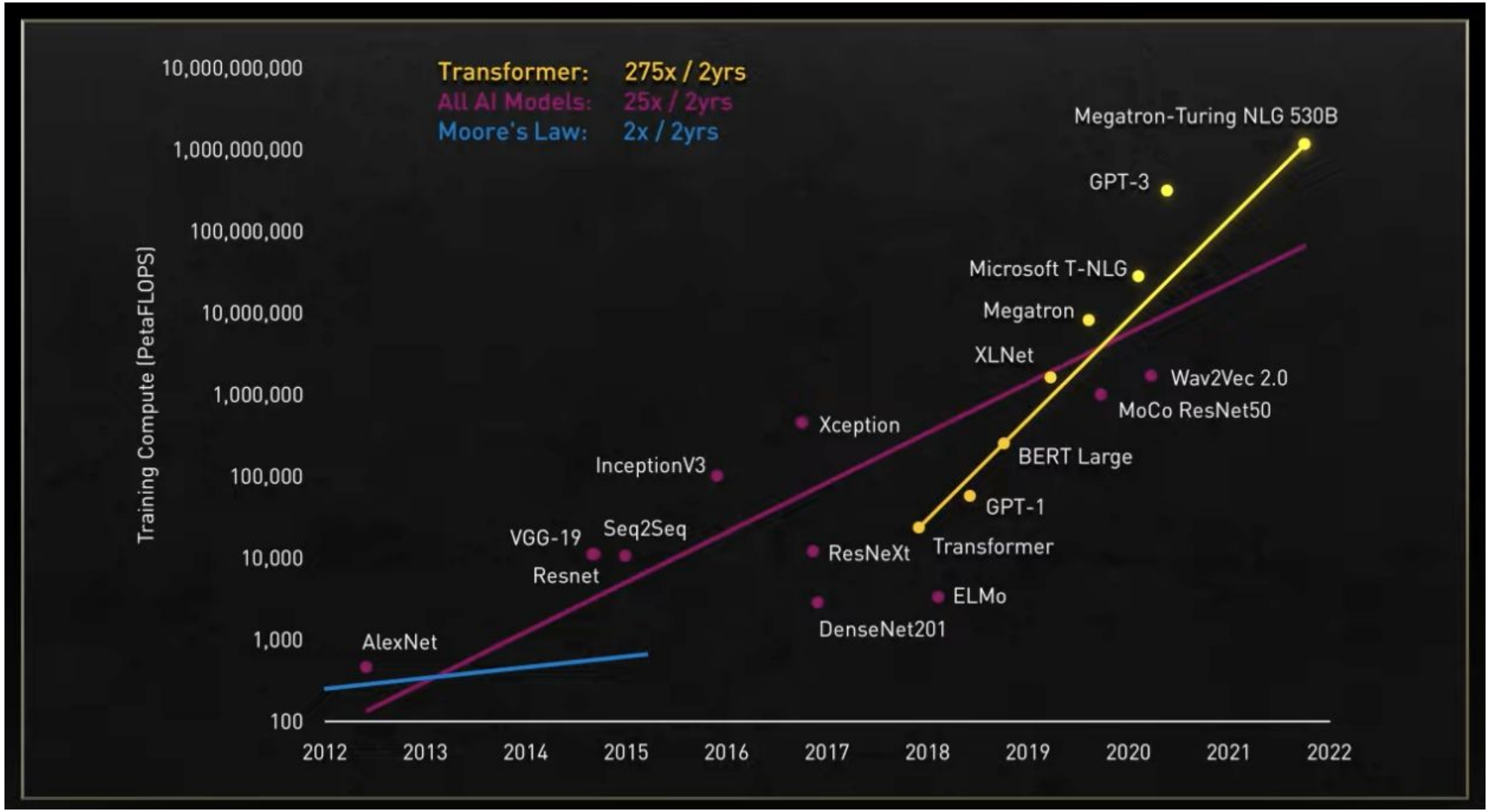
The bitter lesson?

“One thing that should be learned from the bitter lesson is the great power of general purpose methods, of methods that continue to scale with increased computation even as the available computation becomes very great.

The two methods that seem to scale arbitrarily in this way are search and learning.”

[The Bitter Lesson]

Training compute trends



GTC November 2021 Keynote

Summary

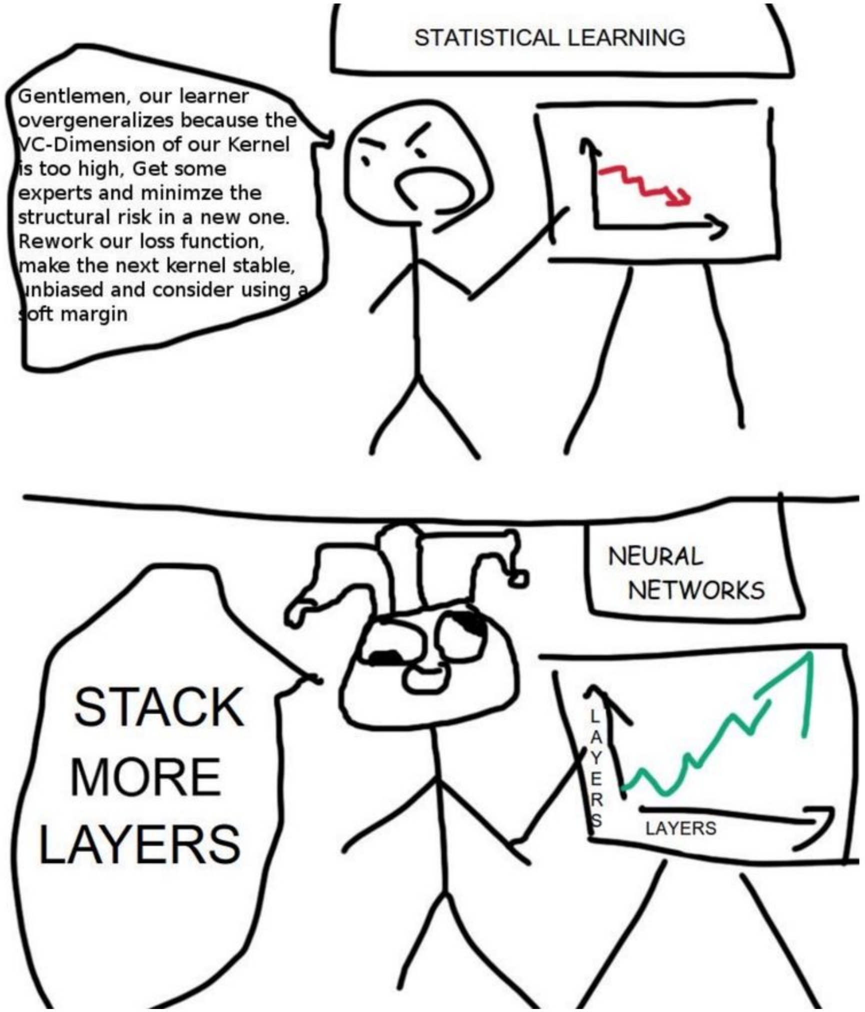
- Larger models are significantly more sample-efficient, such that optimally compute-efficient training involves training very large models on a relatively modest amount of data and stopping significantly before convergence.

- Larger compute budgets should be spent mostly on larger models, rather than much longer training runs.

- Big models may be more important than big data.

- The most effective way to use a given computational budget is to train the biggest model possible for 1 epoch.

Short summary ;-)


STATISTICAL LEARNING
Gentlemen, our learner overgeneralizes because the VC-Dimension of our Kernel is too high, Get some experts and minimze the structural risk in a new one. Rework our loss function, make the next kernel stable, unbiased and consider using a soft margin
NEURAL NETWORKS
STACK MORE LAYERS
LAYERS
LAYERS

StAcK MoRe LaYeRs: ProgrammerHumor

Additional sources

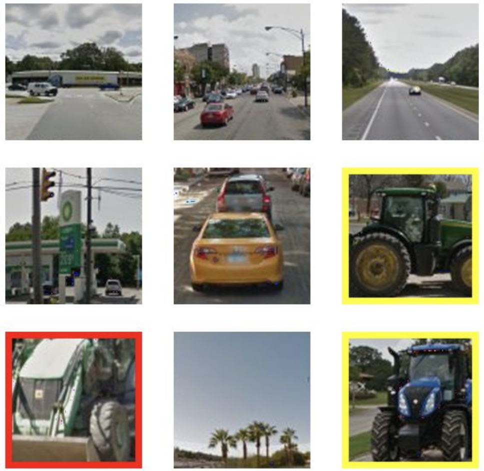
- EleutherAI discord server discussions
- The Scaling Hypothesis · Gwern.net
- Machine Learning Scaling · Gwern.net
- Neural Scaling Laws Slides by Ethan Caballero & Brady Neal
- Scaling laws workshop
- r/mlscaling
- GTC21 Step-by-step Guide to Building Large Custom Language Models PDF

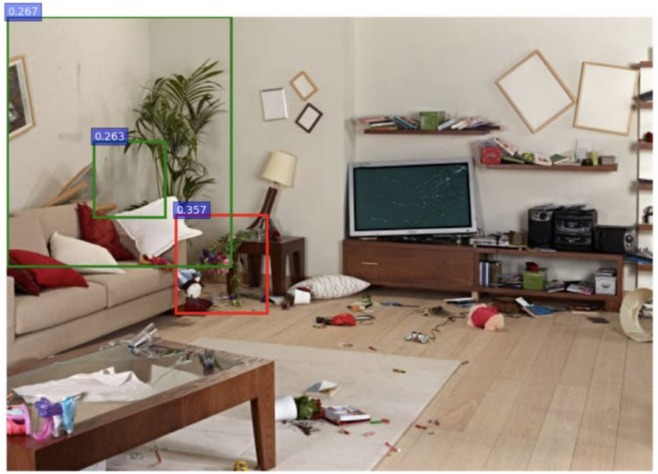
Recent interesting CLIP applications

Michael M. Pieler

Zero-shot object detection

- CLIP: Zero-shot Jack of All Trades & https://github.com/kevinzakka/clip_playground

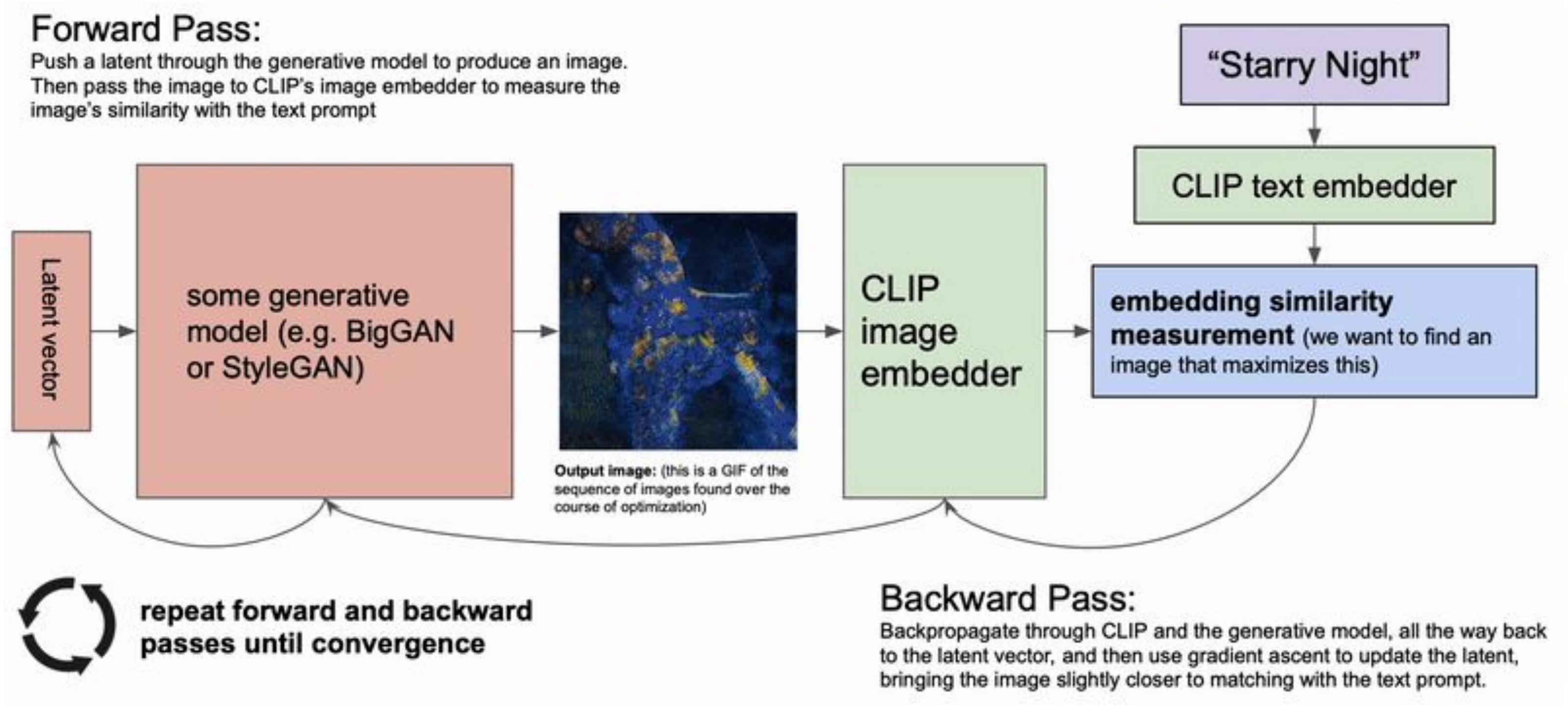

reCAPTCHA prompt: "Select all images with tractors".



The caption I entered: "the plants in the vase".

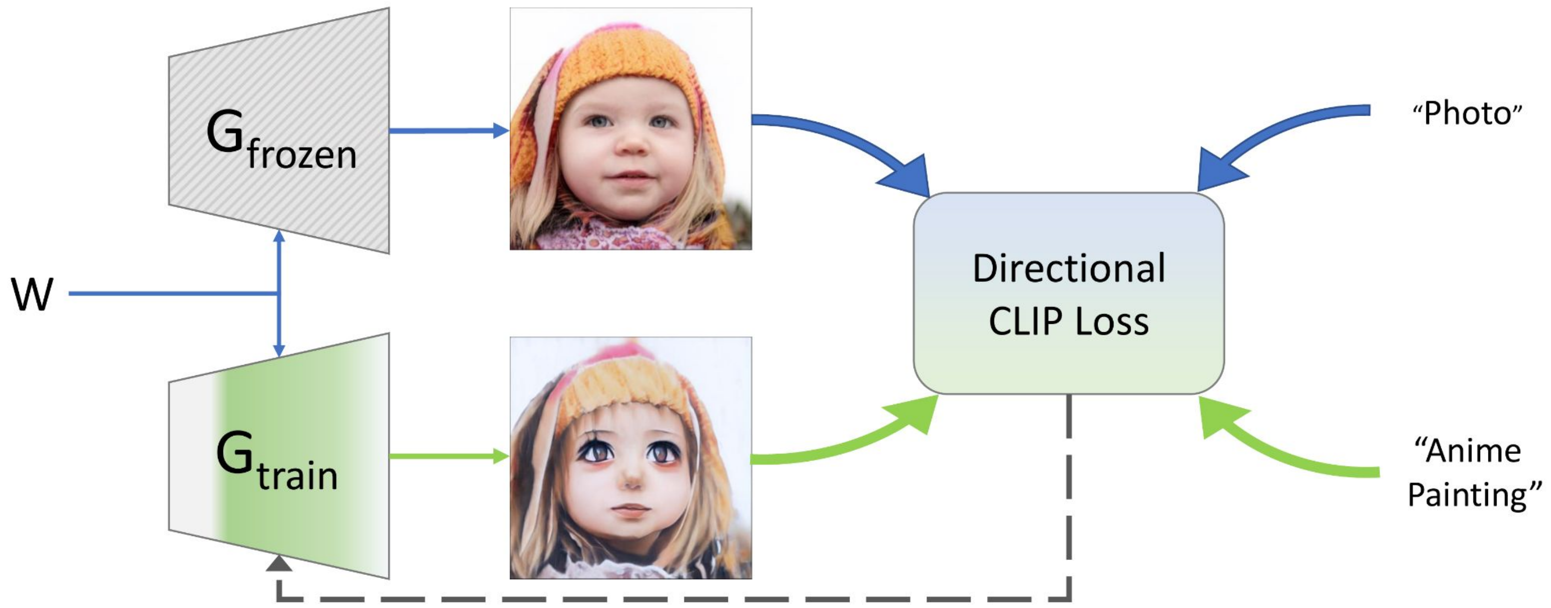
CLIP generated art

- Alien Dreams: An Emerging Art Scene - ML@B Blog


Forward Pass:
Push a latent through the generative model to produce an image. Then pass the image to CLIP's image embedder to measure the image's similarity with the text prompt
"Starry Night"
CLIP text embedder
Latent vector
some generative model (e.g. BigGAN or StyleGAN)
CLIP image embedder
embedding similarity measurement (we want to find an image that maximizes this)
Output image: (this is a GIF of the sequence of images found over the course of optimization)
repeat forward and backward passes until convergence
Backward Pass:
Backpropagate through CLIP and the generative model, all the way back to the latent vector, and then use gradient ascent to update the latent, bringing the image slightly closer to matching with the text prompt.

CLIP-guided domain adaptation

- StyleGAN-NADA: CLIP-Guided Domain Adaptation of Image Generators



G_{frozen}
W
G_{train}
Directional CLIP Loss
"Photo"
"Anime Painting"