Inductive bias and feature-learning, two challenges understanding DNNs

Ard Louis

UNIVERSITY OF OXFORD

1

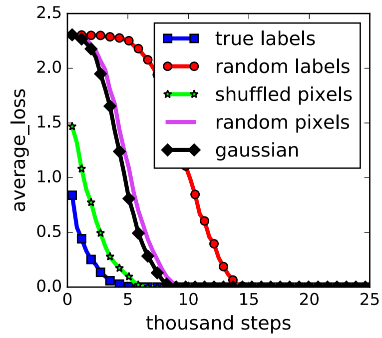
Inductive bias and feature learning

1. Two questions about generalisation
2. Inductive bias towards simplicity
3. Inductive bias and Zipf's law
4. Feature learning

William of Ockham
1287-1347

2

Expressivity: overparameterised DNNs can fit randomized data with zero error



Randomize labels on CIFAR-10

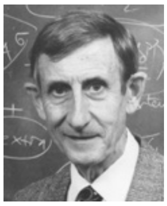
Understanding deep learning requires rethinking generalization, C. Zhang et al, arXiv:1611.03530
(5425 cites by March 2025)

Why do they learn and not just just memorise?

3

More parameters than data points is bad?

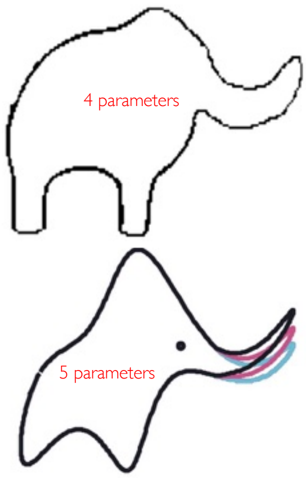
Enrico Fermi
1901-1954



Freeman Dyson
1923-2020

John von Neumann
1903-1957

With four parameters I can fit an elephant, and with five I can make him wiggle his trunk
-- John von Neuman (according to Fermi)

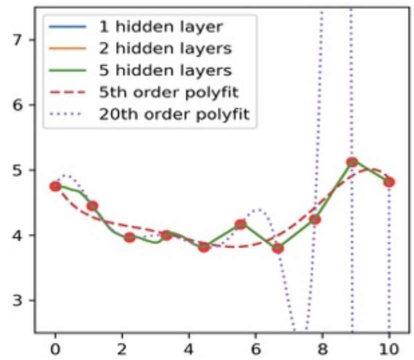


F. Dyson, *A meeting with Enrico Fermi,* Nature. **427**, 287 (2004)

Drawing an elephant with four complex parameters
Jürgen Mayer; Khaled Khairy; Jonathon Howard; American Journal of Physics 78, 648 (2010)

4

Central theoretical conundrum: why do DNNs choose the solutions that work?



polynomial fit : $y(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots a_n x^n$

compared to

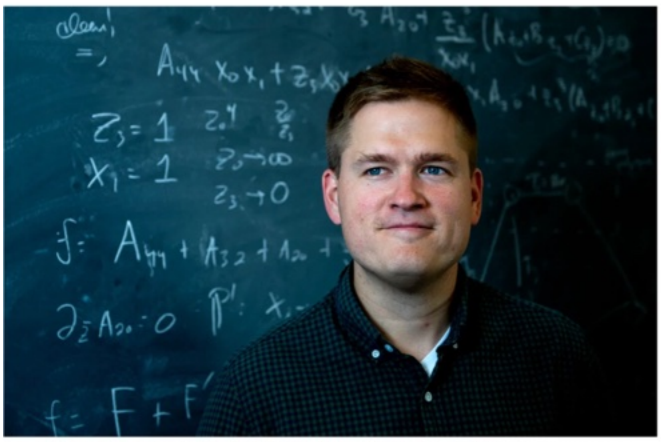
simple DNNs (FCN with layer width of 1000 units)

Inductive bias: why do overparameterized DNNs choose the solutions they do?

5

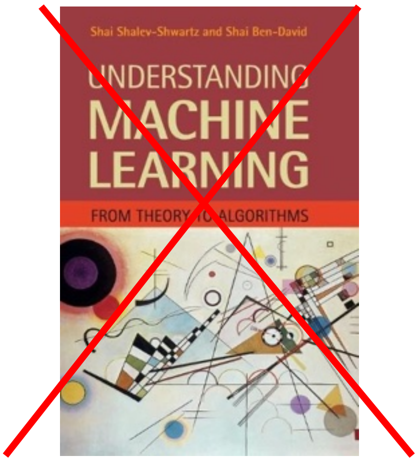
Central theoretical conundrum: why do DNNs choose the solutions that work?

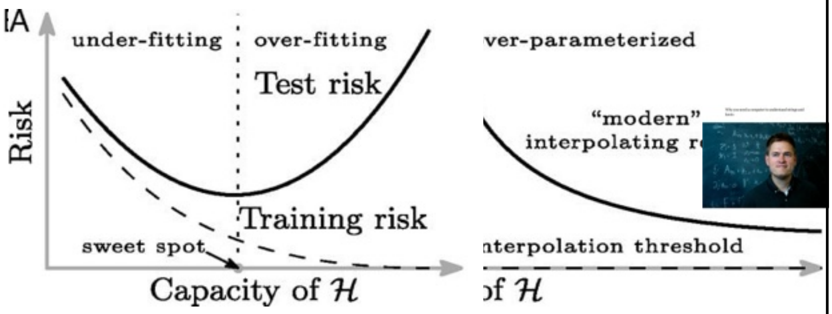
Why you need a computer to understand strings and knots



6

Classical learning theory





M Belkin, PNAS 116.15849 (2019)

7

A traditional Pac-Bayes bound in function

Posterior for functions conditioned on training set S follows from Bayes rule

$$P(f|S) = \frac{P(S|f)P(f)}{P(S)},$$

Prior over functions $P(f)$

If we wish to infer (i.e. no noise) at some points, then we need a 0-1 likelihood on training data $S = \{(x_i, y_i)\}_{i=1}^m$

$$P(S|f) = \begin{cases} 1 \text{ if } \forall i, \quad f(x_i) = y_i \\ 0 \text{ otherwise .} \end{cases}$$

P(S) = marginal likelihood or evidence

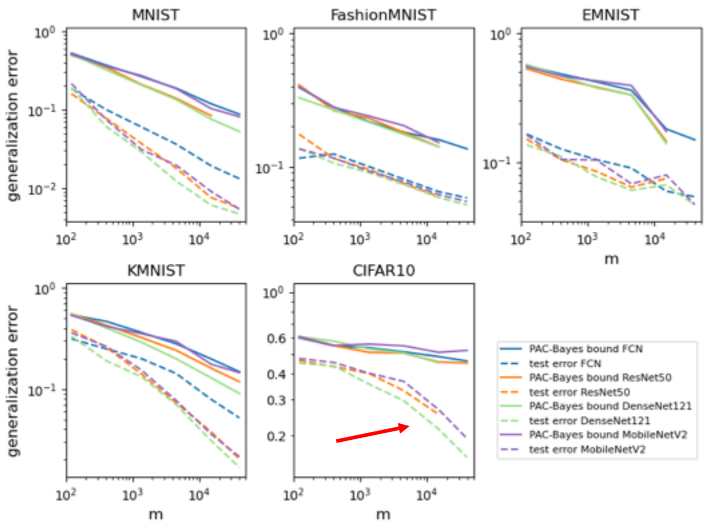
Functions that fit S

$$P(S) = \sum_f P(S|f)P(f) = \sum_{f \in C(S)} P(f)$$

P(f|S) = P(f)/P(S) or 0, so bias in prior translates over to bias in posterior

8

PAC-Bayes bound reproduces learning curve scaling with m



Guillermo
Valle Perez

1) P(S) measures inductive bias
2) We use infinite width limit GP to calculate P(S)
3) No SGD in the bound.
4) P(f) is bias at initialization,

Marginal-likelihood bound

$$-\ln\left(1-\epsilon(h)\right) < \frac{\ln\frac{1}{P(C(S))}+\ln m+\ln\frac{1}{\delta}+\ln\frac{1}{\gamma}}{m-1}$$

Generalization bounds for deep learning Guillermo Valle-Pérez and AAL, arxiv:arXiv:2012.04115

9

2 questions about generalisation?

Leo Breiman
1928 – 2005

Our fields would be better off with far fewer theorems, less emphasis on faddish stuff, and much more scientific inquiry and engineering. But the latter requires real thinking.

For instance, there are many important questions regarding neural networks which are largely unanswered. There seem to be conflicting stories regarding the following issues:

- Why don't heavily parameterized neural networks overfit the data?
- What is the effective number of parameters?
- Why doesn't backpropagation head for a poor local minima?
- When should one stop the backpropagation and use the current parameters?

Ising models

Breiman, L. **Reflections after refereeing papers for nips.** (1995).

Question 1: Why don't heavily parameterized neural networks overfit the data?

Question 2: Given a DNN that works well, what hyperparameters, etc… should we use to get better generalization and why?

10

Why can theory offer to understanding DNNs?

"My best guess is divine benevolence [...] Nobody really understands what's going on. This is a very experimental science [...] It's more like alchemy or whatever chemistry was in the Middle Ages." -- Noam Shazeer 2024

Deep learning is more like biology -- Zohar Ringel 2025

We should be realistic about what theory can do, think of a jet engine – Boris Hanin 2025

11

Inductive bias and feature learning

1. Two questions about generalisation
2. Inductive bias towards simplicity
3. Inductive bias and Zipf's law
4. Feature learning

William of Ockham
1287-1347

12

Boolean functions, an Ising-like model for supervised learning:

Doctor's truth table for COVID-19

Boolean function

Send to hospital?	Fever?	Cough?	Lost sense of smell?	Over 60?	Heart problem?	Obese?	Diabetes?
1	1	1	1	1	1	1	1
1	1	1	1	0	1	1	1
0	1	1	1	0	0	0	0
1	1	1	1	1	0	1	1
0	1	1	0	1	0	1	1

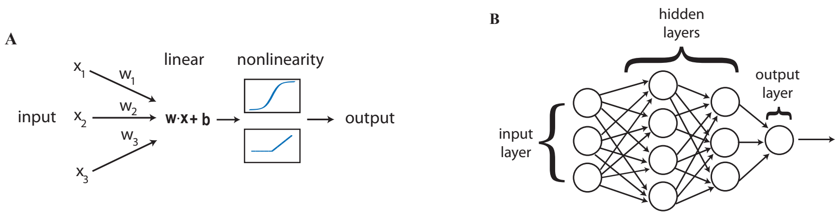
Given some examples, can we learn the rest of the function ?
E.g. can we learn a full truth table from a partial one?

A function maps all possible answers to outputs.

n questions; 2^n possible answers; 2^{2^n} possible Boolean functions
For n=7 $2^7 = 128$ answers; $2^{128} = 3.4 \times 10^{38}$ possible functions

13

Parameter-function map



Let the space of functions that the model can express be $\mathcal{F}$. If the model has p real valued parameters, taking values within a set $\Theta \subseteq \mathbb{R}^p$,
the parameter-function map, $\mathcal{M}$, is defined as:

$$\begin{aligned}\mathcal{M} : \Theta &\to \mathcal{F}\\ \theta &\mapsto f_\theta\end{aligned}$$

where f_θ is the function implemented by the model with choice of parameter vector θ.

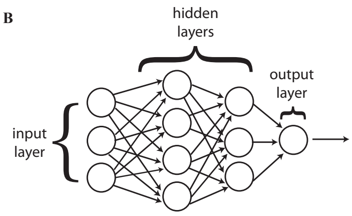
G. Valle-Perez, C. Camargo and A.A. Louis, arxiv:1805.08522 – ICLR 2019

14

If we randomly sample DNN parameters, what Boolean functions do we get?

2^{2^n} possible Boolean functions

For n=7, $N_f = 2^{128} = 3.4 \times 10^{38}$



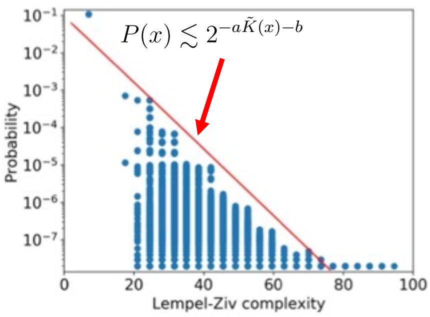
15

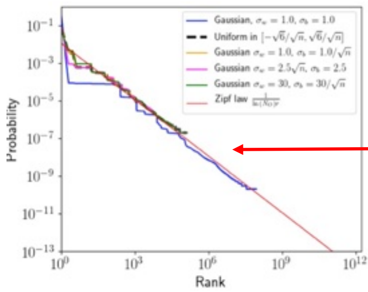
DNNs have an inbuilt Occam's razor: inductive bias towards simple functions

Prior P(f): upon randomly sampling parameters, how likely to find Boolean function f?

Simple functions exponentially more likely to occur

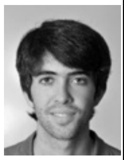
10^8 samples of parameters (7,40,40,1) DNN (FCN) with ReLU.





Zipf's law

Boolean functions for n=7. $2^7 = 128$ possible answers & $2^{128} \simeq 3.4 \times 10^{38}$ possible functions



Guillermo Valle Perez

G. Valle Perez, C. Camargo and A.A. Louis, arxiv:1805.08522 – ICLR 2019

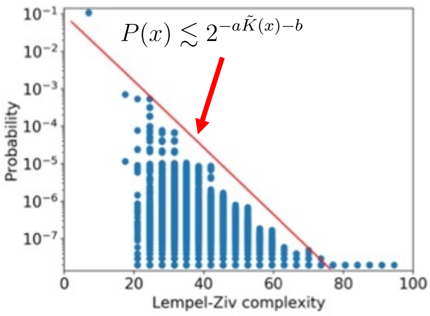
16

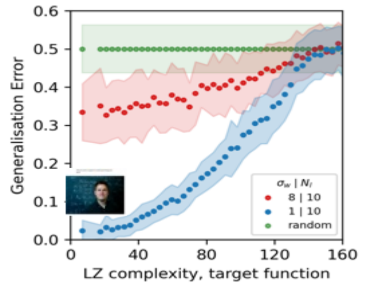
DNNs have an inbuilt Occam's razor: inductive bias towards simple functions

Prior P(f): upon randomly sampling parameters, how likely to find Boolean function f?

Simple functions exponentially more likely to occur

10^8 samples of parameters (7,40,40,1) DNN (FCN) with ReLU.





(c) Generalisation error vs complexity

Boolean functions for n=7. $2^7 = 128$ possible answers & $2^{128} \simeq 3.4 \times 10^{38}$ possible functions

The magic of DNNs: Under supervised learning with |S|=64, there are still $2^{64} \simeq 3.4 \times 10^{38} \simeq 2 \times 10^{19}$ possible functions that fit S with zero error. So why does the DNN chose one that generalizes well?

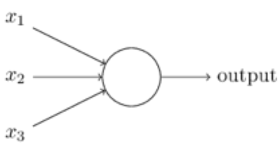
Guillermo Valle Perez

G. Valle Perez, C. Camargo and A.A. Louis, arxiv:1805.08522 – ICLR 2019

17

Proving entropy bias for the perceptron

P(f): If we randomly sample parameters θ, how likely are we to produce a particular function f?



Entropy bias:

Theorem 4.1. *For a perceptron f_θ with $b = 0$ and weights w sampled from a distribution which is symmetric under reflections along the coordinate axes, the probability measure $P(\theta : \mathcal{T}(f_\theta) = t)$ is given by*

$$P(\theta : \mathcal{T}(f_\theta) = t) = \begin{cases} 2^{-n} & \text{if } 0 \le t < 2^n \\ 0 & \text{otherwise} \end{cases}.$$

We can also prove theorems that bias towards simple function gets stronger with more layers.

Chris Mingard

Neural networks are a priori biased towards Boolean functions with low entropy, Chris Mingard, Joar Skalse, Guillermo Valle-Pérez, David Martínez-Rubio, Vladimir Mikulik, Ard A. Louis arxiv:1909.11522

18

Why simplicity bias?

19

Two infinite monkey theorems

1) Borel's infinite monkey theorem – every sequence is equally likely or unlikely

To type out Hamlet's 28 letter sequence METHINKS IT IS LIKE A WEASEL
on a typewriter with 27 keys (26 letters + space) would take about 27^{28} key-strokes.

2) Algorithmic monkey theorem: random typing into a computer language

print 01 50 times
0101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101

For three infinite monkey theorems in the context of evolution see

Nora S. Martin, Chico Q. Camargo, Ard A. Louis, , Bias in the arrival of variation can dominate over natural selection in Richard Dawkins' biomorphs, PLOS Computational Biology, **20**, e1011893. (2024)

20

Formalising the Monkey Intuition using AIT: Kolmogorov complexity

A.N. Kolgomorov
1903-1987

G.J. Chaitin
1947--

Kolmogorov/Chaitin complexity K(X) is the length in bits of the shortest program on a UTM that generates X

K is universal, (not UTM dependent) because you can always write a compiler => O(1) terms.

$K\_U(X) = K\_W(X) + O(1) \approx K(X)$ asymptotically

K is not computable due to Halting problem.

0101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101 simple

011101010011001011110111101101010000100010111010101001101011110111010010100011101110110110111010 complex

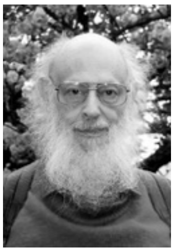
Warning: you don't know for sure that it is complex, t could be encoding **π= 3.**141592653589793238462 ….. =
11.00100100001111110110101010001000100001011010001100001000110100110 00100110001100110001010001011100000

new intuitions
-- A random number is one for which K(X) ≳ |X|
-- The complexity of a set can be << than complexity of elements of the set

21

Formalising the Monkey Intuition using AIT: Algorithmic Probability

Algorithmic Probability P(x) = probability a random program on a (prefix) UTM generates x



R. Solomonoff
1926-2009

$$P_U(x) = \sum_{p:U(p)=x} \left(\frac{1}{2}\right)^{l(p)} = 2^{-K(x)} + \ldots$$

First term is the biggest one

Sum all binary codes that generate X
on a prefix machine

Intuition: simpler (small K(X)) outputs are much more likely to appear

It seems to me that the most important discovery since Gödel was the discovery by Chaitin, Solomonoff and Kolmogorov of the concept called Algorithmic Probability,. Everybody should learn all about that and spend the rest of their lives working on it.
Marvin Minsky (2014)
https://www.youtube.com/watch?v=DfY-DRsE86s&feature=youtu.be&t=1h30m02s

Solomonoff, R., "A Preliminary Report on a General Theory of Inductive Inference", Report V-131, Zator Co., Cambridge, Ma. Feb 4, 1960, revision, Nov., 1960.

22

Formalising the Monkey Intuition using AIT: Levin's Coding Theorem

L. Levin, 1948 --

We should teach this much more widely!

$$2^{-K(x)} \leq P(x) \leq 2^{-K(x)+O(1)}$$

Intuitively: simpler (small K(x)) outputs are much more likely to appear

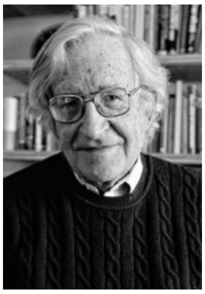
Serious problems for applying coding theorem

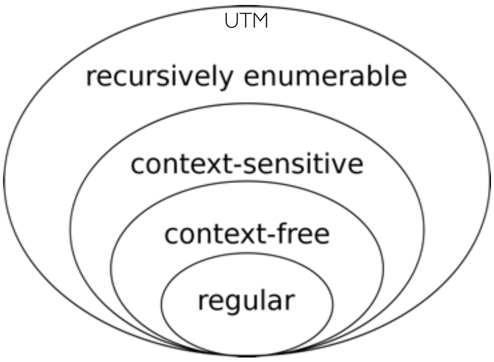
1) Many systems of interest are not Universal Turing Machines
2) Kolmogorov complexity K(x) is formally incomputable
3) Only holds in in the asymptotic limit of large x…

L. A. Levin. Laws of information conservation (non-growth) and aspects of the foundation of probability theory. Problems of Information Transmission, 10:206, (1974)

23

Chomsky hierarchy



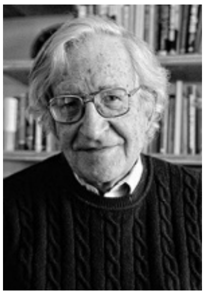


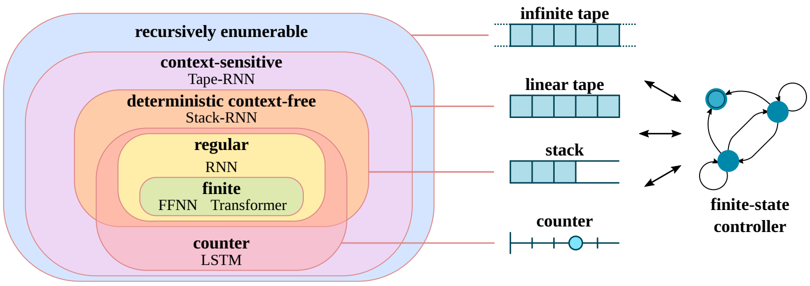
Markov chains

Noam Chomsky (1956). "Three models for the description of language" (PDF). IRE Transactions on Information Theory. **2** (3): 113–124.

24

Chomsky hierarchy





G Delétang et al, Neural networks and the chomsky hierarchy, arXiv:2207.02098

25

Simplicity bias for computable (non-UTM) input-output maps

Kamal Dingle

(2 Dphils of work)

$$\boxed{P(x) \lesssim 2^{-a\tilde{K}(x)-b}}$$

NOTE: upper bound only!

Chico Camargo

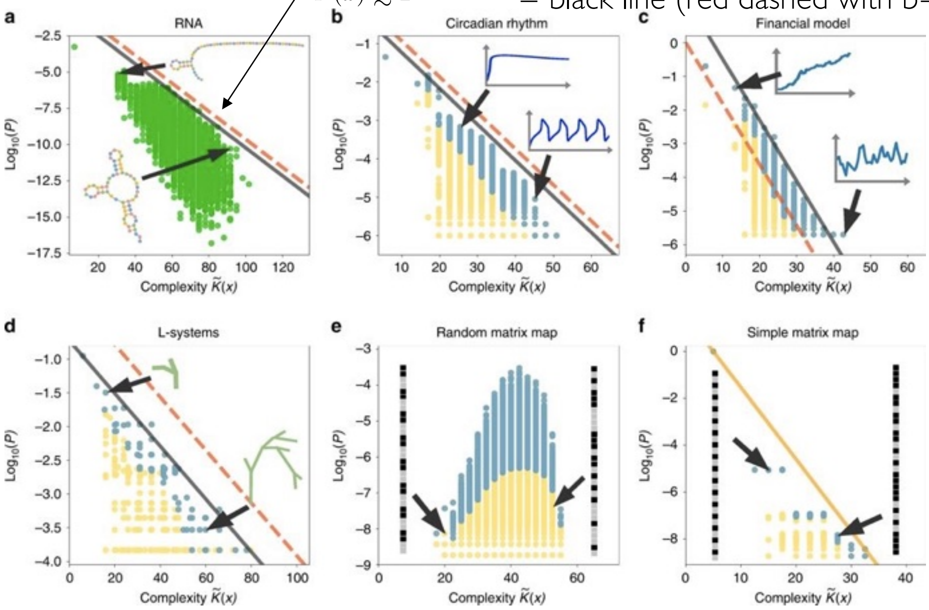
1) Computable input-output map f: I → O
2) Map f must be simple – e.g. K(f) grows slowly with system size – then
   K(x|f,n) ≈ K(x) + O(1)
3) K(x) is approximated, for example by Lempel Ziv compression or some other suitable measure.
4) Bound is tight for most inputs, but not most outputs.
5) Maps must be a) simple, b) redundant, c) non-linear, d) well-behaved (e.g. not a pseudorandom number generator) – many maps satisfy these conditions.
6) There is also a statistical lower bound.

K. Dingle, C. Camargo and A.AL, Nature Comm 9, 761 (2018); K. Dingle, G. Valle-Perez, AAL, Sci. Rep. 10, 4415 (2020)

26

Simplicity bias holds for many different maps

$P(x) \lesssim 2^{-aK(x)-b}$ = black line (red dashed with b=0)



K. Dingle, C. Camargo and A.AL, Nature Communications 9, 761 (2018)

27

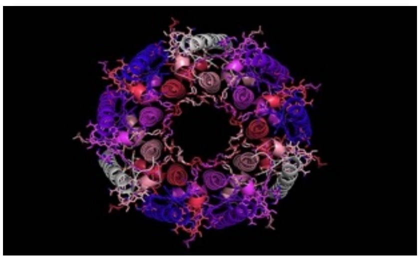
Evolution also has an inbuilt Occam's razor. this helps explain why protein complexes are so symmetric

The New York Times

TRILOBITES

Life's Preference for Symmetry Is Like 'A New Law of Nature'

Techniques from computer science may help explain the tendency in biology for structures to repeat themselves.



By Kate Golembiewski
March 24, 2022

I.G. Johnston, K. Dingle et al., Symmetry and simplicity spontaneously emerge from the algorithmic nature of evolution, PNAS 119, e2113883119 (2022)

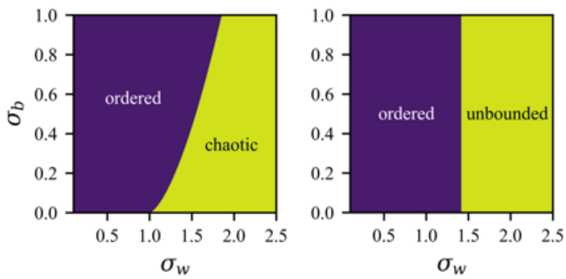
28

Can we break simplicity bias?

William of Ockham
1287-1347

29

DNNs can exhibit an order-to-chaos transition



(a) *Tanh* **(b)** *ReLU*

Figure 3: *Mean field phase diagrams for tanh and ReLU activation functions showing various phase regimes as a function of* σ_w *and* σ_b.

Chaotic regime for some activation functions (not ReLU) – for wider initial parameters

Deep Information Propogation, S. S. Schoenholz et al. arXiv:1611.01232

30

Chaotic regime reduces bias (strength of Occam's razor) in prior P(f)

FCN on Boolean system

J. Empirical probability versus LZ complexity plots

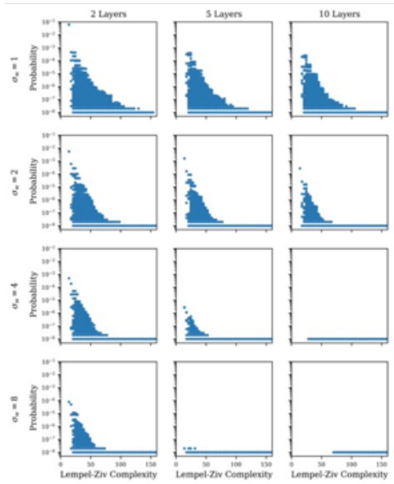
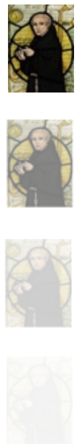


Figure 17: *Empirical probability of individual functions versus their LZ complexity for networks initialised with various σ_w and numbers of layers. Despite suffering from finite-size effects, points with a probability of 10^{-8} are not removed since in plots ($\sigma_w = 4, d = 10$) and ($\sigma_w = 8, d = 10$) only points of this type are found. Details are the same as Figure 6.*

More biased

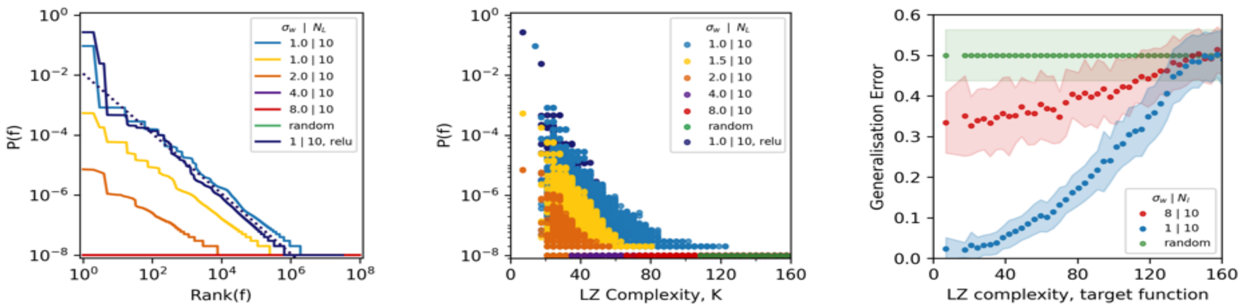


Less biased
No Occam

Greg Yang and Hadi Salman. A fine-grained spectral perspective on neural networks. arXiv preprint arXiv:1907.10599, 2019.

31

Chaotic regime changes the bias in prior P(f)



(a) Prior $P(f)$ versus rank. (b) Prior $P(f)$ versus complexity (c) Generalisation error vs complexity

Do deep neural networks have an inbuilt Occam's razor? Chris Mingard, Henry Rees, Guillermo Valle-Pérez, AAL Nat Comm **16, 220 (2025)**

32

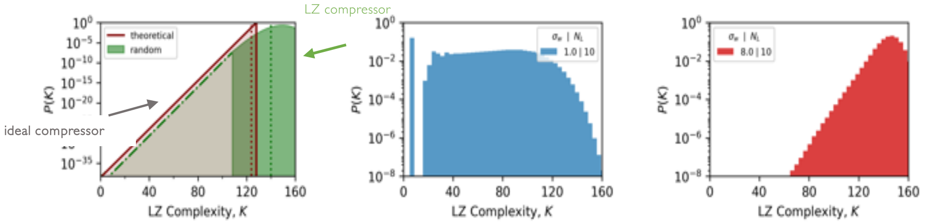
Priors over complexity P(K)

For an ideal compressor, the fraction of strings compressible by X bits = ½^X

2^{10} = 1024
ideal compressor
2^9 = 512
2^8 = 256
2^7 = 128
2^6 = 64
2^5 = 32
2^4 = 16
2^3 = 8
2^2 = 4
2^1 = 2

Priors over complexity

Random strings length 128 | Normal DNN | Chaotic DNN



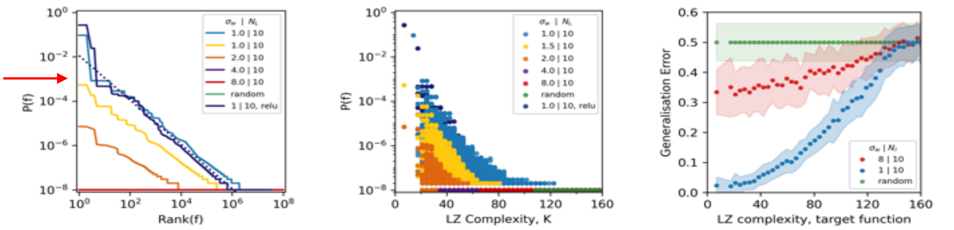
(g) Prior $P(K)$ for uniform sampling (h) Prior $P(K)$ for $\sigma_w = 1$ (i) Prior $P(K)$ for $\sigma_w = 8$

Do deep neural networks have an inbuilt Occam's razor? Chris Mingard, Henry Rees, Guillermo Valle-Pérez, AAL Nat Comm **16**, 220 (2025)

33

To generalise well DNNs need a specific kind of Occam's razor

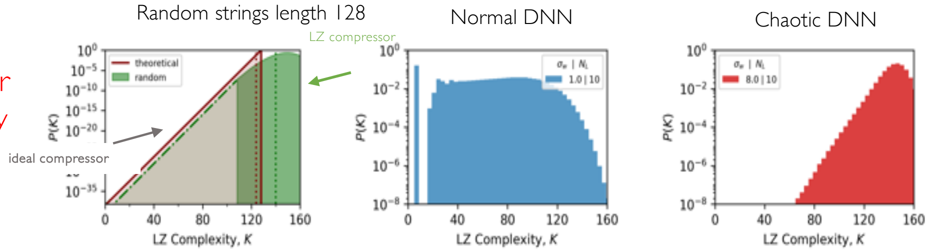
we'll discuss rank plots later in Zipf section



(a) Prior $P(f)$ versus rank. (b) Prior $P(f)$ versus complexity (c) Generalisation error vs complexity

Priors over complexity

Random strings length 128 | Normal DNN | Chaotic DNN



(g) Prior $P(K)$ for uniform sampling (h) Prior $P(K)$ for $\sigma_w = 1$ (i) Prior $P(K)$ for $\sigma_w = 8$

Do deep neural networks have an inbuilt Occam's razor? Chris Mingard, Henry Rees, Guillermo Valle-Pérez, AAL Nat Comm **16**, 220 (2025)

34

Questions about inductive bias towards simplicity

1) Do we need a new statistical learning theory based on inductive bias?

2) What other natural inductive biases aid generalisation in DNNs?

3) What does DNN inductive bias tell us about data on which they generalise well?

35

Questions about inductive bias towards simplicity

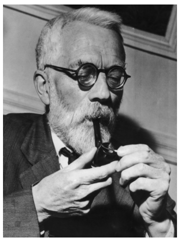
So far we looked at **question 1**: why do DNNs generalise at all?

Can we look at **question 2**: Given a DNN, can me make it work better?

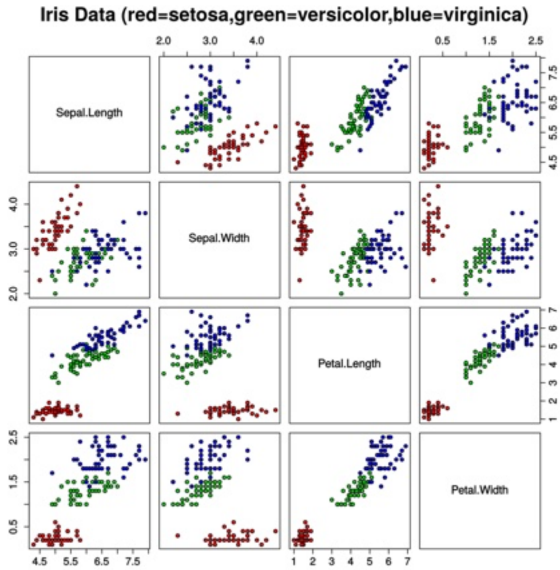
36

Feature learning

Iris Versicolor **Iris Setosa** **Iris Virginica**



R.A. Fisher, 1890-1962



R. A. Fisher "The use of multiple measurements in taxonomic problems". Annals of Eugenics. 7: 179 (1934)

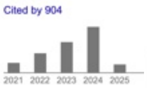
37

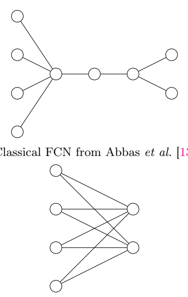
Moan: Quantum neural network literature and quantum advantage

The power of quantum neural networks

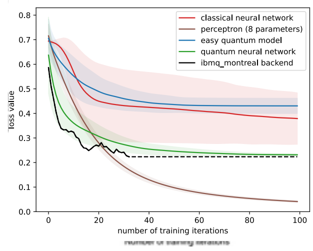
Amira Abbas, David Sutter, Christa Zoufal, Aurelien Lucchi, Alessio Figalli & Stefan Woerner

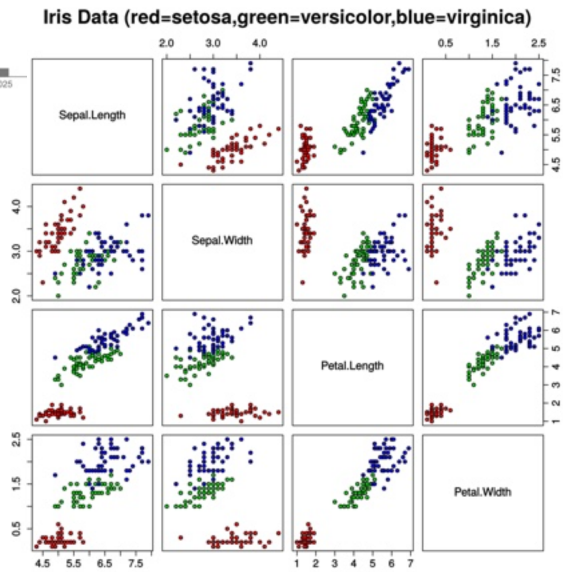
Nature Computational Science 1, 403–409 (2021) | Cite this article





a) Classical FCN from Abbas et al. [13].

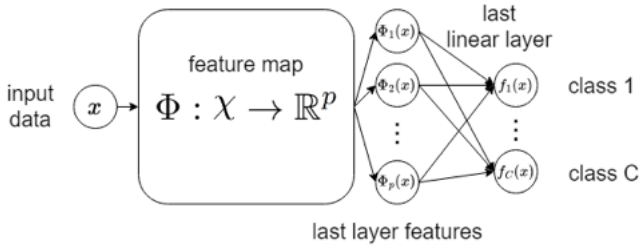




C. Mingard et al. Exploiting the equivalence between quantum neural networks and perceptrons arXiv:2407.04371

38

DNN as a feature map + linear final layer of width p



(a) Abstraction of DNN architecture

Outputs of the penultimate layer are the inputs (features) of the final layer

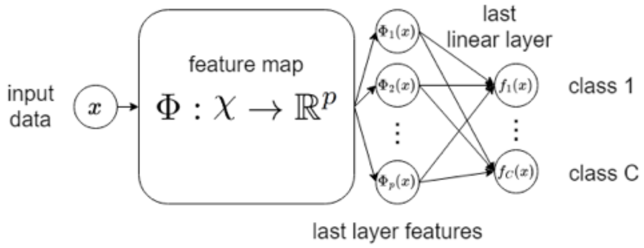
$$f(x) = \sum_{k=1}^{p} \theta_k \Phi_k(x). \qquad x \sim q, \quad \Phi : \mathcal{X} \to \mathbb{R}^p$$

Yoonsoo Nam

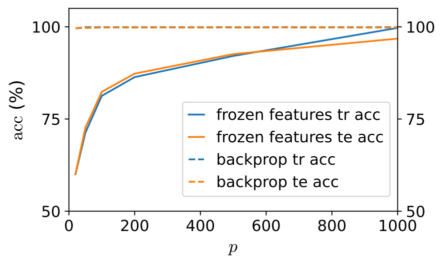
Y. Nam et al, Visualising Feature Learning in Deep Neural Networks by Diagonalizing the Forward Feature Map arXiv:2410.04264

39

Small final layers force feature learning



(a) Abstraction of DNN architecture



5-layer CNN on the standard full MNIST dataset as a function of the width p of the final linear layer for
(**solid line**) fixed feature map
(**dashed lines**) full SGD with backpropagation

On MNIST, p ≅ 1,000
On CIFAR10, p ≅ 50,000

Final layer:

$$f(x) = \sum_{k=1}^{p} \theta_k \Phi_k(x). \qquad x \sim q, \quad \Phi : \mathcal{X} \to \mathbb{R}^p$$

Y. Nam et al, Visualising Feature Learning in Deep Neural Networks by Diagonalizing the Forward Feature Map arXiv:2410.04264

40

Toy model I: feature learning the Heaviside function

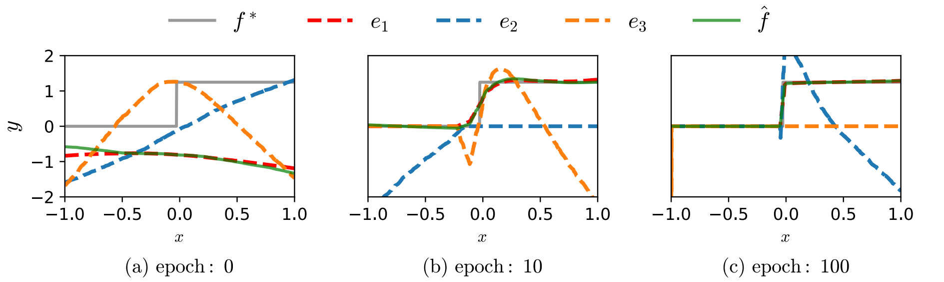


Figure 2: **Toy model demonstrating feature learning by a DNN** A 3-layer FCN with width 100, having scalar input and output, is trained to learn the Heaviside step function f^* over the domain $[-1, 1]$.

$$f(x) = \sum_{k=1}^{p} \theta_k \Phi_k(x). \qquad x \sim q, \quad \Phi : \mathcal{X} \to \mathbb{R}^p$$

41

Projections on the target function and learned function

we can define projection operator $P_{\mathcal{H}^*} : L^2(\mathcal{X}) \to L^2(\mathcal{X})$ onto the target function space $\mathcal{H}^*$ and define the quality of a feature Q_k^* for e_k as follows:

$$P_{\mathcal{H}^*}[g] := \sum_{j=1}^{C} \frac{1}{\|f_i^*\|^2} |f_i^*\rangle \langle f_i^*|g\rangle , \qquad Q_k^* := \frac{\langle e_k|P_{\mathcal{H}^*}[e_k]\rangle}{C} = \sum_{i=1}^{C} \langle e_k|f_i^*\rangle^2 . \tag{7}$$

Projection onto target function

$$\Pi^*(k) := \sum_{j=1}^{k} Q_k^* = \sum_{j=1}^{k} \sum_{i=1}^{C} \langle e_j|f_i^*\rangle^2 ,$$

Projection onto learned function

$$\hat{\Pi}(k) = \sum_{j=1}^{k} \hat{Q}_k,$$

42

Some operators and other definitions

Integral operator version

$$T[f](x') := \int_{\mathcal{X}} \Phi(x)^T \Phi(x') f(x) q(x) dx, \qquad T[e_k] = \rho_k e_k,$$

eigenvalues

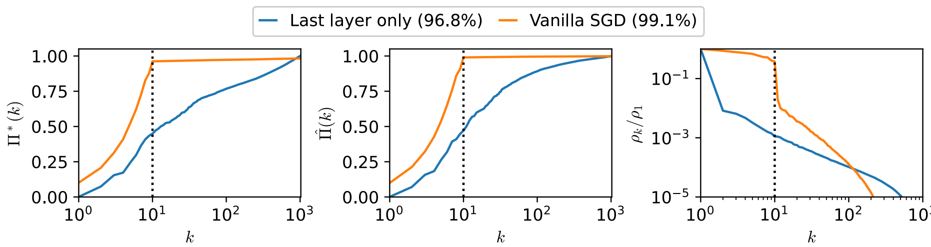
$$\rho_k = \langle e_k | T[e_k] \rangle = \langle e_k | \sum_{j=1}^{p} |\Phi_j\rangle \langle \Phi_j | e_k \rangle = \sum_{j=1}^{p} \langle \Phi_j | e_k \rangle^2 ,$$

$$CKA(T, T_{MP}) = \frac{\mathrm{Tr}(c(T)c(T_{MP}))}{\|c(T)\|_F \|c(T_{MP})\|_F},$$ (how close to the minimal representation)

Definition 5 (CKA Minimum Feature Regime Measure κ_{CKA})**.** *For a distribution q and a class-balanced learned function $\hat{f}$, a DNN is in MF regime if $\kappa_{CKA} = 1 - \text{CKA}(T, T_{MP}) < \epsilon$, where T is the feature kernel (operator) of a DNN and T_{MP} is the MP-operator.*

43

Toy model 2: coefficient learning v.s. feature learning.



(a) Target projection (b) Self projection (c) Eigenvalue

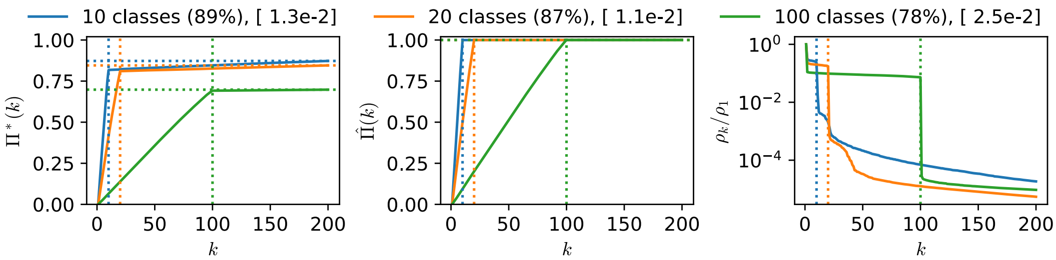
Figure 3: **Comparing feature learning to coefficient learning for a wide CNN on MNIST** A CNN of width $p = 1024$ can be trained to 100% training accuracy on the full training set of MNIST in two different

Big differences:
Frozen features uses all 1024 features
Feature learning mainly uses just 10 features

Is strong feature learning a dynamical effect?
SGD could have minimized the loss of the full DNN by coefficient learning, why did it not do so?

44

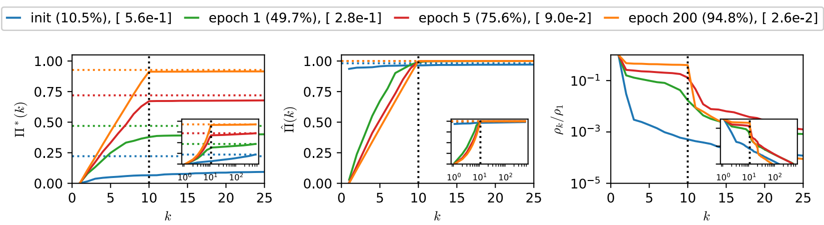
Minimum feature regime (MFR) for 10, 20 and 100 classes on CIFAR



(a) Target projection (b) Self projection (c) Eigenvalue

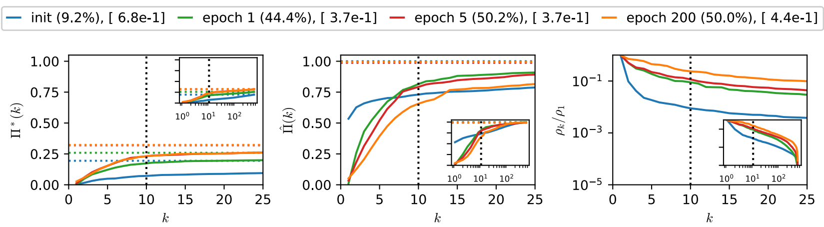
45

Minimum feature regime (MFR) and extended feature regime (EFR)



(i) Target Projection (ii) Self Projection (iii) Eigenvalue

(a) ResNet18 on CIFAR10 (Minimum Feature Regime (MFR))



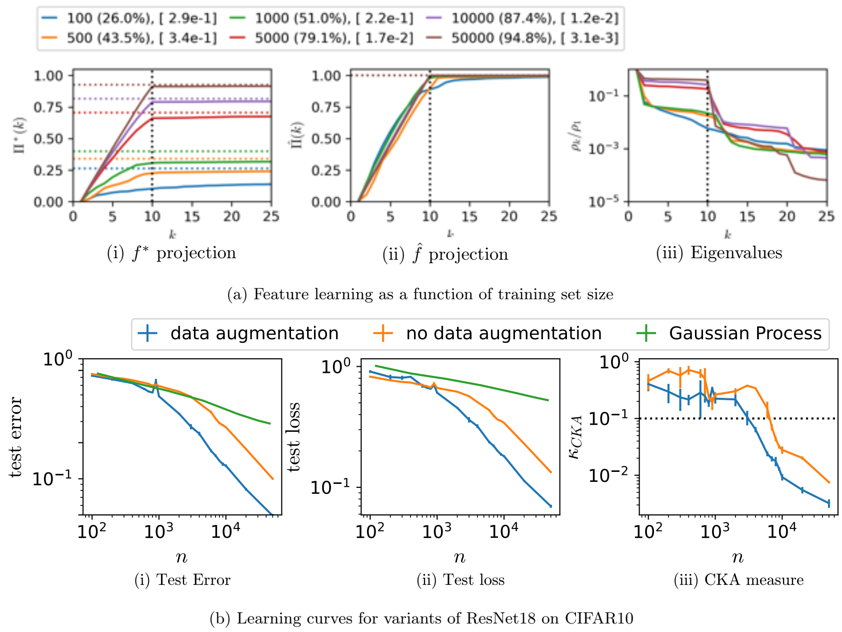
(i) Target Projection (ii) Self Projection (iii) Eigenvalue

(b) FCN on CIFAR10 (Extended Feature Regime (EFR))

MFR is like Neural Collapse (NC) but at much earlier timescales

46

Feature-learning as a function of data (learning curves)

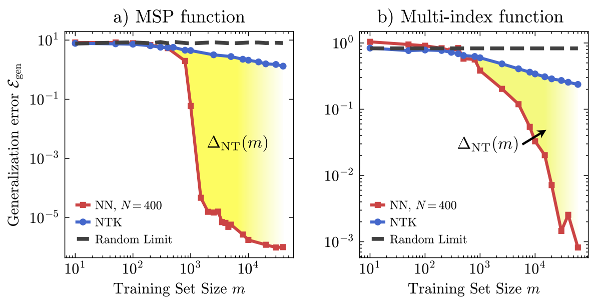


(a) Feature learning as a function of training set size

(b) Learning curves for variants of ResNet18 on CIFAR10

47

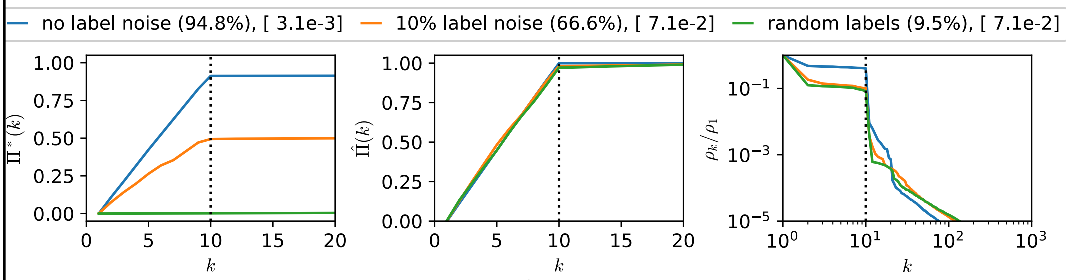
Is image data special?



Niclas Göring

48

Strong feature learning for random data.

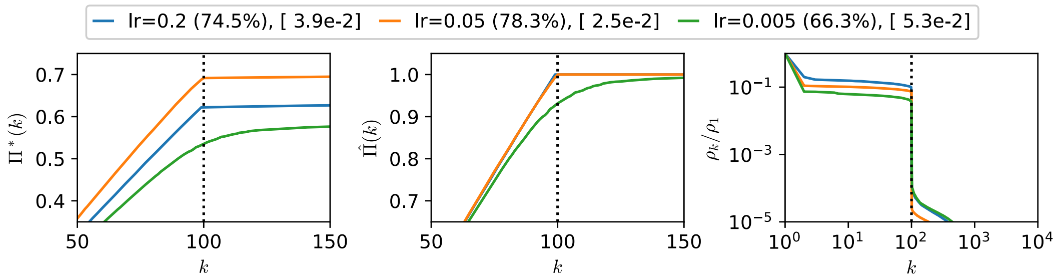


(i) f^* projection (ii) $\hat{f}$ projection (iii) Eigenvalues

(a) ResNet18 on CIFAR10 with label randomization

49

Learning rate and feature learning regimes

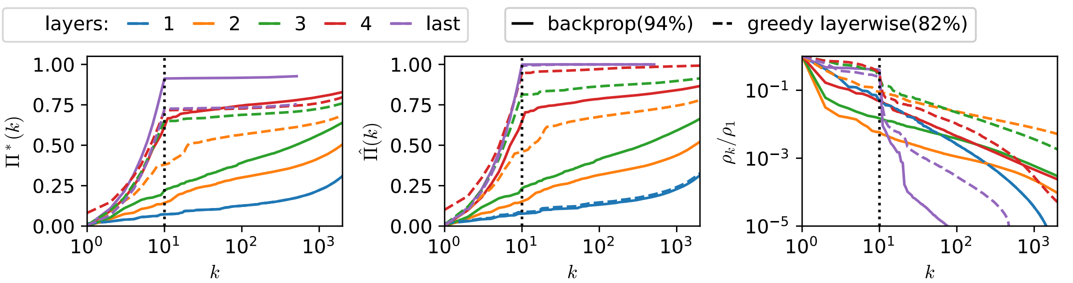


(i) Target projection (ii) Self projection (iii) Eigenvalue

(a) ResNet18 on CIFAR100 with various learning rates

50

Greedy layerwise training v.s. full backpropogation

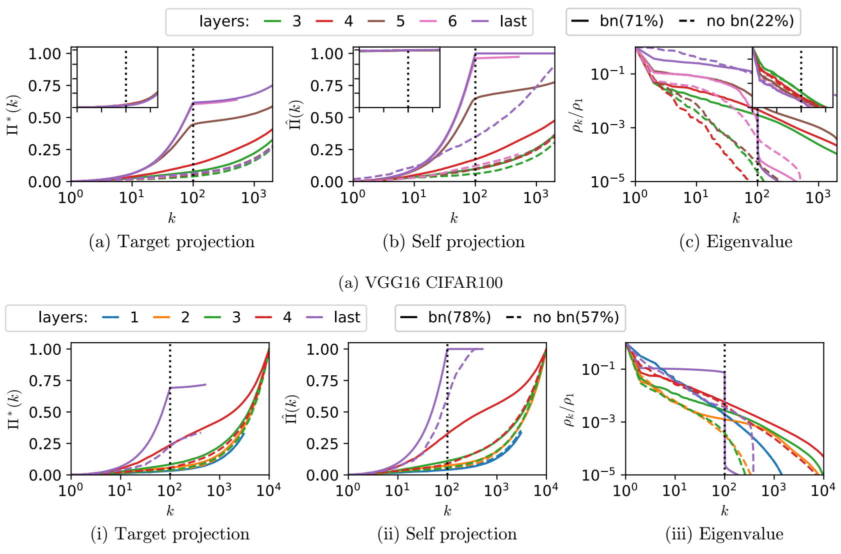


(a) Target projection (b) Self projection (c) Eigenvalue

Figure 15: **Greedy layerwise training leads to a collapse of features in earlier layers.** In contrast to

51

Batchnorm's effect on intermediate layers



(a) Target projection (b) Self projection (c) Eigenvalue

(a) VGG16 CIFAR100

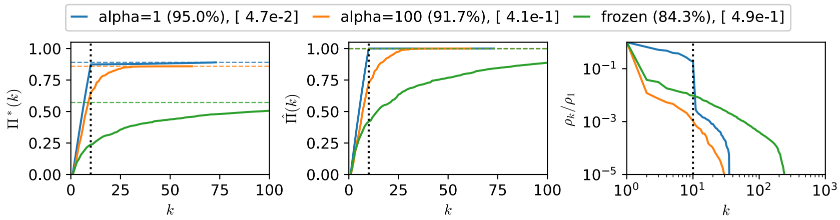
(i) Target projection (ii) Self projection (iii) Eigenvalue

(b) WideResNet18 CIFAR100

Figure 22: **Effect of batch normalization in CIFAR100.** We clearly observe more alignment in the

52

Lazy regime for finite size DNNs



(a) Target projection (b) Self projection (c) Eigenvalue

Figure 7: **Features in the rich and lazy regimes compared to fixed features** A width $p = 256$ 4-layer FCN is trained on 10,000 images of MNIST. We compare the standard case ($\alpha = 1$) to $\alpha = 100$, which is well above the threshold $\alpha/\sqrt{p} > 1$ thought to herald the so-called lazy regime, and the coefficient learning regime,

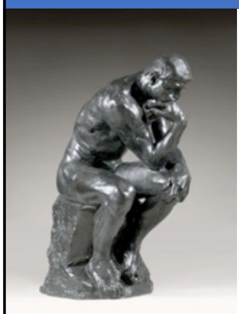
rescale outputs by a factor alpha to enter lazy regime – here lazy regime = EFR, not coefficient learning due to small p

$$\mathbb{L}(x) = \frac{1}{\alpha^2 n} \sum_{i}^{n} \ell(\alpha(f(x^{(i)}) - f^*(x^{(i)})).$$

L. Chizat, E. Oyallon, and F. Bach. On lazy training in differentiable programming. Advances in neural information processing systems, 32, 2019.
M. Geiger, S. Spigler, A. Jacot, and M. Wyart. Disentangling feature and lazy training in deep neural networks. Journal of Statistical Mechanics: Theory and Experiment, 2020(11):113301, 2020.

53

questions raised by feature-learning

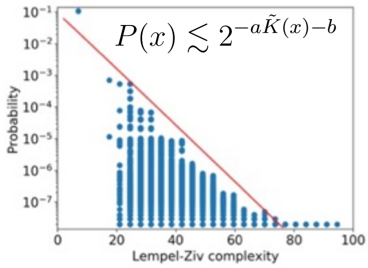


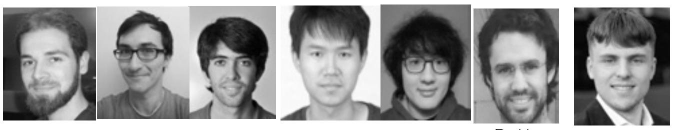
- How should one measure feature-learning?
- Why does a DNN feature learn?
- Can we perturb around NTK to reach the feature-learning regime?
- What does feature-learning tell us about generalisation?
- When is feature-learning bad for generalisation?

54

Who did all the hard work: machine learning

Occam's razor





Kamal Dingle Chico Camargo Guillermo Valle Perez Shuofeng Zhang Yoonsoo Nam David Martinez Niclas Göring

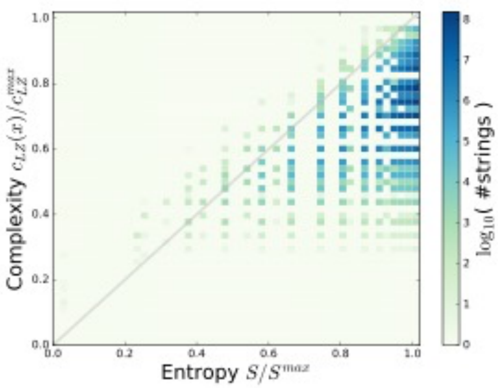
Jed Burkat Nivedita Hanzhi Jiang Ouns El Harzli Henry Rees Chris Mingard Joar Skalse Vlad Mikulik Isaac Reid

55

Lempel Ziv complexity of binary strings

101010101010101010101010101010101010101010101010101010101010

- Has high entropy but low LZ complexity



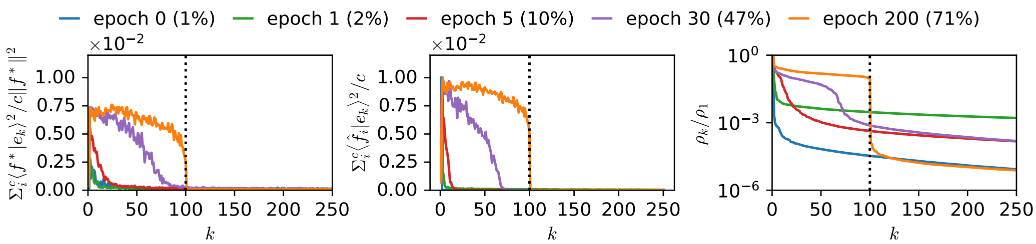
LZ Complexity v.s. Entropy S = p log p + (1-p) log (1-p) for binary strings of length 30

$$C_{LZ}(x) = \begin{cases} \log_2(n), & x = 0^n \text{ or } 1^n \\ \log_2(n)[N_w(x_1...x_n) + N_w(x_n...x_1)]/2, & \text{otherwise} \end{cases}$$

56

Matthew effect dynamics

Matthew effect in feature learning
During the training of a DNN via gradient descent- based algorithms, the quality of features increases faster for smaller k (larger eigenvalues ρk).



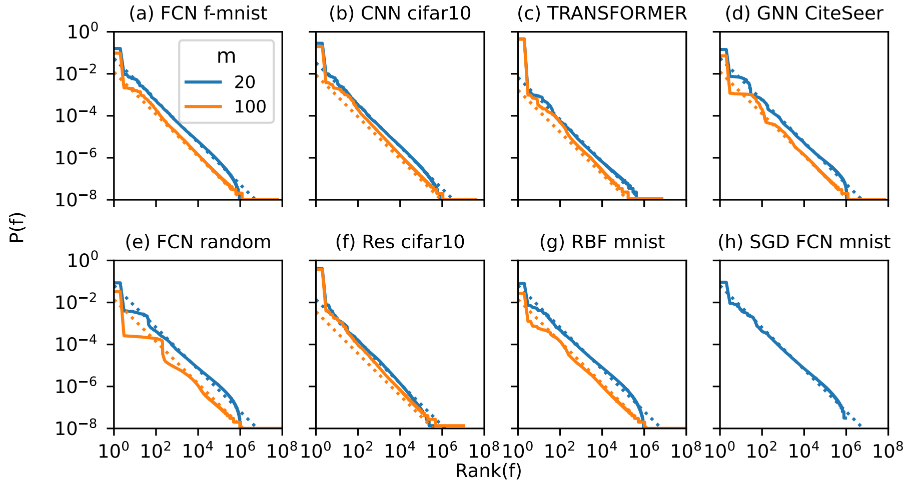
(i) Target Projection (ii) Self Projection (iii) Eigenvalue

(d) VGG16, CIFAR100

So far we can only prove its dynamical origin for the unconstrained feature model of Mixon et al [2020].

57

Prior of high-capacity models exhibits a universal Zipf's law



Schwab, D. J., Nemenman, I. & Mehta, P. Zipf's law and criticality in multivariate data without fine-tuning. PRL 113, 068102 (2014).
Aitchison, L., Corradi, N. & Latham, P. E. Zipf's law arises naturally when there are underlying, unobserved variables. PLoS comp. Biol 12, e1005110 (2016).

C. Mingard et al., The priors of successful high capacity machine learning models exhibit a universal Zipf's law

58

Same global Zipf law, but different function orderings

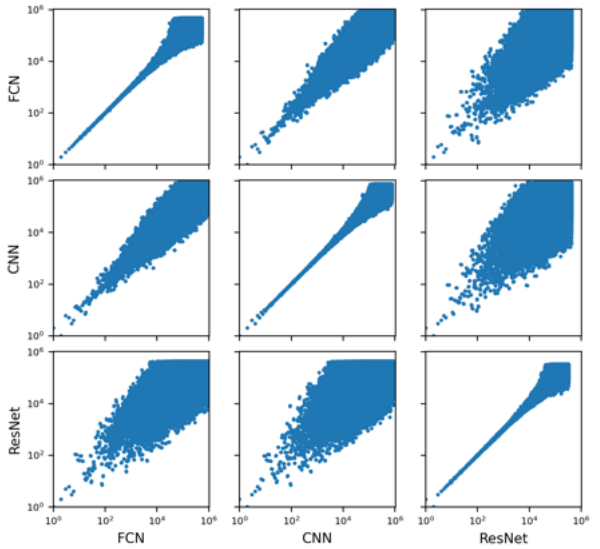
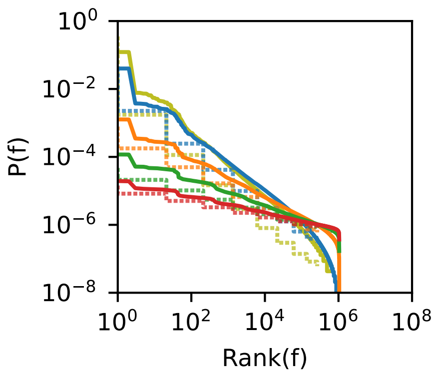


Fig. S3: **Effect of architecture on the ordering of functions in the prior.** We use cifar10, $m = 20$, with data

C. Mingard et al., The priors of successful high capacity machine learning models exhibit a universal Zipf's law

59

Why is pure Zipf so special ?



$P(f) = A/r^{1+\delta}$

N_f = total number of functions.

----

if $\delta = 0$ (pure Zipf)

$r_{1/2} = \sqrt{N_f}$

----

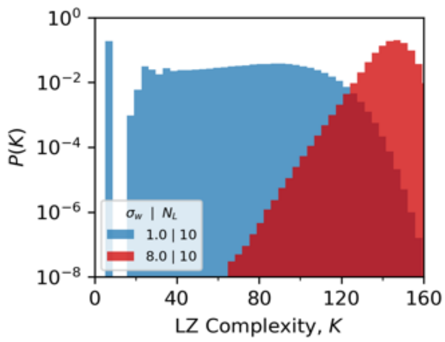
if $\delta > 0$ (super-Zipf)

$r_{1/2} \approx \frac{1}{2^{\delta}}$

----

if $\delta < 0$ (sub-Zipf)

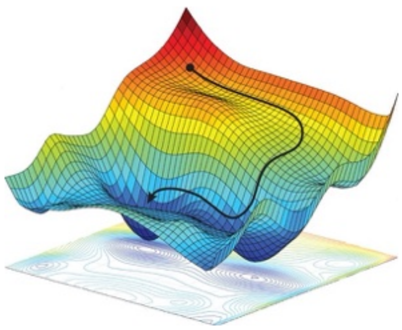
$r_{1/2} \approx \frac{1}{2^{\delta}} N_f$



C. Mingard et al., The priors of successful high capacity machine learning models exhibit a universal Zipf's law

60

Hold on: why should parameter function map predict DNN outcomes?



DNNs are trained using Stochastic gradient descent (SGD) on a loss function.

~~Dominant hypothesis in the field is that SGD has special properties that enhance generalization~~

Is SGD a Bayesian sampler? Well, almost, Chris Mingard, Guillermo Valle-Pérez, Joar Skalse, Ard A. Louis, arxiv.org: 2006.15191

61

Bayesian function picture for supervised learning on S

Posterior for functions conditioned on training set S follows from Bayes rule

$$P(f|S) = \frac{P(S|f)P(f)}{P(S)},$$

Prior over functions $P(f)$

If we wish to infer (i.e. no noise) at some points, then we need a 0-1 likelihood on training data $S = \{(x_i, y_i)\}_{i=1}^{m}$

$$P(S|f) = \begin{cases} 1 \text{ if } \forall i, \quad f(x_i) = y_i \\ 0 \text{ otherwise .} \end{cases}$$

P(S) = marginal likelihood or evidence

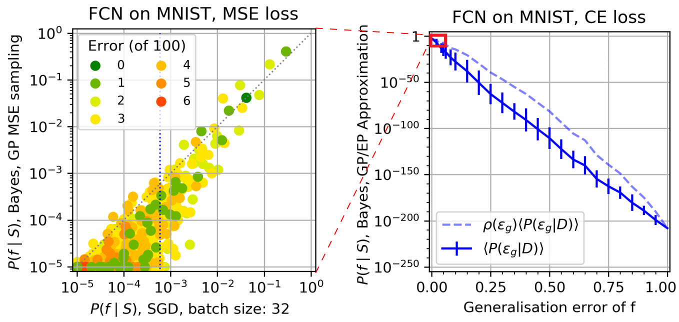
Functions that fit S

$$P(S) = \sum_f P(S|f)P(f) = \sum_{f\in C(S)} P(f)$$

P(f|S) = P(f)/P(S) or 0, so bias in prior translates over to bias in posterior

62

SGD acts like a Bayesian optimiser ….



(a) $P_B(f|S)$ v.s. $P_{SGD}(f|S)$ (b) $P_B(f|S)$ v.s. ϵ_G

FCN on binarized MNIST – training set=10,000, test set=100 images $2^{100} = 10^{30}$ possible functions fit the test set.

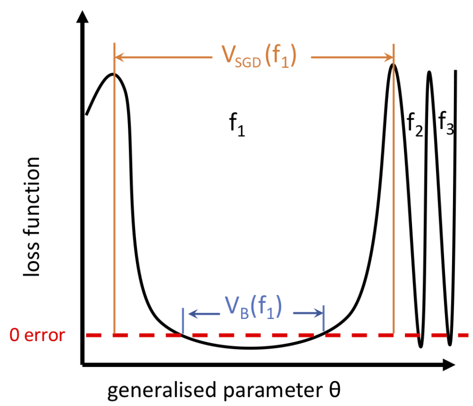
We use Gaussian Processes (GP)s to calculate $P_B(f|S)$ –

Is SGD a Bayesian sampler? Well, almost, Chris Mingard, Guillermo Valle-Pérez, Joar Skalse, Ard A. Louis, arxiv.org: JMLR 22(79):1−64, 2021

63

Problem: why should parameter function map predict outcomes?

Intuition: for very strong bias: Basin of attraction ~ Basin size (P(f))



Similar effect in evolutionary theory under strong bias:
The arrival of the frequent: how bias in genotype-phenotype maps can steer populations to local optima
Steffen Schaper and Ard A. Louis, PLoS ONE 9 (2): e86635 (2014)

Is SGD a Bayesian sampler? Well, almost, Chris Mingard, Guillermo Valle-Pérez, Joar Skalse, Ard A. Louis, arxiv.org: 2006.

64

Questions about Zipf's law

1)Can you find a super-Zipfian learner?

2) What is the link between Zipf and complexity measures?

65

A function based picture for DNNs

Definition (Representation of Functions)**.** Consider a DNN $\mathcal{N}$, a training set $S = \{(x_i, y_i)\}_{i=1}^{m}$ and test set $E = \{(x'_i, y'_i)\}_{i=1}^{|E|}$. We *represent* the function $f(\mathbf{w})$ with parameters $\mathbf{w}$ associated with $\mathcal{N}$ as a string of length $(|S| + |E|)$, where the values are the labels $\hat{y}_i$ and $\hat{y}'$ that $\mathcal{N}$ produces on the concatenation of training inputs and testing inputs.

Example: labels predicted on 5 MNIST inputs:

f(**w**) = (5,0,4,1,9) (0 errors)
f(w) = (5,0,4,7,9) (1 error)

66

Bayesian function picture for supervised learning

Posterior for functions conditioned on training set S follows from Bayes rule $S = \{(x_i, y_i)\}_{i=1}^{m}$

$$P(f|S) = \frac{P(S|f)P(f)}{P(S)},$$

0-1 likelihood on training data S

$$P(S|f) = \begin{cases} 1 \text{ if } \forall i, \quad f(x_i) = y_i \\ 0 \text{ otherwise} . \end{cases}$$

P(f|S) = P(f)/P(S) or 0,

bias in prior translates over to bias in posterior

P(S) = marginal likelihood or evidence

Functions that fit S

$$P(S) = \sum_f P(S|f)P(f) = \sum_{f \in C(S)} P(f)$$

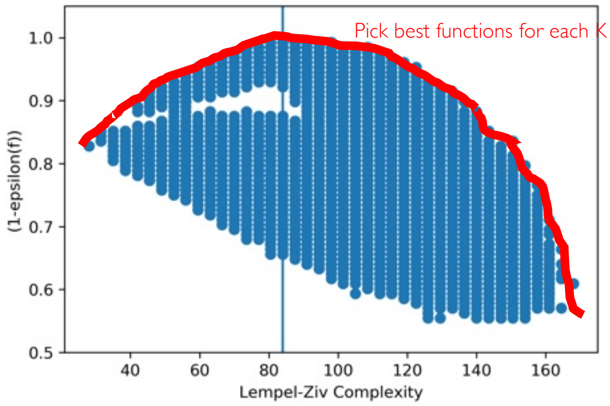
Do deep neural networks have an inbuilt Occam's razor? Chris Mingard, Henry Rees, Guillermo Valle-Pérez, Ard A. Louis, arxiv 2304.06670

67

Bayesian picture and the data

$$\langle P(f|\mathcal{S})\rangle_S = P(f)\left\langle \frac{P(\mathcal{S}_i|f)}{P(\mathcal{S}_i)}\right\rangle_{\mathcal{S}_i} \approx \frac{P(f)\boxed{(1-\epsilon(f))^m}}{\langle P(\mathcal{S})\rangle}.$$

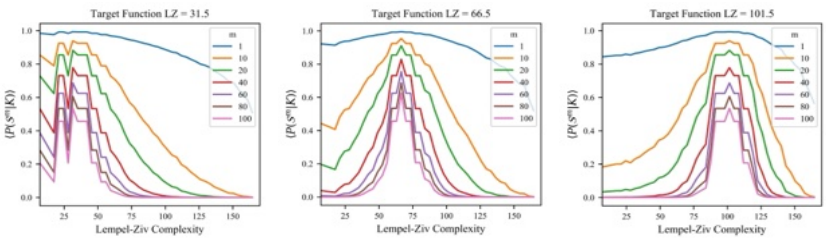
$(1 - \varepsilon(f))$



68

Bayesian picture and data

$$\langle P(f|\mathcal{S})\rangle_S = P(f)\left\langle \frac{P(\mathcal{S}_i|f)}{P(\mathcal{S}_i)}\right\rangle_{\mathcal{S}_i} \approx \frac{P(f)\boxed{(1-\epsilon(f))^m}}{\langle P(\mathcal{S})\rangle}\,.$$



69

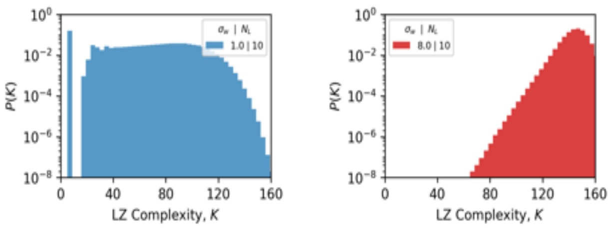
Bayesian picture and prior P(K)

$$\langle P(f|\mathcal{S})\rangle_S = P(f)\left\langle \frac{P(\mathcal{S}_i|f)}{P(\mathcal{S}_i)}\right\rangle_{\mathcal{S}_i} \approx \frac{\boxed{P(f)}(1-\epsilon(f))^m}{\langle P(\mathcal{S})\rangle}\,.$$

$P(K)$

Instead of

$P(f)$

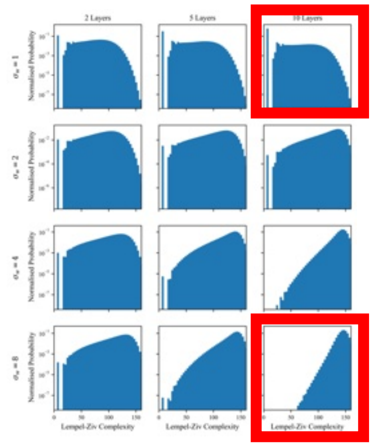


(h) Prior $P(K)$ for $\sigma_w = 1$ (i) Prior $P(K)$ for $\sigma_w = 8$

70

Bayesian picture and prior P(K)

$$\langle P(f|\mathcal{S})\rangle_S = P(f)\left\langle \frac{P(\mathcal{S}_i|f)}{P(\mathcal{S}_i)}\right\rangle_{\mathcal{S}_i} \approx \frac{\boxed{P(f)}\,(1-\epsilon(f))^m}{\langle P(\mathcal{S})\rangle}.$$



Ordered Regime:
10 Layers, σ_w = 1.0

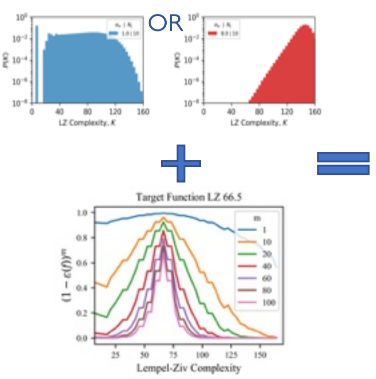
Chaotic Regime:
10 Layers, σ_w = 8.0

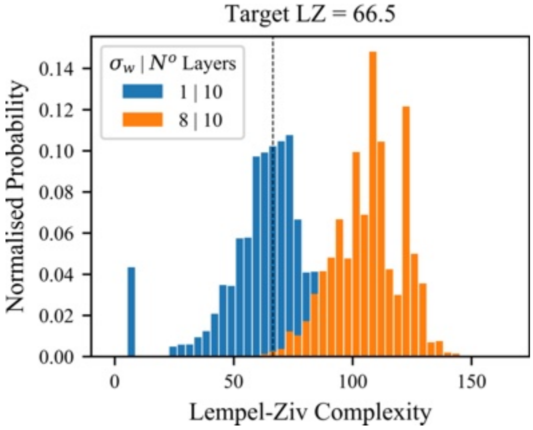
71

Bayesian picture: combining data and prior

$$\langle P(f|S)\rangle_S = P(f)\left\langle \frac{P(S_i|f)}{P(S_i)}\right\rangle_{S_i} \approx \frac{P(f)\,(1-\epsilon(f))^m}{\langle P(S)\rangle} \propto P(K)(1-\epsilon(f))^m$$

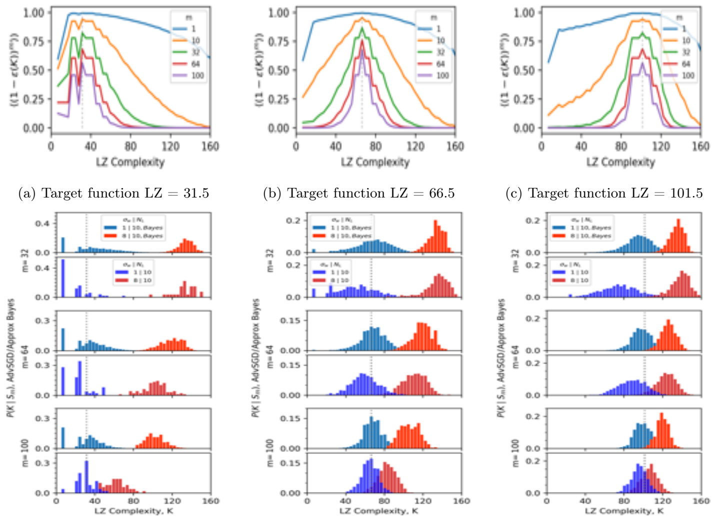
decoupling approximation





72

Bayesian picture combining data and prior



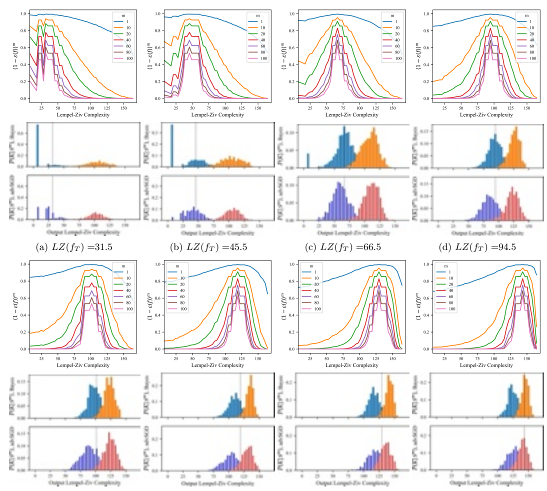
(a) Target function LZ = 31.5 (b) Target function LZ = 66.5 (c) Target function LZ = 101.5

(d) Target function LZ = 31.5 (e) Target function LZ = 66.5 (f) Target function LZ = 101.5

light blue/light red = decoupling approximation, dark blue dark red = SGD

73

Bayesian picture combining data and prior



(a) $LZ(f_T)$ =31.5 (b) $LZ(f_T)$ =45.5 (c) $LZ(f_T)$ =66.5 (d) $LZ(f_T)$ =94.5

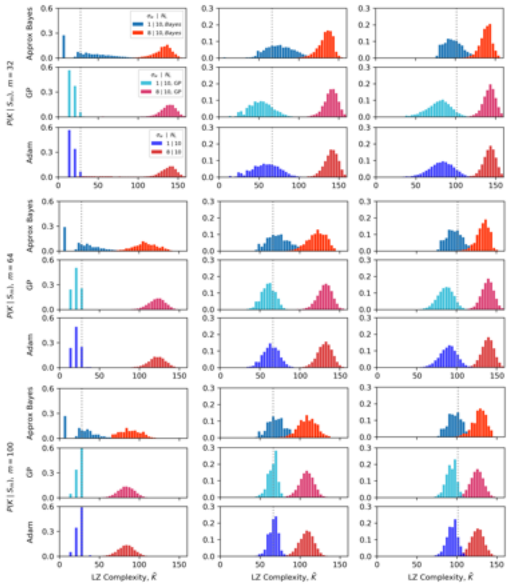
(e) $LZ(f_T)$ =101.5 (f) $LZ(f_T)$ =119.0 (g) $LZ(f_T)$ =129.5 (h) $LZ(f_T)$ =143.5

FIG. 8: **Top row** $\langle 1-\epsilon(f)\rangle^m$ versus LZ complexity. **Middle row** Approx Bayes using results from the top row. See Appendix A 4 f for full experimental details. **Bottom row** AdvSGD trained with CE loss. See Appendix A 4 e for full experimental details. $LZ(f_T)$ denotes the LZ complexity of the target function, m the number of training examples, and all other symbols have their usual meaning.

other target functions

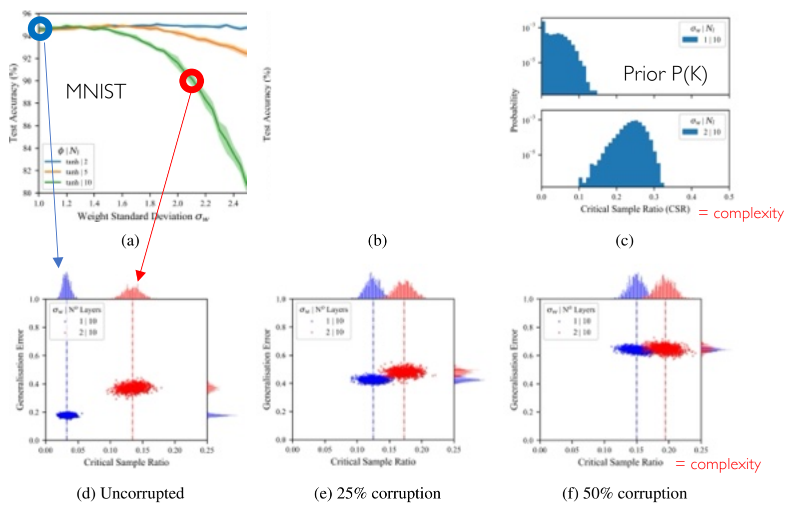
74

Full Bayesian/SGD comparison



75

Bayesian picture: prior and data for MNIST/CIFAR-10



CSR complexity, Arpit et al. arXiv:1706.05394

76

What about SGD?

<u>Hold on:</u> why should parameter function map predict DNN outcomes?

77

Two kinds of questions about generalisation:

1) Why do DNNs generalise at all in the overparameterised regime?

Because the parameter-function map is highly biased towards simple solutions.

2) Given DNNs that generalise, can we further fine-tune the hyperparameters to improve generalisation? (engineers).

Is SGD a Bayesian sampler? Well, almost, C. Mingard, G.Valle-Pérez, J. Skalse, AAL, arxiv.org: 2006.15191

78

Inductive bias and feature learning

1. Two questions about generalisation
2. Inductive bias towards simplicity
3. Bayesian function based picture of generalisation
4. Inductive bias and Zipf's law

William of Ockham
1287-1347

79

Function based picture and generalisation bounds

	Algorithm-independent (section 4.1)	Algorithm-dependent (section 4.2)		
	Based on uniform convergence	Based on non-uniform convergence		Other
Data-independent	VC dimension bound* (section 4.1.1)	SRM-based bounds† (section 4.2.1.1)	-	uniform stability bounds‡ and compression bounds§ (section 4.3.1)
Data-dependent	Rademacher complexity bound¶ (section 4.1.2)	data-dependent SRM-based bounds** (section 4.2.1.1)	margin bounds†† (4.2.1.2), sensitivity-based bounds‡‡ (section 4.2.1.4), NTK-based bounds§§ (section 4.2.1.3), other PAC-Bayes bounds¶¶ (section 4.2.2)	non-uniform stability bounds*** (section 4.3.1), marginal-likelihood PAC-Bayes bound††† (section 5)

Table 1: Classification of the main types of generalization bounds treated in this paper. Roughly speaking, the number of assumptions grows going from left to right, and from top to bottom. Note that, as we discussed in section 3.3.4, algorithm dependent bounds based on non-uniform convergence are automatically data-dependent, which is why there is an empty cell.

*Vapnik and Chervonenkis (1974); Blumer et al. (1989); Harvey et al. (2017)
†Vapnik (1995); McAllester (1998)
‡Bousquet and Elisseeff (2002); Hardt et al. (2016); Mou et al. (2018)
§Littlestone and Warmuth (1986); Brutzkus et al. (2018)
¶Bartlett and Mendelson (2002)
**Shawe-Taylor et al. (1998); Shawe-Taylor and Williamson (1997)
††Bartlett (1997, 1998); Bartlett et al. (2017); Neyshabur et al. (2018a); Golowich et al. (2018); Neyshabur et al. (2018b); Barron and Klusowski (2019)
‡‡Neyshabur et al. (2017); Dziugaite and Roy (2017); Arora et al. (2018); Banerjee et al. (2020)
§§Arora et al. (2019); Cao and Gu (2019)
¶¶Zhou et al. (2018); Dziugaite and Roy (2018)
***Kuzborskij and Lampert (2017)
†††Valle-Pérez et al. (2018)

Big review paper on generalization bounds, includes 7 desiderata bounds should satisfy and a classification

Generalization bounds for deep learning Guillermo Valle-Pérez and AAL, arxiv:arXiv:2012.04115

80

Function based picture and PAC-Bayes bounds

$$\forall \mathcal{D},\ \mathbf{P}_{S\sim\mathcal{D}^m}\left[\forall Q\ KL(\mathbf{E}_{h\sim Q}[\epsilon(h)], \mathbf{E}_{h\sim Q}[\hat{\epsilon}(h)]) \leq \frac{KL(Q||P) + \ln\frac{1}{\delta} + \ln(2m)}{m-1}\right] \geq 1-\delta \tag{13}$$

where $KL(Q||P)$ is the KL-divergence between Q and P. On the left hand side we use the standard abuse of notation to define $KL(a,b) \equiv a\ln(a/b) + (1-a)\ln((1-a)/(1-b))$, for $a, b \in [0,1]$.

David McAllester COLT (1998)

We prove that function based bounds will (in principle) always be better than parameter based PAC-Bayes bounds

$$KL(Q||P) \leq KL(Q_{\text{par}}||P_{\text{par}})$$

Generalization bounds for deep learning Guillermo Valle-Pérez and AAL, arxiv:arXiv:2012.04115

81

Function based picture and PAC-Bayes bounds

Theorem 5.1. *(marginal-likelihood PAC-Bayes bound)*
For any distribution P on any hypothesis space $\mathcal{H}$ and any realizable distribution $\mathcal{D}$ on a space of instances we have, for $0 < \delta \leq 1$, and $0 < \gamma \leq 1$, that with probability at least $1-\delta$ over the choice of sample S of m instances, that with probability at least $1-\gamma$ over the choice of h:

$$-\ln(1-\epsilon(h)) < \frac{\ln\frac{1}{P(C(S))} + \ln m + \ln\frac{1}{\delta} + \ln\frac{1}{\gamma}}{m-1}$$

where h is chosen according to the posterior distribution $Q(h) = \frac{P(h)}{\sum_{h\in C(S)} P(h)}$, $C(S)$ is the set of hypotheses in $\mathcal{H}$ consistent with the sample S, and where $P(C(S)) = \sum_{h\in C(S)} P(h)$

1) Marginal-likelihood P(c(S)) = weighted sum over functions (hypotheses) h consistent with training set S.

2) P(c(S)) can also be interpreted as the probability of obtaining zero error on S upon random sampling of parameters.

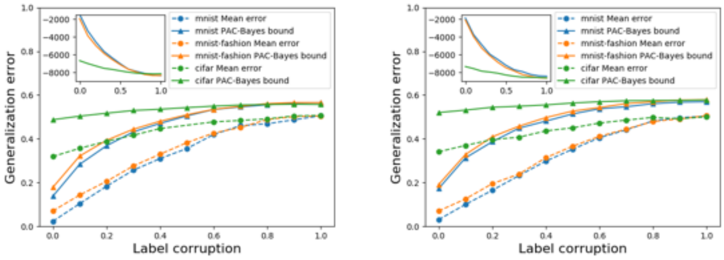
3) P(c(S)) is a natural measure of the "fit" of the inductive bias of the network with the data.

4) Larger P(c(S)) means smaller PAC-Bayes generalisation bound.

Generalization bounds for deep learning Guillermo Valle-Pérez and AAL, arxiv:arXiv:2012.04115

82

Tight PAC-Bayes bounds: error with complexity



(a) for a 4 hidden layers convolutional network

(b) for a 1 hidden layer fully connected network

Guillermo
Valle
Perez

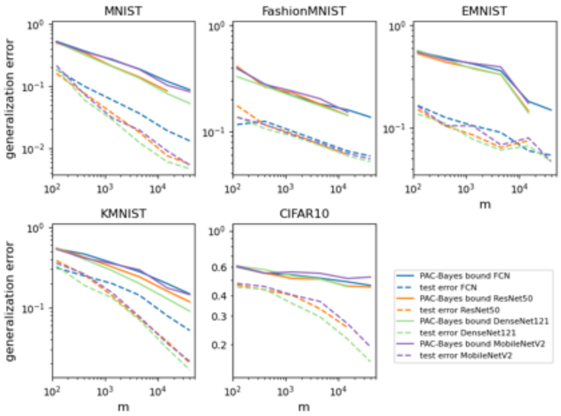
Marginal-likelihood PAC-Bayes bound

$$-\ln\left(1-\epsilon(h)\right) < \frac{\ln\frac{1}{P(C(S))}+\ln m+\ln\frac{1}{\delta}+\ln\frac{1}{\gamma}}{m-1}$$

Generalization bounds for deep learning Guillermo Valle-Pérez and AAL, arxiv:arXiv:2012.04115

83

Tight PAC-Bayes bounds: learning curves with m



Guillermo
Valle
Perez

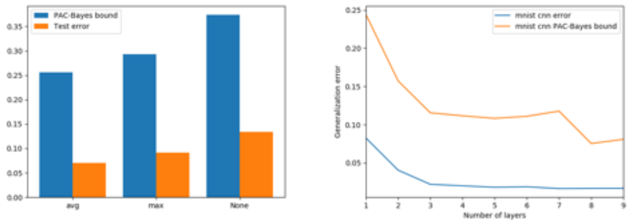
Marginal-likelihood bound

$$-\ln\left(1-\epsilon(h)\right) < \frac{\ln\frac{1}{P(C(S))}+\ln m+\ln\frac{1}{\delta}+\ln\frac{1}{\gamma}}{m-1}$$

Generalization bounds for deep learning Guillermo Valle-Pérez and AAL, arxiv:arXiv:2012.04115

84

Tight PAC-Bayes bounds: comparing architectures



(a) (b)

Figure 6: **PAC-Bayes bound and generalization error versus different architecture hyperparameters**. (a) Error versus pooling type, for a CNN trained on a sample of 1k images from KMNIST. (b) Error versus number of layers for a CNN trained on a sample of size 10k from MNIST. Training set error is 0 in all experiments. We used SGD with batch 32 for both of these experiments.

Guillermo Valle Perez

Marginal-likelihood bound

$$-\ln\left(1-\epsilon(h)\right) < \frac{\ln\frac{1}{P(C(S))}+\ln m+\ln\frac{1}{\delta}+\ln\frac{1}{\gamma}}{m-1}$$

Generalization bounds for deep learning Guillermo Valle-Pérez and AAL, arxiv:arXiv:2012.04115

85