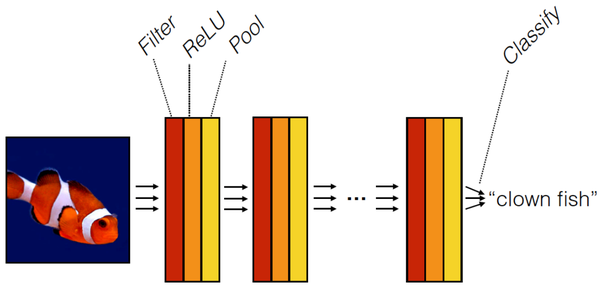
Dynamics and Generalization in deep neural networks

TOMASO POGGIO, MIT, CBMM

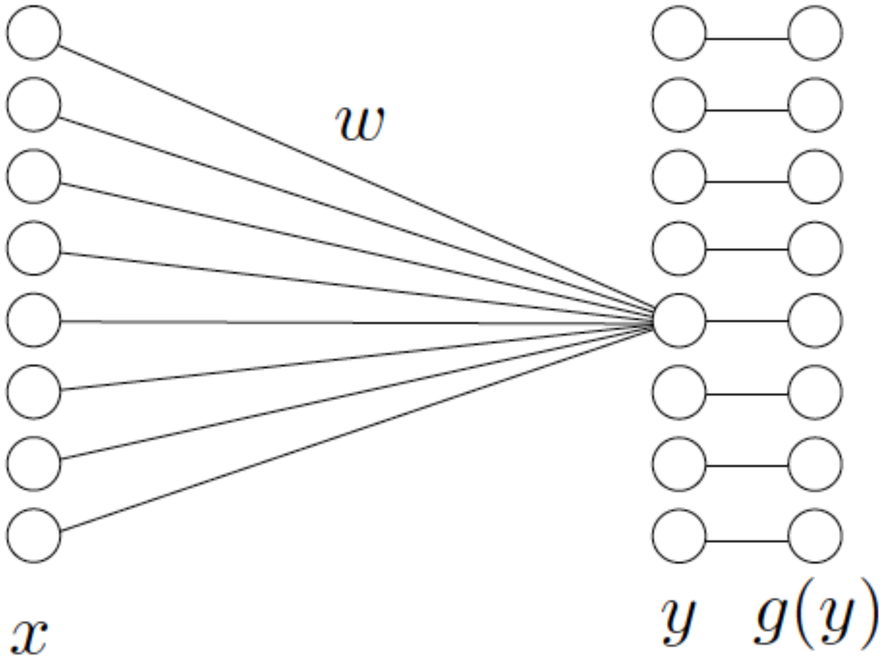
*WITH HELP FROM HRUSHIKESH MHASKAR, **ANDY BANBUSKI**, QIANLI LIAO, BRANDO MIRANDA AND LORENZO ROSASCO*

Computation in a deep neural net

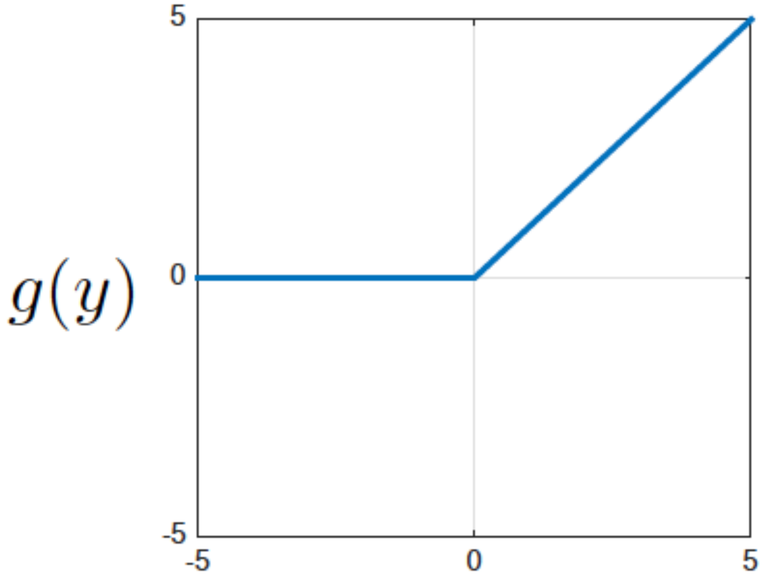
Computation in a neural net



$$f(\mathbf{x}) = f_L(\dots f_2(f_1(\mathbf{x})))$$



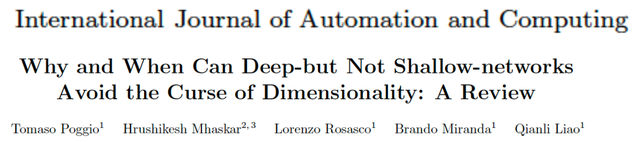
Rectified linear unit (ReLU)



$$g(y) = \max(0, y)$$

It is a bit of a mystery why deep learning works

- *Approximation Theory:* When and why are deep networks better than shallow networks?

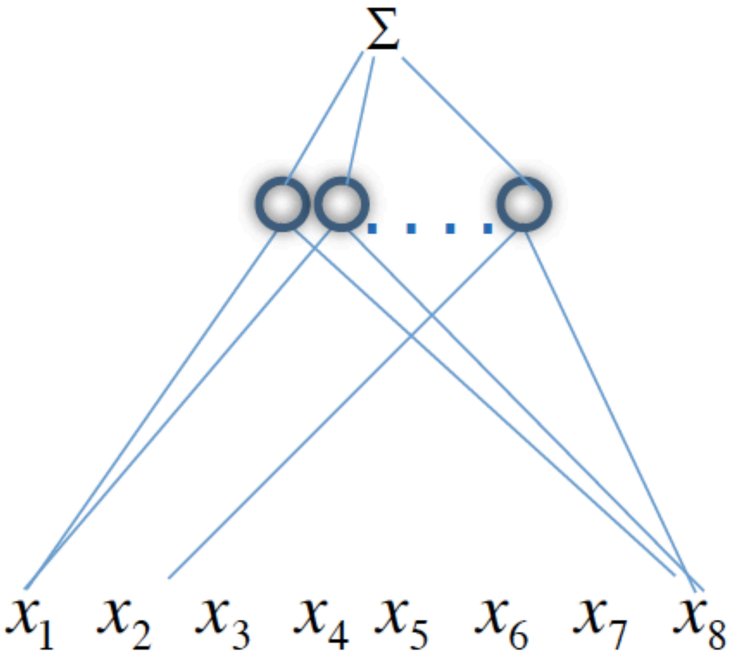


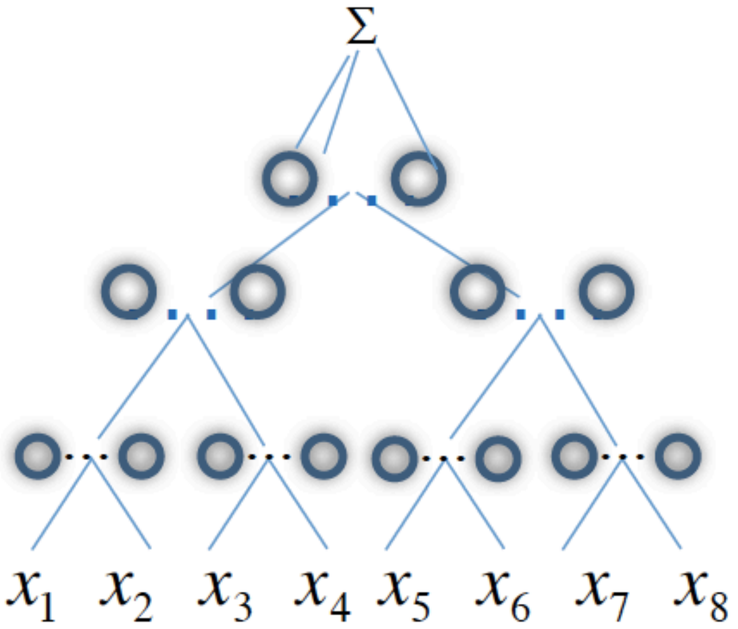
- *Optimization:* Where is complexity control in deep nets?

- *Learning Theory:* Classical and Modern regime, new foundations

Deep and shallow networks: universality

Theorem *Shallow, one-hidden layer networks with a nonlinear $\phi(x)$ which is not a polynomial are universal. Arbitrarily deep networks with a nonlinear $\phi(x)$ (including polynomials) are universal.*



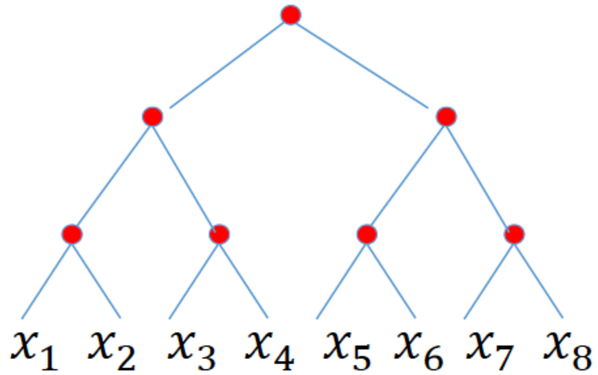


$$\phi(x) = \sum_{i=1}^{r} c_i \left| < w_i , x > + b_i \right|_+$$

Cybenko, Girosi, ….

A new way to avoid the curse for compositional functions:
deep networks

$$f(x_1,x_2,...,x_8) = g_3(g_{21}(g_{11}(x_1,x_2),g_{12}(x_3,x_4))g_{22}(g_{11}(x_5,x_6),g_{12}(x_7,x_8)))$$



Theorem, Mhaskar, Poggio, Liao 2016

Suppose that a function of d variables is hierarchically locally compositional . Both shallow and deep network can approximate f equally well. The number of parameters of the shallow network depends exponentially on d as $O(\varepsilon^{-d})$ whereas the deep networks show $O(d\varepsilon^{-2})$ vs $O(\varepsilon^{-d})$

- Convolutional networks are a special case;
- *Locality, not weight sharing,* avoids the curse of dimensionality

H. N. Mhaskar and Poggio, T., "Deep vs. shallow networks: An approximation theory perspective", Analysis and Applications, vol. 14, no. 06, pp. 829 - 848, 2016.

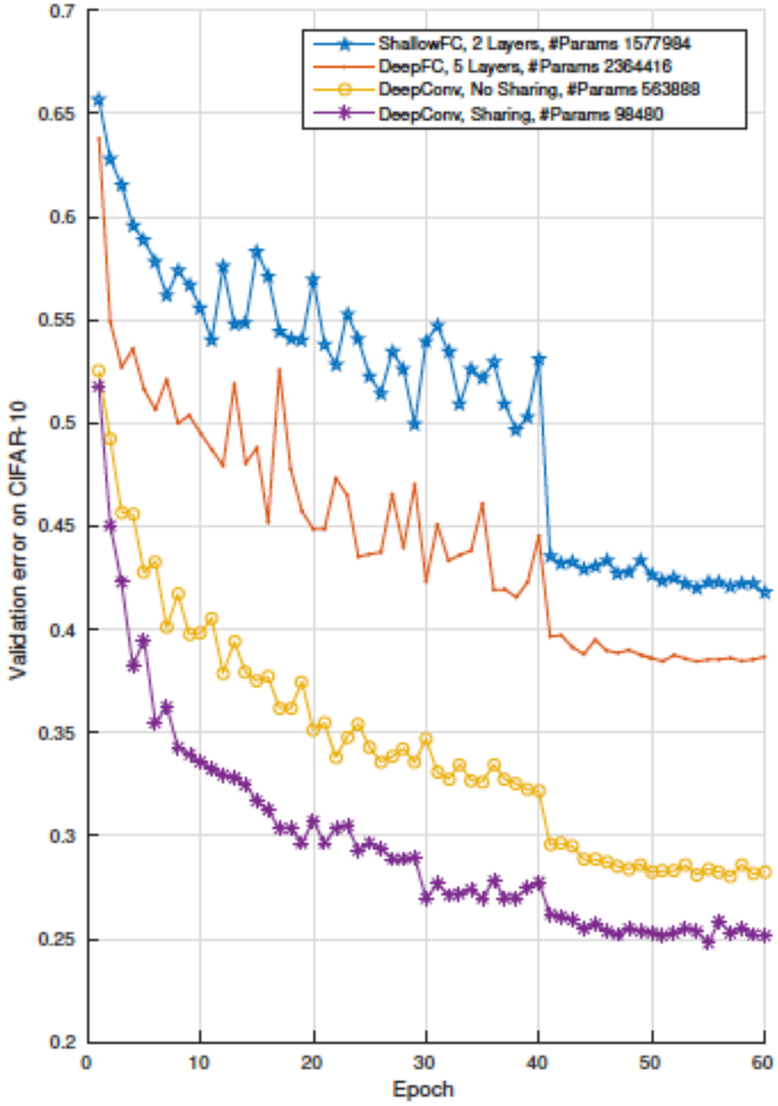
Proof

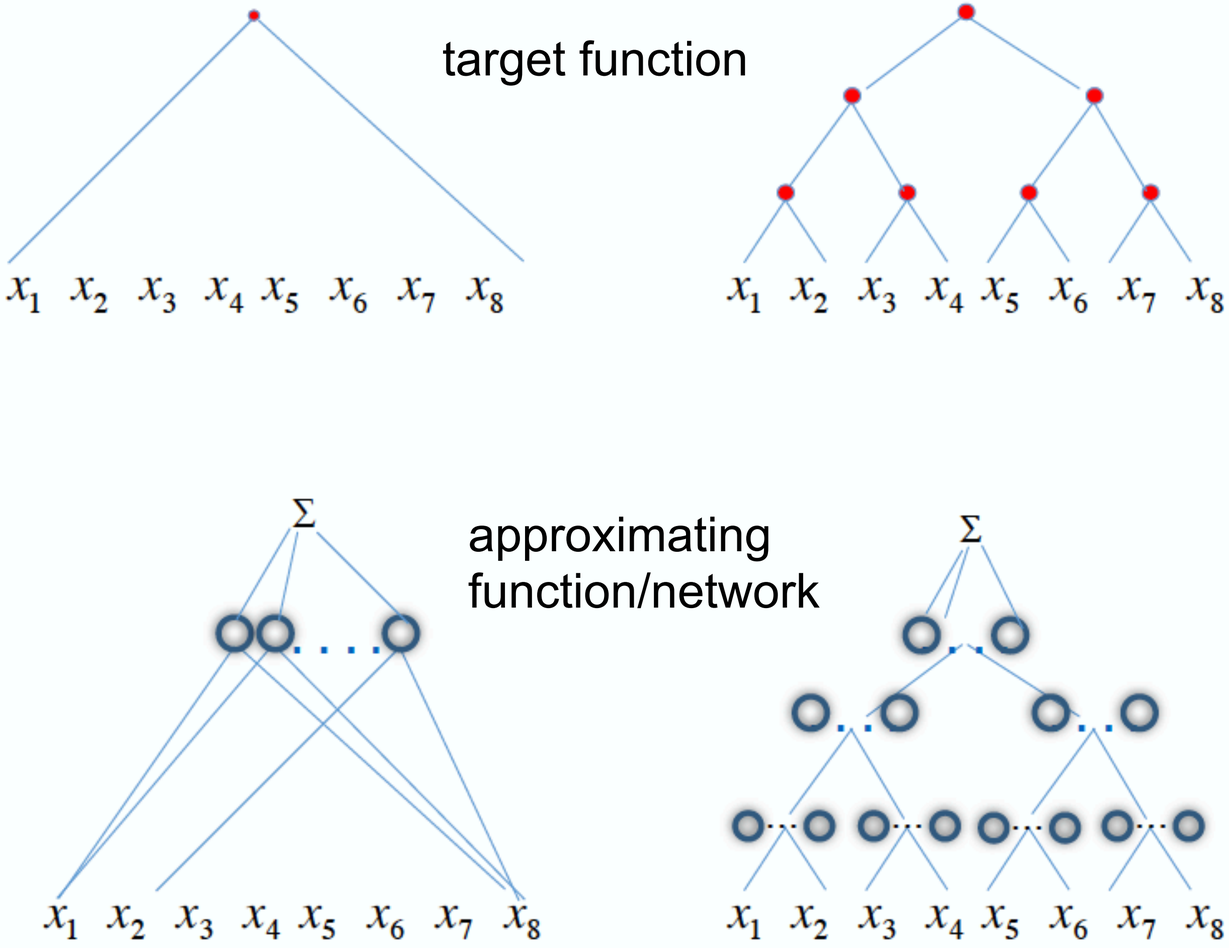
Proof To prove Theorem 2, we observe that each of the constituent functions being in W_m^2, (1) applied with $n = 2$ implies that each of these functions can be approximated from $\mathcal{S}_{N,2}$ up to accuracy $\epsilon = cN^{-m/2}$. Our assumption that $f \in W_m^{N,2}$ implies that each of these constituent functions is Lipschitz continuous. Hence, it is easy to deduce that, for example, if P, P_1, P_2 are approximations to the constituent functions h, h_1, h_2, respectively within an accuracy of ϵ, then since $\|h - P\| \leq \epsilon$, $\|h_1 - P_1\| \leq \epsilon$ and $\|h_2 - P_2\| \leq \epsilon$, then $\|h(h_1, h_2) - P(P_1, P_2)\| = \|h(h_1, h_2) - h(P_1, P_2) + h(P_1, P_2) - P(P_1, P_2)\| \leq \|h(h_1, h_2) - h(P_1, P_2)\| + \|h(P_1, P_2) - P(P_1, P_2)\| \leq c\epsilon$ by Minkowski inequality. Thus

$$\|h(h_1, h_2) - P(P_1, P_2)\| \leq c\epsilon,$$

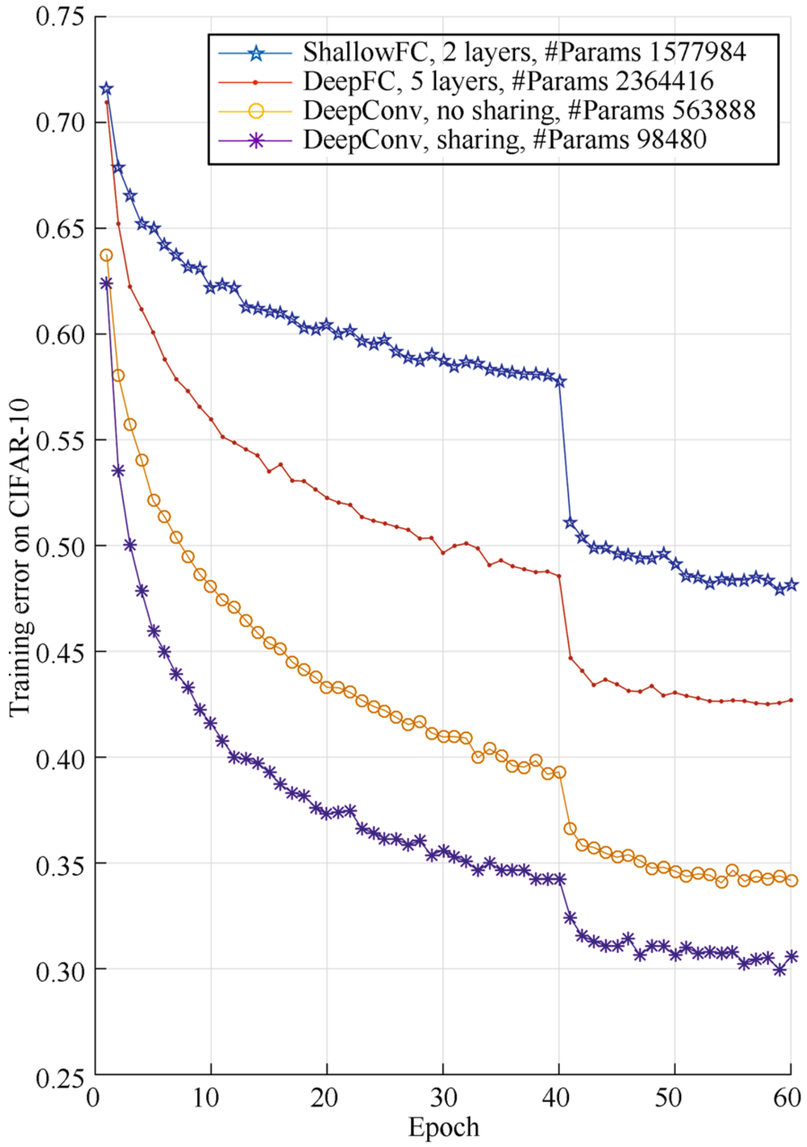
for some constant $c > 0$ independent of the functions involved. This, together with the fact that there are $(n - 1)$ nodes, leads to (6). □

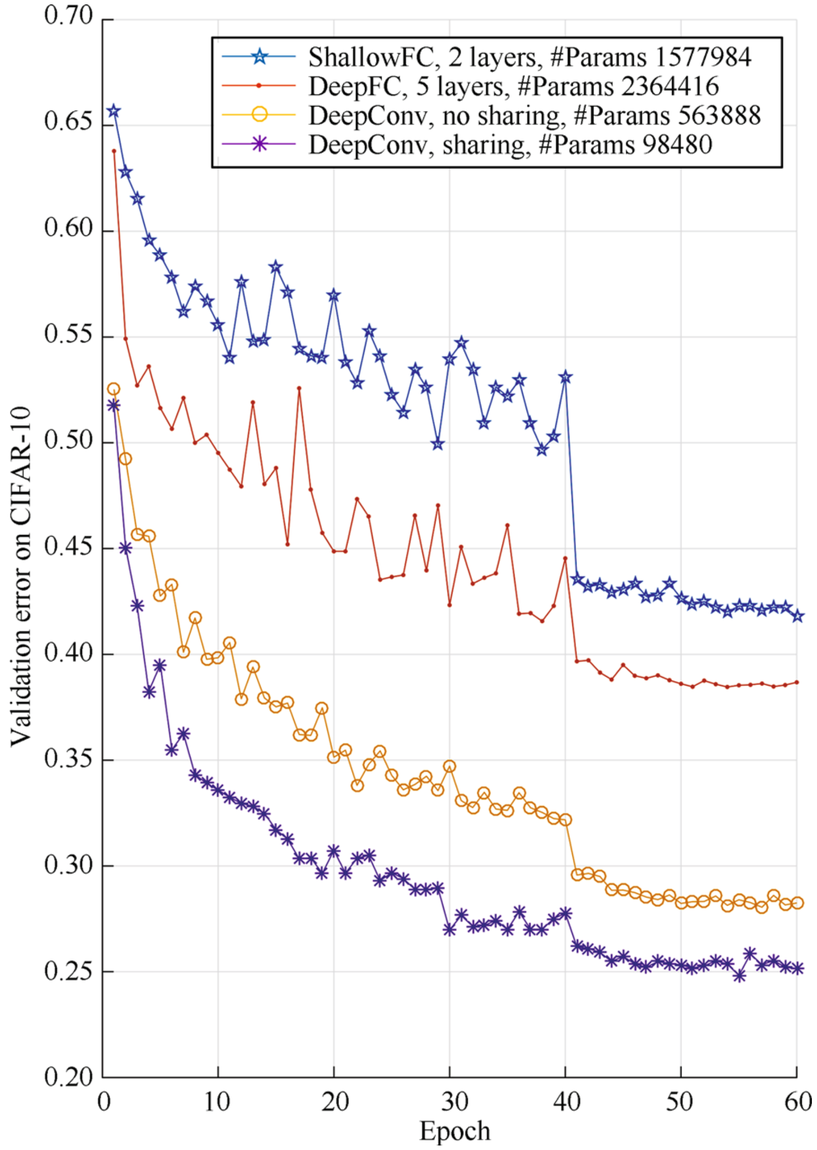
Functions and approximating networks





a b


ShallowFC, 2 layers, #Params 1577984
DeepFC, 5 layers, #Params 2364416
DeepConv, no sharing, #Params 563888
DeepConv, sharing, #Params 98480
Training error on CIFAR-10
Epoch
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0
10
20
30
40
50
60


ShallowFC, 2 layers, #Params 1577984
DeepFC, 5 layers, #Params 2364416
DeepConv, no sharing, #Params 563888
DeepConv, sharing, #Params 98480
Validation error on CIFAR-10
Epoch
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0
10
20
30
40
50
60

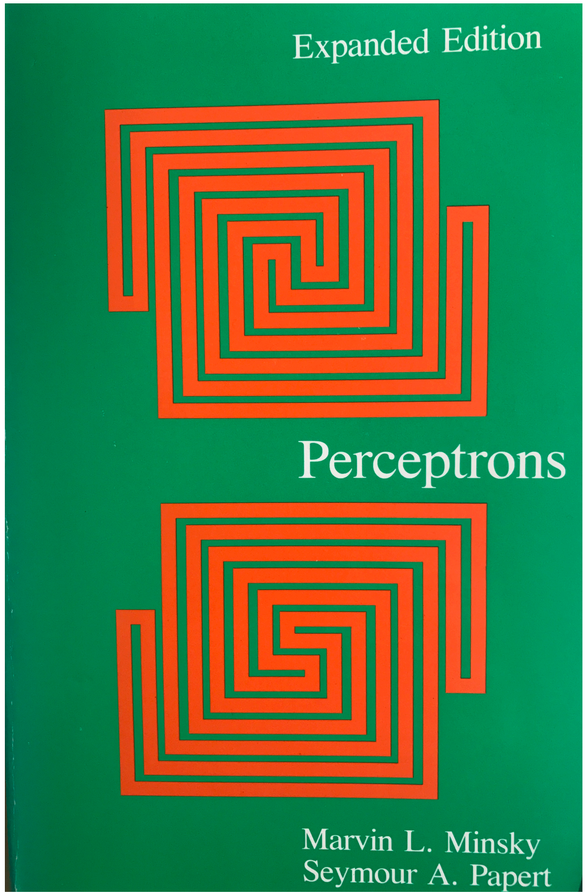
Why are compositional functions important?

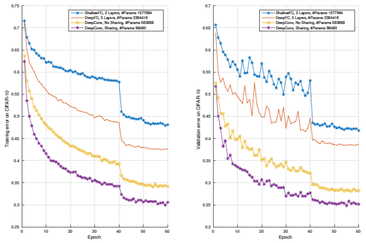
Which one of these reasons:
Physics?
Neuroscience? <===
Evolution?

Locality of Computation

What is special about locality of computation?

Locality in “space”?
Locality in “time”?





Project: work to be done

It is a bit of a mystery why deep learning works

- *Approximation Theory:* When and why are deep networks better than shallow networks?

- *Optimization:* Where is complexity control in deep nets?

- *Learning Theory:* Classical and Modern regime, new foundations

It is a bit of a mystery why deep learning works

- *Optimization:* Where is complexity control in deep nets?

Theory III: Dynamics and Generalization in Deep Networks*

Andrzej Banburski [1], Qianli Liao[1], Brando Miranda[1], Tomaso Poggio[1], Lorenzo Rosasco[1], and Jack Hidary[2]

Poggio, Liao, Banburski, NatCom., 2020

Complexity control by gradient descent in deep networks

Tomaso Poggio[1✉], Qianli Liao[1] & Andrzej Banburski[1]

Poggio Banburski Liao, PNAS, 2020

Theoretical issues in deep networks

Tomaso Poggio[a,1], Andrzej Banburski[a], and Qianli Liao[a]

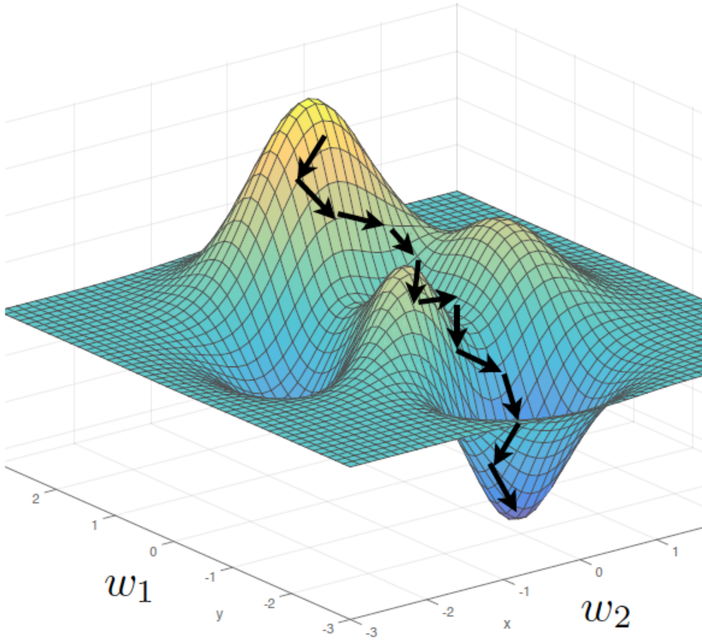
Gradient descent

$$\underset{\mathbf{w}}{\operatorname{argmin}} \quad \sum_i \ell(\mathbf{z}_i, f(\mathbf{x}_i; \mathbf{w})) = L(\mathbf{w})$$

One iteration of gradient descent:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta_t \frac{\partial L(\mathbf{w}^t)}{\partial \mathbf{w}}$$

learning rate

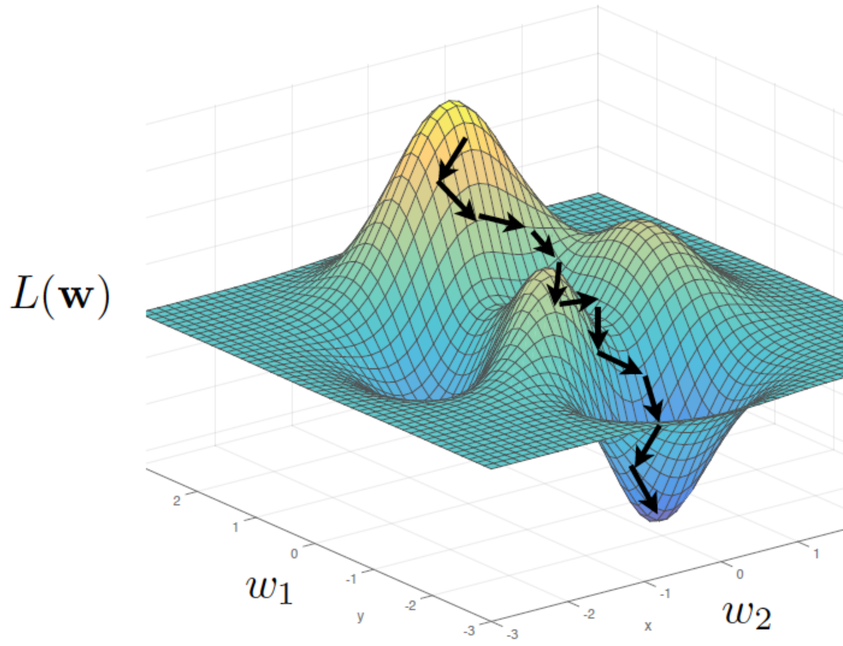


Training in a deep neural net

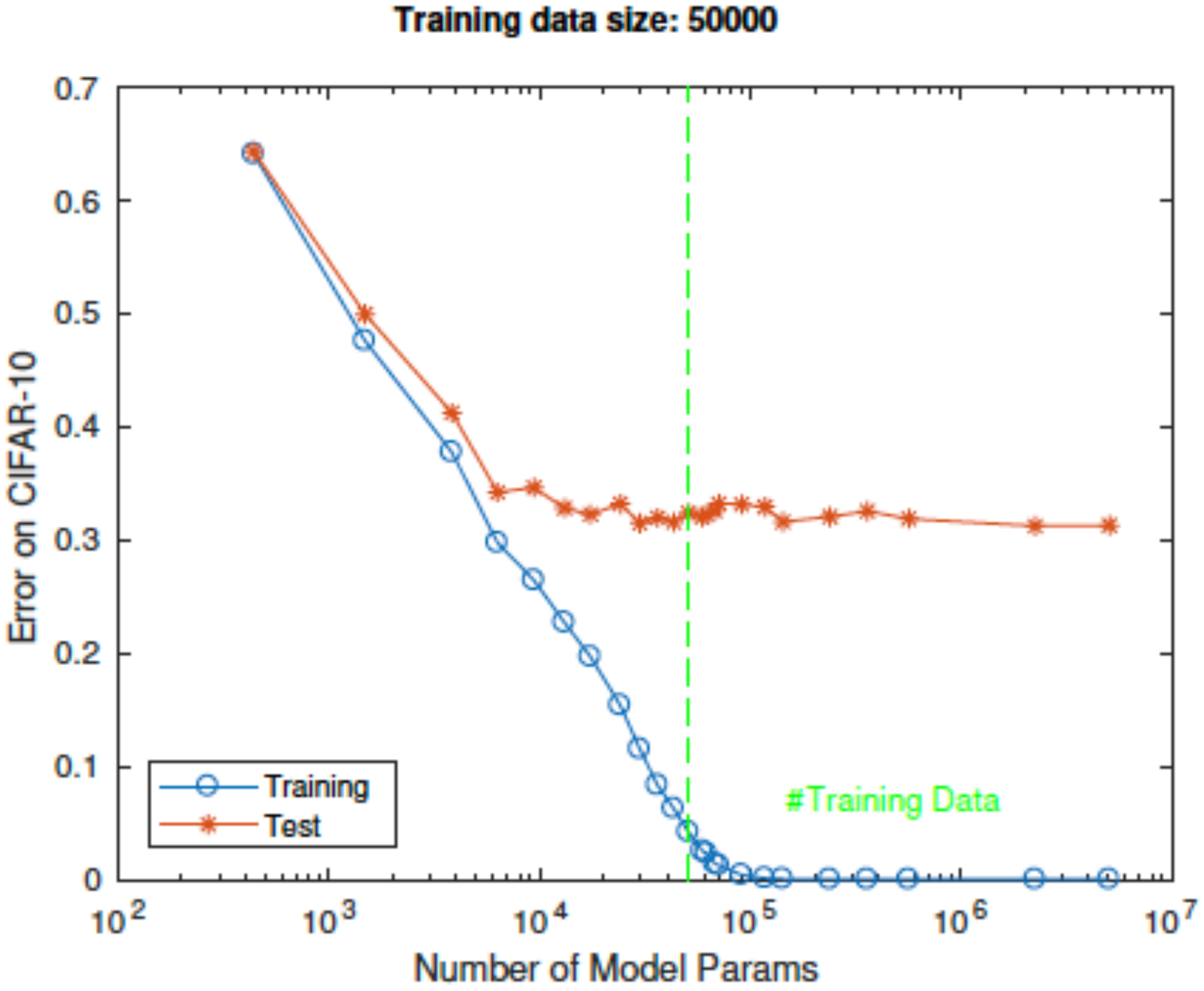
$$\arg\min_{W_k} L = \arg\min_{W_k} \sum_{n}^{N} e^{-y_n f(W_K,\ldots,W_1;x_n)} \quad \rightarrow$$

$$\dot{W}_k == -\nabla_w L$$

$$\dot{w}_k^{i,j} = \sum_{n}^{N} e^{-y_n f(x_n)} \frac{\partial f}{\partial w_k^{i,j}}$$

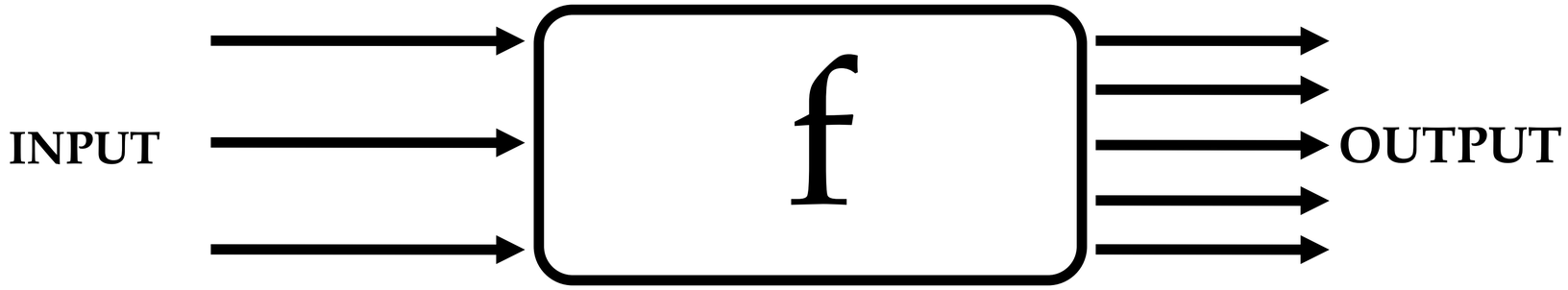


Overparametrized, underconstrained solutions are predictive: how is this possible?



Trained on cross entropy loss

Historical parenthesis about foundations of Statistical Learning Theory: **supervised** learning (~1980-today)



Given a set of l examples (data)

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_\ell, y_\ell)\}$$

Question: find function f such that

$$f(x) = \hat{y}$$

is a **good predictor** of y for a **future** input x (fitting the data is **not** enough!)

Statistical Learning Theory: part of mainstream math not just statistics (Valiant, Vapnik, Smale, Devore...)

BULLETIN (New Series) OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 39, Number 1, Pages 1–49
S 0273-0979(01)00923-5
Article electronically published on October 5, 2001

ON THE MATHEMATICAL FOUNDATIONS OF LEARNING



FELIPE CUCKER AND STEVE SMALE

The problem of learning is arguably at the very core of the problem of intelligence, both bi

T. Poggio and C.R. Shelton

Introduction

(1) A main theme of this report is the relationship of approximation to learning and the primary role of sampling (inductive inference). We try to emphasize relations of the theory of learning to the mainstream of mathematics. In particular, there are large roles for probability theory, for algorithms such as *least squares*, and for tools and ideas from linear algebra and linear analysis. An advantage of doing this is that communication is facilitated and the power of core mathematics is more easily brought to bear.

Statistical Learning Theory: supervised learning

There is an unknown **probability distribution** on the product space $Z = X \times Y$, written $\mu(z) = \mu(x, y)$. We assume that X is a compact domain in Euclidean space and Y a bounded subset of $\mathbb{R}$. The **training set** $S = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\} = \{z_1, ...z_n\}$

consists of n samples drawn i.i.d. from μ.

$\mathcal{H}$ is the **hypothesis space**, a space of functions $f : X \to Y$.

A **learning algorithm** is a map $L : Z^n \to \mathcal{H}$ that looks at S and selects from $\mathcal{H}$ a function $f_S : \mathbf{x} \to y$ such that $f_S(\mathbf{x}) \approx y$ *in a predictive way*.

Statistical Learning Theory

Given a function f, a loss function V, and a probability distribution μ over Z, the **expected or true error** of f is:

$$I[f] = \mathbb{E}_Z V[f, z] = \int_Z V(f, z) d_\mu(z) \tag{1}$$

which is the **expected loss** on a new example drawn at random from μ.

The **empirical error** of f is:

$$I_S[f] = \frac{1}{n} \sum V(f, z_i) \tag{2}$$

A very natural requirement for f_S is distribution independent **generalization**

$$\forall \mu, \lim_{n \to \infty} |I_S[f_S] - I[f_S]| = 0 \ \textit{in probability} \tag{3}$$

In other words, the training error for the solution must converge to the expected error and thus be a "proxy" for it.

Statistical Learning Theory: generalization

The ERM problem does not have a *predictive* solution in general (just fitting the data does not usually predict well).

Choosing an appropriate hypothesis space *H* (for instance a compact set of continuous functions) can guarantee generalization. A necessary and sufficient condition for generalization is that *H is uGC*.

Related concept, measuring complexity of the hypothesis space, are:

VC dimension, V_gamma dimension, Rademacher numbers..

Statistical Learning Theory: foundational theorems

One of the key msgs of the 80’-90’ from learning theory:
if you do not control capacity/complexity of the predictors in an implicit or explicit way
you will not predict well!
The capacity of models fitting the data must be controlled!

How can deep networks generalize? Where is the complexity control?

- Observation: classical learning theory has made clear that the number of parameters is not the key to predictivity. The norm of the weights and related quantities such as Rademacher complexity and covering numbers are what should be controlled.

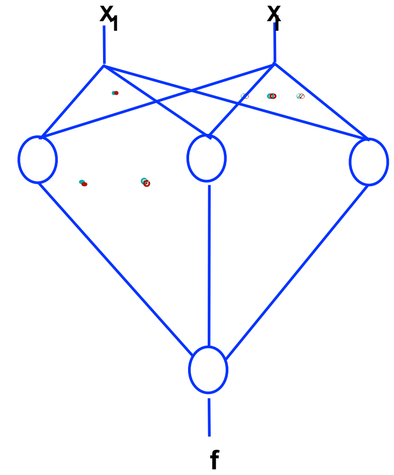
- Control of the norm can be done via regularization (the λ term).

As example a classical algorithm: Regularization in RKHS (eg. kernel machines)

$$\min_{f \in H} \left[\frac{1}{n} \sum_{i=1}^{n} V(f(x_i) - y_i) + \lambda \|f\|_K^2 \right]$$

The regularization term controls the complexity of the function in terms of its RKHS norm

implies $f(\mathbf{x}) = \sum_i^n \alpha_i K(\mathbf{x}, \mathbf{x}_i)$ equivalent to



“Equivalence” of Tikhonov, Morozov and Ivanov Regularization

Ivanov is $\min_{W} L(W) \quad s.t. \mathbf{R}(W) \leq \delta$

Morozov is $\min_{W} \mathbf{R}(W) \quad s.t. L(W) \leq \epsilon$

Tikhonov is $\min_{W} L(W) + \lambda \mathbf{R}(W)$

Under mild conditions these 3 optimization problems are “equivalent”.

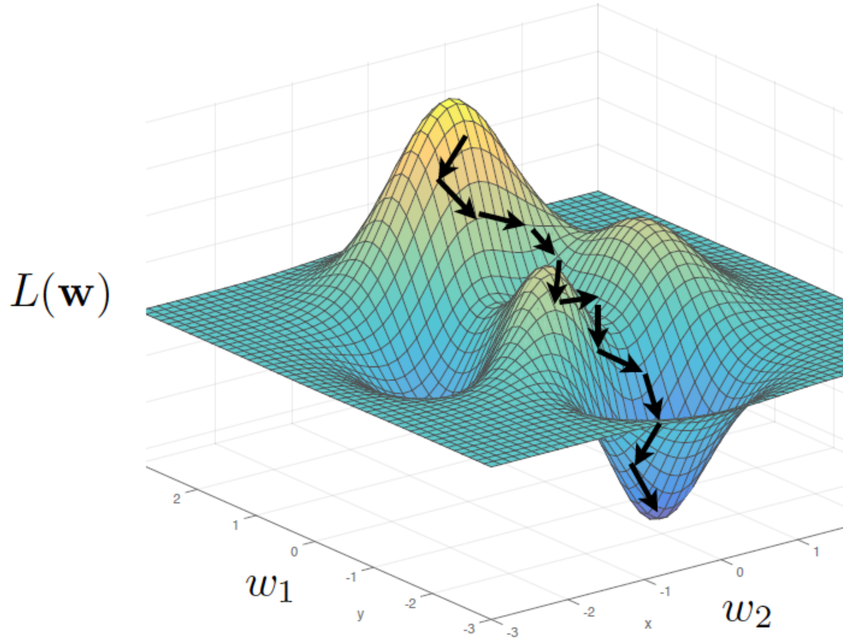
Gradient descent on cross entropy loss

$$\hat{L}(f) = -\sum_{n=1}^{N} \log\left(\frac{e^{-f_{y_n}(W;x)}}{\sum_{j=1}^{C} e^{-f_j(W;x)}}\right).$$

$$\arg\min_{W_k} L = \arg\min_{W_k} \sum_{n}^{N} e^{-y_n f(W_K,\ldots,W_1;x_n)} \quad \rightarrow$$

$$\dot{W}_k == -\nabla_w L$$

$$\dot{w}_k^{i,j} = \sum_{n}^{N} e^{-y_n f(x_n)} \frac{\partial f}{\partial w_k^{i,j}}$$



The puzzle of deep learning

More parameters than training data…

No explicit norm control in training…

Where is the complexity control?

Are deep nets so magic

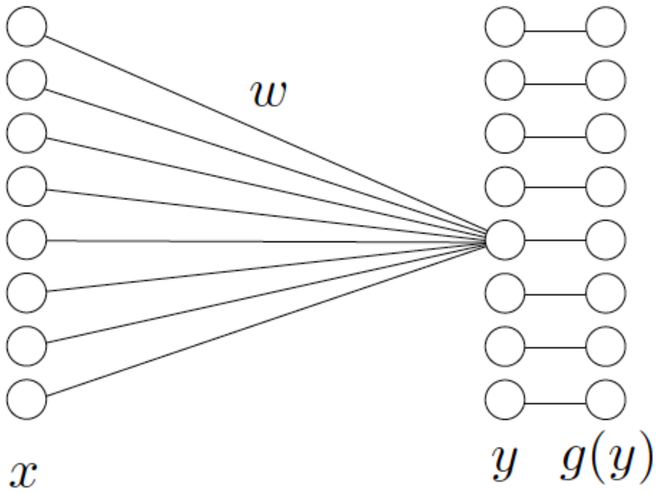
that we have to

throw away classical statistic?

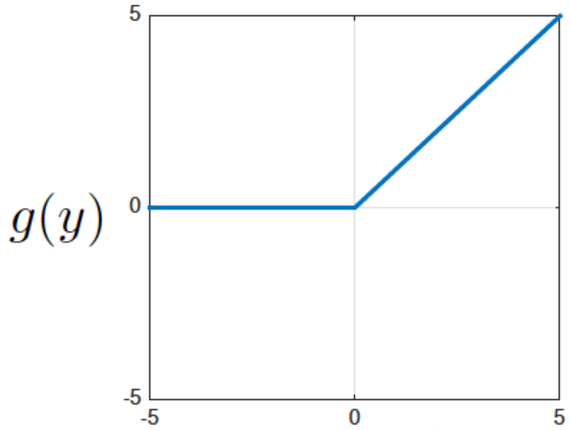
Deep RELU networks: definitions and homogeneity

We define a deep K-layer network with the usual RELU function σ

$$\mathrm{f}(\mathrm{W};\mathrm{x}) : R^d \to R, \quad f(W_K,...,W_1;x) = W_K\sigma(W_{K-1}...\sigma(W_1x)...)$$

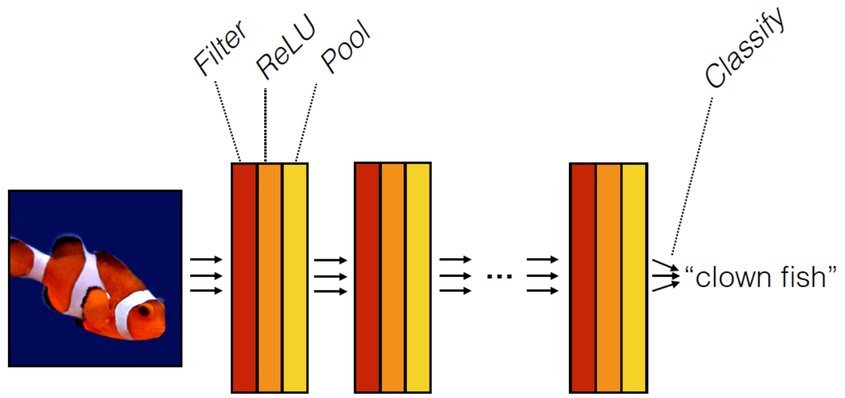


Rectified linear unit (ReLU)



$$g(y) = \max(0, y)$$

Computation in a neural net



$$f(\mathbf{x}) = f_L(\ldots f_2(f_1(\mathbf{x})))$$

Deep RELU networks: definitions and homogeneity

For RELUs $\sigma(z) = z\,\frac{\partial \sigma}{\partial z}$ and homogeneity holds that is

$$f(W; x) = \rho\, \tilde{f}(x) \quad \text{where } \rho = \prod_k \rho_k \text{ and } \rho_k V_k^{i,j} = W_k^{i,j}, \quad \rho_k^2 = \sum_{i,j} (W_k^{i,j})^2,$$

$$\tilde{f}(x) = f(V; x) \text{ with } ||V_k|| = 1 \text{ and } ||\tilde{f}|| = 1$$

Notice that what matters for classification is $\tilde{f}$

We make hypothesis of separability that is $\tilde{f}(x_n) y_n > 0, \quad \forall n = 1, \ldots N$

GD optimization induces a gradient dynamical system

$$L = \sum_{n}^{N} (y_n - f(W_K, \ldots, W_1; x_n))^2$$ for regression

$$L = \sum_{n}^{N} e^{-y_n f(W_K, \ldots, W_1; x_n)}$$ for classification

Exponential loss
Deep network with RELUs,

$$\dot{W}_k^{i,j} = \sum_{n}^{N} e^{-y_n f(x_n)} \frac{\partial f}{\partial W_k^{i,j}}$$

Exponential loss: approach 2

$$L = \sum_{n}^{N} e^{-y_n f(W_K,\ldots,W_1;x_n)}$$

$$\dot{W}_k^{i,j} = \sum_{n}^{N} e^{-y_n f(x_n)} \frac{\partial f}{\partial W_k^{i,j}}$$

Analysis of GD for the exponential loss

$$L = \sum_{n}^{N} e^{-y_n f(W_K,\ldots,W_1;x_n)}$$

Exponential loss
Deep network with RELUs,

$$\dot{W}_k^{i,j} = \sum_{n}^{N} e^{-y_n f(x_n)} \frac{\partial f}{\partial W_k^{i,j}}$$

Standard Unconstrained Gradient Descent

$$\dot{W}_k^{i,j} = -\nabla_w L = \sum_n^N e^{-y_n f(x_n)} \frac{\partial f}{\partial W_k^{i,j}} \quad \text{from} \quad L = \sum_n^N e^{-y_n \rho\, \tilde{f}(V_K,\ldots,V_1;x_n)}$$

Do $\frac{W_k^{i,j}}{\|W_k\|} = V_k^{i,j}$ converge?

Reparametrize with $W_k = \rho_k V_k$ and $\|V_k\| = 1$ to get the same dynamical system

$$\dot{\rho}_k = \frac{\rho}{\rho_k} \sum_n^N e^{-y_n \rho\, \tilde{f}(x_n)} y_n \tilde{f}(x_n)$$

$$\dot{V}_k = \frac{\rho}{\rho_k} \sum_n^N e^{-y_n \rho\, \tilde{f}(x_n)} \left(\frac{\partial \tilde{f}(x_n)}{\partial V_k} - V_k \tilde{f}(x_n)\right)$$

Standard Unconstrained Gradient Descent

The critical points of V_k maximize margin and are given by

$$\sum_n e^{-\rho y_n f_V(x_n)} y_n \frac{\partial f_V(x_n)}{\partial V_k} = \sum_n e^{-\rho y_n f_V(x_n)} V_k y_n f_V(x_n)$$

In the limit of large t, if one group of surviving SVs with the same margin remain then

$$V_k f_V(x_*) = \frac{\partial f_V(x_*)}{\partial V_k}$$

Constrained Gradient Descent

Consider the loss $L = \sum_n e^{-y_n \rho\, \tilde{f}(V_K,\ldots,V_1;x_n)} + \lambda \sum_k \left|\left|V_k\right|\right|^2$

and choose λ so that $\left|\left|V_k\right|\right| = 1$ at convergence. Then the gradient flow is

$$\dot{\rho}_k = \frac{\rho}{\rho_k} \sum_n e^{-y_n \rho\, \tilde{f}(V_K,\ldots,V_1;x_n)} y_n \tilde{f}(x_n) \qquad \dot{V}_k = \rho(t) \sum_n e^{-y_n \rho\, \tilde{f}(V_K,\ldots,V_1;x_n)} y_n \frac{\partial \tilde{f}(x_n)}{\partial V_k} - 2\lambda V_k$$

Multiplying last equation by V^T and imposing $\mathrm{V}^T V = 1$

yields $2\lambda = \rho(t) \sum_n e^{-y_n \rho\, \tilde{f}(V_K,\ldots,V_1;x_n)} y_n \frac{\partial \tilde{f}(x_n)}{\partial V_k}$

and thus $\dot{V}_k = \frac{\rho}{\rho_k} \sum_n e^{-y_n \rho\, \tilde{f}(x_n)} y_n \left(\frac{\partial \tilde{f}(x_n)}{\partial V_k} - V_k \tilde{f}(x_n)\right)$

Constrained Gradient Descent

Thus minimization of L subject to $\left|\left|V_k\right|\right| = 1$ that is

$$\dot{V}_k^{i,j} = -\nabla_w (L + \sum_k \lambda_k (\left|\left|V_k\right|\right|^2 - 1)) \quad \text{with} \quad L = \sum_n^N e^{-y_n \rho \, \tilde{f}(V_K,\ldots,V_1;x_n)}$$

gives the dynamical system

$$\dot{\rho}_k = \frac{\rho}{\rho_k} \sum_n^N e^{-y_n \rho \, \tilde{f}(V_K,\ldots,V_1;x_n)} y_n \tilde{f}(x_n)$$

$$\dot{V}_k = \rho \sum_n^N e^{-y_n \rho \, \tilde{f}(x_n)} y_n \left(\frac{\partial \tilde{f}(x_n)}{\partial V_k} - V_k \tilde{f}(x_n)\right)$$

which is the same as Weight Normalization!

Constrained Gradient Descent

Thus minimization of L subject to $\left|\left|V_k\right|\right| = 1$ that is

$$\dot{V}_k^{i,j} = -\nabla_w (L + \sum_k \lambda_k (\left|\left|V_k\right|\right|^2 - 1)) \quad \text{with} \quad L = \sum_n^N e^{-y_n \rho \, \tilde{f}(V_K,\ldots,V_1;x_n)}$$

gives the dynamical system

$$\dot{\rho}_k = \frac{\rho}{\rho_k} \sum_n^N e^{-y_n \rho \, \tilde{f}(V_K,\ldots,V_1;x_n)} y_n \tilde{f}(x_n)$$

$$\dot{V}_k = \rho \sum_n^N e^{-y_n \rho \, \tilde{f}(x_n)} y_n \left(\frac{\partial \tilde{f}(x_n)}{\partial V_k} - V_k \tilde{f}(x_n)\right)$$

which is the same (apart from $\boldsymbol{\rho_k^2}$ factor) as standard unconstrained gradient descent!

In particular they have the *same stationary points*!

Constrained and _unconstrained_
optimization of deep nets
with exponential loss
by gradient descent
correspond to different dynamical systems
converging towards
the same critical points

Similarly to GD on a linear net under the square loss
GD here performs an implicit (vanishing) regularization. The underlying mechanism is different and more robust.

Convergence

For $\rho \to \infty$ or $t \to \infty$ the stationary points coincide with maximum margin solutions

$$V^{i,j}{}_k \tilde{f}(x^*) = \frac{\partial \tilde{f}(x^*)}{\partial V_k}$$

which are minimum norm solutions similar to pseudoinverse solutions.

Parenthetical Results

The norm of the various layers remains the same if it was equal at initialization

The rate of change for $||\rho_k||^2$ is

$$\frac{\partial \rho_k^2}{\partial t} = 2\rho_k \dot{\rho}_k = 2\rho_k \frac{\rho}{\rho_k} \sum_n e^{-\rho y_n f_V(x_n)} y_n f_V(x_n) = 2\rho \sum_n e^{-\rho y_n f_V(x_n)} y_n f_V(x_n)$$

which is independent of k

When one group of SVs dominates (for large t) then margin is not decreasing

$y_1 f(V; x_1)$ increases monotonically or is constant because

$$y_* \frac{\partial f(V; x_1)}{\partial t} = \sum_k \left(\frac{\partial y_1 f(V; x_1)}{\partial V_k}\right)^T \dot{V}_k =$$

$$\sum_k \frac{\rho}{\rho_k^2} e^{-\rho y_1 f(V; x_1)} \left(\left\| \frac{\partial f(V; x_1)}{\partial V_k} \right\|_F^2 - f(V; x_1)^2 \right).$$

Remarks: rate of convergence (for 1 support vector!)

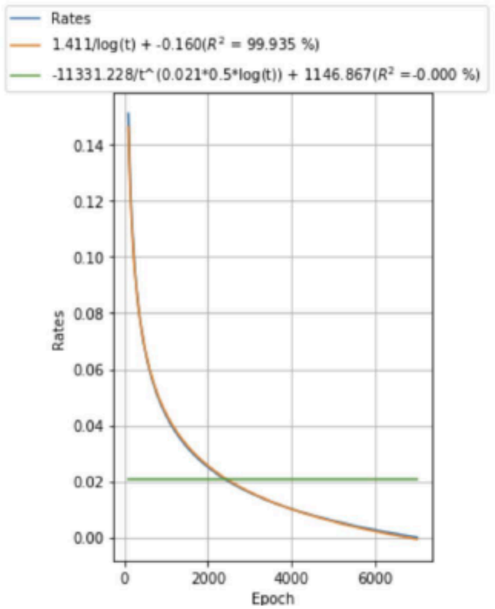
The *rate of convergence of gradient descent* for linear one-layer networks

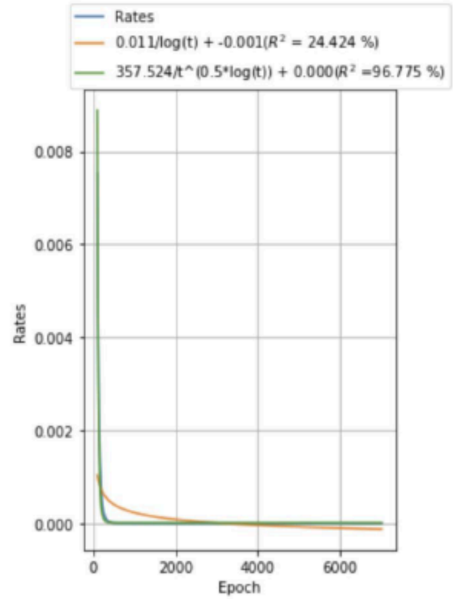
$$\dot{V} = \frac{1}{\rho}(I - VV^T)\sum_n^N e^{-y_n \rho V^T x_n} y_n x_n$$

is $\frac{1}{\log\ t}$, consistent with Srebro, very slow. The rate *of convergence of Weight Normalization*

$$\dot{V} = \rho(I - VV^T)\sum_n^N e^{-y_n \rho V^T x_n} y_n x_n$$

is $t^{-\sqrt{\log t}}$ thus faster!

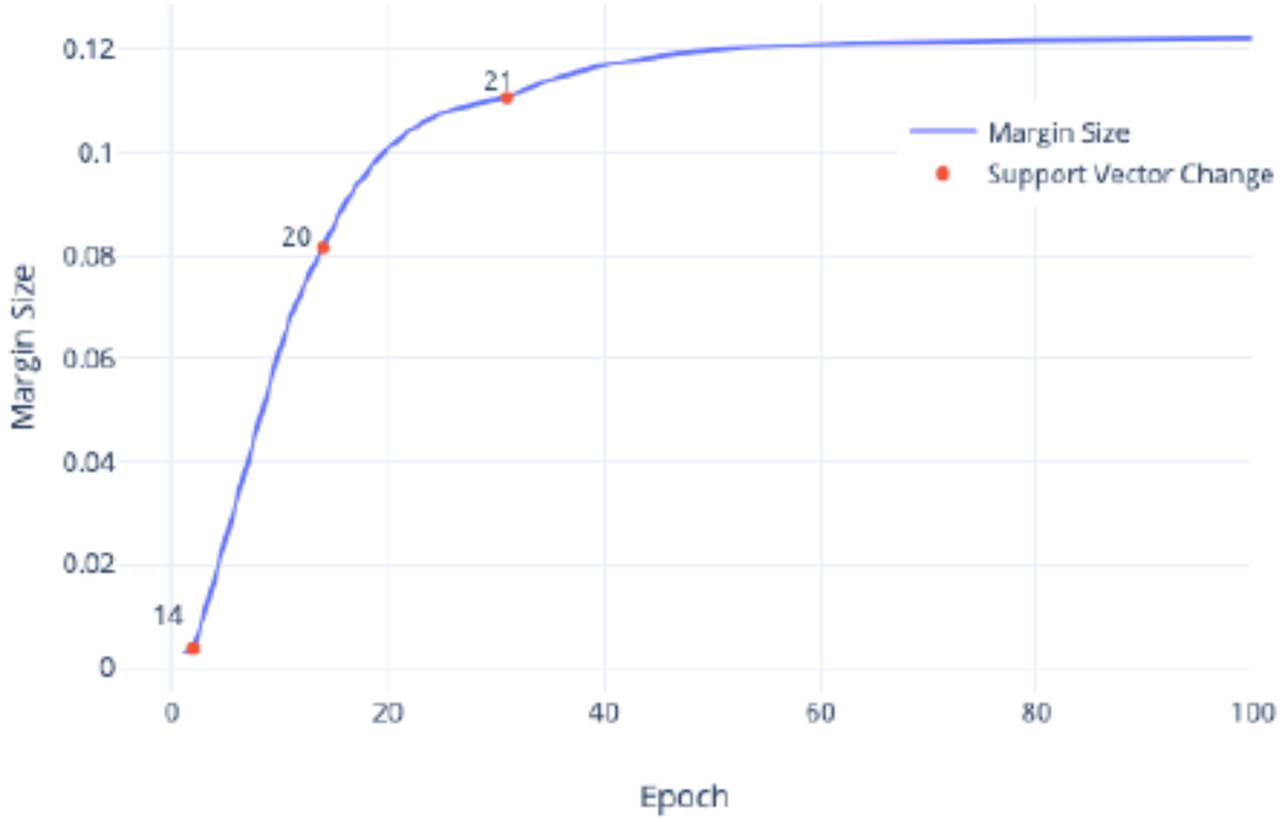




Parenthesis closed

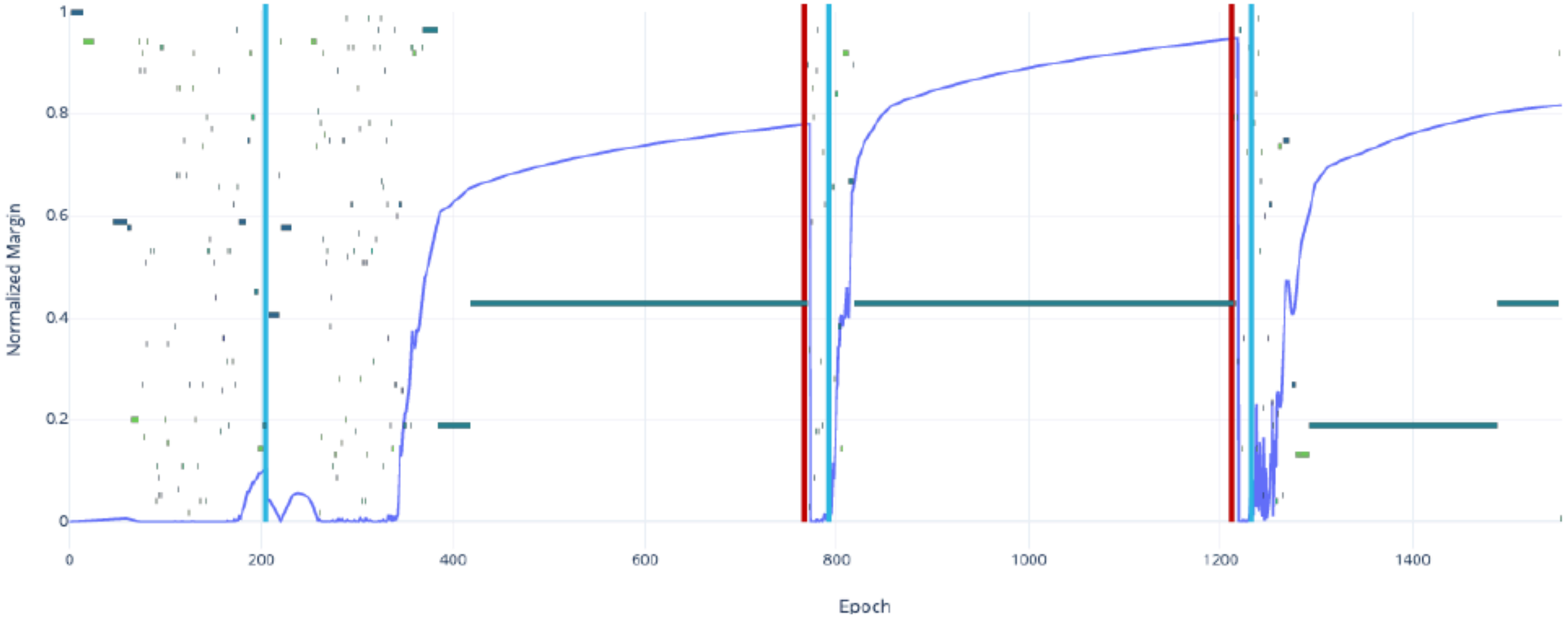
Margin Maximization

Lemma: $\tilde{f}(x^*)$ increases monotonically



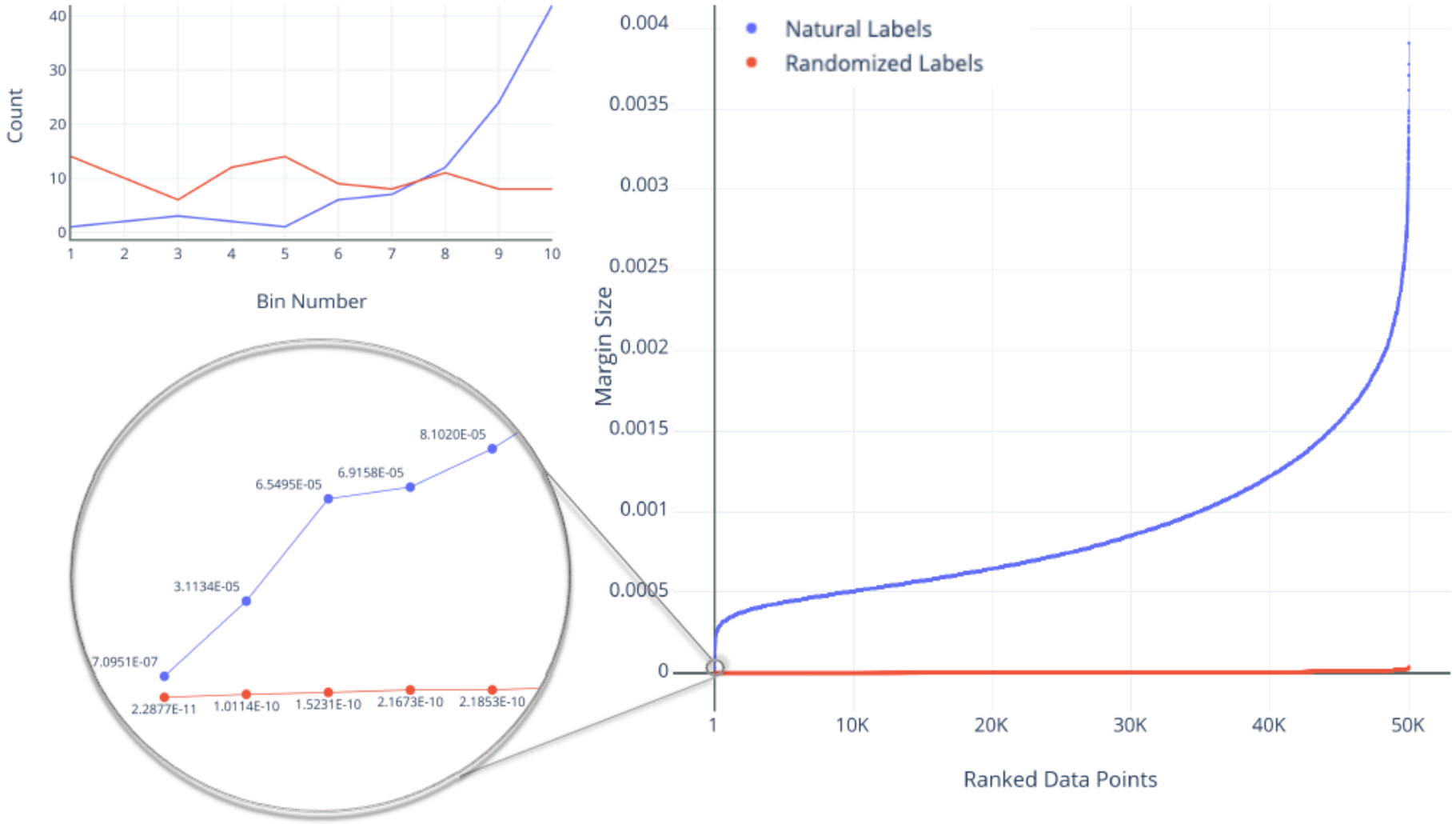
Fernanda +Andy

Trajectory of convergence with perturbations

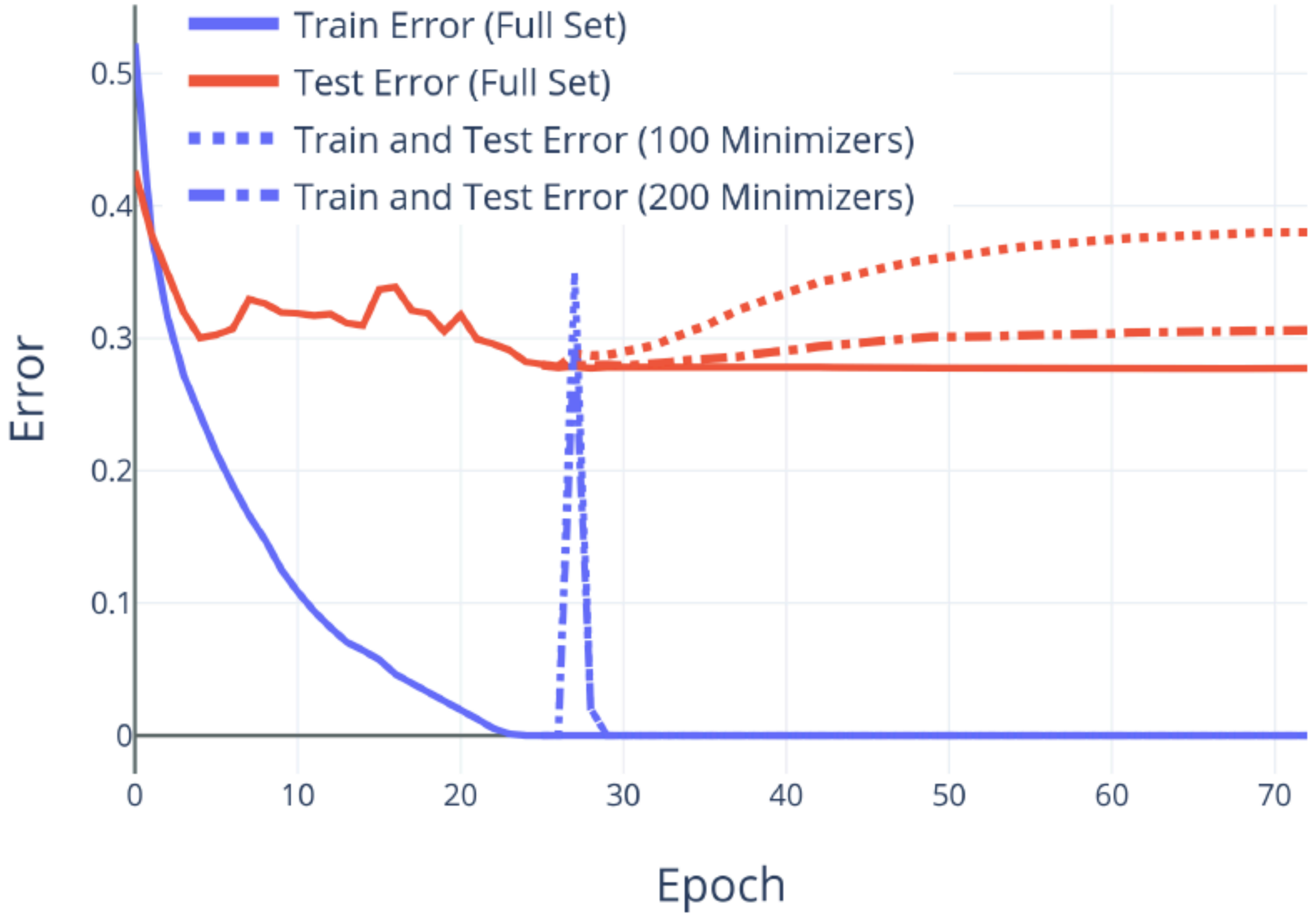


Fernanda +Andy

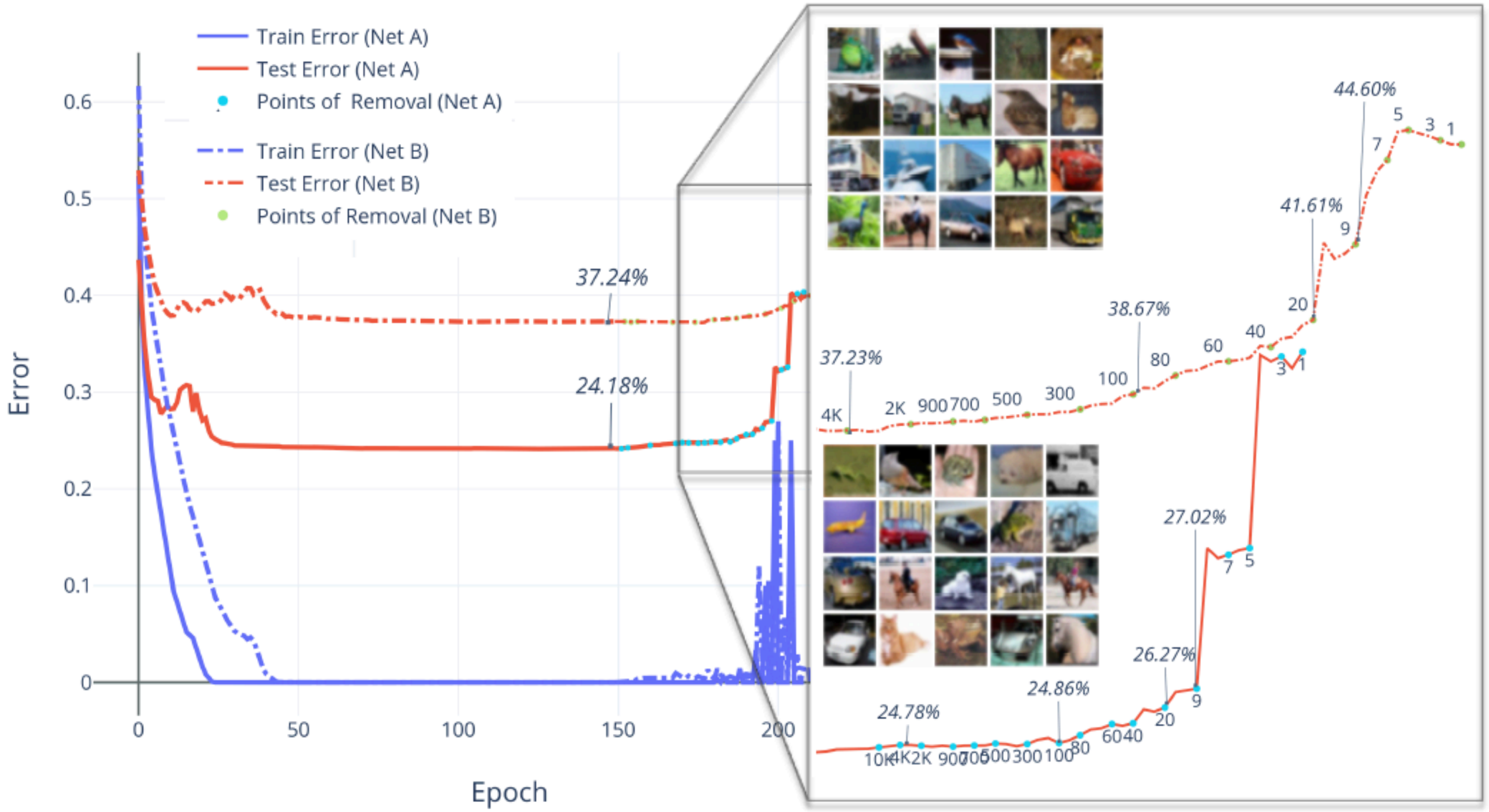
Margin: natural and random labels



Compression (after separation from 50K training data points to 200)



Compression: removing training points



The puzzle of overparametrization and generalization: a solution?

Theorem (surprising):
Variants of SGD - Batch Normalization and Weight Normalization - perform minimization with explicit unit norm constraint, which is equivalent to maximize (normalized) margin.

$$\dot{\rho}_k = \frac{\rho}{\rho_k} \sum_n^N e^{-y_n \rho \ \tilde{f}(V_K,...,V_1;x_n)} y_n \tilde{f}(x_n) \qquad L = \sum_n^N e^{-y_n \rho \ \tilde{f}(V_K,...,V_1;x_n)}$$

$$\dot{V}_k = \rho(t) \sum_n^N e^{-y_n \rho \ \tilde{f}(V_K,...,V_1;x_n)} y_n \frac{\partial \tilde{f}(x_n)}{\partial V_k}$$

Theorem (very surprising):
Standard gradient descent maximize L_2 margin without explicit normalization or regularization

Explanation of hidden complexity control by gradient descent in deep networks!

A. Banburski, Liao, Q., Miranda, B., Poggio, T., Rosasco, L., Hidary, J., and De La Torre, F., "Theory III: Dynamics and Generalization in Deep Networks". 2018.

Caveats

- The analysis so far assumes vanilla GD: no momentum, no WN, no BN, no data augmentation
- It assumes exponential loss: same for square loss?
- GD and SGD converge towards a minimum norm solution but this is *not unique* (unique only for linear networks)
- What does the minimum norm solution imply wrt generalization?
- Can we have a more general theory valid for Kernel Machines and Deep Learning?