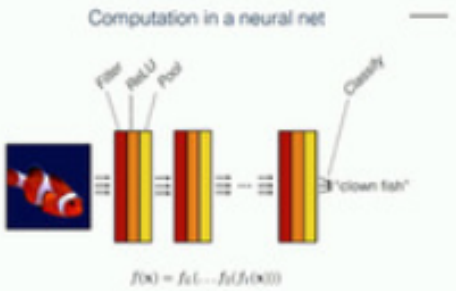
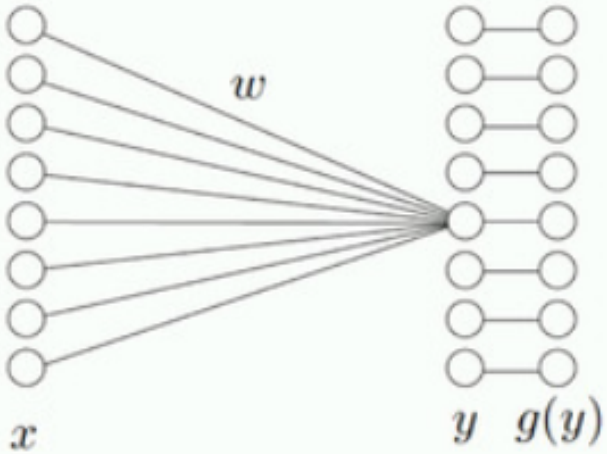
Theory of Deep Learning

Tomaso Poggio

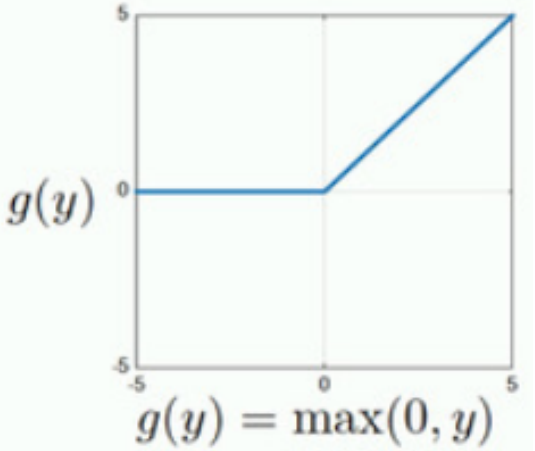
Centre for Brains, Minds & Machines

Training and computation in a neural net





Rectified linear unit (ReLU



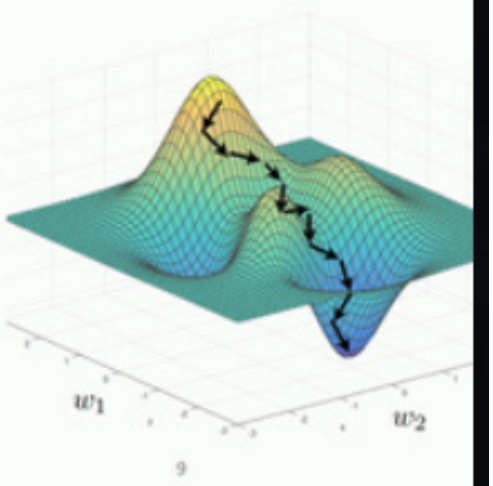
Gradient descent

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

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



Deep Networks: three theory questions

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

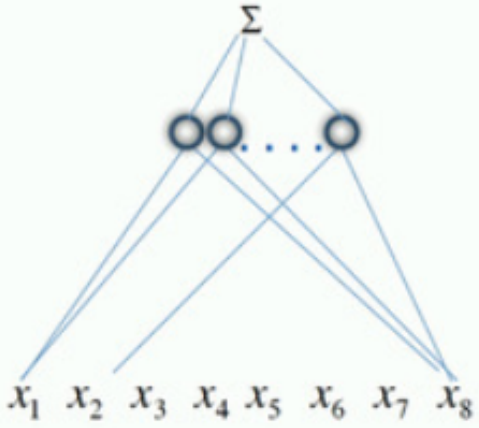
- *Optimization:* What is the landscape of the empirical risk?

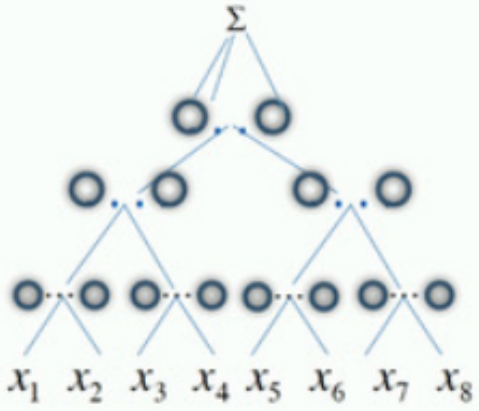
- *Dynamical System Approach:* Characterizing SGD solutions. Are they stable? Control of norm? Maximum margin?

- L*earning Theory:* How can deep learning not overfit?

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, ...

Curse of dimensionality

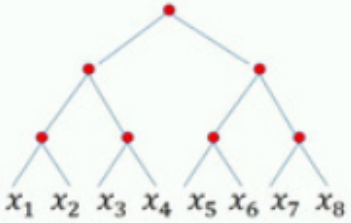
$$y = f(x_1, x_2, \ldots, x_8)$$

Both *shallow* and *deep* network can approximate a function of d variables equally well. The number of parameters in both cases depends exponentially on d as $O(\varepsilon^{-d})$. Both suffer from the *curse of dimensionality.*

Mhaskar, Poggio, Liao, 2016

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 (informal statement)**

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 $O(\varepsilon^{-d})$

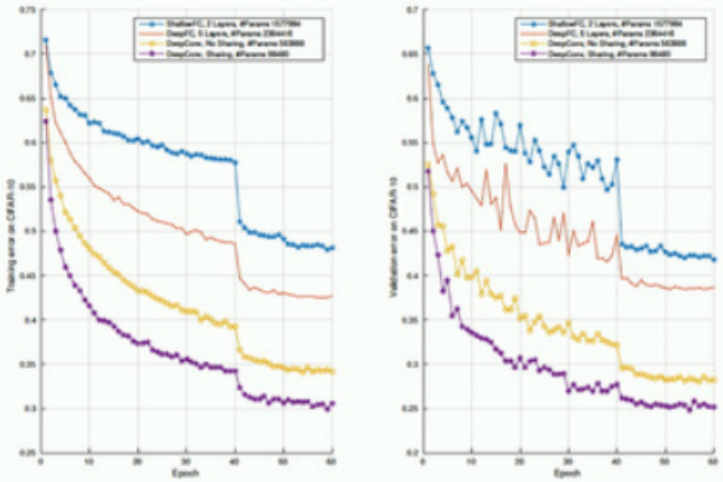
Note: *Locality, not weight sharing,* avoids the curse of dimensionality

Compositional functions

Notes

- *Locality, not weight sharing,* avoids the curse of dimensionality
- Linear combinations of compositional functions are universal approximators
- Are compositional functions similar to elementary recursive functions in the theory of computation?

Theory: locality of constituent functions — *not weight sharing* — is key



CENTER FOR
Brains
Minds+
Machines

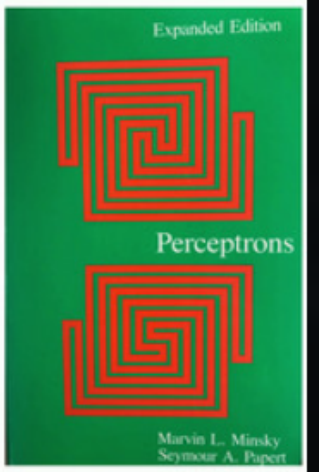
Why are compositional functions important?

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

Locality of Computation

What is special about locality of computation?

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



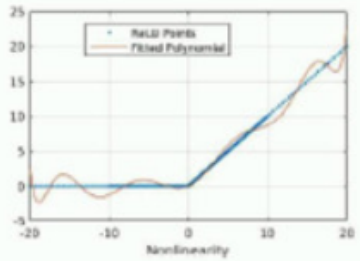
Approximation: summary

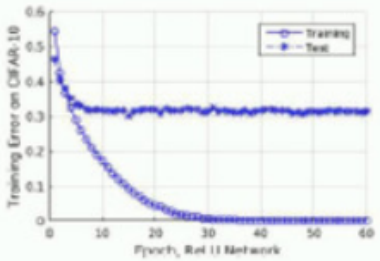
- *Locality, not weight sharing,* avoids the curse of dimensionality
- *Notice that ConvNets rather than dense nets* are the main empirical success so far
- Why do *compositional functions* underly image and speech recognition?

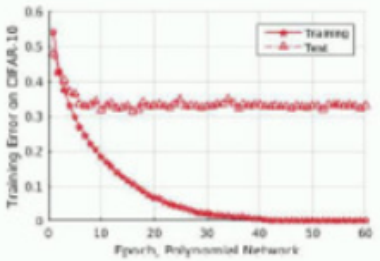
Deep Networks: three theory questions

- *Approximation Theory:* When and why are deep networks better than shallow networks?
- *Optimization:* What is the landscape of the empirical risk?
- *Dynamics of learning:* What are the solutions? Are they stable? Maximum margin?
- *Learning Theory:* How can deep learning not overfit?

RELU approximation by univariate polynomial preserves deep nets properties







CENTER FOR
Brains
Minds+
Machines

Bezout theorem

$$p(x_n) - y_n = 0 \quad \text{for n} = 1,...,N$$

The set of polynomial equations above with *k= degree of p(x)* has a number of distinct zeros (counting points at infinity, using projective space, assigning an appropriate multiplicity to each intersection point, and excluding degenerate cases) equal to

$$Z = k^N$$

the product of the degrees of each of the equations.

CENTER FOR Brains Minds+ Machines

More important: Bezout and local+global critical points

$f(x_i) - y_i = 0$ for $i = 1,...,n$ — N equations in W unknowns with $W >> N$

$\nabla_w \sum_{i=1}^{N} (f(x_i) - y_i)^2) = 0$ — W equations in W unknowns

Global minima are degenerate

Other critical points of the gradient are isolated (generically)

Langevin equation

$$\frac{df}{dt} = -\gamma_t \nabla V(f(t)) + dB(t)$$

$$f_{t+1} = f_t - \gamma_n \nabla V(f_t, z_t) + \gamma'_t W_t.$$

with the Boltzmann equation as asymptotic "solution"

$$p(f) \sim \frac{1}{Z} = e^{-\frac{V(x)}{T}}$$

CENTER FOR Brains Minds+ Machines

SGD

$$f_{t+1} = f_t - \gamma_t \nabla V(f_t, z_t), \qquad \nabla V(f_t, z_t) = \frac{1}{|z_t|} \sum_{z \in z_t} \nabla V(f_t, z).$$

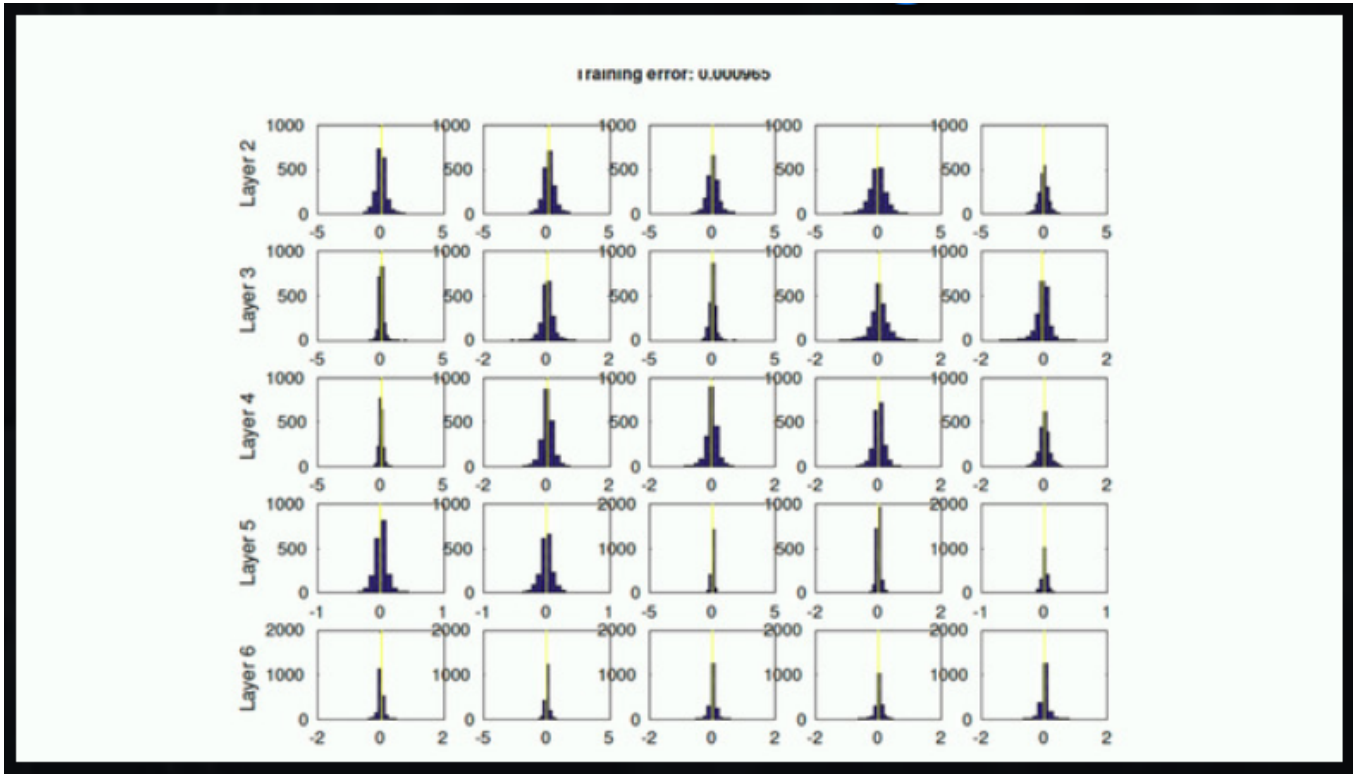
We define a noise "equivalent quantity"

$$\xi_t = \nabla V(f_t, z_t) - \nabla I_{S_n}(f_t),$$

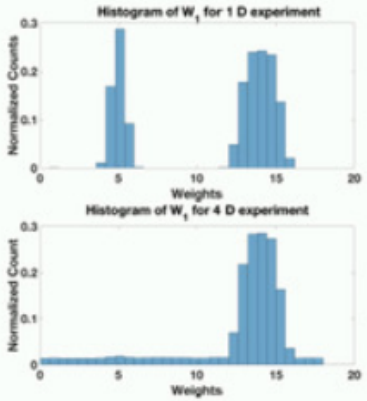
and it is clear that $\mathbb{E}\xi_t = 0$.

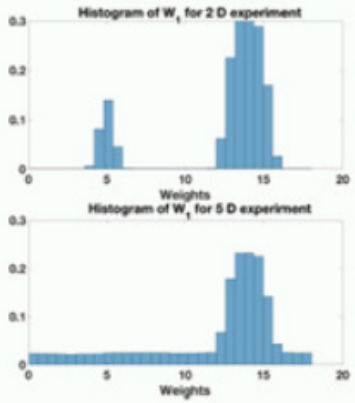
We write Equation 6 as

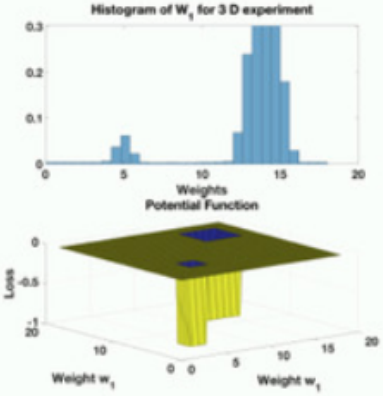
$$f_{t+1} = f_t - \gamma_t(\nabla I_{S_n}(f_t) + \xi_t).$$


Training error: 0.000965
Layer 2
Layer 3
Layer 4
Layer 5
Layer 6

GDL selects larger volume minima

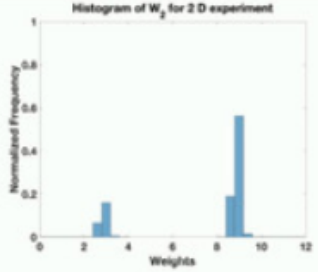


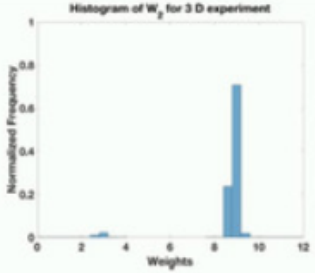


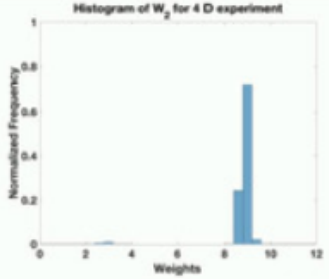


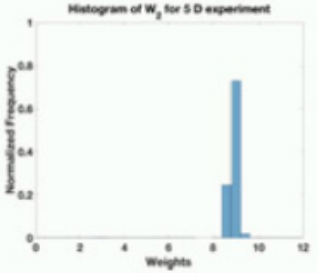
CENTER FOR
Brains
Minds+
Machines

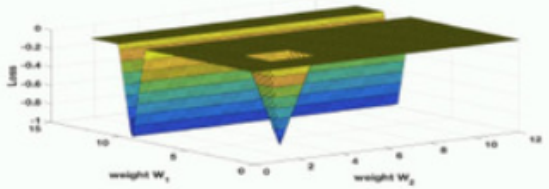
Concentration because of high dimensionality





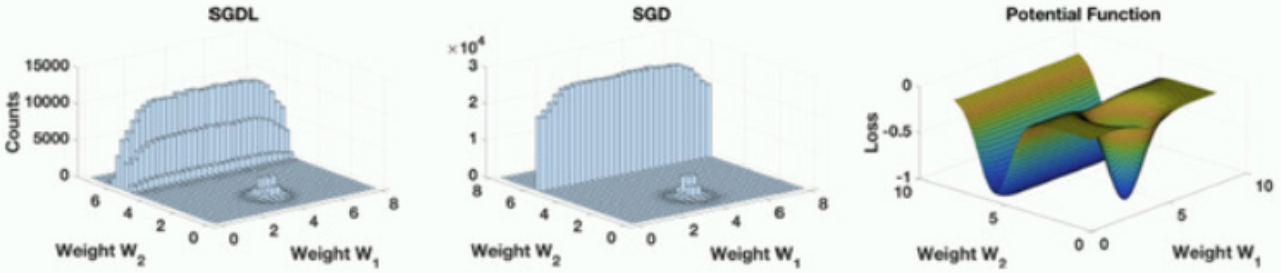






Minds+
Machines

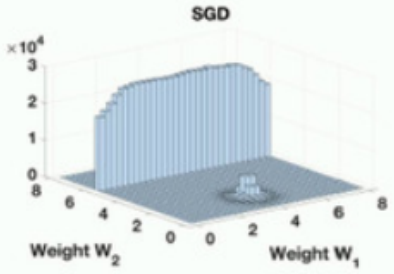
SGD behaves as SGDL (or GDL)

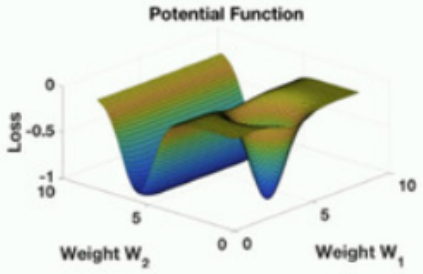


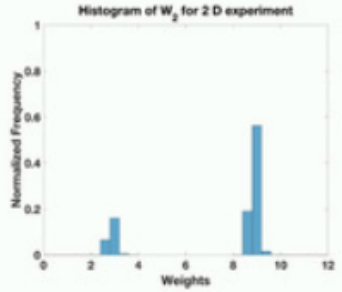
CENTER FOR
Brains
Minds+
Machines

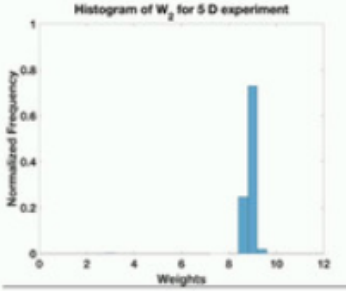
SGDL and SGD: summary

- Overparametrized deep networks have many global minimizers that are generically degenerate; other critical points of the gradient are generically isolated.
- SGDL finds with very high probability large volume, **zero-minimizers**; empirically SGD behaves in a similar way.









Poggio, Rakhlin, Golovitc, Zhang, Liao, 2017

Deep Networks: three theory questions

- *Approximation Theory:* When and why are deep networks better than shallow networks?
- *Optimization:* What is the landscape of the empirical risk?
- *Dynamical System Approach:* Characterizing SGD solutions: are they stable? control of norm? maximum margin?
- *Learning Theory:* How can deep learning not overfit?

GD optimization induces a gradient dynamical system

$$L = \sum_n^N (y_n - f(W_K, ..., W_1; x_n))^2$$ for regression

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

---

Linear one-layer case under square loss
we analyze various nonlinear cases:
perhaps in discussion...

$$L = \sum_n^N (y_n - w_1^{1,j} x^j{}_n)^2$$

$$\dot{w}_1^{1,j} = 2 \sum_n^N E_n x^j{}_n$$

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

---

Linear one-layer case under square loss
we analyze various nonlinear cases:
perhaps in discussion...

$$L = \sum_n^N (y_n - w_1^{1,j} x^j{}_n)^2$$

$$\dot{w}_1^{1,j} = 2 \sum_n^N E_n x^j{}_n$$

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

GD optimization induces a gradient dynamical system

$$L = \sum_n^N (y_n - f(W_K, ..., W_1; x_n))^2$$ for regression

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

Exponential loss
Deep network with RELUs,
with Halpern term

$$\dot{w}_k^{i,j} = \sum_n^N e^{-y_n f(x_n)} \frac{\partial f}{\partial w_k^{i,j}} - \lambda(t) \sum_k \| w_k - w_k^0 \|^2$$

$$H_{abk'}^{ijk} = \sum_n^N e^{-y_n f(x_n)} \left(-\frac{\partial f}{\partial w_k^{i,j}} \frac{\partial f}{\partial w_{k'}^{a,b}} + \frac{\partial^2 f}{\partial w_k^{i,j} \partial w_{k'}^{a,b}}\right) - \lambda(t) I$$

Dynamical Systems approach

What happens in the simple cases, eg linear nets?

In the deep case, usually degenerate equilibria,
degenerate Hessian, no direct norm control!

What can we say?

Deep RELU networks: definitions and homogeneity

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

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

There are no biases. The top layer matrix is actually a vector.

For RELUs $\sigma(z) = z$ $\dfrac{\partial\sigma}{\partial z}$ and homogeneity holds

f (W; x) = $\prod \rho_k \tilde{f}(V;x)$ where $\rho_k v_k^{i,j} = w_k^{i,j}$, $\rho_k^2 = \sum_{i,j}(w_k^{i,j})^2$

CENTER FOR Brains Minds+ Machines

Generalization bounds for regression: minimize empirical loss with minimum norm

(*a*) For deep RELU networks $\mathbb{R}_N(\mathbb{F}) = \rho_1 \ldots \rho_K \mathbb{R}_N(\tilde{\mathbb{F}})$

(*b*) Classical bounds: with probability $\geq (1\text{-}\delta)$

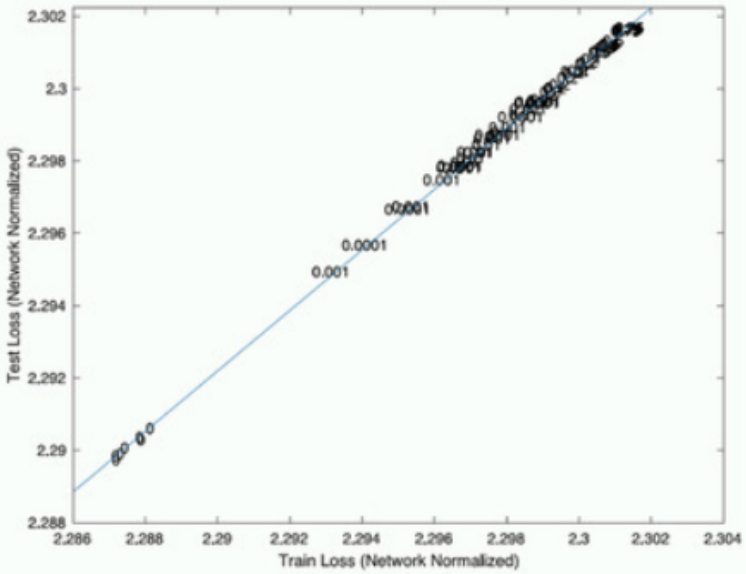
$$\left|L(f) - L_S(f)\right| \leq 2\mathbb{R}_N(\mathbb{F}) + \sqrt{\frac{\ln\frac{1}{\delta}}{2N}}$$

The norm dependency of Rademacher complexity (a) together with the bound (b) implies that we should *minimize the empirical error and select the minimum norm minimizer* if the minimum is degenerate (which is the case in deep networks).

Qianli Liao, Andrzej Barbuski

A parenthesis on regression and generalization

The magic of layer-wise normalization



Generalization in regression (not classification)

Consider typical generalization bounds for regression: they have the following form:

With probability $\geq (1\text{-}\delta)$ $\quad |L(f) - L_S(f)| \leq 2\mathbb{R}_N(\mathbb{F}) + \sqrt{\frac{\ln \frac{1}{\delta}}{2N}}$

$$\mathbb{R}_N(\mathbb{F}) = \rho_1 \cdots \rho_K \mathbb{R}_N(\tilde{\mathbb{F}}).$$

Then, for the unnormalized cross-entropy loss the bound gives

$$L(f) \leq \prod_{k=1}^{K} \rho_k \mathbb{R}_N(\tilde{\mathbb{F}}) + c_2 \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}},$$

since $\hat{L}(f) \approx 0$, whereas for the normalized cross-entropy loss it yields

$$|L(\tilde{f}) - \hat{L}(\tilde{f})| \leq \mathbb{R}_N(\tilde{\mathbb{F}}) + c_2 \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}}.$$

Closing parenthesis on regression and generalization

Generalization bounds for classification: maximize margin at unit norm

(*a*) For deep RELU networks $\mathbb{R}_N(\mathbb{F}) = \rho_1 \ldots \rho_K \mathbb{R}_N(\tilde{\mathbb{F}})$

(*b*) Classical bounds: with probability $\geq (1\text{-}\delta)$

$$E(L_{01}(f)) \leq L_{\exp}(f) + \frac{2}{\eta}\Re(f) + \sqrt{\frac{\ln\frac{1}{\delta}}{2N}}$$

Thus the goal is to minimize $\frac{\Re(f)}{\eta} = \frac{\rho}{y\rho\tilde{f}} = \frac{1}{\tilde{f}}$

that is to maximize the margin of f on the support vectors, while controlling the norm $\|\tilde{f}\|$.

Qianli Liao, Andrzej Barbuski

A unified view...

Lagrange multiplers, weight normalization, batch normalization, are all trying to do GD with unit norm constraints: Halpern iterations

A unifying view of weight normalization techniques – such as the Lagrange multiplier approach of section 5.1, the weight normalization algorithms as well as batch normalization – is to regard them as instances of Halpern iterations. Appendix 9 describes the technique. The gradient flow corresponds to an operator T which is non-expansive. The fixed points are degenerate. Minimization with a regularization term in the weights that vanishes at the appropriate rate (Halpern iterations) converges to the minimum norm minimizer associated to the local minimum.

Maximize margin of normalized network

We define the loss

$$L = \sum_{n=1}^{N} e^{-\rho \tilde{f}((x_n)y_n} + \sum_{k=0}^{K} \lambda_k ||V_k|| \tag{11}$$

where the Lagrange multipliers λ_k are chosen to satisfy $||V_k|| = 1$ at convergence or when the algorithm is stopped (the constraint can also be enforced at each iteration, see later).

We perform gradient descent on L with respect to ρ, V_k. We obtain for $k = 1, \cdots, K$

$$\rho(t+1) - \rho(t) = \sum_n \rho(t) e^{-\rho(t)\tilde{f}(x_n)y_n} \tilde{f}(x_n), \tag{12}$$

and for each layer k

$$\dot{V_k} \approx V_k(t+1) - V_k(t) = \rho(t) \sum_n e^{-\rho(t)\tilde{f}(x_n)} \frac{\partial \tilde{f}(x_n)}{\partial V_k}(t) - 2\lambda_k(t) V_k(t). \tag{13}$$

The sequence $\lambda_k(t)$ must satisfy $\lim_{t\to\infty} ||V_k|| = 1$.

Proposition 3

The GD equations 12 and 13 converge to maximum margin solutions with fixed complexity $\mathbb{R}_N$.

Deep Networks: three theory questions

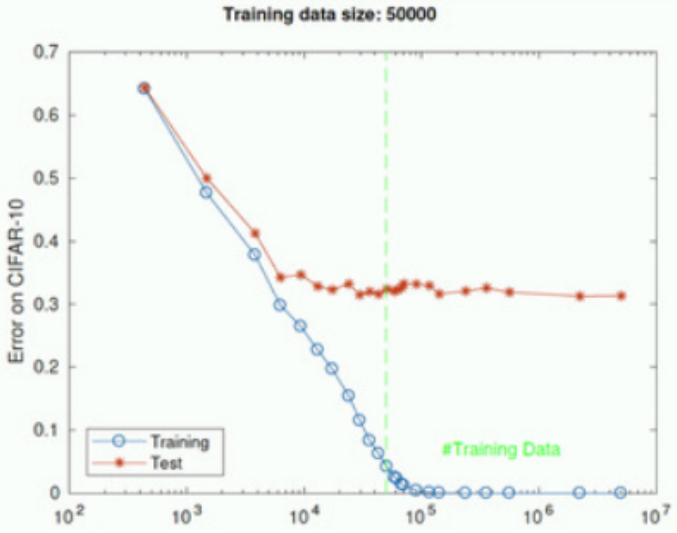
- *Approximation Theory:* When and why are deep networks better than shallow networks?
- *Optimization:* What is the landscape of the empirical risk?
- *Dynamical System Approach:* Characterizing SGD solutions. Are they stable? Control of norm? Maximum margin?
- *Learning Theory:* Generalization and non-overfitting

The key (equivalent to implicit regularization)
is optimization of *normalized* network

$$\tilde{f}$$

Weight normalization and batch normalization are two of the
(at least four) techniques
to do gradient descent under unit norm constraint

Deep nets "puzzle" explained: ρ grows and $\tilde{f}$ does not



The algorithm minimizes $\frac{\Re(f)}{\eta} = \frac{\rho}{y\rho\tilde{f}} = \frac{1}{\tilde{f}}$;

thus ρ increases but $\tilde{f}$ does not change...

General musings

The evolution of computer science

- there were programmers
- there are now labelers
- there may be schools for bots…

Today's science, tomorrow's engineering: learn like children learn

The first phase (and successes) of ML:
supervised learning, big data: $n \to \infty$

from programmers…
…to labelers…
…to computers that learn like children…

The next phase of ML: implicitly supervised learning,
learning like children do, small data: $n \to 1$

Musings on Near Future Breakthroughs

- new architectures/class of applications from basic DCN block (example GAN + RL/DL + …)

- Implicit labeling: evolution is opportunistic…few bits…face area…motion machinery…bootstrapping…predicting next "frame"…

- Learning and representing *symbols…with networks of neurons …*abstract concepts, relations, routines…new circuit motif in addition to DCN?

- New learning algorithm — more biologically plausible than SGD …