Why NN has (# params) capacity to memorize but choose to learn?

How well NN learn is measured by how well it generalize

test data should have same dist as training test

out of sample generalization : generalize to data with different dist as training

Deep Learning Theory and Applications

Learning, Memorization and Generalization

CPSC/AMTH 663

Yale

Outline

1. Classic Generalization Bound
2. Effective Capacity
3. Overparameterization
4. Double Descent Curve
5. Memorization vs Interpolation
6. Smoothness
7. Norm-based regularization
8. Implicit SGD regularization
9. Coherent Gradients

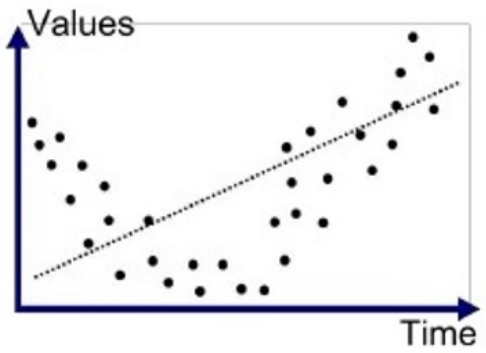
Generalization

- Underfit: training error high, test error high
- Good fit: training error low, test error low
- Overfit: training error low, test error high

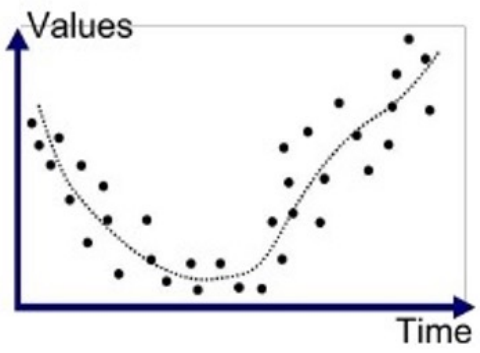
- Generalization is the ability for a model to perform well on test data, i.e. generalize its results from training to test

- Thus classically people have linked generalization to model capacity
- Ex: regularizations motivated as ability to reduce model capacity and therefore increase generalizations

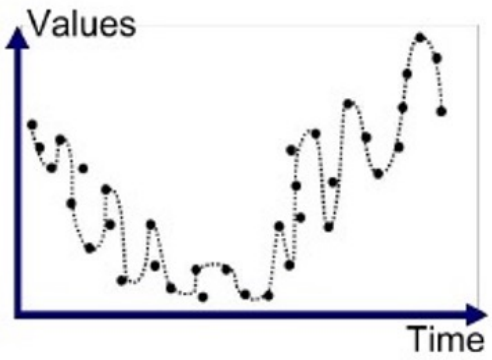
Classic notion of over/underfit



Underfitted



Good Fit/Robust



Overfitted

bad extrapolate

solution : decrease model capacity

Connects generalization of a fit to model capacity

Classic optimization

Model or function complexity, e.g., VC, margin or $\|f\|_{\mathcal{H}}$

Expected risk: what you get

Empirical risk: what you see

$$E(L(f^*_{ERM}, y)) \leq \frac{1}{n}\sum L\left(f^*_{ERM}(x_i), y_i\right) + O^*\left(\sqrt{\frac{c}{n}}\right)$$

i.i.d assumption

regularization

Effective capacity

Capacity (complexity / expressive power / richness / flexibility)

- The effective capacity of neural networks is sufficient to "memorize" the entire dataset
- Deep networks do fine at reducing training error on random samples (noise)
  - Reshuffled labels (random labels)
  - Random images

- They form a model that is "over-parameterized", i.e. the number of parameters approaches the size of the data

params = # data

- Yet they generalize!

[Zhang et al ICLR 2017]

Rademacher complexity

Given a set $A \subseteq \mathbb{R}^m$, the **Rademacher complexity of** A is

$$\mathrm{Rad}(A) := \frac{1}{m} \mathrm{E}\left[\sup_{a \in A} \sum_{i=1}^{m} \sigma_i a_i\right]$$

$\sigma_1, \ldots, \sigma_m$ are independent random variables uniformly chosen from $\{-1, 1\}$

Rademacher complexity refers to the ability to fit random functions

loss of fit noise A

Random noise is fit with or without regularization !

model	# params	random crop	weight decay	train accuracy	test accuracy
Inception	1,649,402	yes	yes	100.0	89.05
		yes	no	100.0	89.31
		no	yes	100.0	86.03
		no	no	100.0	85.75
(fitting random labels)		no	no	100.0	9.78
Inception w/o BatchNorm	1,649,402	no	yes	100.0	83.00
		no	no	100.0	82.00
(fitting random labels)		no	no	100.0	10.12
Alexnet	1,387,786	yes	yes	99.90	81.22
		yes	no	99.82	79.66
		no	yes	100.0	77.36
		no	no	100.0	76.07
(fitting random labels)		no	no	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
(fitting random labels)		no	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
(fitting random labels)		no	no	99.34	10.61

[Zhang et al 2017]

VC Dimension

D the cardinality of the largest set of points an algo can shatter

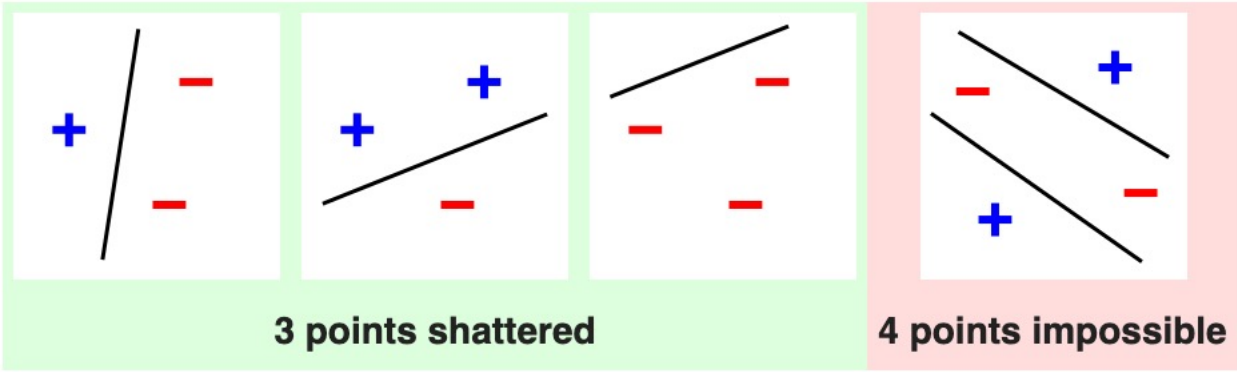
Let H be a set family (a set of sets) and C a set. Their *intersection* is defined as the following set-family:

$$H \cap C := \{h \cap C \mid h \in H\}$$

We say that a set C is *shattered* by H if $H \cap C$ contains all the subsets of C, i.e:

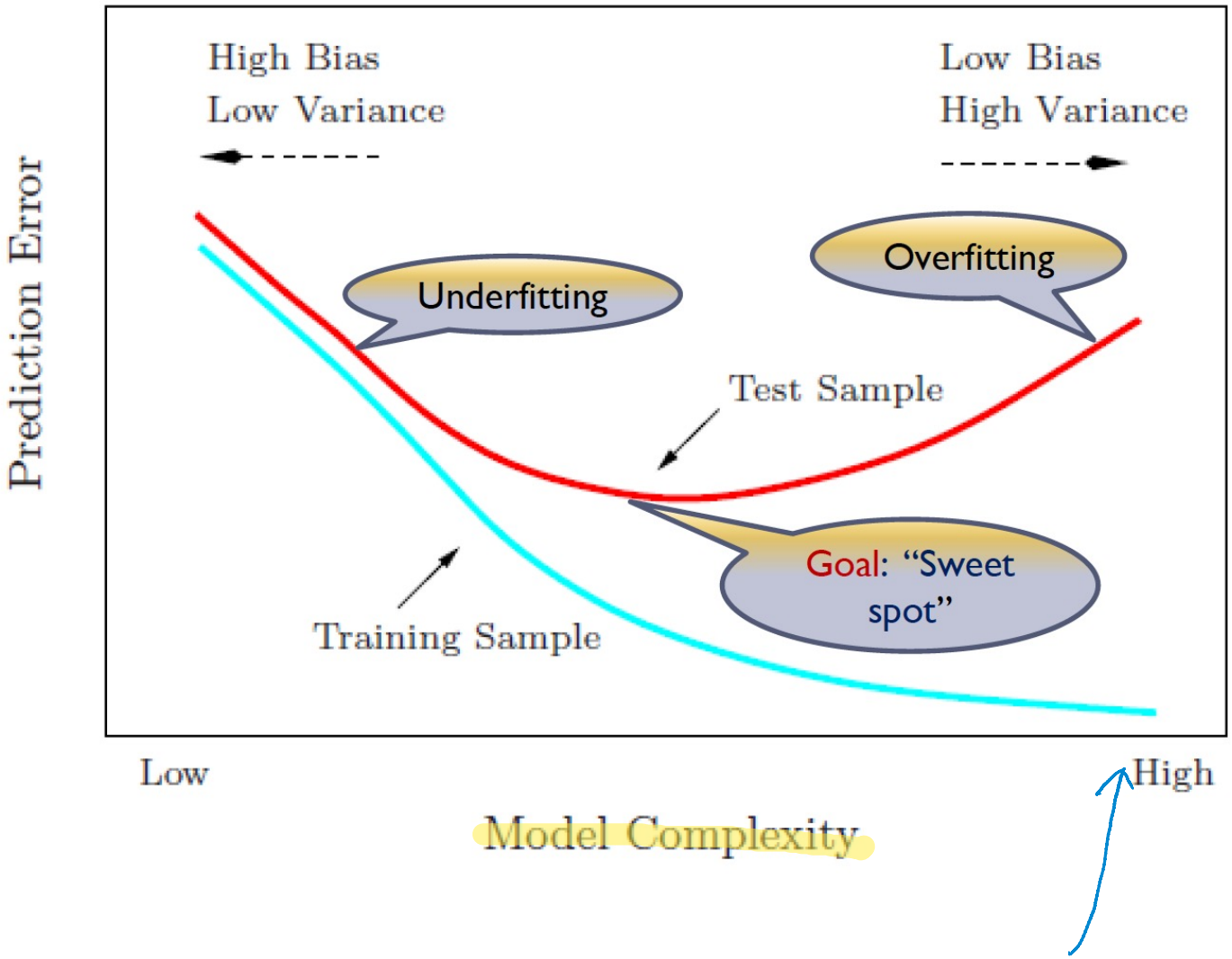
$$|H \cap C| = 2^{|C|}$$

打碎



D = 2

Classic U Curve



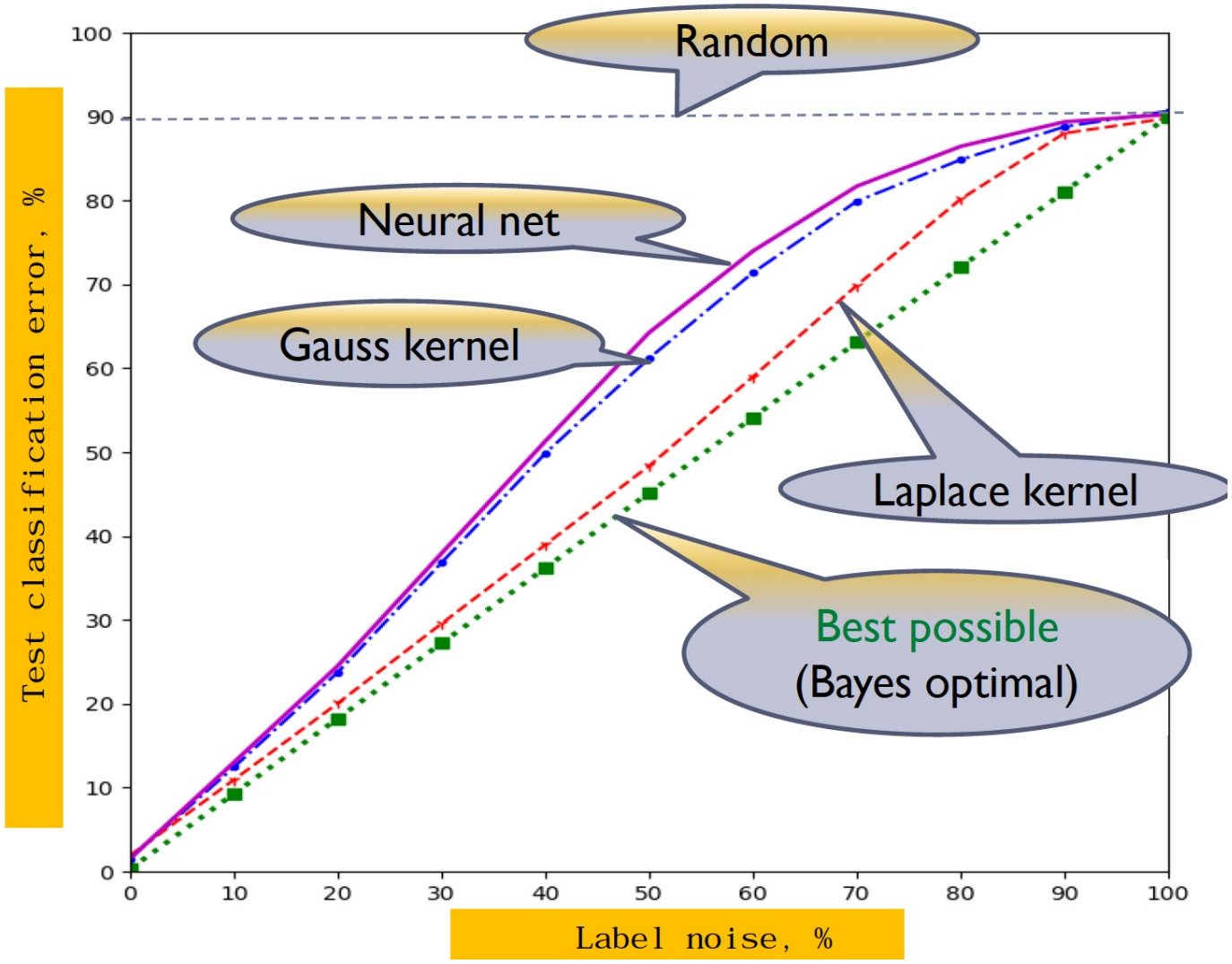
Zero training error does not generalize according to this model, it is overfit

Neural Network Behavior

model	# params	random crop	weight decay	train accuracy	test accuracy
Inception	1,649,402	yes	yes	100.0	89.05
		yes	no	100.0	89.31
		no	yes	100.0	86.03
		no	no	100.0	85.75

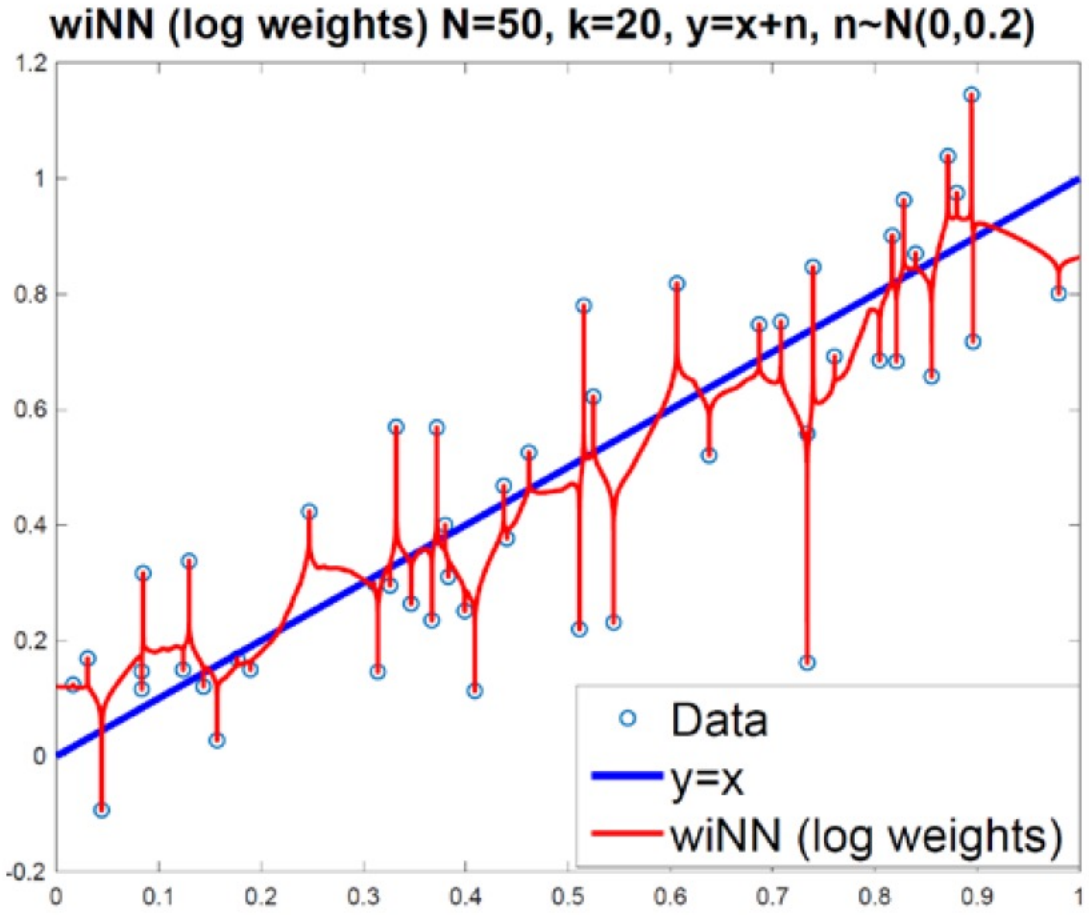
Zhang et al. 2017

Overparameterization does not overfit noisy data



1 Nearest Neighbor KNN Classifier

memorize label but generalize well bc over-parameterized



Yann Lecun:

IPAM talk, 2018

Deep learning breaks some basic rules of statistics.

Leo Breiman
Statistics Department, University of California, Berkeley, CA 94305;
e-mail: leo@stat.berkeley.edu

Reflections After Refereeing Papers for NIPS

Written in 1995

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

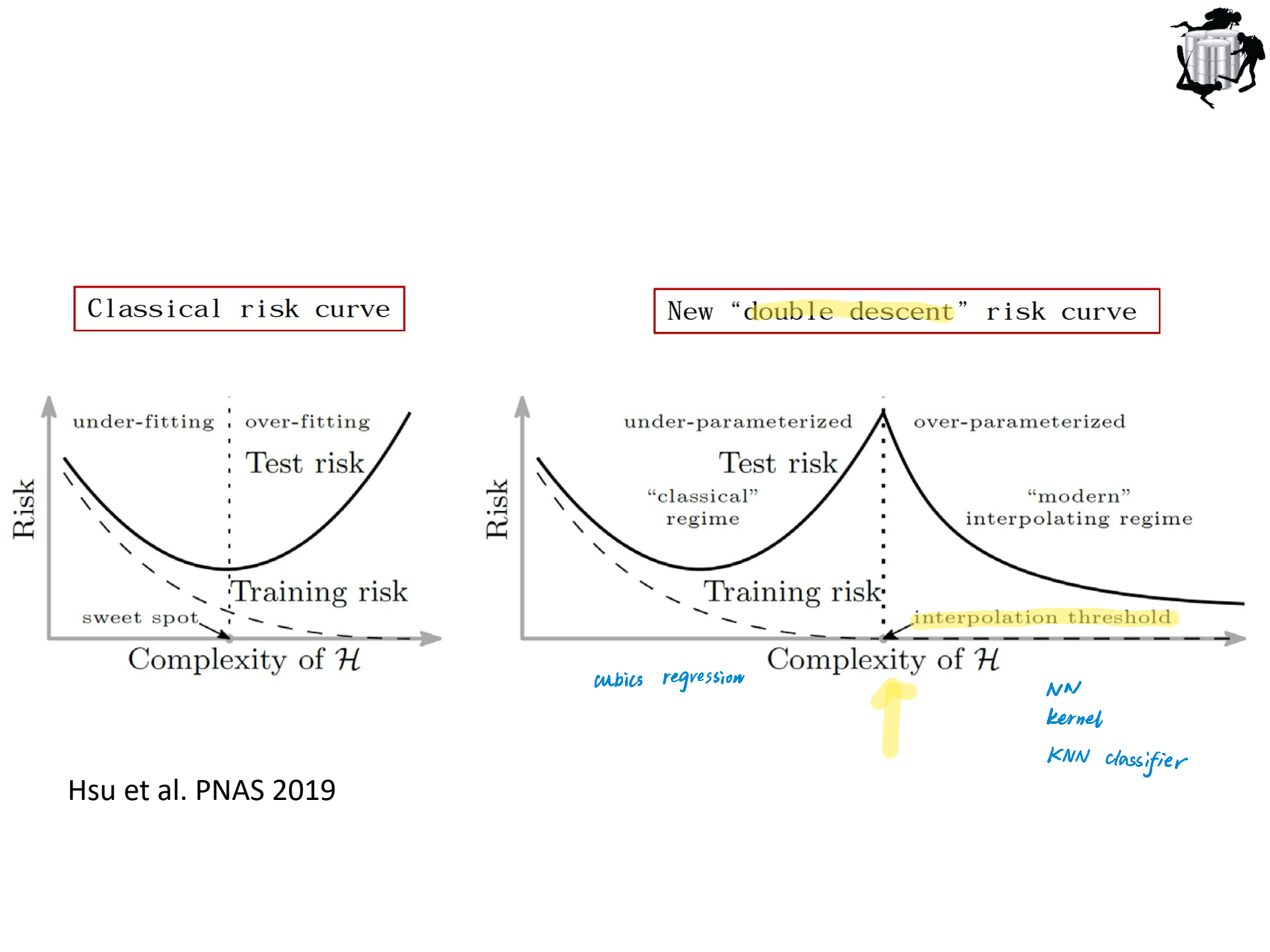
- Why don't heavily parameterized neural networks overfit the data?

Boosting the margin: A new explanation for the effectiveness of voting methods

Written in 1998

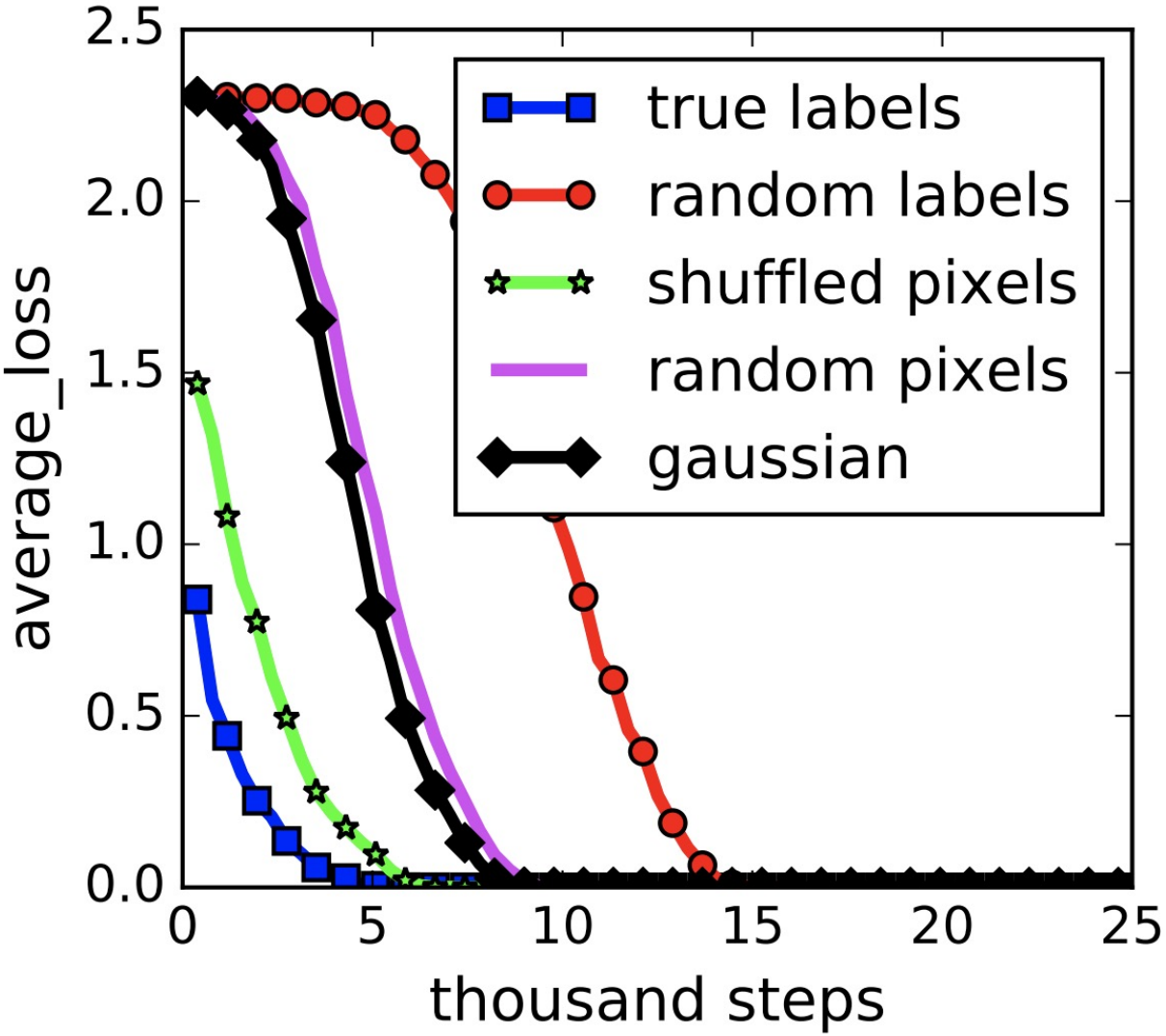
Robert E. Schapire Yoav Freund Peter Bartlett Wee Sun Lee

Abstract. One of the surprising recurring phenomena observed in experiments with boosting is that the test error of the generated hypothesis usually does not increase as its size becomes very large, and often is observed to decrease even after the training error reaches zero.



Hsu et al. PNAS 2019

Memorization = fitting to random data

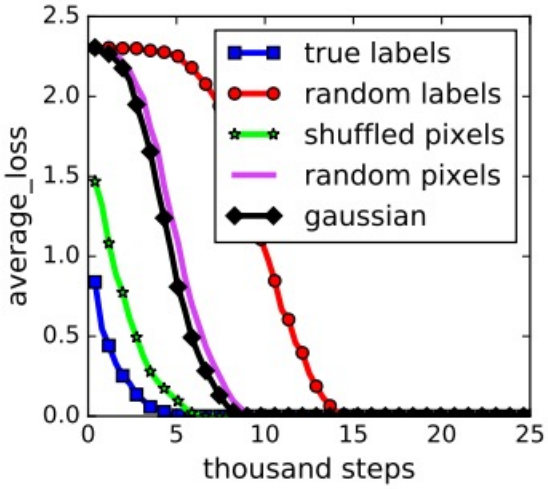


[Zhang et al. ICLR 2017]

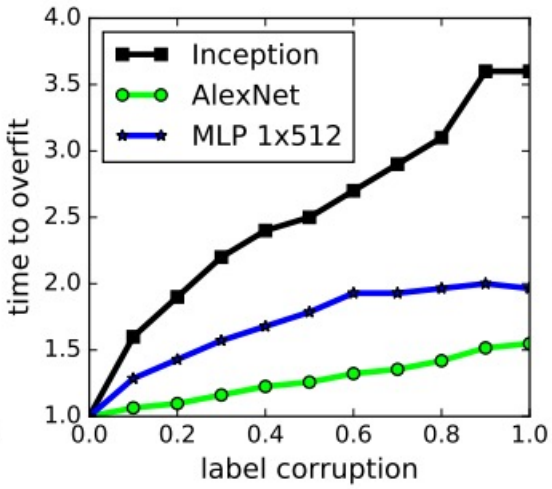
Memorization does not generalize

NN can generalize if data and label are meaningful

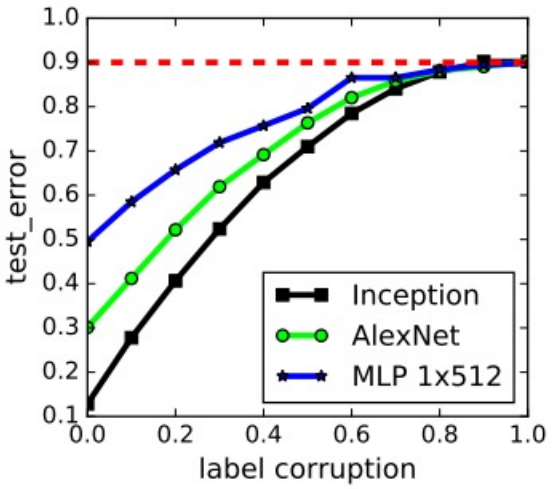
NN only memorize if data and label don't match



(a) learning curves



(b) convergence slowdown

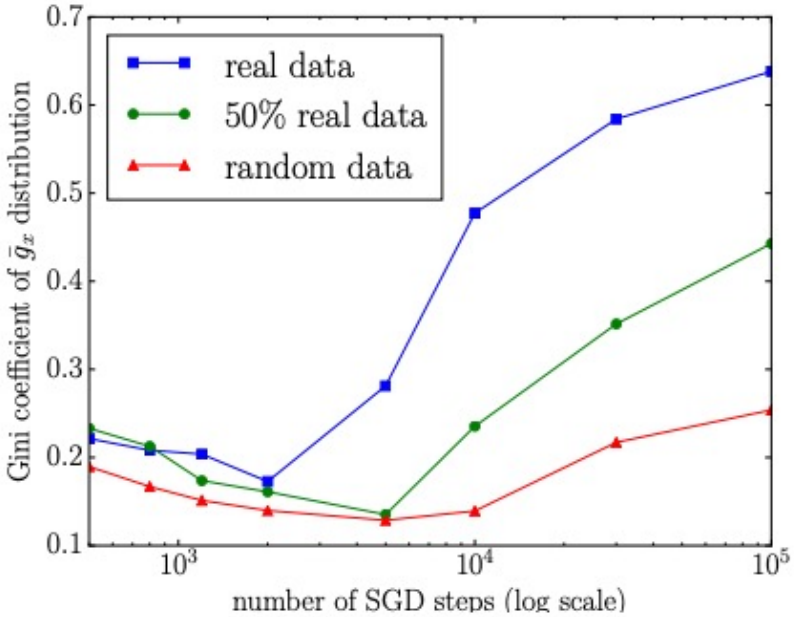


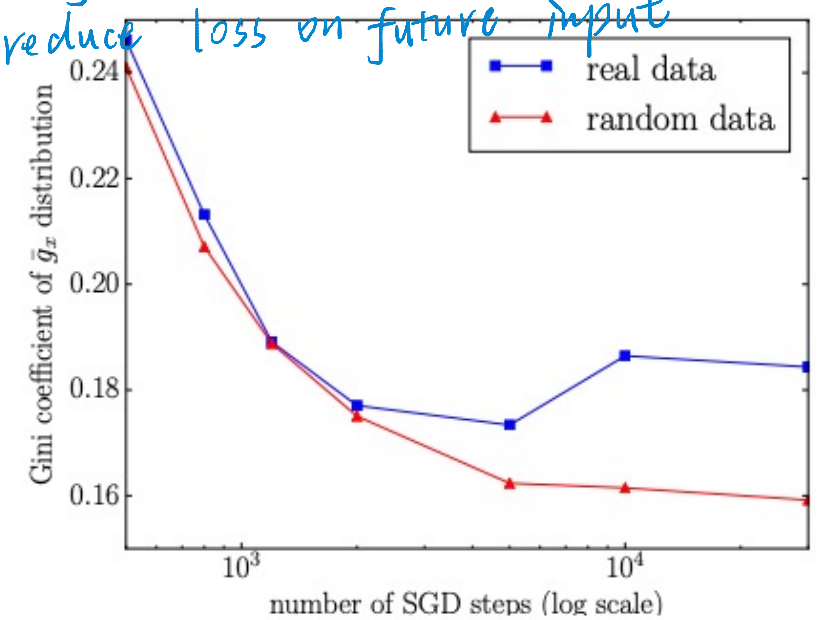
(c) generalization error growth

random label

What exactly are neural networks doing that's different?

- If data is random, SGD converge slower than real data
- bc reducing loss on current input may not reduce loss on future input



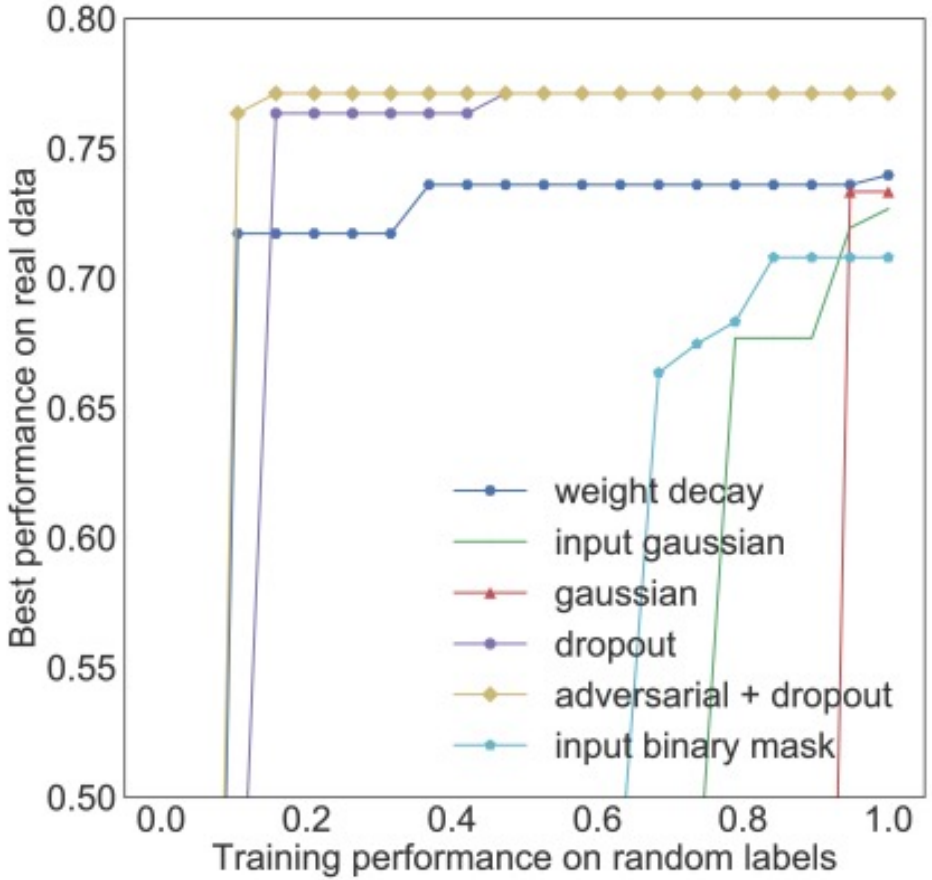


[Arpit et al 2017]

Loss gradient with each input looks different, i.e., the impact that an input has on future loss looks very different

What are regularizations doing?

overfit → goodfit



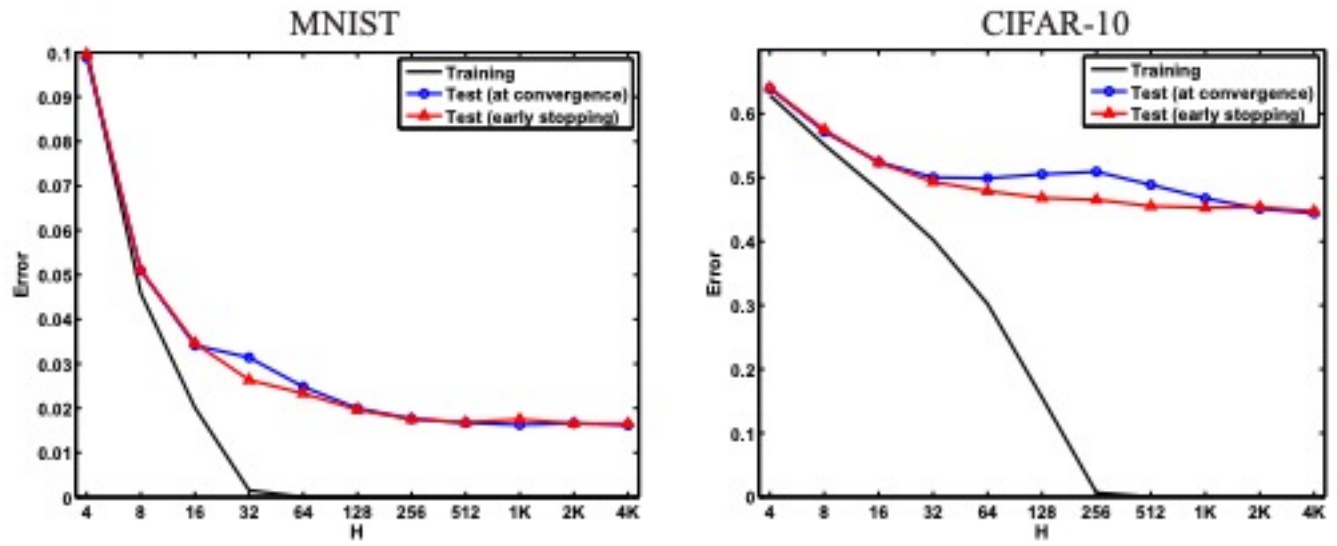
[Arpit et al. 2017[

They affect the ability to generalize but not on the ability model capacity

Rethinking Generalization

- Why do models generalize?
- When do they generalize vs memorize?
- What does regularization do---because it doesn’t really restrict capacity?

Inductive bias 归纳偏见



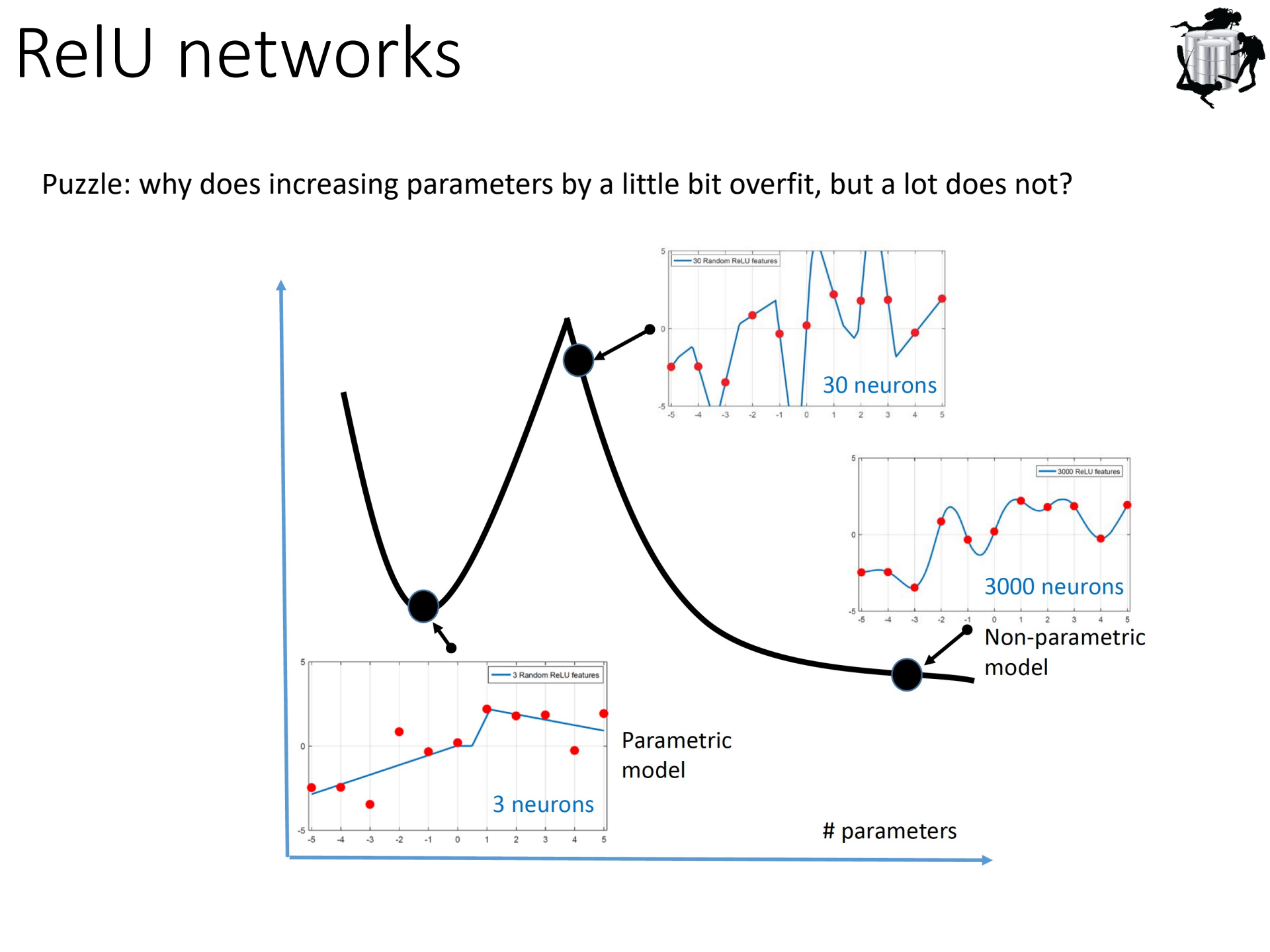
[Neyshabur et al 2015]

An inductive bias that induces some sort of capacity control (i.e. restricts or encourages predictors to be "simple" in some way), which in turn allows for generalization eg. l_2-norm restrict norm of coefficients
smoother gradient descent space (less jumps)

The success of learning then depends on how well the inductive bias captures reality

The inductive bias is not the size of the neural network, what could it be?

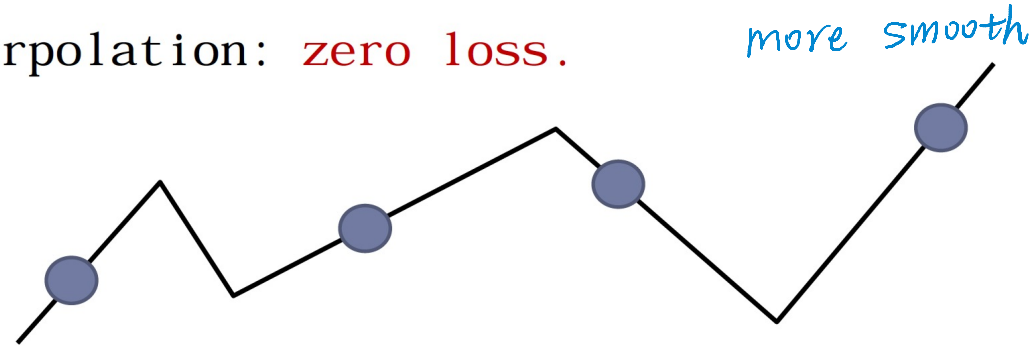
RelU networks



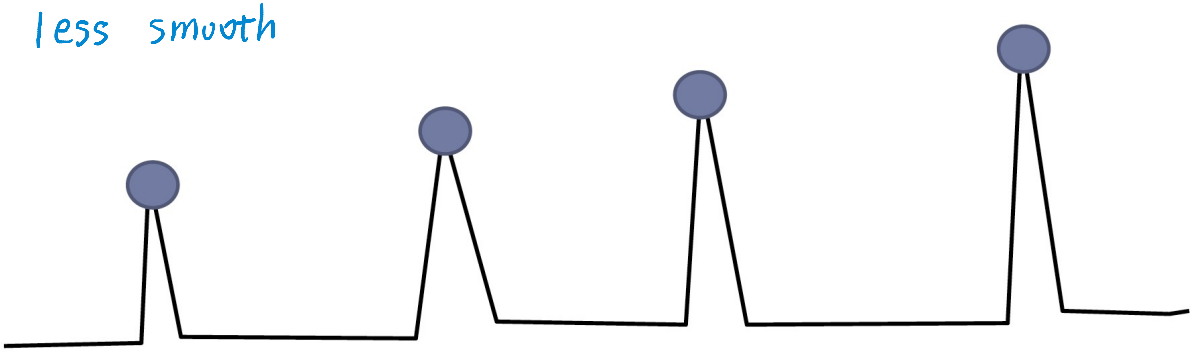
Puzzle: why does increasing parameters by a little bit overfit, but a lot does not?

Memorization vs Interpolation

Interpolation: zero loss.



Memorization: interpolation + retrieval mechanism.



- Difference between memorization and interpolation has to do with smoothness

N-->infinity

params

- Choosing the smoothest function that perfectly fits observed data is a form of Occam's razor: *The simplest explanation compatible with the observations should be preferred*
- Larger function classes (with more parameters) contain more candidate predictors compatible with the data
- => We can find interpolating functions that have smaller norm and are thus "simpler."
- I.e., increasing function class capacity improves performance of classifiers.

Smooth func : low degree polynomial with small coefficient

Kernel Methods

- Kernel methods memorize a label for a training instance (x_i, y_i) and extend beyond training data to test data x' using a similarity function k(x, x')

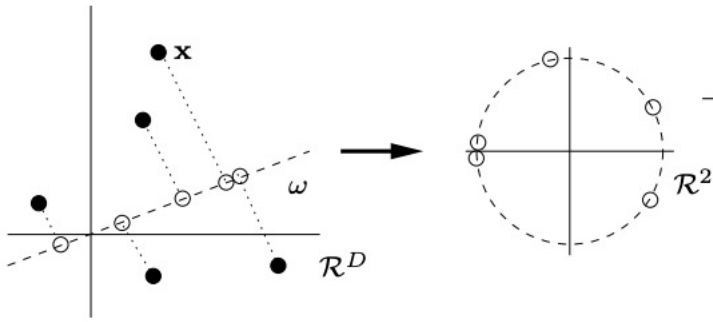
kernel function is smooth

$$\hat{y} = \operatorname{sgn} \sum_{i=1}^{n} w_i y_i k(\mathbf{x}_i, \mathbf{x}'),$$

- The Kernel trick is that this kernel function can be shown to be an inner product in a higher dimensional space as long as the similarity matrix is positive semi-definite (Mercer's Theorem)

Random Fourier Features (RFF)

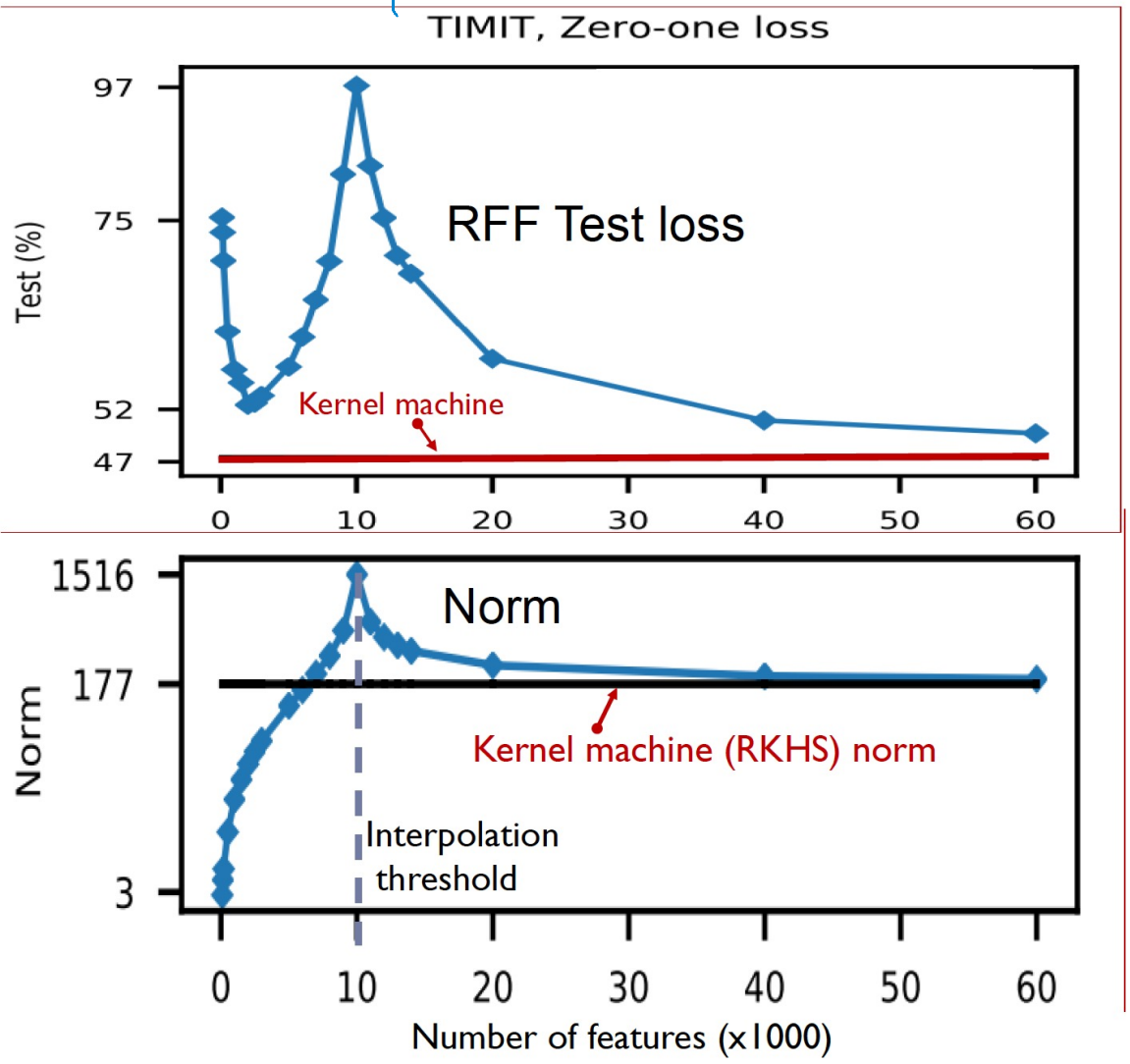
- Instead of using a similarity function kernel use a random projection into a different space and use inner products there to approximate kernels quickly



Kernel Name	$k(\Delta)$	$p(\omega)$
Gaussian	$e^{-\frac{\|\Delta\|_2^2}{2}}$	$(2\pi)^{-\frac{D}{2}} e^{-\frac{\|\omega\|_2^2}{2}}$
Laplacian	$e^{-\|\Delta\|_1}$	$\prod_d \frac{1}{\pi(1+\omega_d^2)}$
Cauchy	$\prod_d \frac{2}{1+\Delta_d^2}$	$e^{-\|\Delta\|_1}$

Kernel Methods

overparametrization



$N \to \infty$ -- infinite neural net

=

kernel machine

=

minimum norm solution

$argmin_{h \in \mathcal{H},\, h(x_i)=y_i} \; ||h||_{\mathcal{H}}$

More features ⇒

better approximation
to minimum norm solution

Richer classes have better fitting fns

- For the classes that we consider, there is no guarantee that the most regular, smallest norm predictor consistent with training data is contained in the class H with features for any finite N
- But increasing N allows us to construct progressively better approximations to that smallest norm function
- Thus highly parameterized function classes have more members where you can choose smooth solutions that fit the data
- Rich class + better inductive bias = better generalization

Other Ways to Increase Smoothness

- Explicit: Minimal functional norm solutions
- Implicit: SGD minimization
- Averaging: (boosting, bagging)

decrease sum of coefficient norm

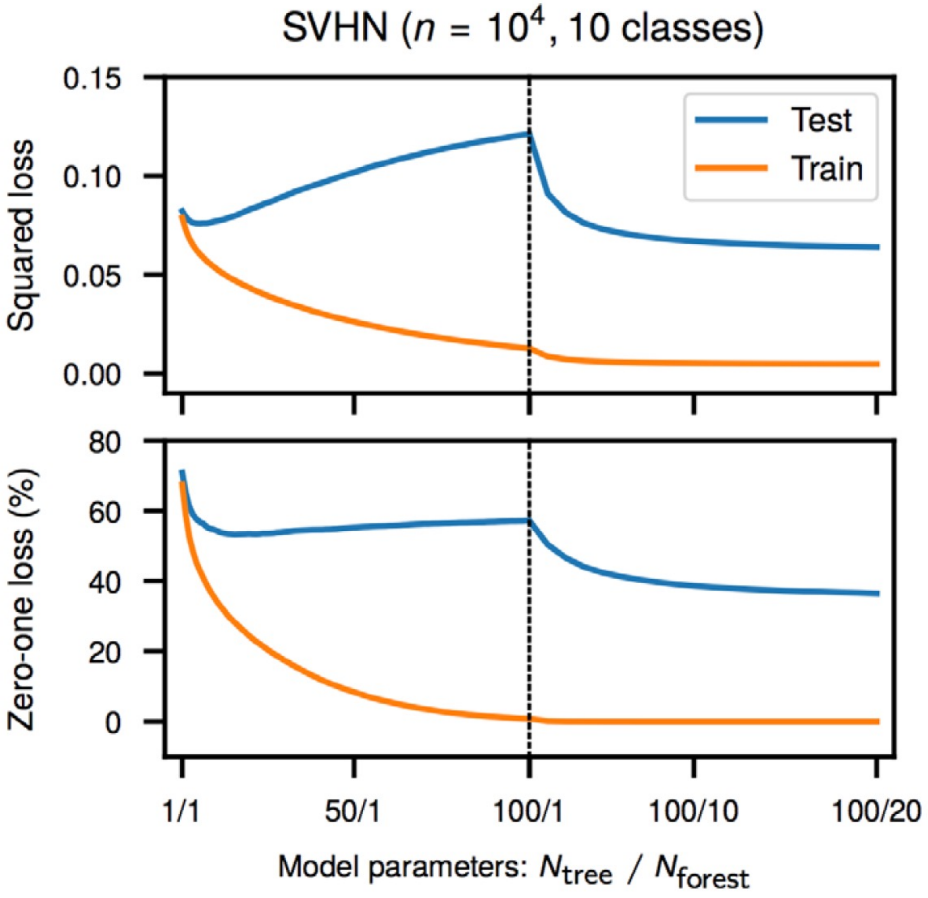
So sometimes NN w/out regularization can generalize well

over jiggy funcs

Random Forest, XGB

Drop off (train lots of NN at same time)

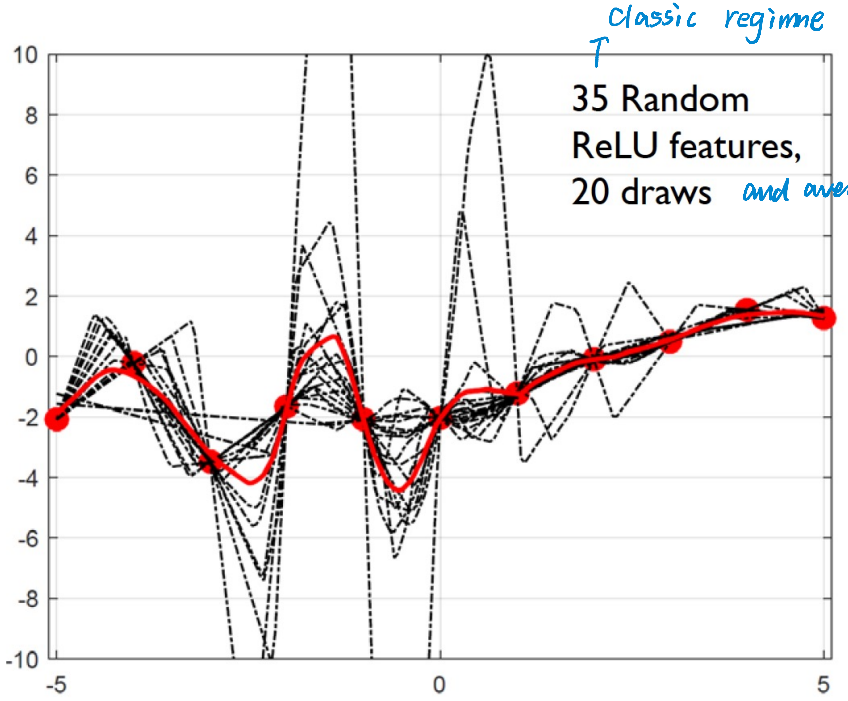
Non-neural Network Methods: Trees

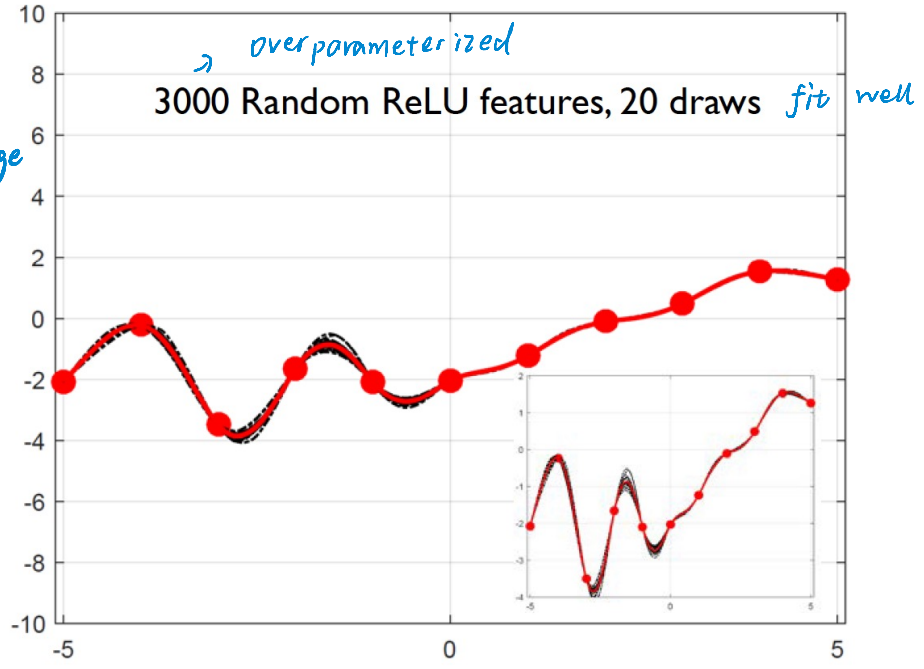


An average of interpolating trees is interpolating and better than any individual tree.

Cf. PERT [Cutler, Zhao 01]

Averaging

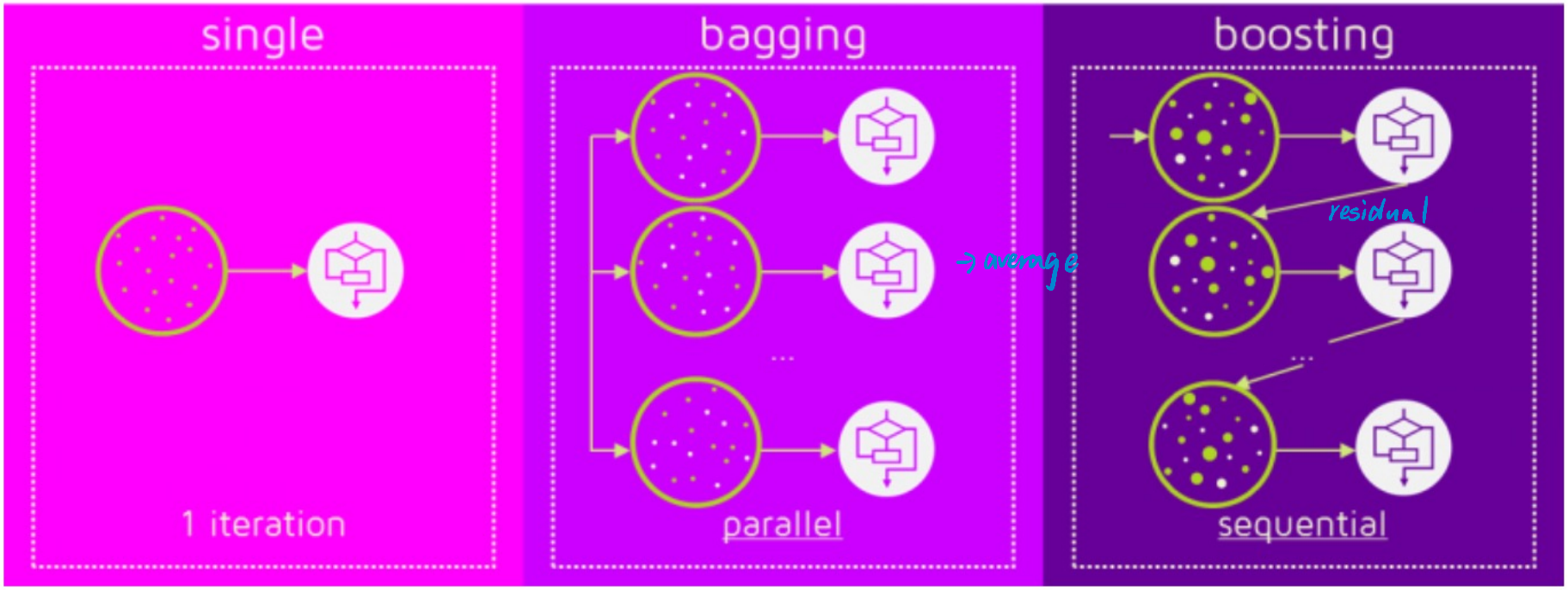




Bagging

Boosting

Bagging, Boosting



Source: Quantdare

similar to dropoff

Residual Net

similar to residual connection

New Notion of Complexity that's "norm-based"

- Neyshabur. et al. ICLR 2019 define a new notion of complexity that increases with the number of hidden units in a restricted setting (2-layer Rel U Network)
- The idea is that with a low-norm initialization hidden units only step away from the initialization by a limited amount
- Tdistance Frobenius norm, measured w.r.t. initialization $||Uk - U_0k||_F$ decreases with width of hidden layer h

An analogy to matrix factorization

矩阵分解

- Consider a neural network with a single hidden layer consisting of linear activations

↑

often 1 layer to experience

$$y[j] = \sum_{h=1}^{H} v_{hj} \langle \boldsymbol{u}_h, \boldsymbol{x} \rangle .$$

- y=Wx where $W = UV^T$
- Limiting the number of hidden units corresponds to a low rank factorization (zeros in columns of V)
- However, recently there has been a lot of success in low *norm factorizations*
- *Low Frobenius norm:*

$$\|\boldsymbol{W}\|_{tr} = \min_{\boldsymbol{W}=\boldsymbol{V}\boldsymbol{U}^\top} \frac{1}{2}(\|\boldsymbol{U}\|_F^2 + \|\boldsymbol{V}\|_F^2).$$

Frobenius norm of a matrix

$$\|\mathbf{A}\|_F \equiv \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$$

L2 norm of all matrix entries, corresponds to “volume of matrix”

Low rank vs Low norm

- Unlike the rank, the trace-norm (as well as other factorization norms) is convex, and leads to tractable learning problems
- In fact, even if learning is done by a local search over weights of the network, if the dimensionality is high enough and the norm is regularized, we can ensure convergence to a global minima [Burer, Choi 2006]
- This is in stark contrast to the dimensionality-constrained low-rank situation, where the limiting factor is the number of hidden units, and local minima are abundant [Sreebo, Jakkola, 2003]

Sensible Inductive Bias?

- A. low-trace norm model corresponds to a realistic factor model with many factors of limited overall influence
- What situations is this realistic in?
- The norm of the factorization may be a better inductive bias than the number of weights
- Perhaps learning is succeeding because there is a good representation with small overall norm, and the optimization is implicitly biasing us toward low-norm models

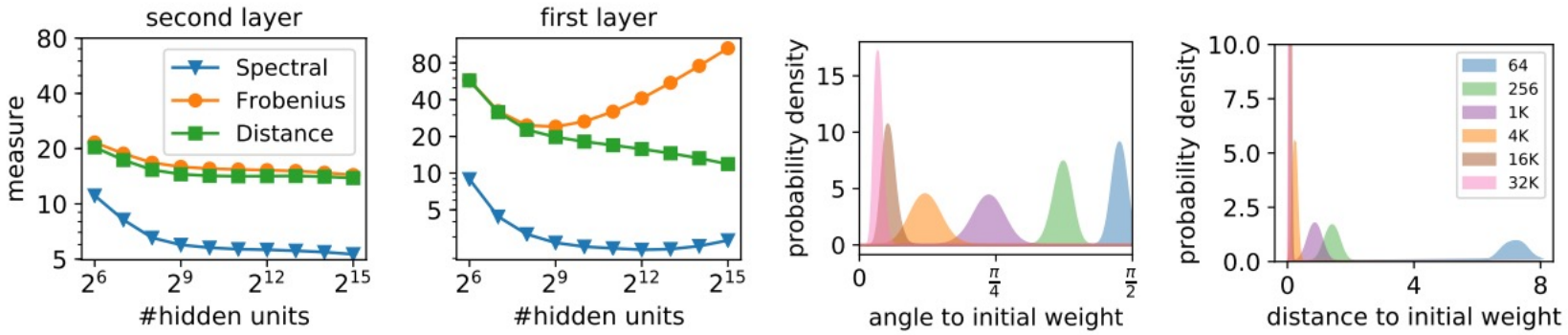
Complexity decreases with increasing hidden units

can use SGD to solve

search space is convex

give search space more smoother to search

RELU network trained on CIFAR 10



[Neyshambur 2019]

It can be proven in certain cases (like matrix factorization) that SGD converges to the lowest norm solution

Gradient Descent itself a regularization?

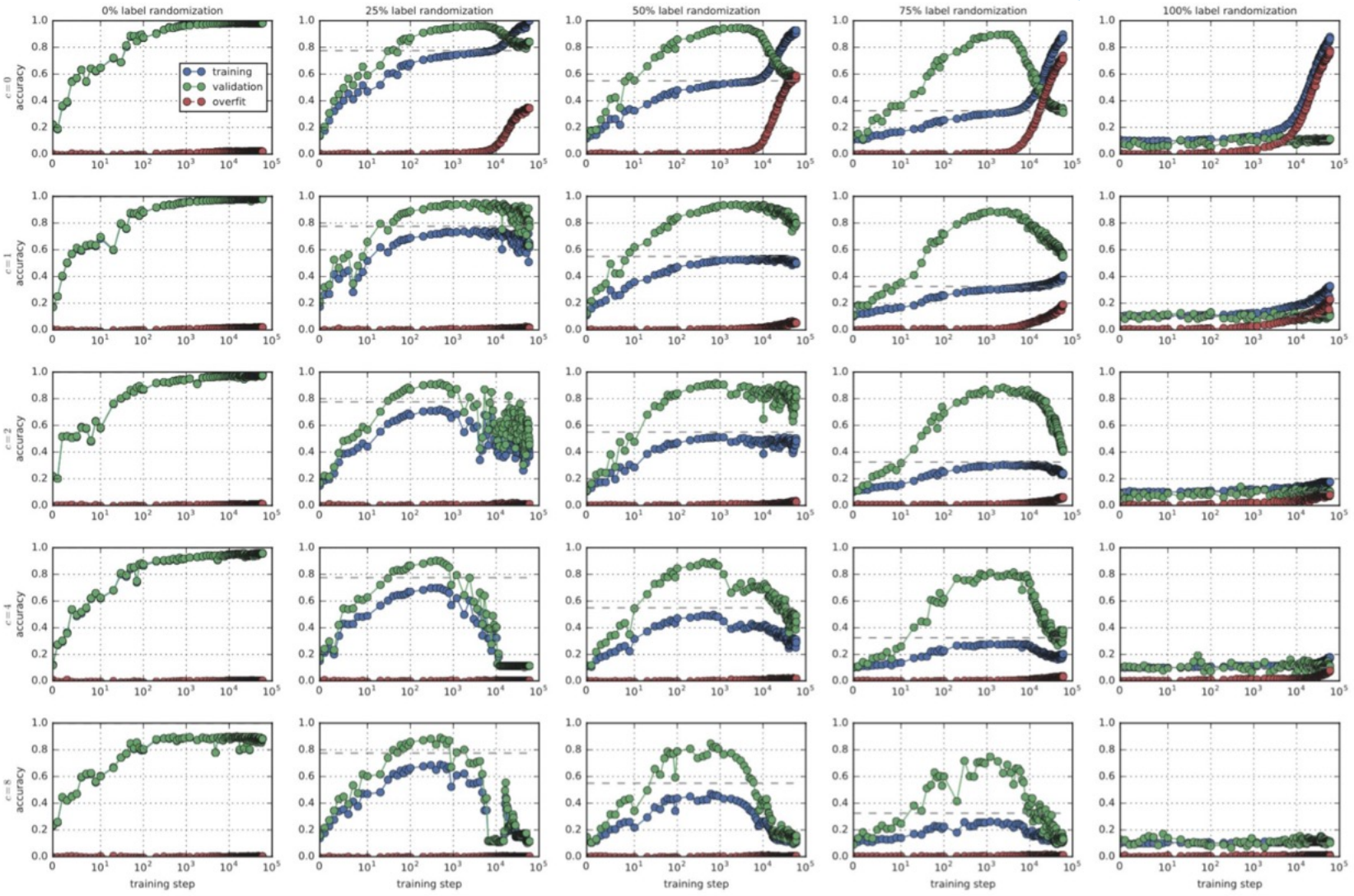
- [Hardt et. Al. 2016] Any model trained with stochastic gradient method in *a reasonable* amount of time attains small generalization error
- A randomized algorithm **A** is *uniformly stable* if for all data sets differing in only one element, the learned models produce nearly the same predictions
- Stochastic gradient descent is uniformly stable

- Results rest on an important theorem that ***uniform stability implies generalization in expectation [***O. Bousquet and A. Elisseeff. 2002 Stability and generalization]

Suppressing weak gradients

cut off part of gradient on some direction which is small

gradient : high dim vector



SGD is uniformly stable for a "reasonable" number of steps

- Proof idea: analyze the output of the algorithm on two data sets that differ in precisely one point. Note that if the loss function is L-Lipschitz for every example z.

$$\mathbb{E}\,|f(w;z) - f(w';z)| \leq L\,\mathbb{E}\,\|w - w'\| \text{ for all } w \text{ and } w'$$

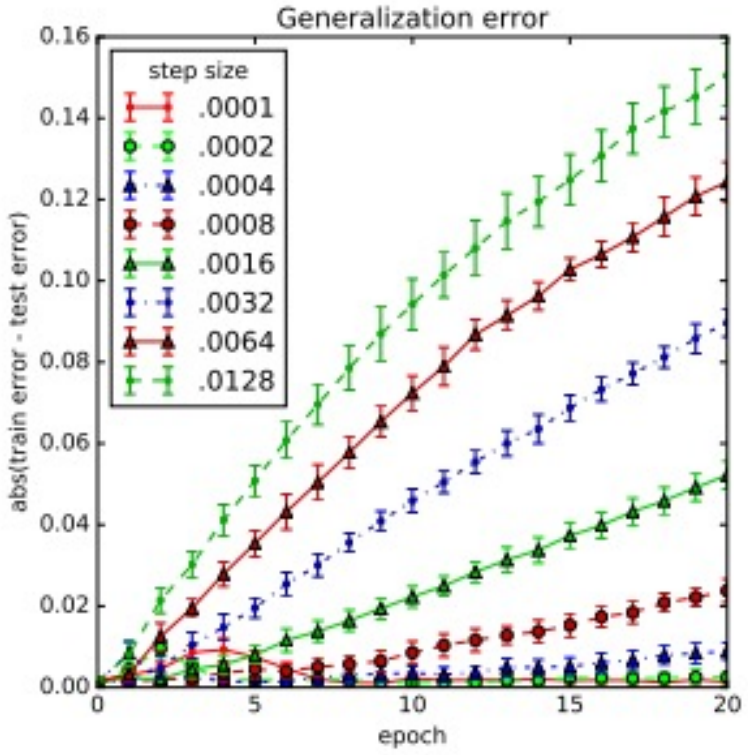
loss — step size

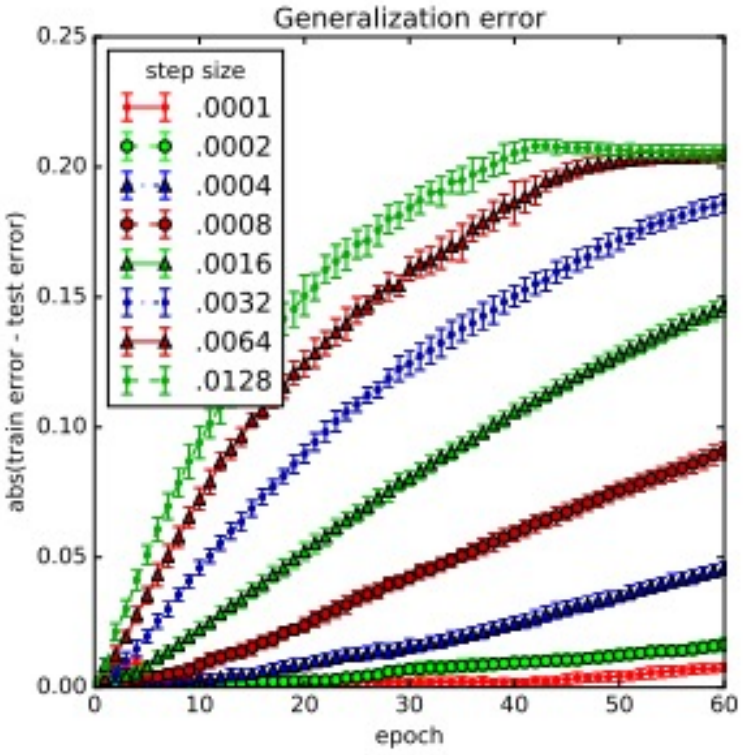
- Halving the step size roughly halves the generalization error

$\frac{1}{2}\eta \rightarrow \frac{1}{2}$ loss

- This behavior is fairly consistent for both generalization error defined with respect to classification accuracy and cross entropy. [Hardt et al. 2016]

Step-size vs generalization error





Infinite sized norm-regularized networks?

- Global weight decay, i.e. adding a regularization term that penalizes the sum of squares of all weights in the network
- Weight decay can often improves generalization but it also improves stability! [Krogh et al. NIPS 1992]

Important Property of SGD

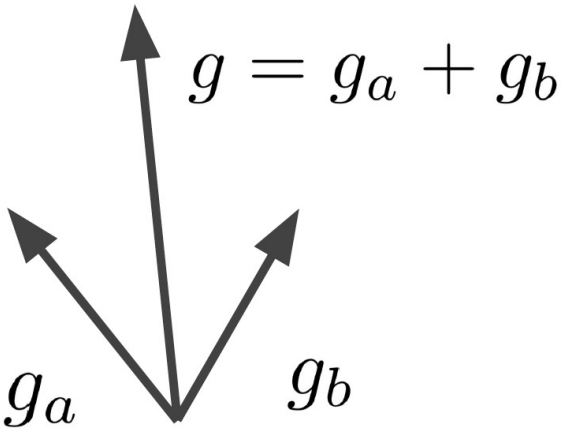
The overall gradient is the sum of gradients of individual examples

Before proceeding, let's simplify notation a bit by using *g* to denote gradients.

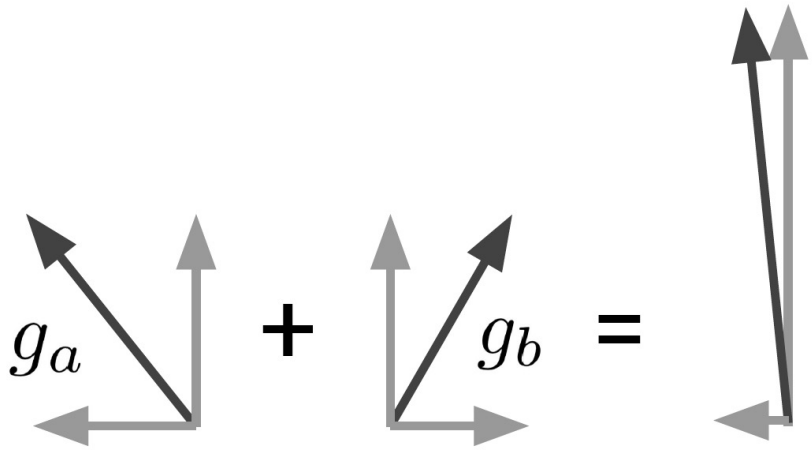
$$g_t = \sum_i g_{ti}$$

Thought Experiment

Consider the overall gradient of two examples *a* and *b*.



Now, in terms of components,



The gradient is stronger in the common direction that improves both *a* and *b*

Gradients Stronger in Common Directions

Per-Example Gradients are Coherent

Gradients of "similar" examples are similar and gradients of "dissimilar" examples are dissimilar.

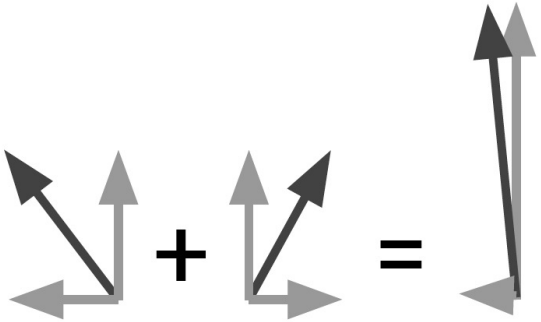
Overall Gradient is Stronger in Some Direction

Gradients of similar examples reinforce each other whereas those of dissimilar examples do not.

Parameter updates are biased to benefit multiple examples

Since parameter updates are proportional to gradients in gradient descent.

Those improvements which favor multiple examples get picked over those that only favor one example



Clipping Gradients

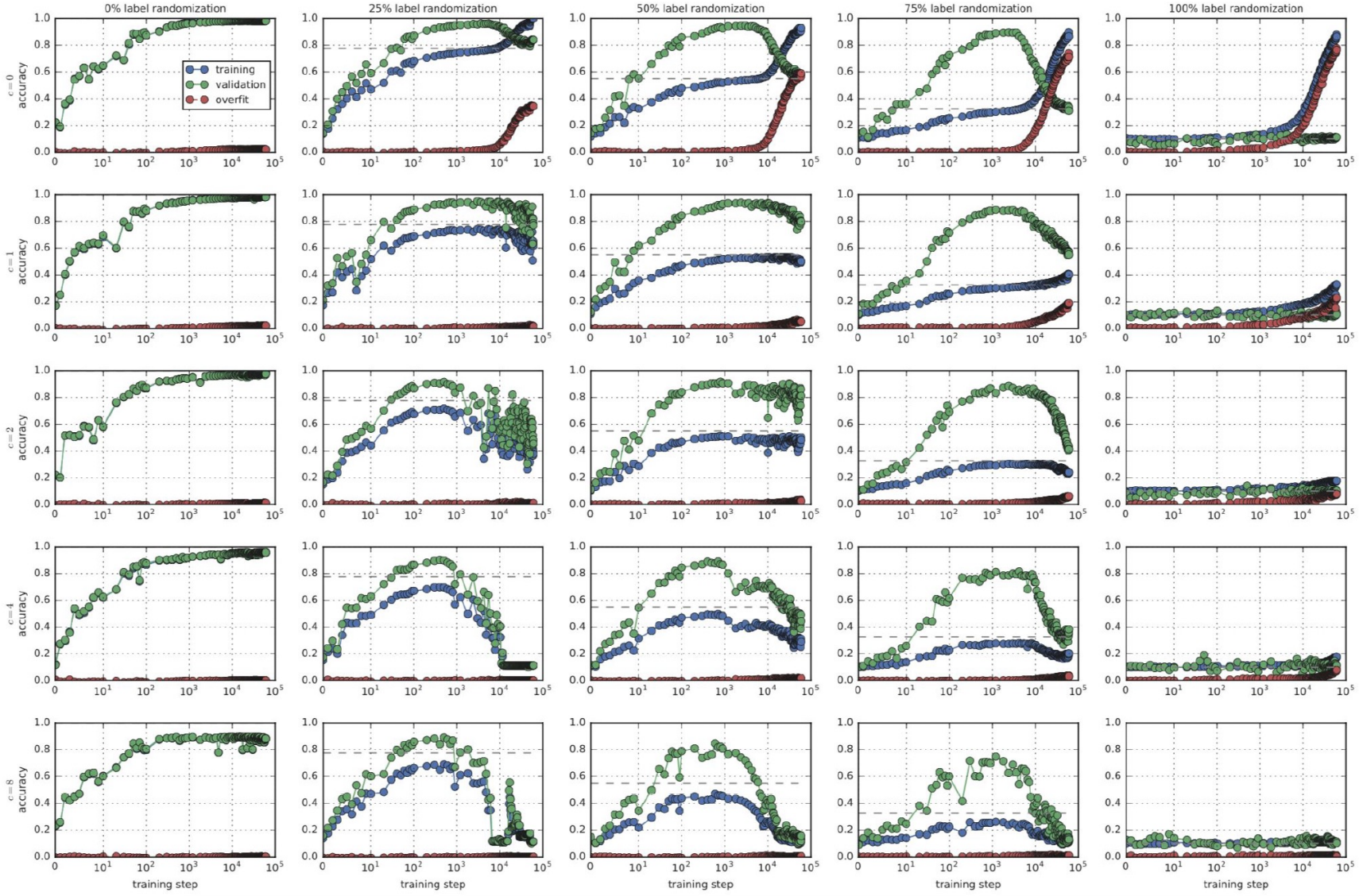
- Working in coordinates we have normally $g_w = \sum g_{we}$
- Instead, we modify it to $g_w^c := \sum_e \text{clip}(g_{we}, l_w, u_w)$

where l_w is the c-th percentile across all examples and u_w is (100-c)-th percentile

prevent over-fitting

converge faster : bc NN will stop learn if you lots of outliers

Reduces Overfitting



Further reading

- Belkin, et al. 2019 PNAS Reconciling Modern Machine-Learning Practice and The Classical Bias-Variance Tradeoff
- Understanding Deep Learning requires rethinking generalization, Zhang et al ICLR 2017 https://arxiv.org/pdf/1611.03530.pdf
- A closer look at memorization in deep networks [Arpit et al 2017] https://arxiv.org/pdf/1706.05394.pdf
- In search of the real inductive bias: on the role of implicit regularization in deep learning [Neyshabur et al, 2015] https://arxiv.org/pdf/1412.6614.pdf
- The role of over-parametrization in generalization of neural networks [Neyshabur et al, 2019] https://arxiv.org/pdf/1805.12076.pdf
- Train faster, generalize better: stability of stochastic gradient descent [Hardt et al, 2016] https://arxiv.org/pdf/1509.01240.pdf
- Stability and generalization [Bousquet and A. Elisseeff 2002] https://www.academia.edu/13743279/Stability_and_generalization
- Rademacher complexity: http://www.cs.cmu.edu/~ninamf/ML11/lect1117.pdf
- Chatterjee S. ICLR 2020 Coherent Gradients: An approach to understanding generalization in gradient descent-based optimization https://arxiv.org/abs/2002.10657