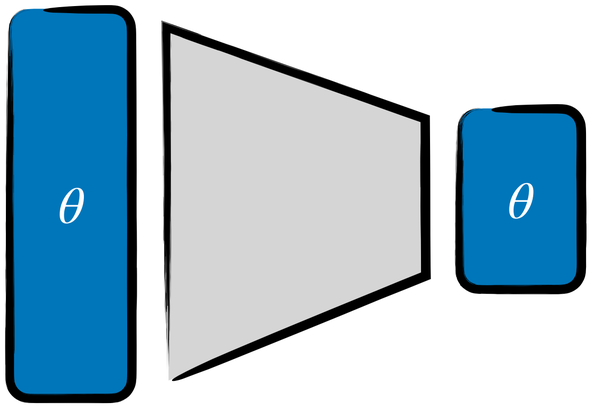
Intro to ML III

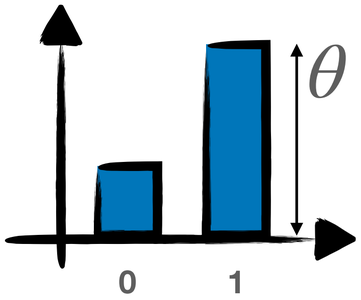
CERN School of Computing 2023

Lukas Heinrich, TUM

Where we where

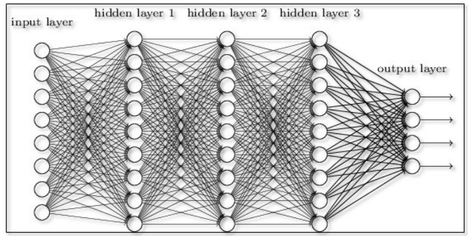
So far we discussed supervised learning + tricks. Gave us a natural learning task to predict latent properties $q(z \mid x)$



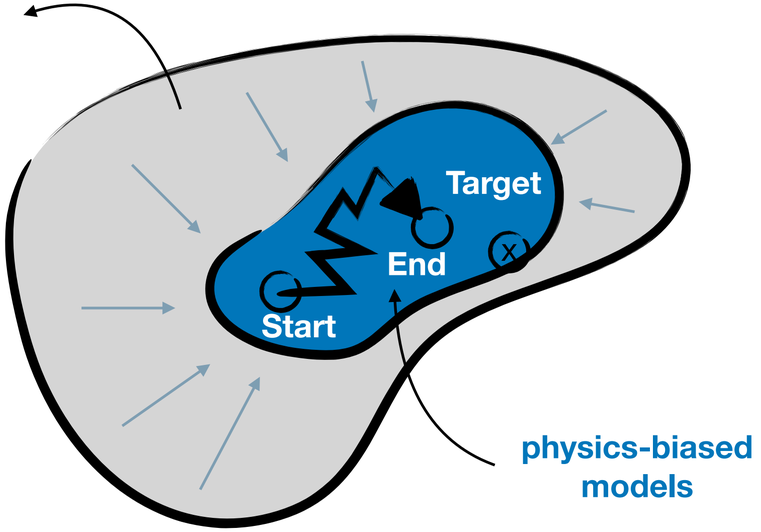


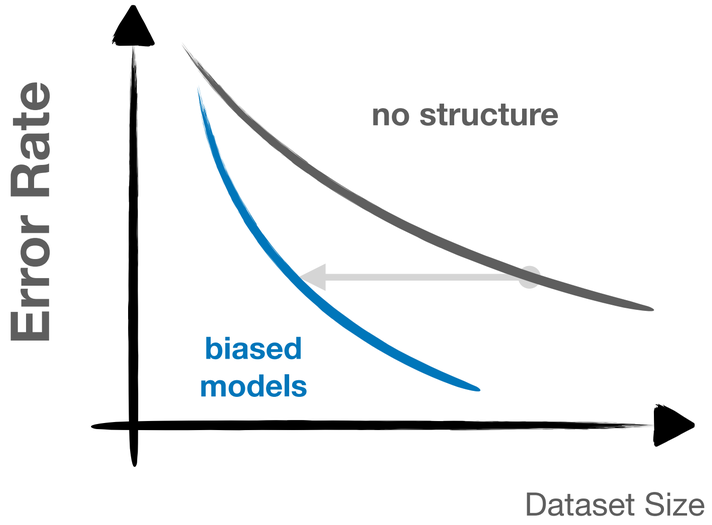
$$q_\phi(z \mid x) = q(z \mid \theta = f_\phi(x))$$

Inductive Bias



unstructured models





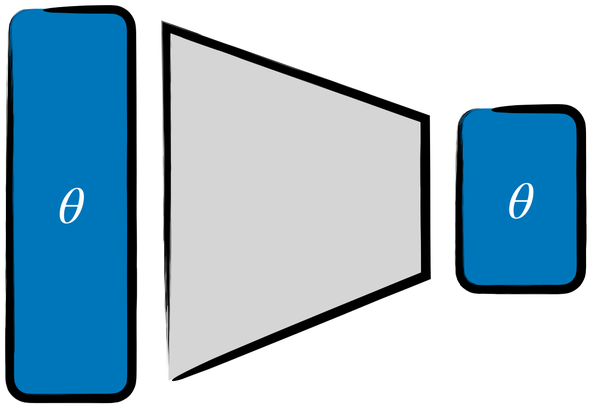
Architectures

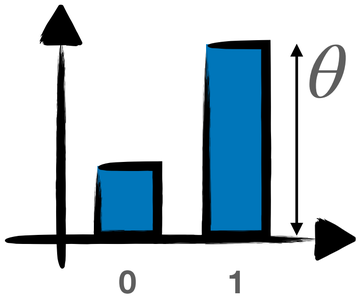
Data Type	***Symmetry***	***Network***
Grids/n-D arrays	*Translation*	*CNN*
Full Graph	*Permutation*	*Transformers*
Any Graphs	*Permutation*	*GNN*
Sets of Objects	*Permutation*	*Deep Set*
Sequences	*Time Warping*	*RNN/LSTM*
…	*…*	*…*

Unsupervised Learning

Unsupervised Learning

So far we discussed supervised learning + tricks. Gave us a natural learning task to predict latent properties $q(z \mid x)$





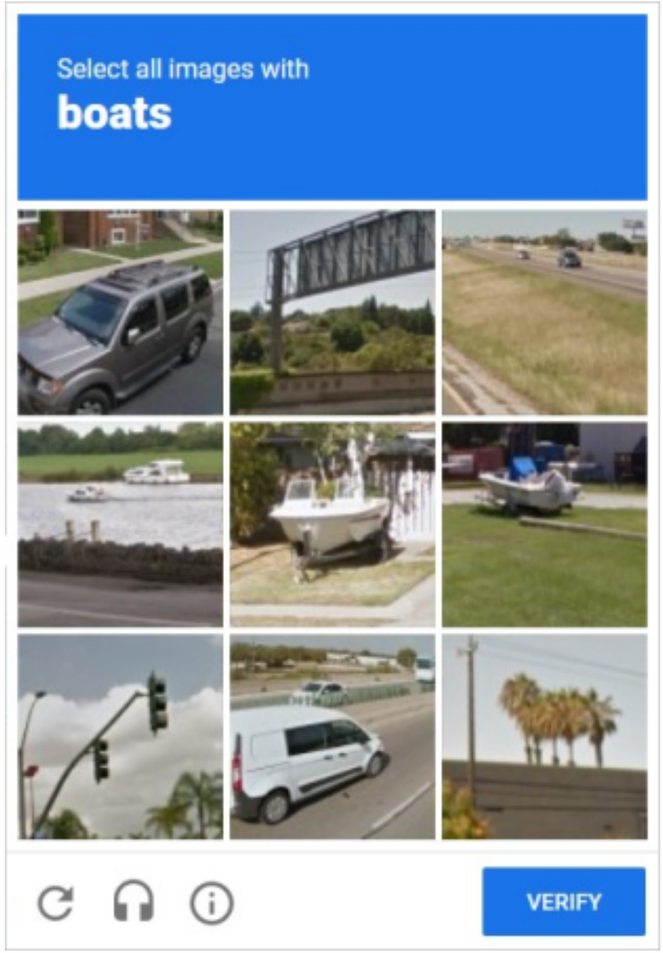
$$q_\phi(z \mid x) = q(z \mid \theta = f_\phi(x))$$

But supervised learning is not everything

Labeled Data is Sparse

Ever wonder:
what's the point of this?

You're providing free labor to produce labeled datasets.



Home > News > Computing

Captcha if you can: how you've been training AI for years without realising it

By James O'Malley published January 12, 2018

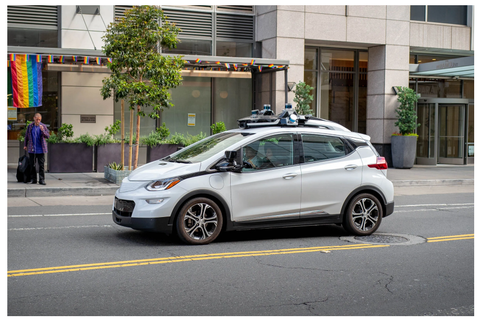
All those little visual puzzles add up to AI advances

Inference as Limited Understanding

We train ML on data to predict something specific

Car

Cow



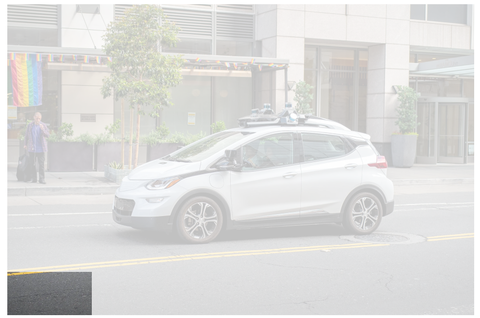
Does this mean it understands what a car or a cow is?

Inference as Limited Understanding

We train ML on data to predict something specific

Car

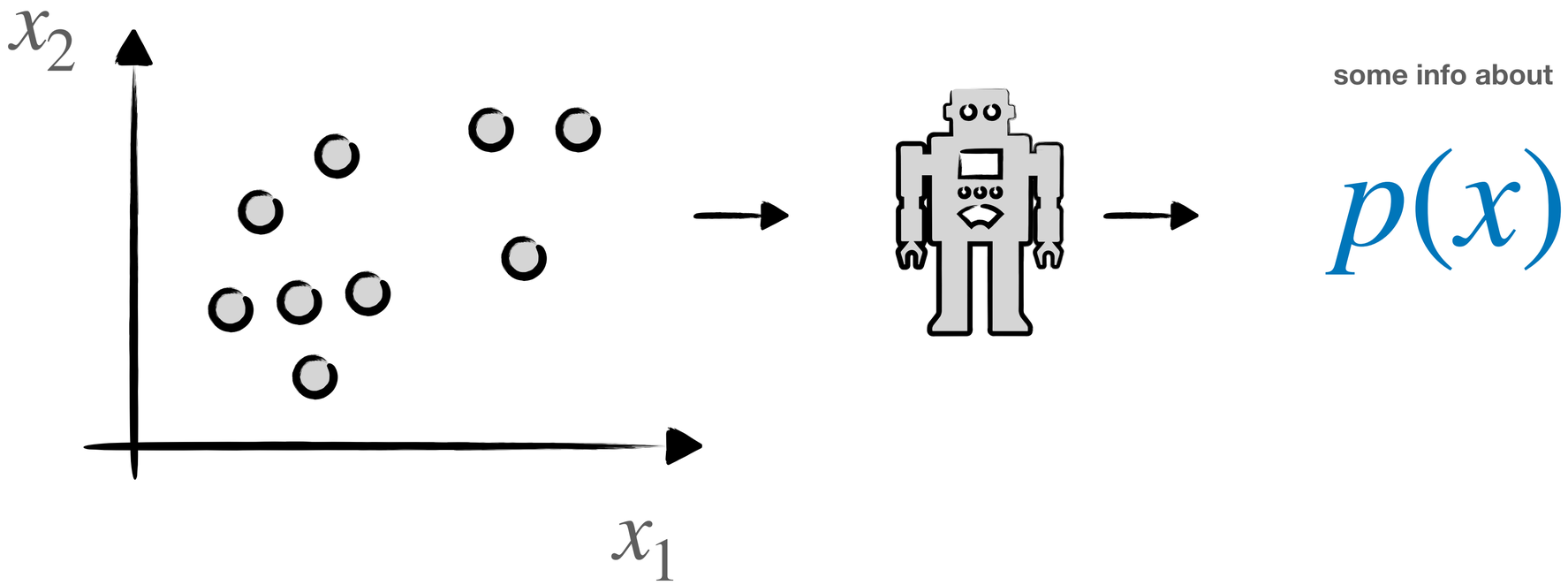
Cow



Does this mean it understands what a car or a cow is?

Unsupervised

The target of our study is $p(x)$ itself. If we don't have labels we at least want to *characterize the data distribution*



Probability models

Understand $p(x)$? It's two things at once:

A process

$$\rightarrow \mathbb{R}^2$$

Generating new samples from randomness

A formula

$$\mathbb{R}^2 \rightarrow \mathbb{R}$$

$$p_{\mu,\Sigma}(x) = \frac{1}{\sqrt{|2\pi\Sigma|}} exp\left(-\frac{1}{2}(x-\mu)^T\Sigma(x-\mu)\right)$$

Evaluating the Probability for a given sample

"Understanding $p(x)$": ability to do either of these or both

 $\rightarrow \mathbb{R}^2$

A lot of the recent headline-grabbing Deep Learning advances are parts of the generative modelling domain

$$\text{face} \sim p(\text{face})$$

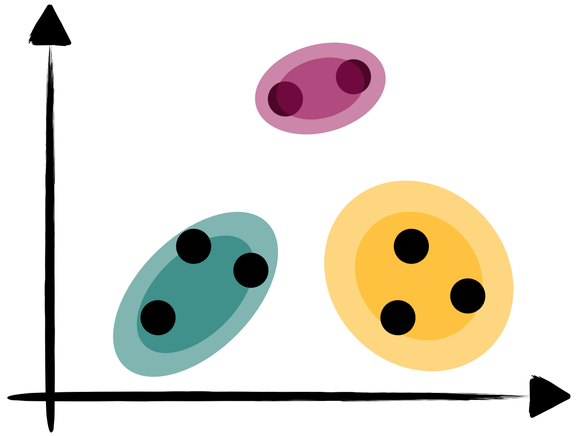


https://thispersondoesnotexist.com

Example: K-Means Clustering

Assumption:

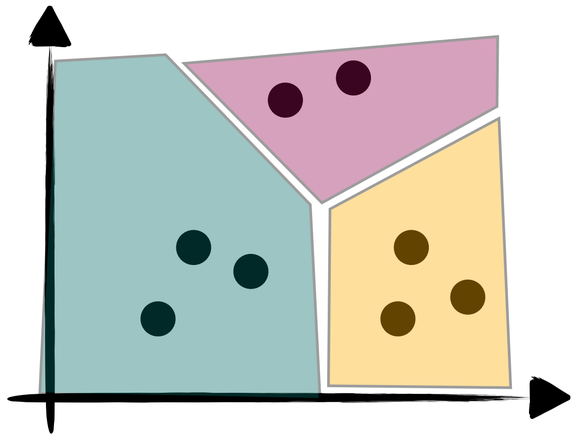
$p(x) = \sum_k w_k p_k(x).$



Learning:

fit w_k w/ some assumptions for $p_k(x)$

Goal: once fit, we can assign a x
to a cluster (e.g. by max. likelihood)



Learning Objectives

Unsupervised learning is more heterogeneous than supervised learning.

Many architectures with their own loss functions

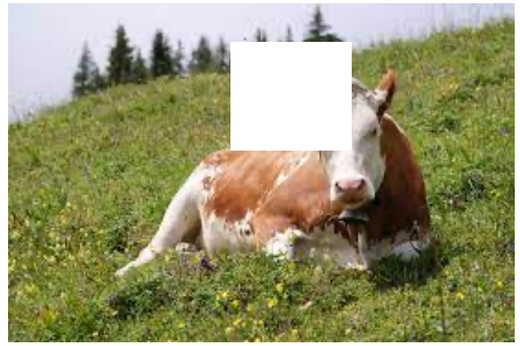
- Often losses constructed to prove either exact convergence to $p(x)$ or by formulating bounds to it

Special Case: “Self-supervised” Learning

One way to do unsupervised learning is to use the data itself to come up with new supervised tasks

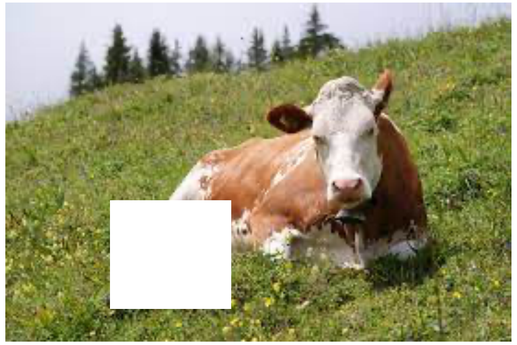
data x_1

label x_2



data x_1

label x_2



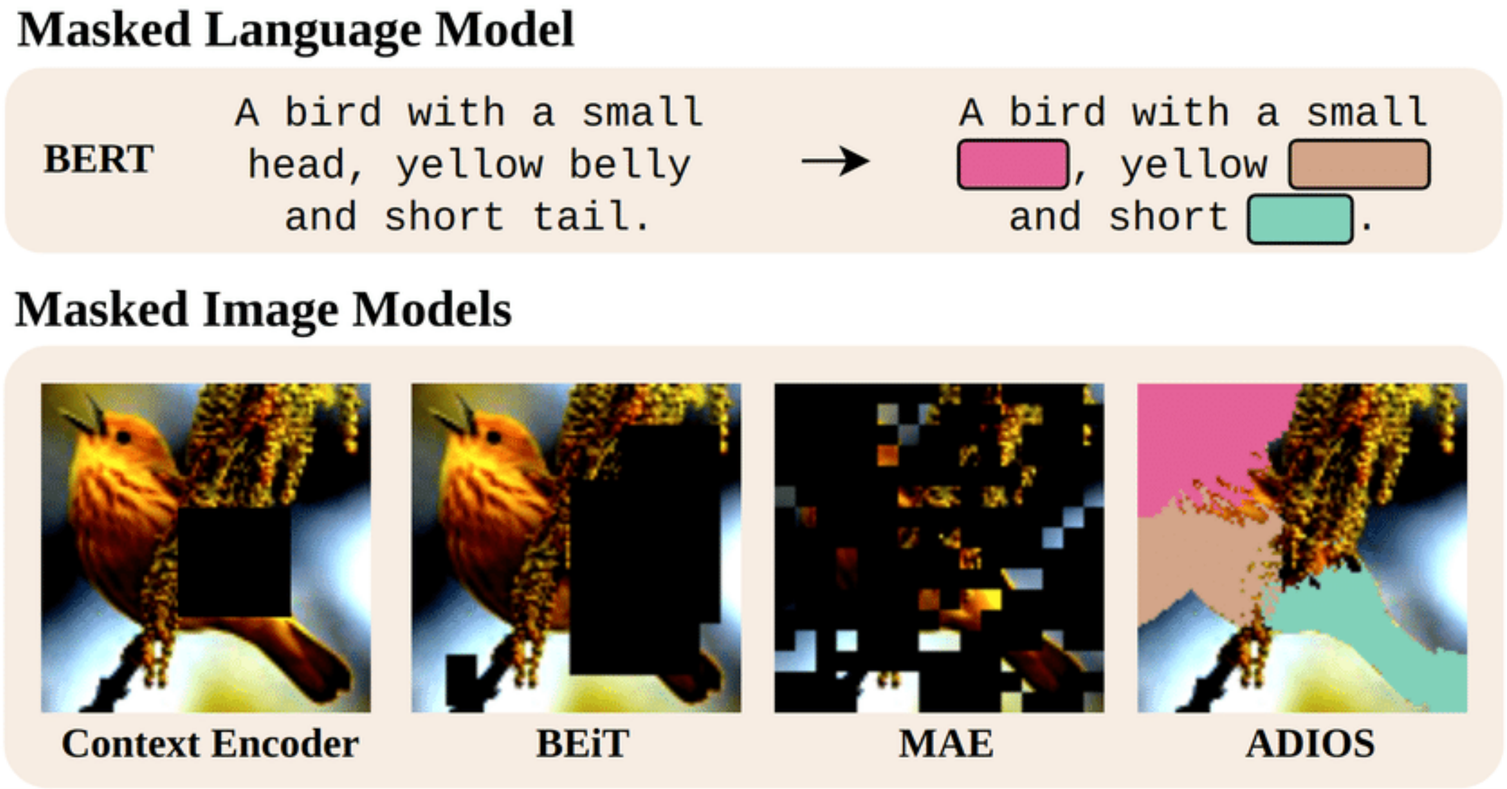
data x_1

label x_2

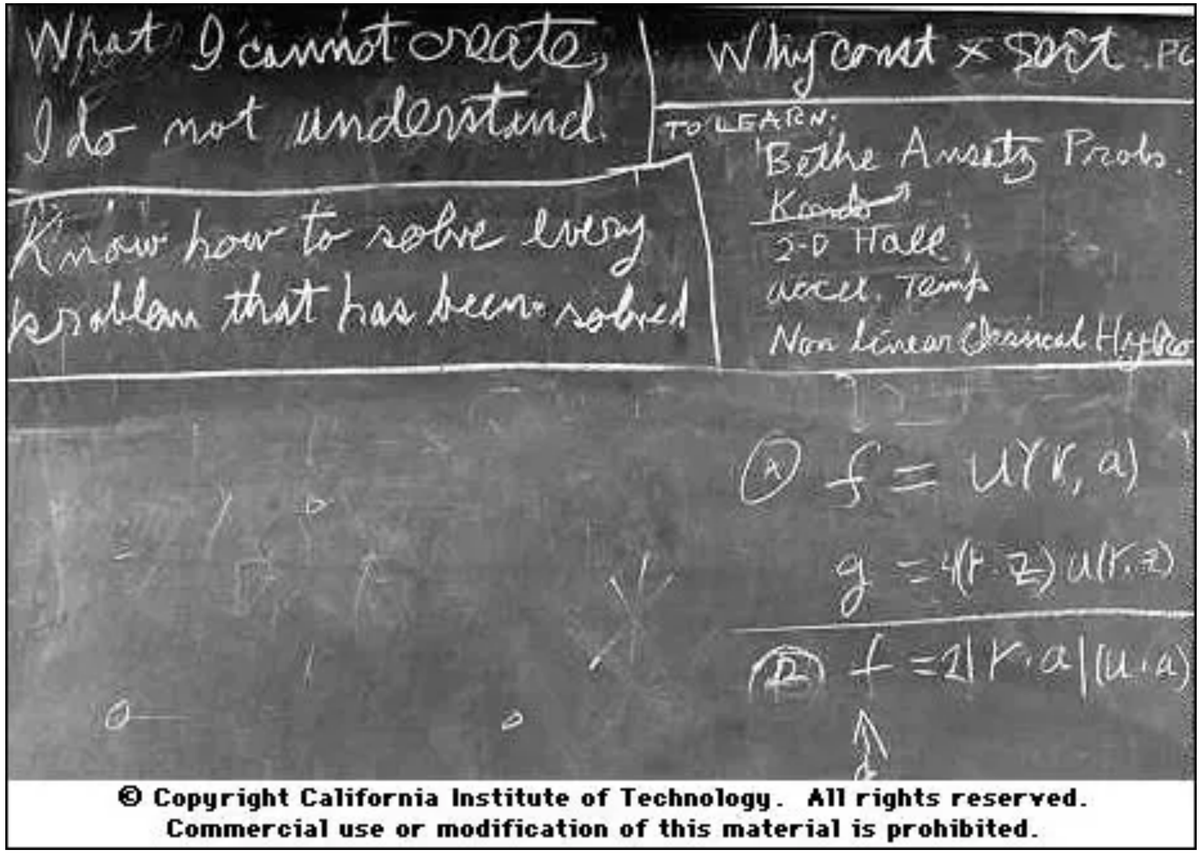
and many more possibilities…

Special Case: “Self-supervised” Learning

We can also recover our “supervised” setup by splitting the data arbitrarily into a observed and a label part $x \rightarrow (x_1, x_2)$



Special Case: “Self-supervised” Learning

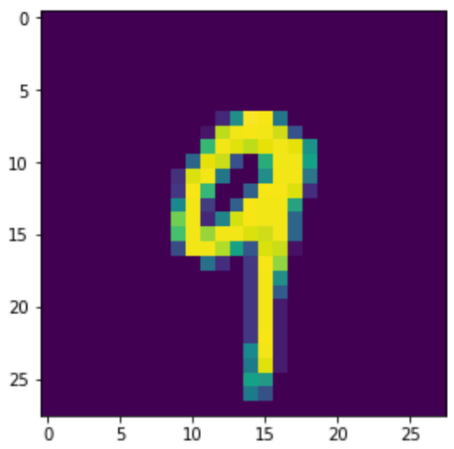


What I cannot create, I do not understand -Feynman

Special Case: “Self-supervised” Learning

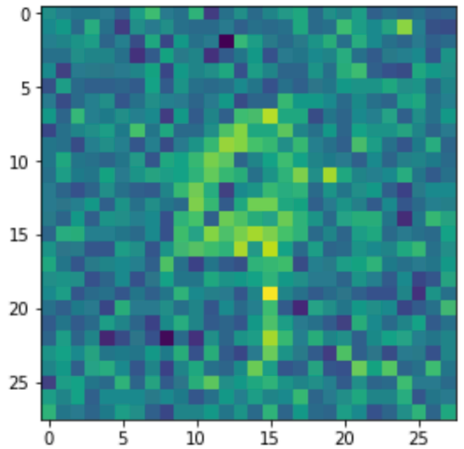
More ways to generate “ad-hoc” inference task from the data.

One alternative: Noising and denoising



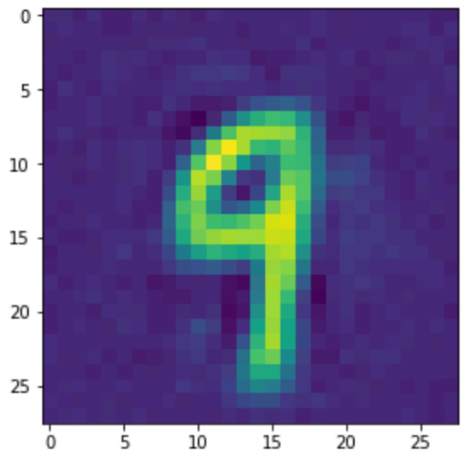
x

add noise



$\tilde{x}$

add noise



$q_\phi(x \mid \tilde{x}) \approx p(x \mid \tilde{x})$

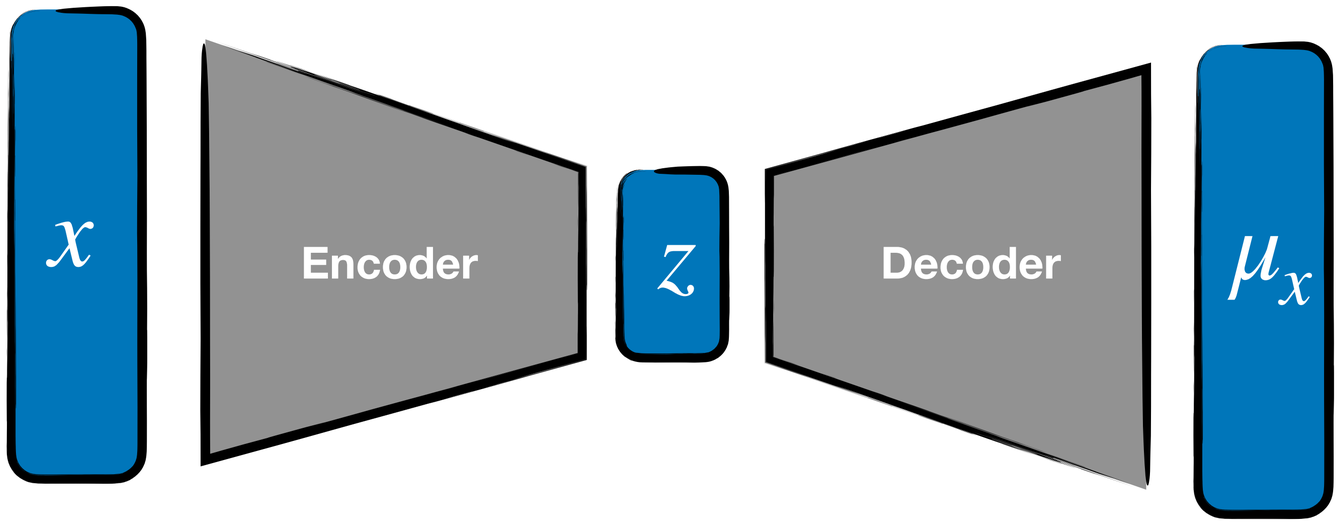
Some Philosophy

When we're learning to see, nobody's telling us what the right answers are — we just look. Every so often, your mother says "that's a dog", but that's very little information. You'd be lucky if you got a few bits of information — even one bit per second — that way. The brain's visual system has 10^{14} neural connections. And you only live for 10^9 seconds. So it's no use learning one bit per second. You need more like 10^5 bits per second. And there's only one place you can get that much information: from the input itself. — Geoffrey Hinton, 1996 (quoted in [Gor06]).

Example: Autoencoder

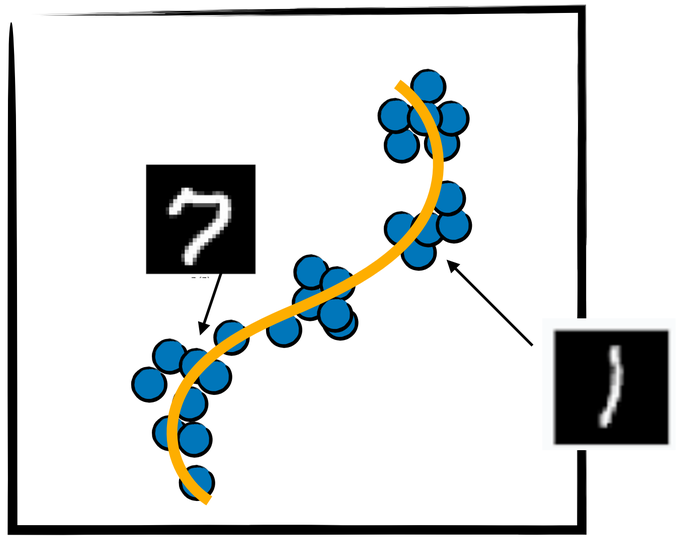
A prototypical example of self-supervised learning. **Idea:** even if we don't know them, probably there *are* only a few "label"-like degrees of freedom.

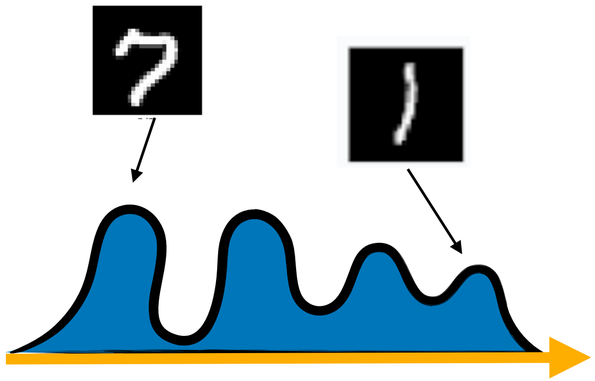
Should be able to compress data into much smaller representation, without much loss of information



Example: Autoencoder

E.g. for MNIST. Most 28x28 array **do not** look like x. The actual data lives on a low dimensional manifold.

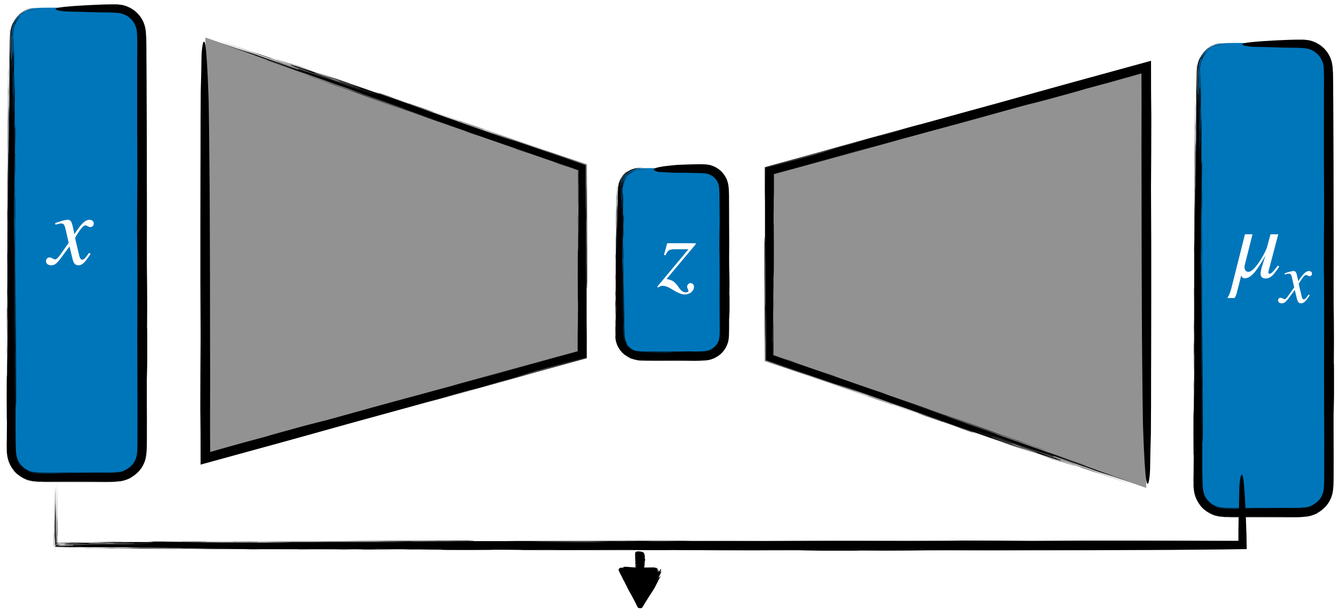




Autoencoder learns a coordinate system of the manifold (the coordinates may or may not be meaningful to us)

Example: Autoencoder

The way to train an autoencoder is to see how well it can reconstruct the original input based on the low-dimensional encoding

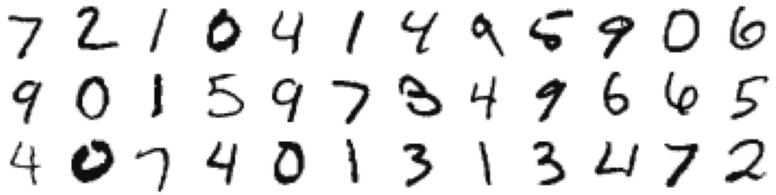


$$\mathbb{E}_x L(x, f_D(f_E(x)) = (x - (f_D(f_E(x)))^2$$

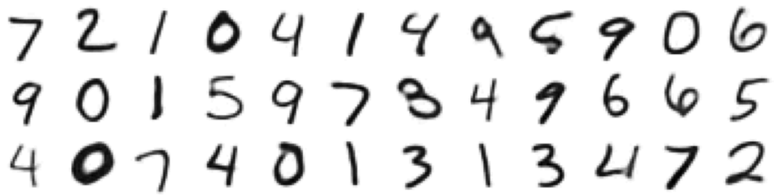
All the tricks apply

All the architectural tricks can be used at an appropriate place within the unsupervised context

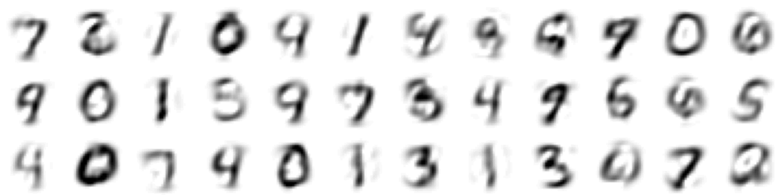
X (original samples)



$g \circ f(X)$ (CNN, $d = 16$)



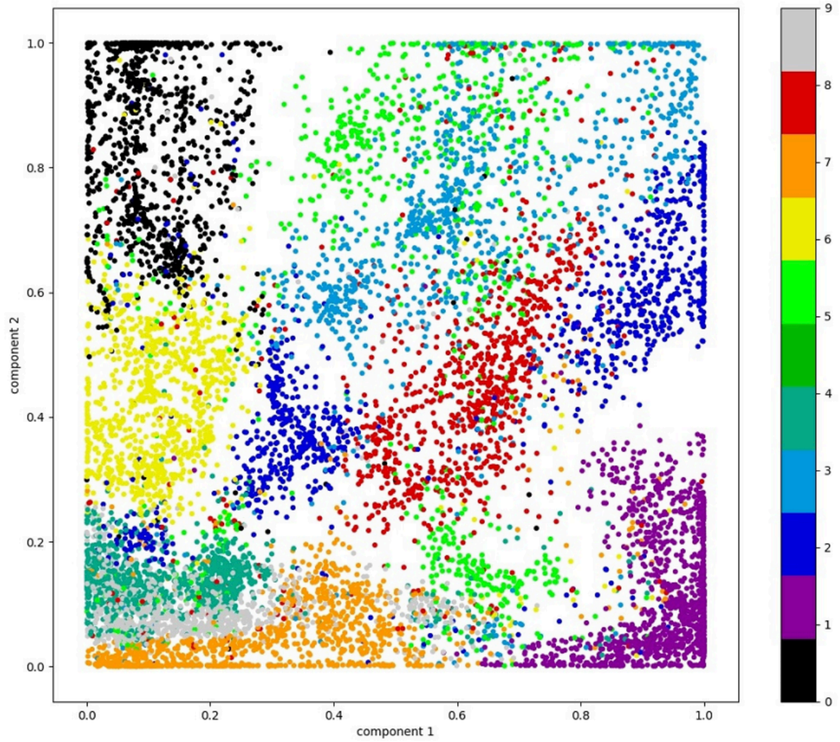
$g \circ f(X)$ (PCA, $d = 16$)



Credit: [Francois Fleuret]

The Latent Space

We can see how a neural net arranges the dataset within the latent coordinates



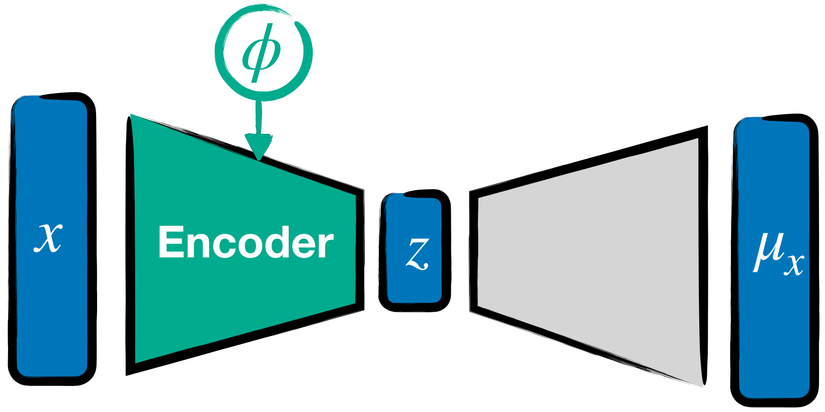
It does seem to pair similar instances (numbers) together, but

the network picks its own arrangement generally not under our control

Self-supervised Pretraining

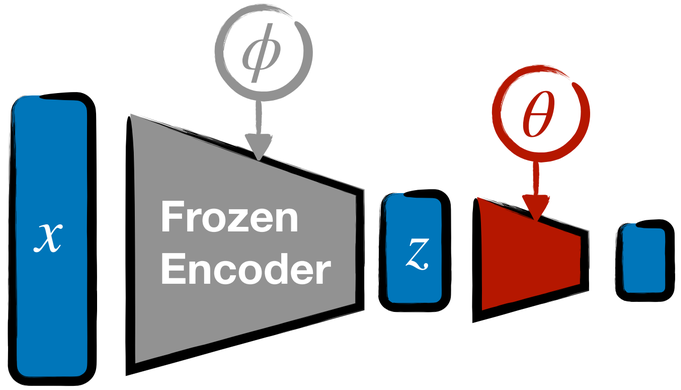
A key use-case for e.g. unsupervised training is to create good representations for data for future supervised tasks

First train unsupervised



Lots of unlabeled data

… use learned representation to train on supervised model on top

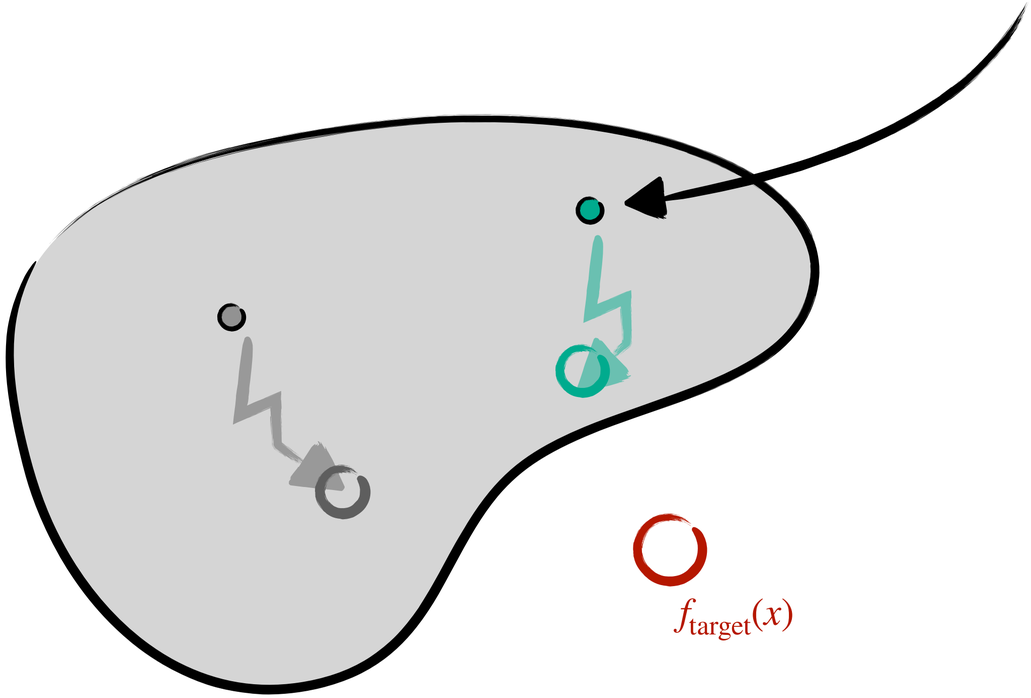


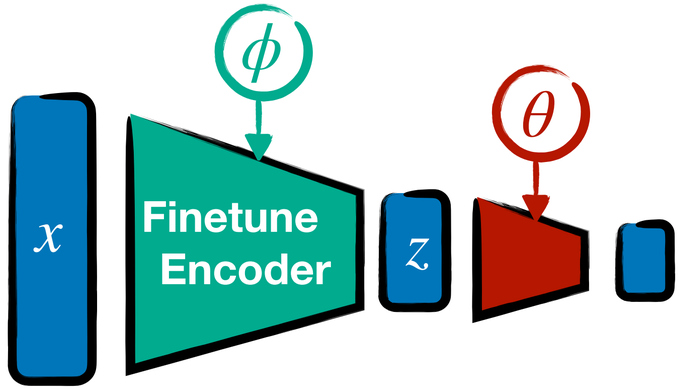
Little labeled Data

Finetuning

Because everything is differentiable, we can even fine-tune the base model parameters for the supervised task

- unsupervised pre-training as **good initialization**



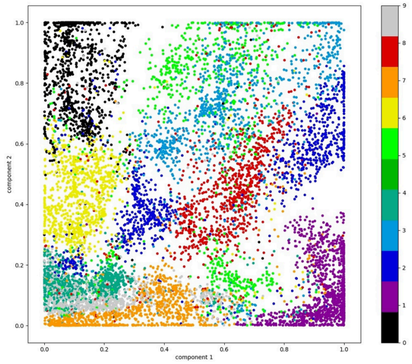


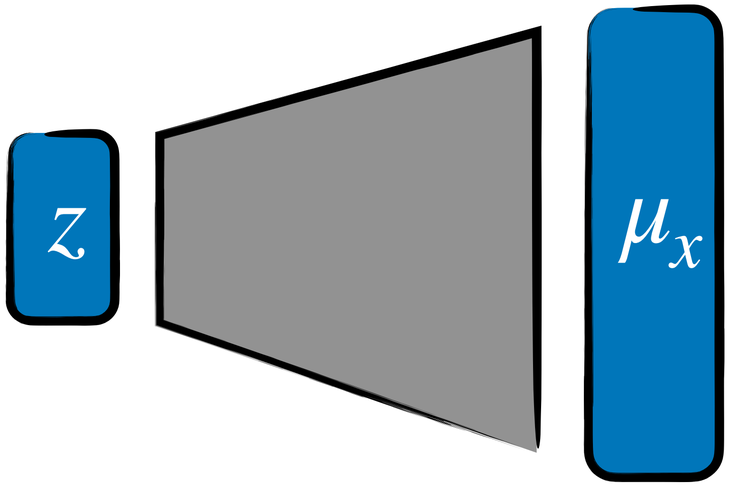
Little labeled Data

What if?

The encoded space could be useful for something else, too

If we did *know* how the data were distributed, we would have a very clear way to generate more data





$$z \sim p(z)$$

$$x \sim p(x \mid z)$$

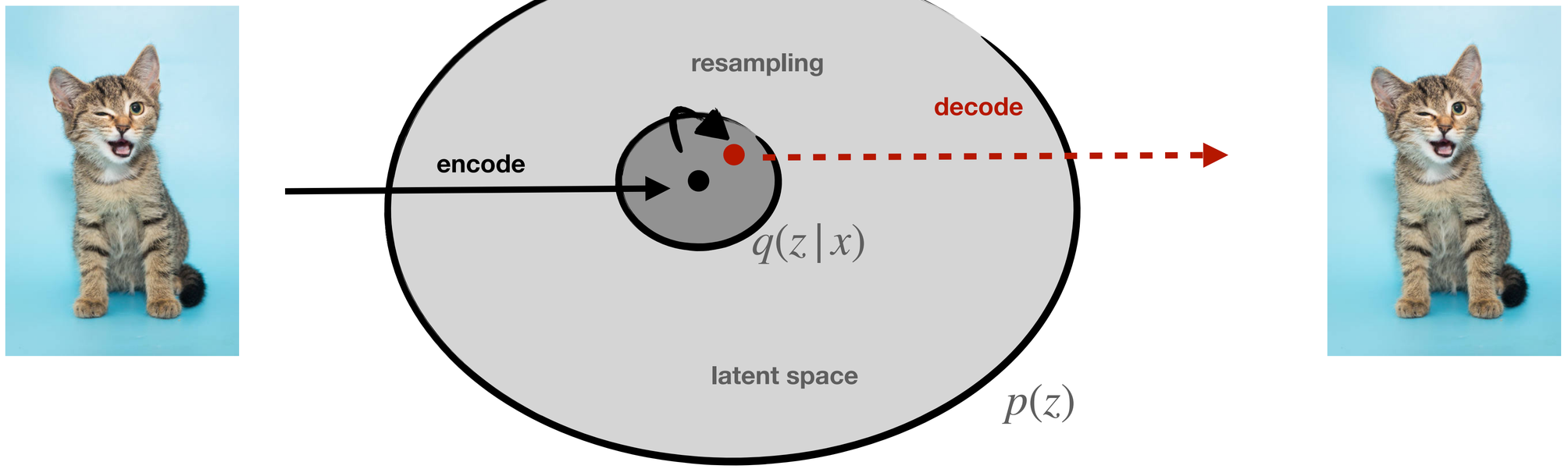
Variational Autoencoder

Variational Autoencoders make changes to the standard Autoencoder setup to force a “more tame” latent space

$$L_{\text{VAE}} = L_{\text{reco}} + \beta L_{\text{latent}}$$

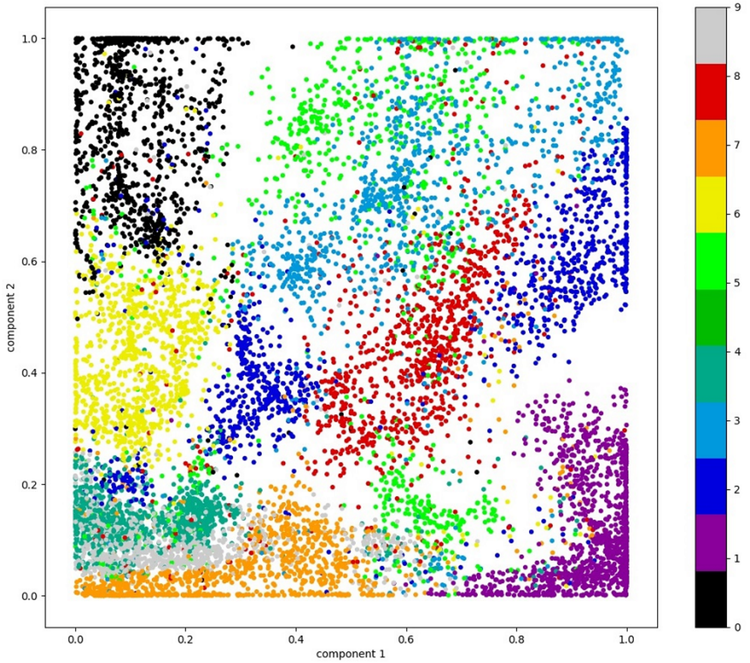
- change encoder to be stochastic $z = f(x) \rightarrow q(z \mid x)$
- add KL term to shape latent term to be Gaussian

Variational Autoencoder

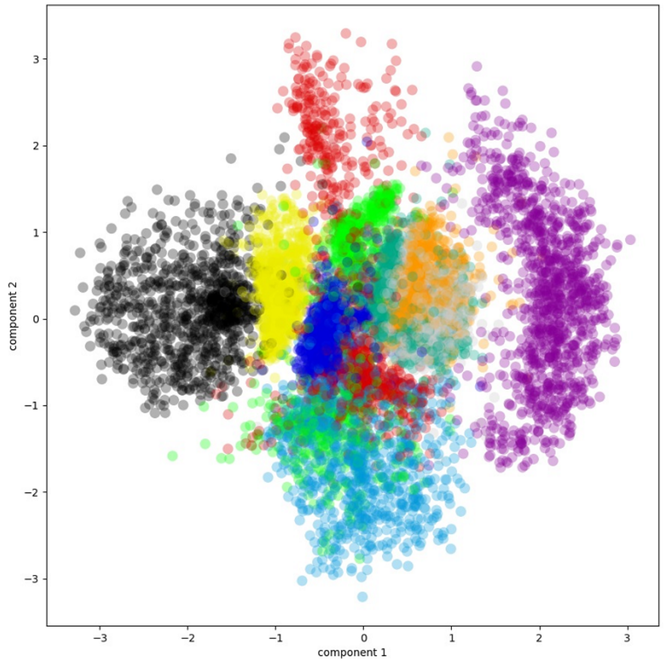


ensures smoothness in latent space (close-by points) must decode to similar images

Variational Autoencoder



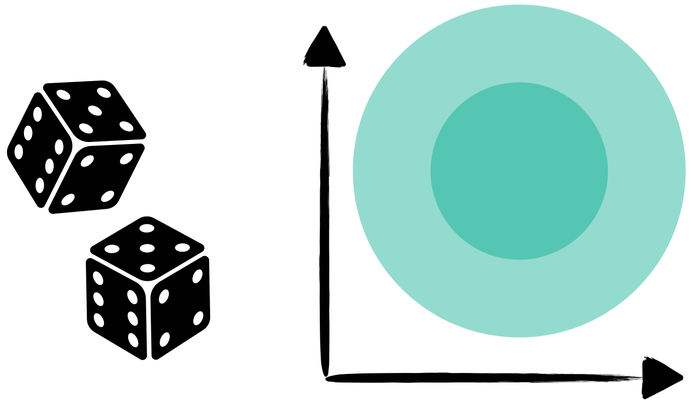
Standard Autoencoder for MNIST

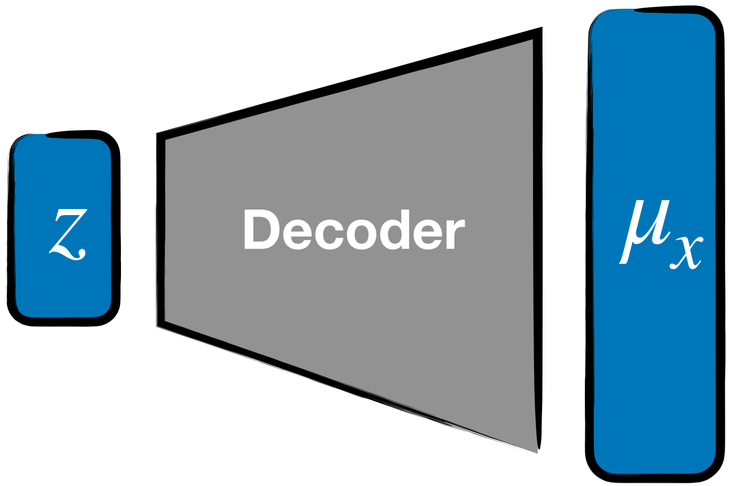


Variational Autoencoder

Generative Model w/ VAE

If we did *know* how the data were distributed, we would have a very clear way to generate more data



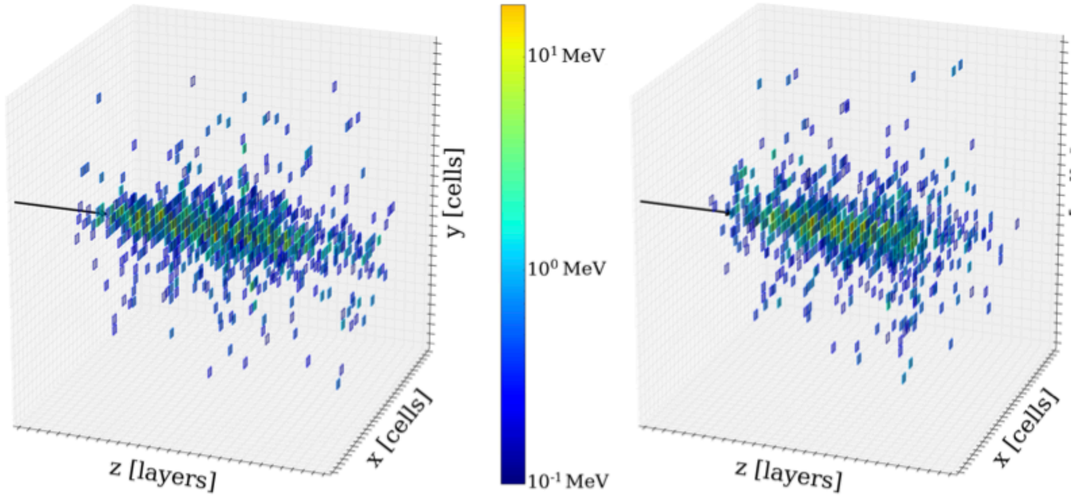


$$z \sim p(z)$$

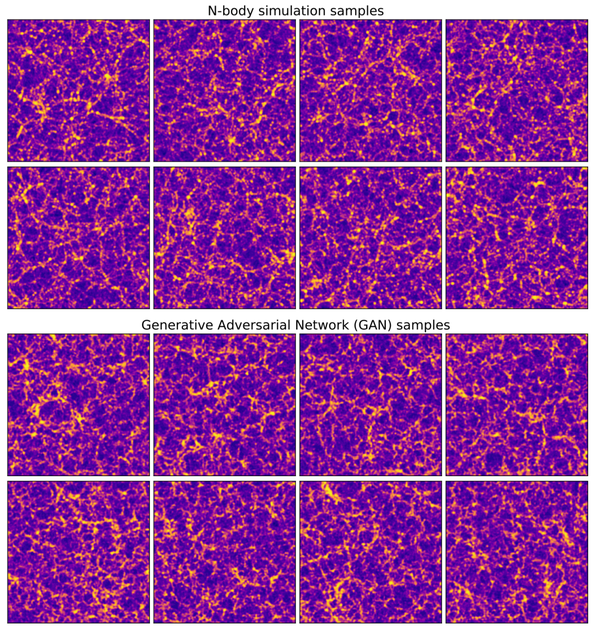
$$x \sim p(x \mid z)$$

Generative Models in Physics

A **key application** of generative models is to have fast, approximate simulation, that otherwise would be prohibitively expensive



1801.09070



1801.09070

Exact Likelihood Models

Often it's useful to access both ends of the unsupervised learning spectrum: **sampling and exact likelihood**

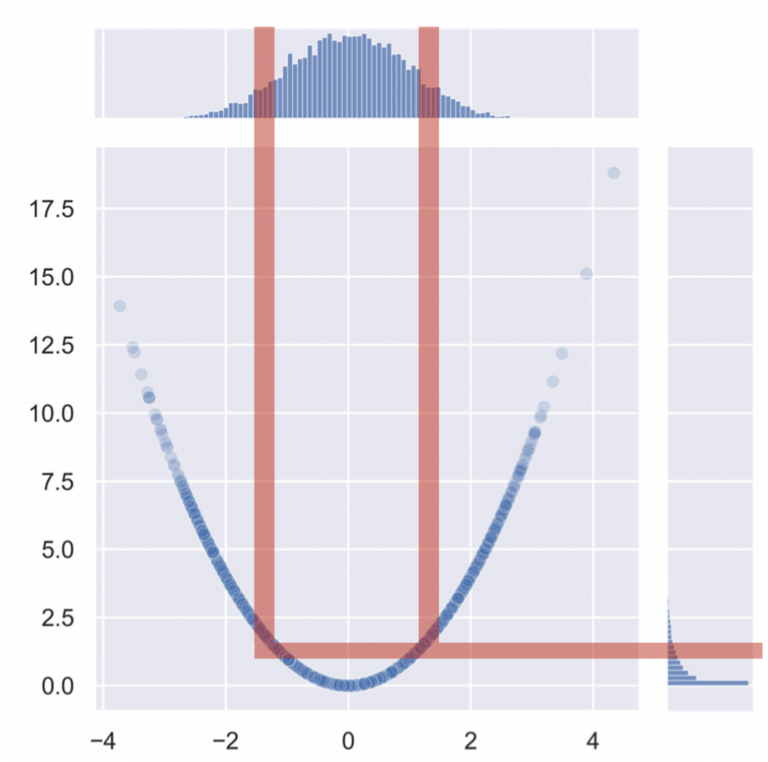
$$x \sim p(x) \qquad \log p(x)$$

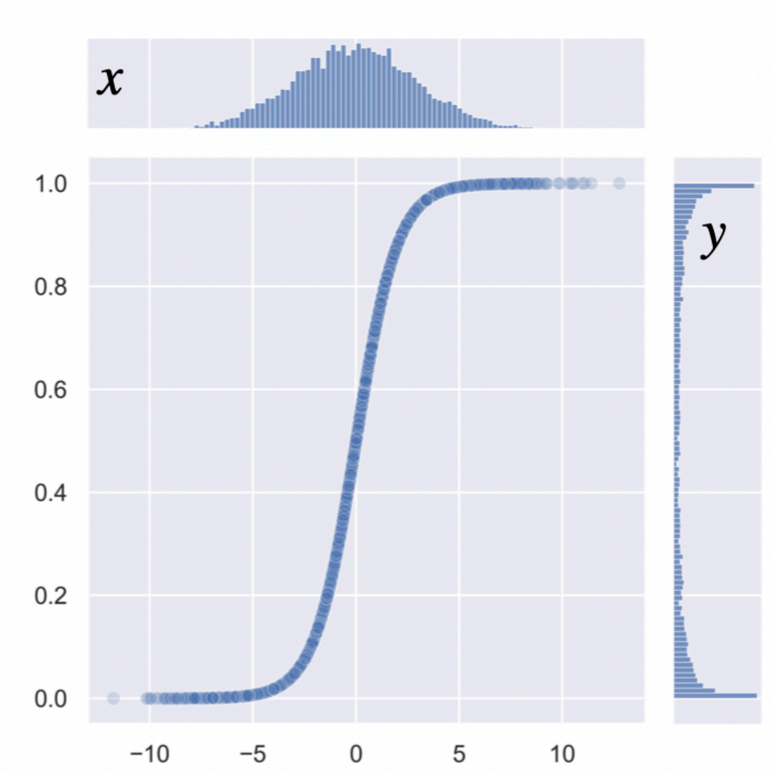
e.g. for statistical analysis (MLE estimates & faithful samples)

Requires dedicated architecture

Normalizing Flows

We can create complex densities from simple ones by passing samples through complicated functions



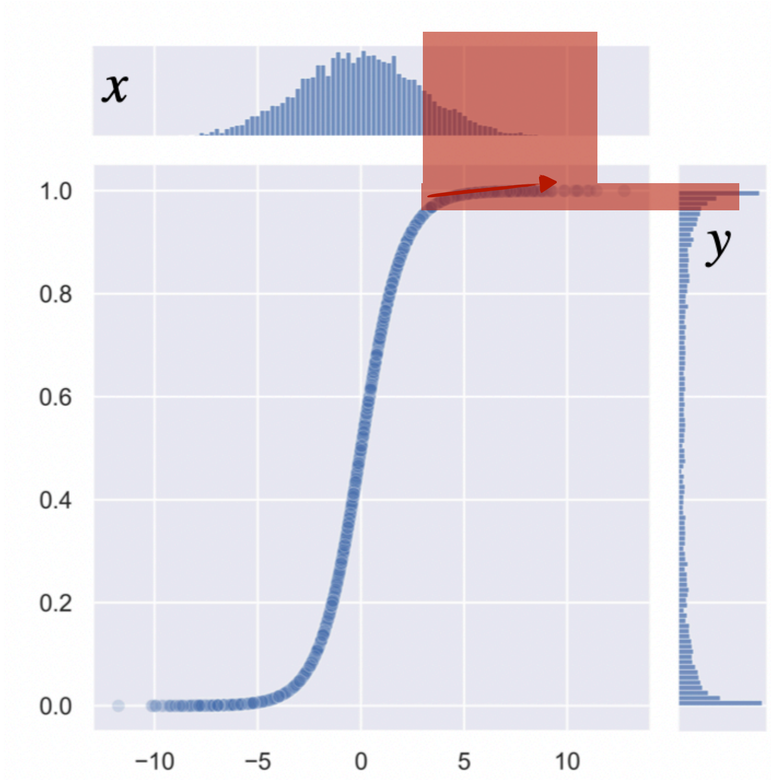


Bijections

If the function is bijective we can **also** evaluate the density of a new sample

Change of Variable Formula

$$p(y) = \frac{1}{f'(x)} p(f^{-1}(y))$$



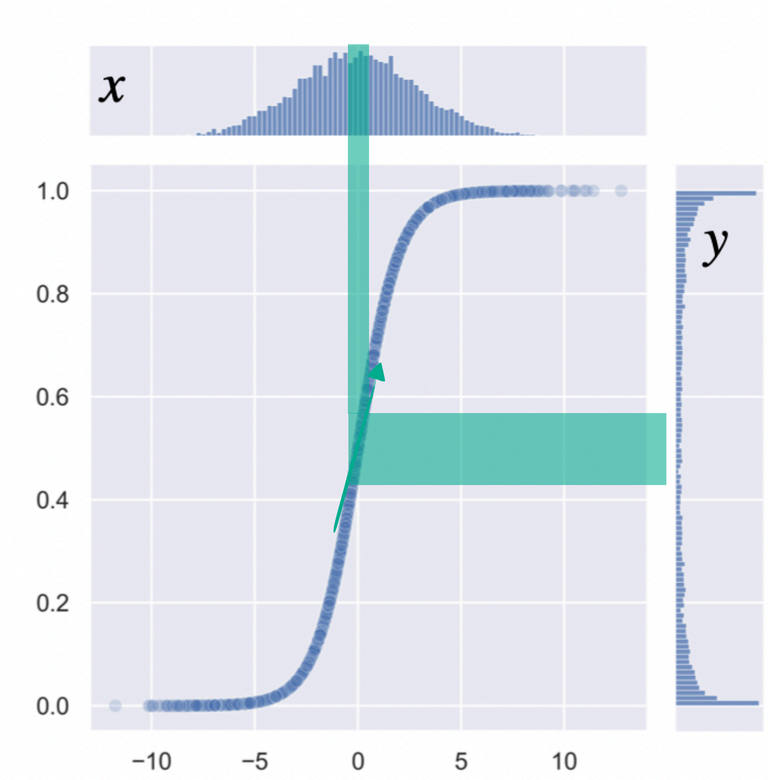
Low derivative → Large increase in density

Bijections

If the function is bijective we can **also** evaluate the density of a new sample

Change of Variable Formula

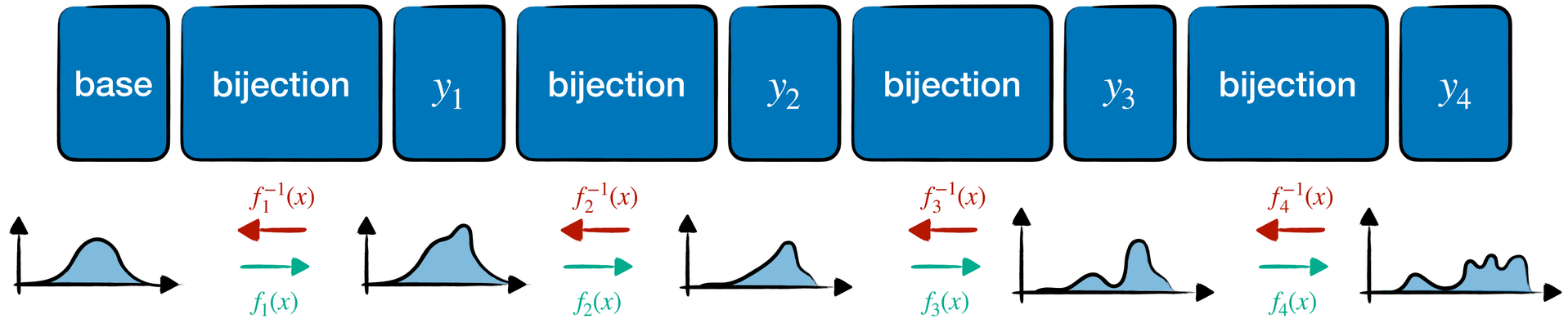
$$p(y) = \frac{1}{f'(x)} p(f^{-1}(y))$$



High derivative → Large decrease in density

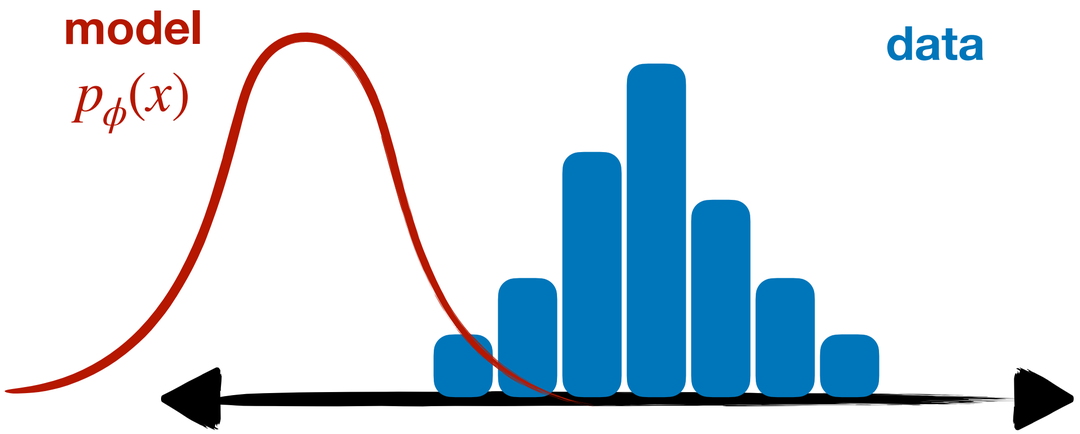
Normalizing Flows

Normalizing Flows take this idea and apply the Deep Learning mantra: compose, compose, compose

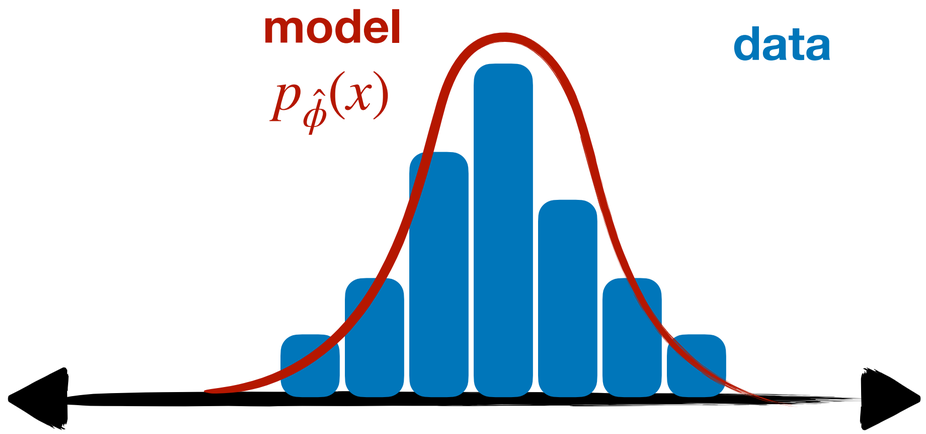


Training Flows on Data

Training of Normalizing Flows then proceeds via standard maximum likelihood (minimum NLL) loss



before the fit

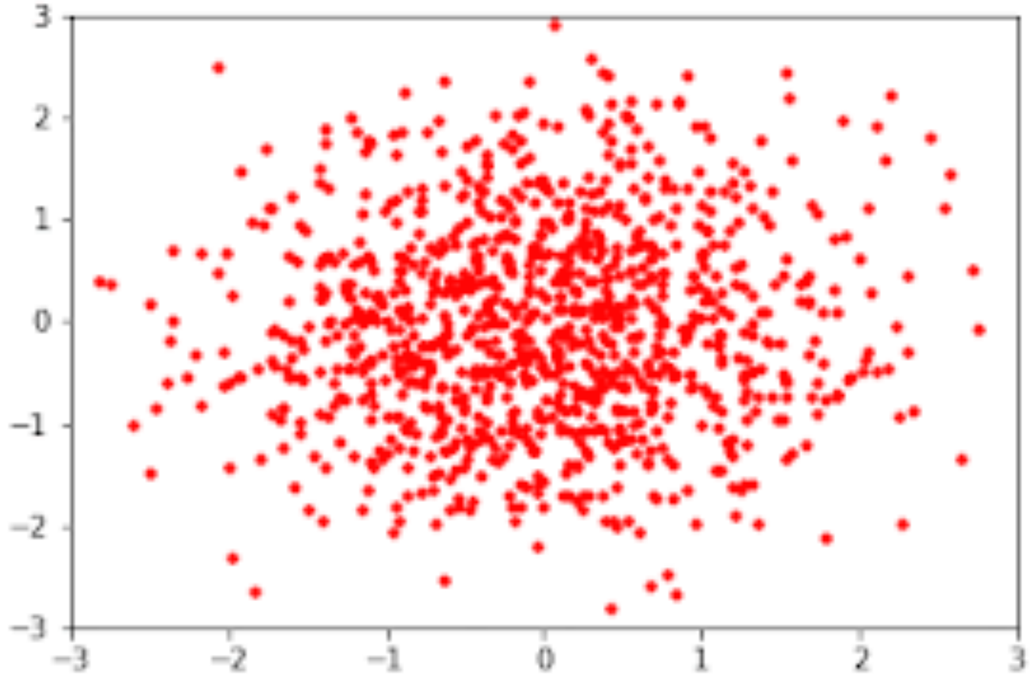


after the fit

$$\log_\phi p(X) = \sum \log p_\phi(x_i)$$

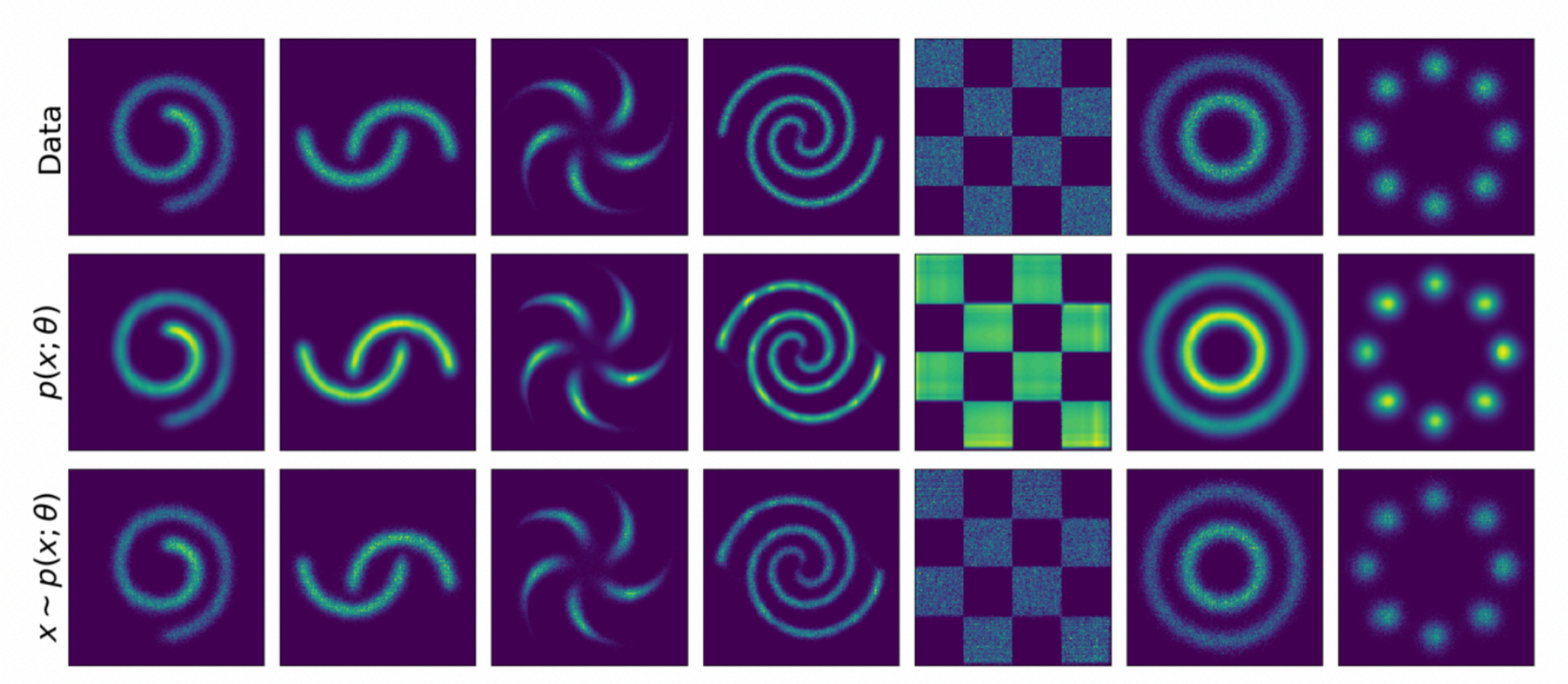
Example

A intuitive idea, but the trick is to get expressive enough bijections, while maintaining bijectivity & tractable Jacobians



One approach: coordinate wise transform (RealNVP)

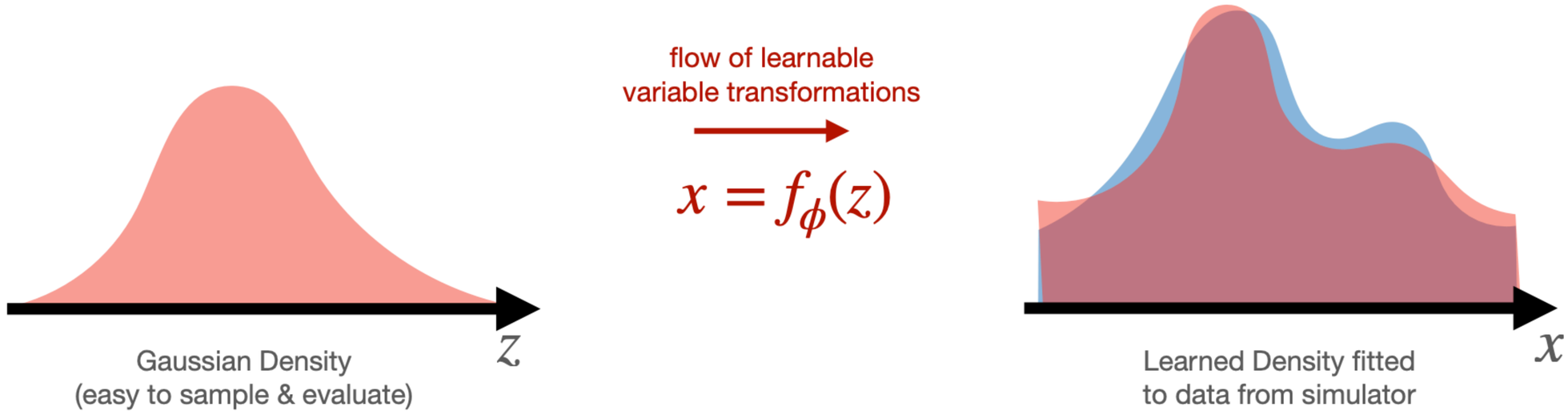
Modern Flows can be very expressive



Applications

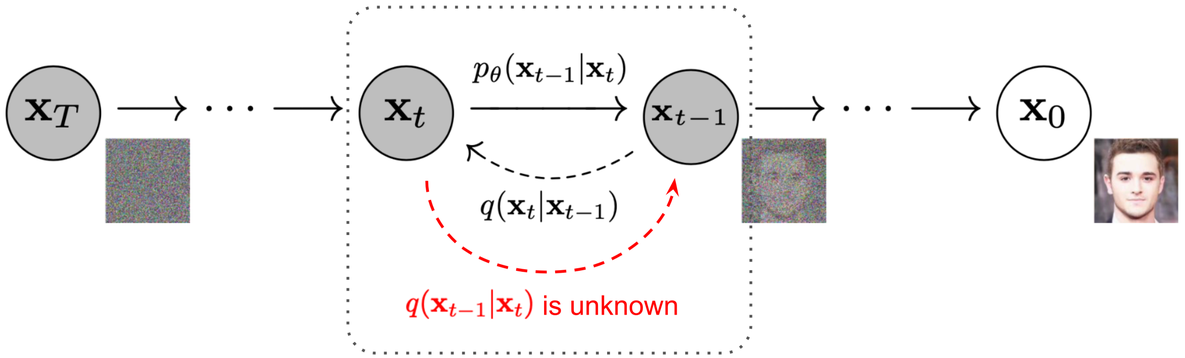
e.g. as proposal distributions in **importance sampling**

$$\mathbb{E}_p[f(x)] \int_\Omega p(x)f(x) = \int_\Omega q_\phi(x)\frac{p(x)}{q_\phi(x)}f(x) = \mathbb{E}_q[w(x)f(x)]$$

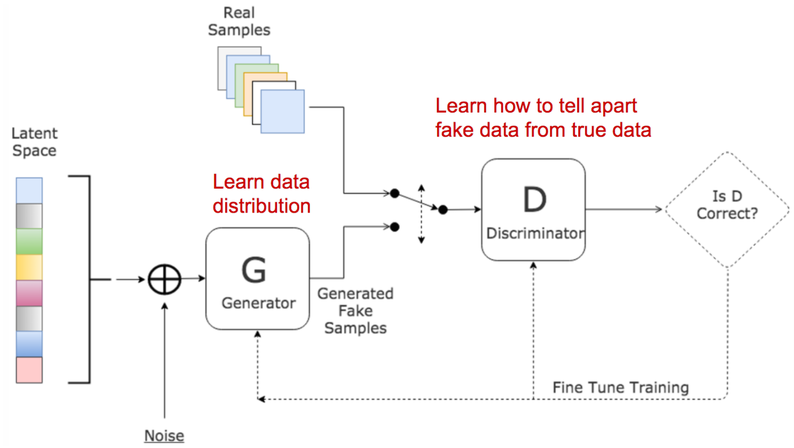


Many more things, we can't cover

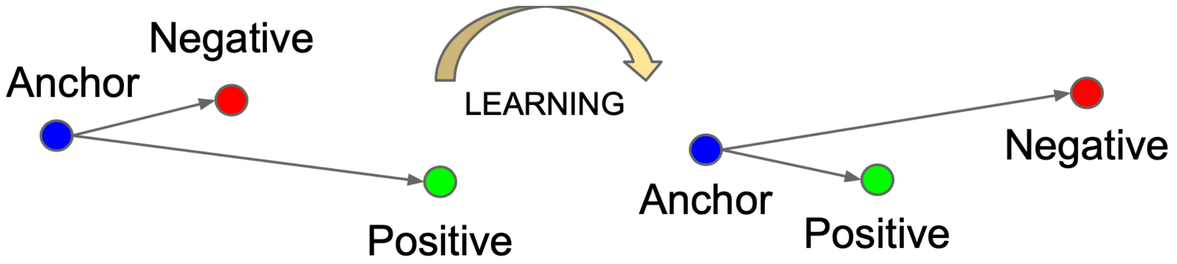
Diffusion Models



Generative Adversarial Networks



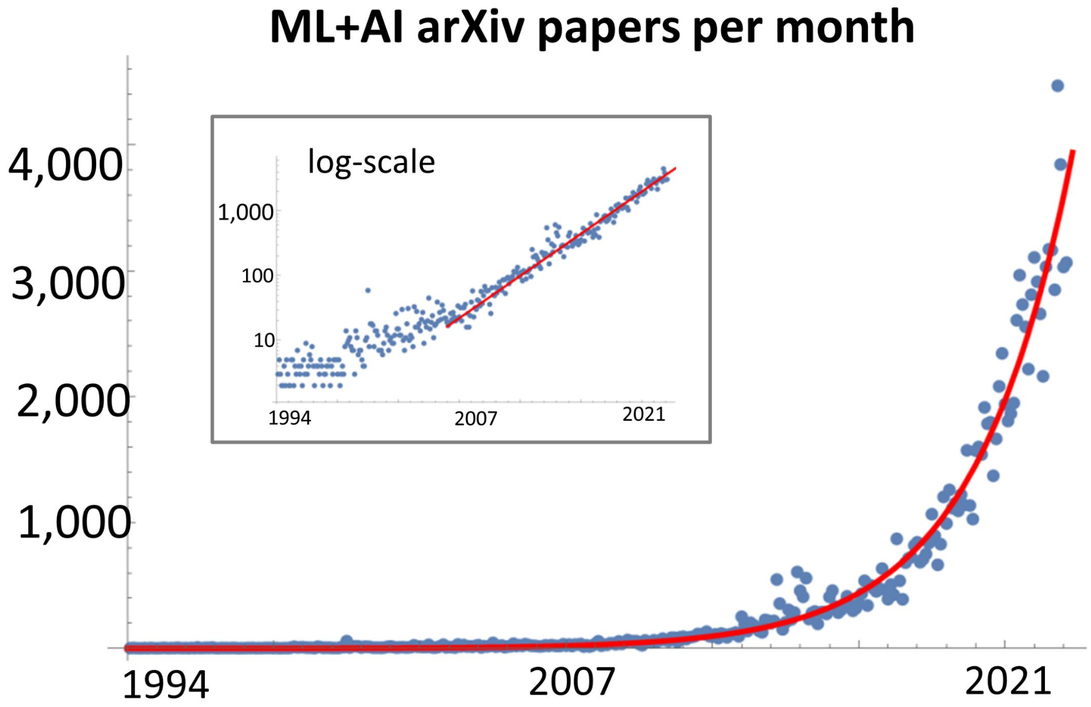
Contrastive Learning:



Closing Thougthts

There is a lot of work going on

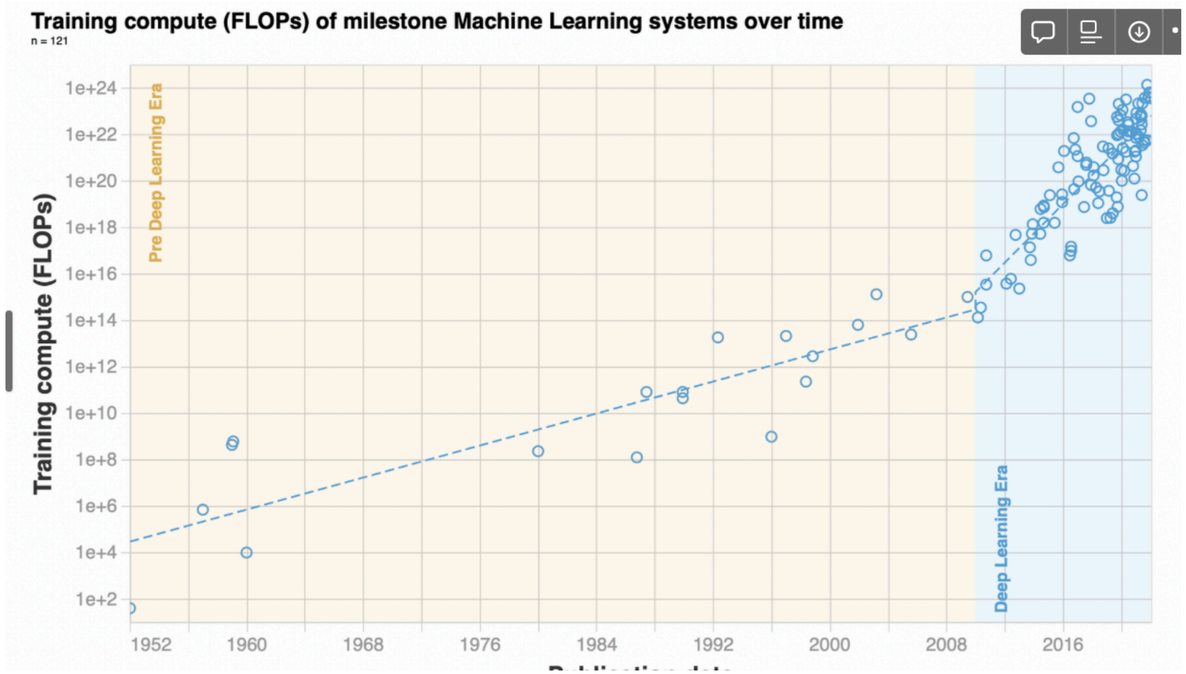
We could really only touch very briefly on the most basic ideas. The field itself is exploding since ~ a decade



Why now?

"Connectionist" ML is a old idea going back to the original percepton, but for many years it just did not work

Since "AlexNet" 2012, a breakthrough



The New York Times

Scientists See Promise in Deep-Learning Programs

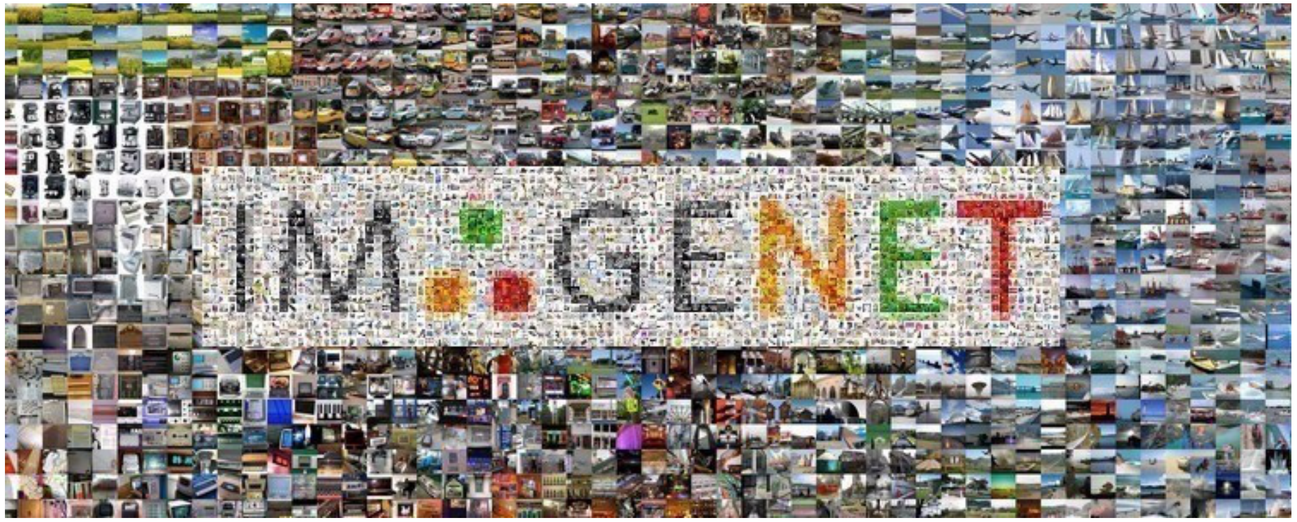
Give this article

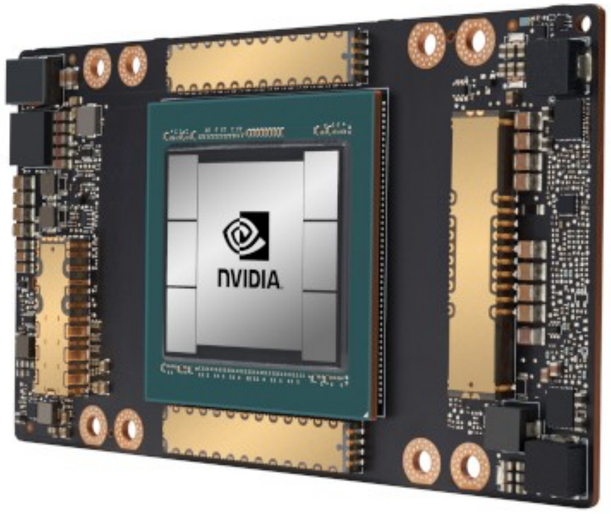
A voice recognition program translated a speech given by Richard F. Rashid, Microsoft's top scientist, into Mandarin Chinese. Hao Zhang/The New York Times

By John Markoff

Why now?

- sufficiently big (web-scale) data (thanks CERN!)
- parallel processings with GPUs
- relentless improvements in small details (initialization, optimizers, regularization, etc…)





Recent Years: Scale is all you need

Neural networks have become extremely large.
Large Language Models: ~ Trillion Parameters

Performance is almost predictable through “scaling laws”

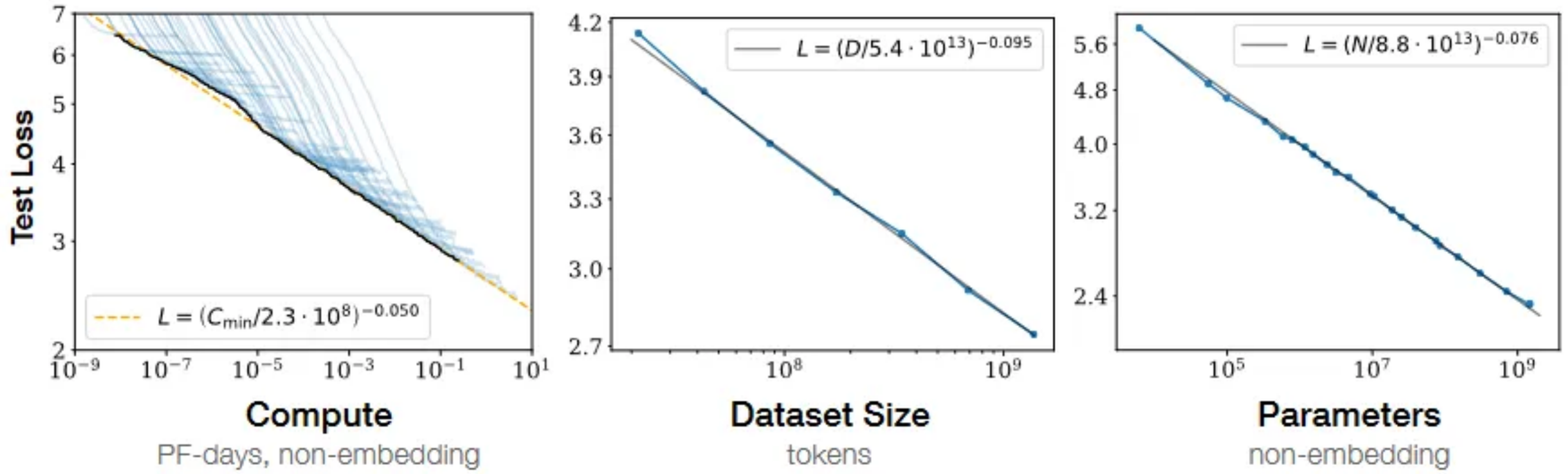
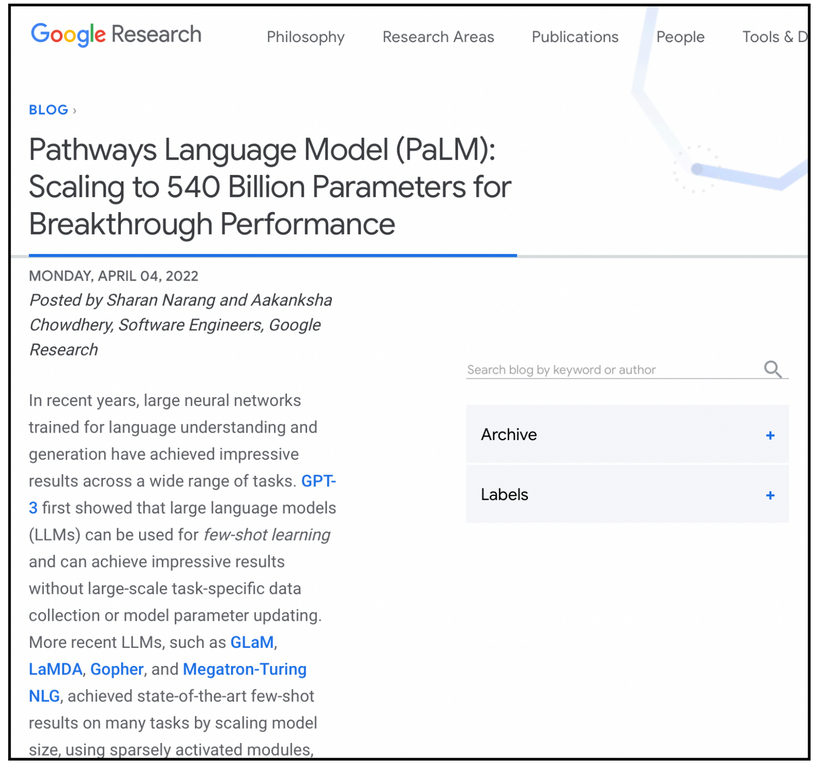


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when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

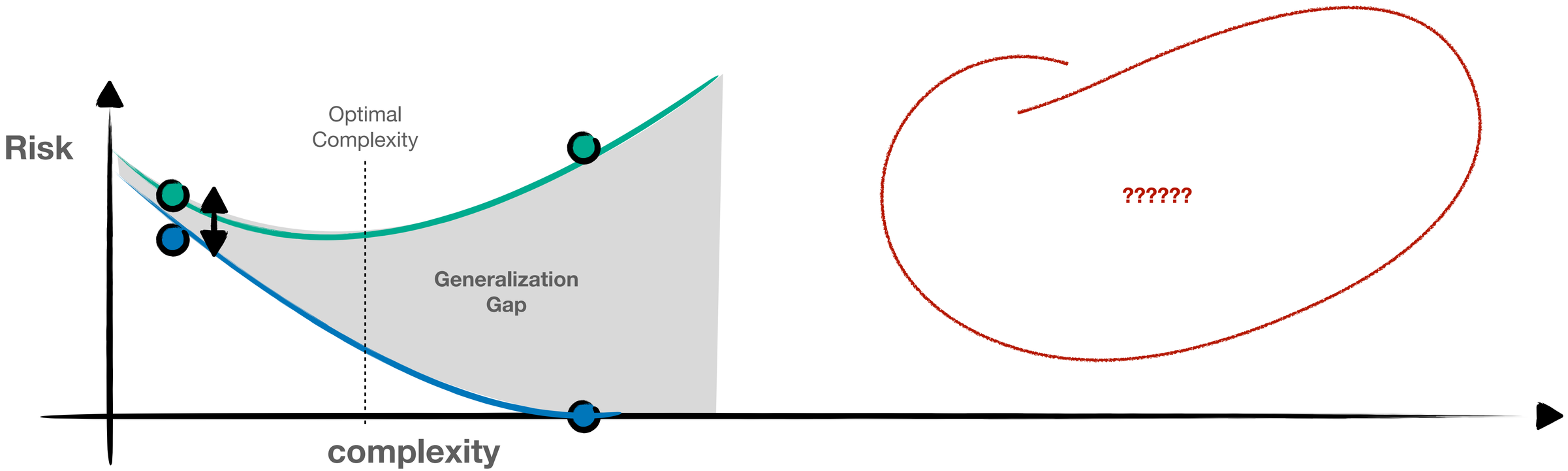
Wait a minute…

Billions or Trillions of parameters!

#pars >> #data points

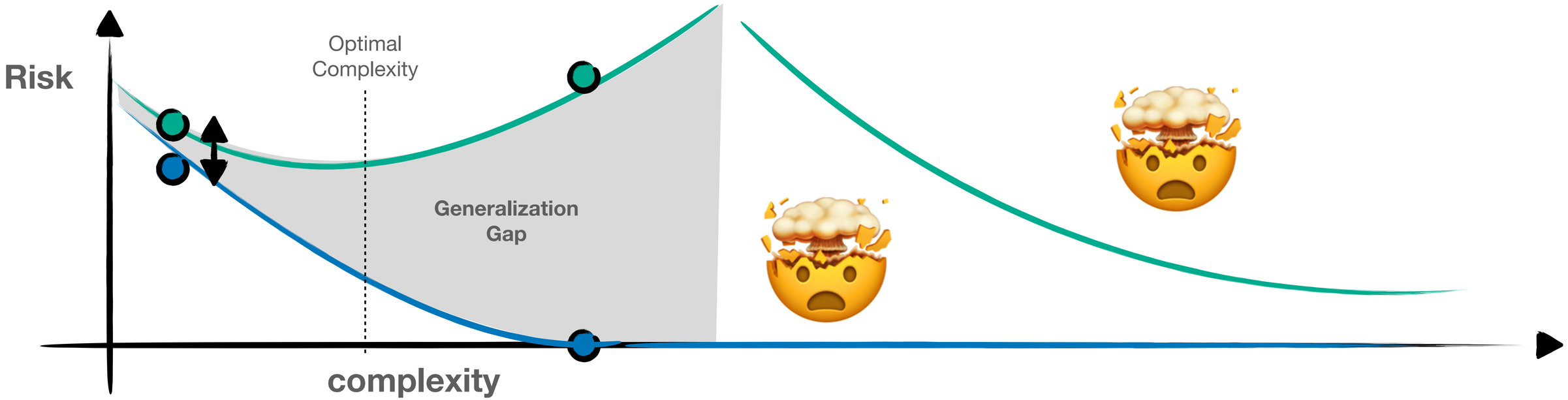


Deep Learning



Double Descent!

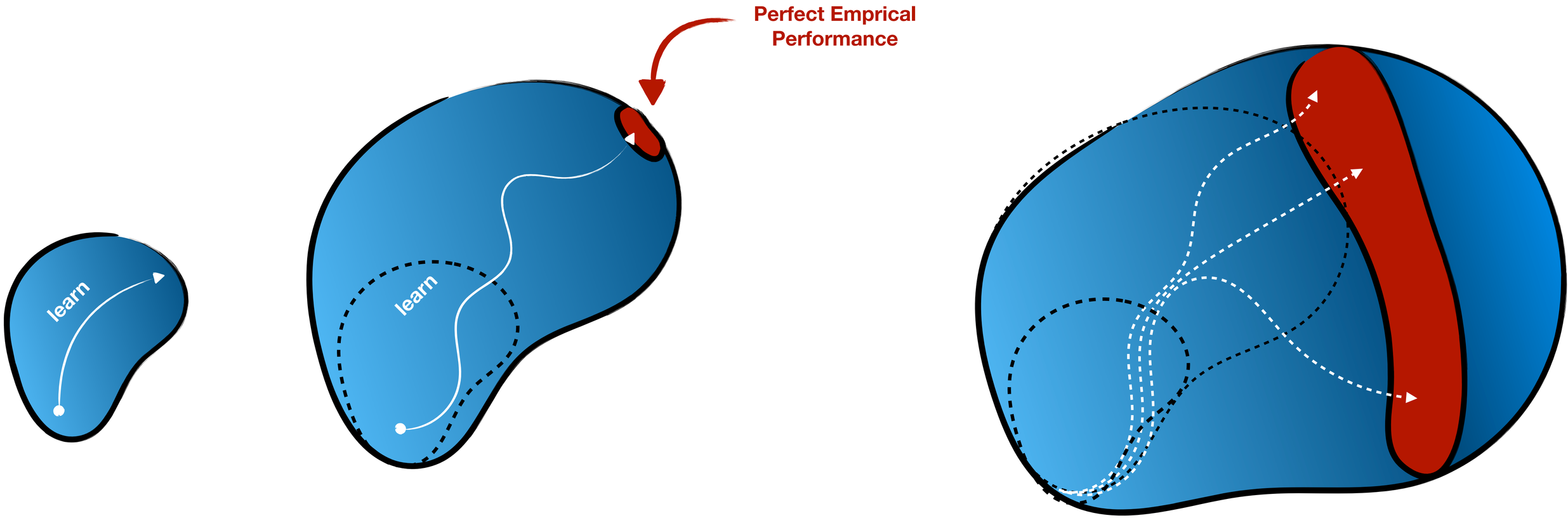
The Test Error can decrease again once you get to VERY big hypothesis sets



Modern Deep Learning Lives here

Understanding Double Descent

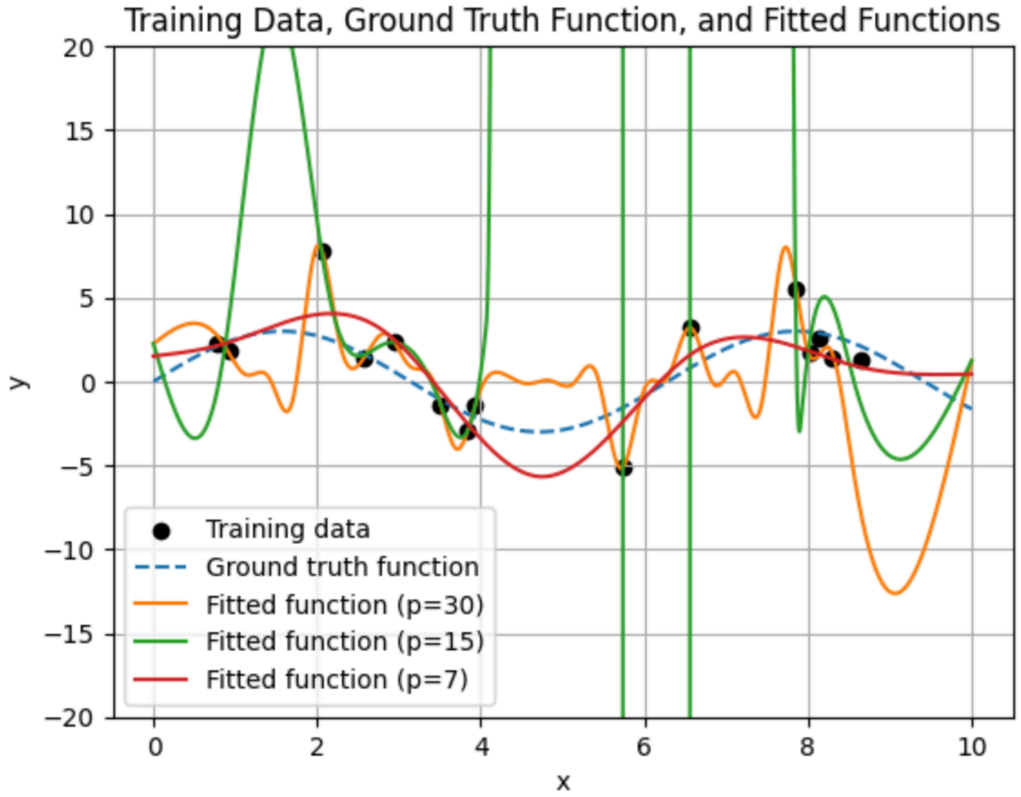
Double Decent might seem surprising, but it’s intuitive



If you increase the hypothesis set even more
a multitude of possible options appears - all with very good empirial performance

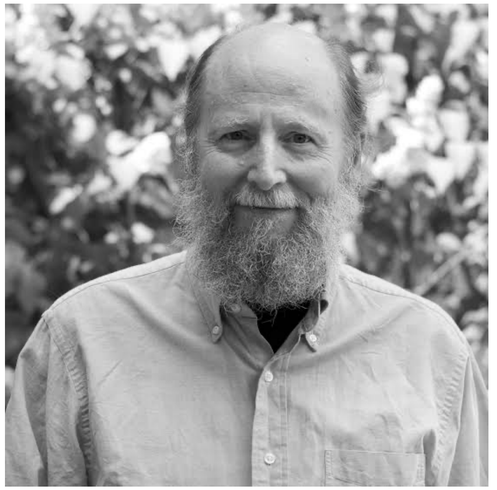
Double Descent in Spline Regression

It’s not a deep learning phenomenon, but appears for many overparametrized systems, e.g. spline regression



Notebook: [Notebook]: Double Descent w/ Splines]

The Bitter Lesson



Rich Sutton

March 13, 2019

Researchers seek to leverage their human knowledge […], but the only thing that matters in the long run is the leveraging of computation

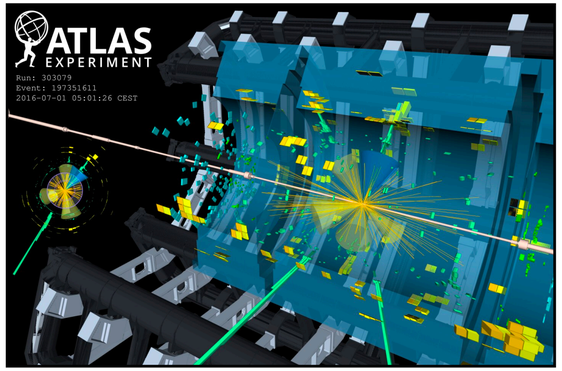
The biggest l…ethods that leverage computation are ultimately the most effectiv…, or rather its generalization of continued exponentially…nducted as if the computation available to the agent were constan… only ways to improve performance) but, over a slightly longer time than a typical research project, massively more computation inevitably becomes available. Seeking an improvement that makes a difference in the shorter term, researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of computation. These two need not run counter to each other, but in practice they tend to. Time spent on one is time not spent on the other. There are psychological commitments to investment in one approach or the other. And the human-knowledge approach tends to complicate methods in ways that make them less suited to taking advantage of general methods leveraging computation. There were many examples of AI researchers' belated learning of this bitter lesson, and it is instructive to review some of the most prominent.

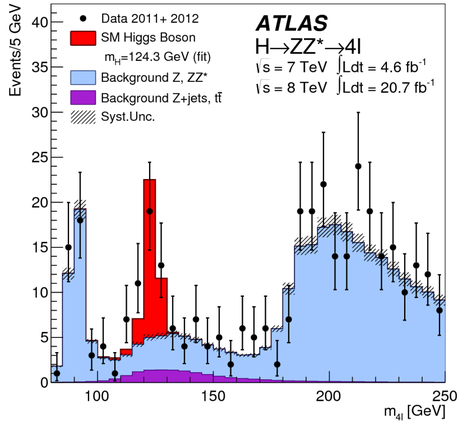
In computer chess, the methods that defeated the world cha…ep search. At the time, this was looked upon with dismay by the majority of …ds that leveraged human understanding of the special structure of chess. Whe…ware and software proved vastly more effective, these human-knowledge-base…hat ``brute force'' search may have won this time, but it was not a general stra…s. These researchers wanted methods based on human input to win and were disappointed when they did not.

… many examples of AI researchers’ belated learning of this bitter lesson

But also:

So far there is no indication, that a raw “Hits to Higgs” workflow could be learned. HEP has not had a AlphaFold / ChatGPT moment: more incremental improvements.





$\mathbb{R}^{10^8}$ → $\mathbb{R}^{1}$ →

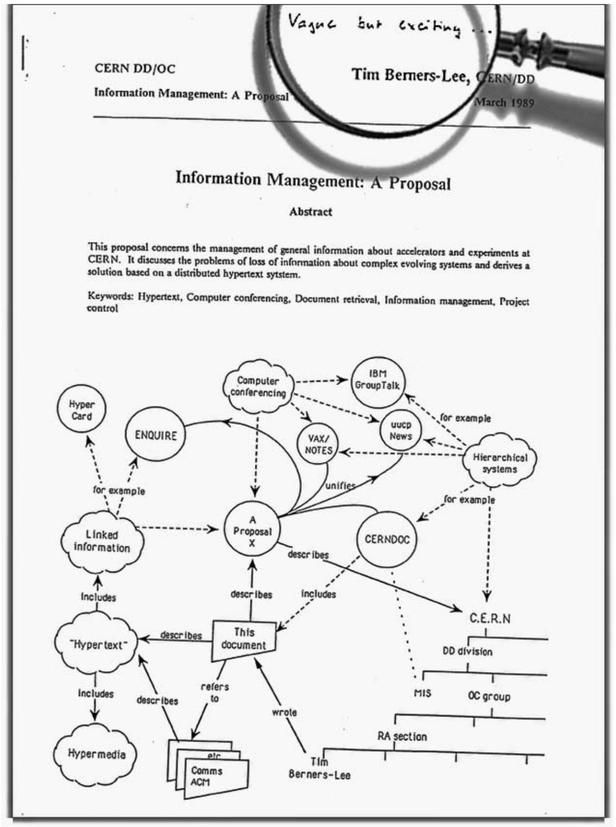
Also: would it be desirable? What does it mean to understand?

and…

“Salmon in River”

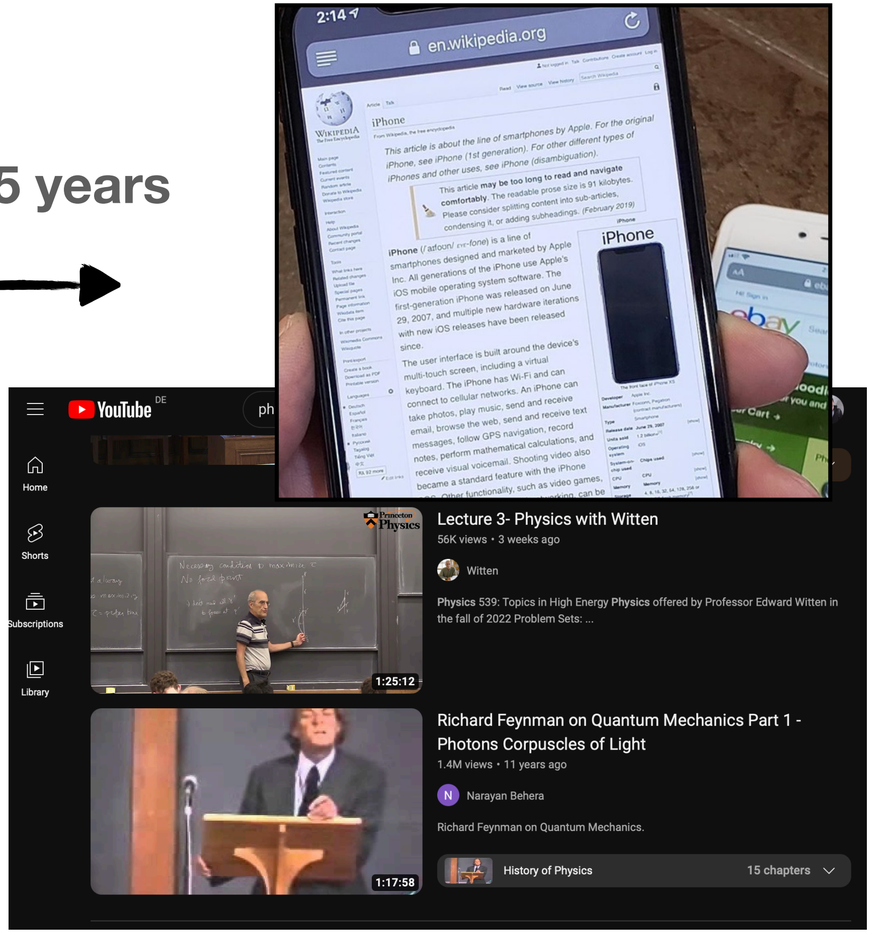
+ more serious Ethical Issues (see Savannah’s Talk)

1999: 10 years after Tim Berners Lee invented the World Wide Web

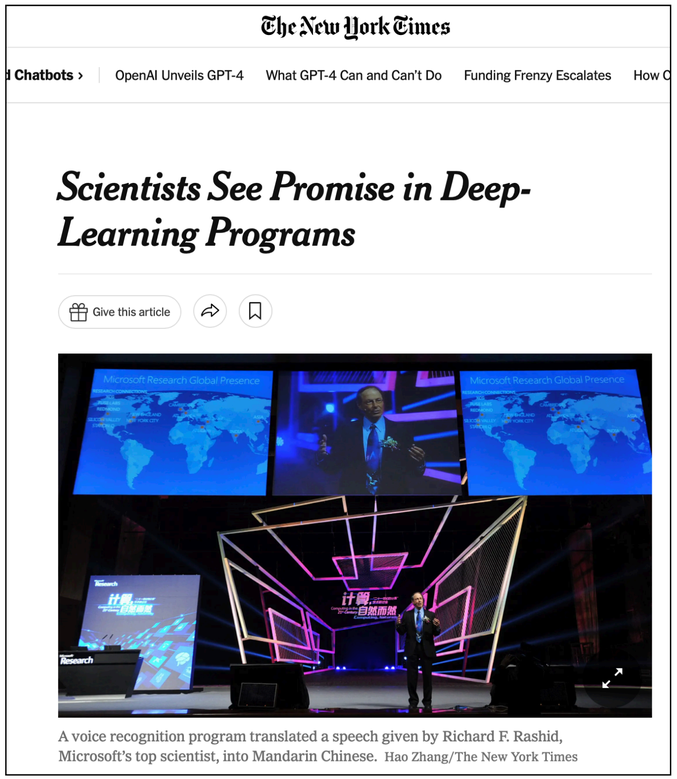


~10 years

~25 years



We’re still early, where will we be in 25y? very dynamic - difficult to predict.

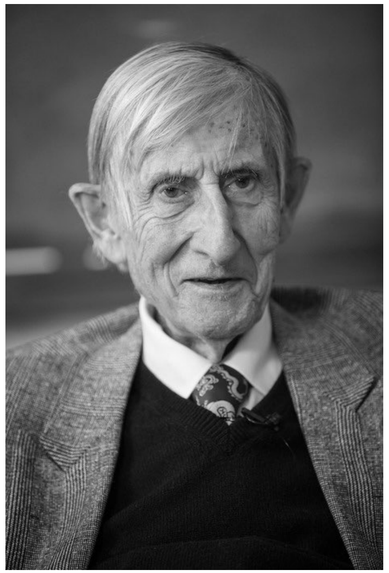


~10 years

~25 years

?

Bowie & Dyson



New directions in science are launched by new tools much more often than by new concepts. - *Dyson*

We are at the cusp of something exhilirating and terrifying
- Bowie on the Internet (1999)