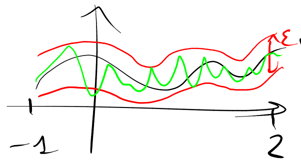
Chapter 1 : Deep Learning vs classical ML & optimization

I Going deep learning or not?

→ no guarantee to obtain a good solution

Universal approximation theorems (expressive power)

[Cybenko, 1989 ; Hornik, 1991] with just one hidden layer, one can approximate any C^0 function (on a compact set) arbitrarily well

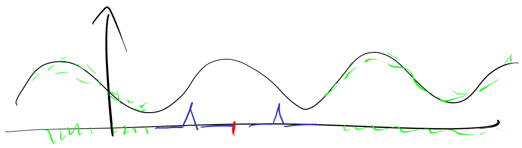


[Sprecher 1965]

[Kolmogorov 1956] $\forall\ C^0\ f,\ \exists N < +\infty,$
$\exists\ \sigma$ $\quad f = \text{Network}_{N,\sigma}$
$\mathbb{R} \to \mathbb{R}'$

↳ hashing functions

⚠ existence theorems ⟹ doesn't provide the solution
⟹ doesn't tell | whether the solution is easy to find
| anything about the optim°
⟹ " " " " generaliz°

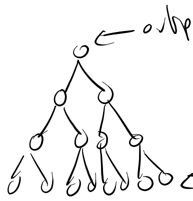


Depth simplifies the approximation / estimation task

. [Lin & co, 2017] multiplication of n variables

→ multiplicat° of 2 variables ≃ 4 neurons

→ m of n variables :



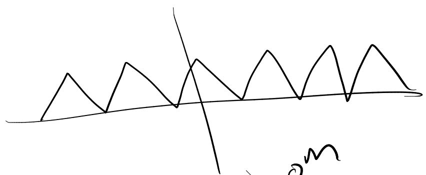
binary tree ↕ log n

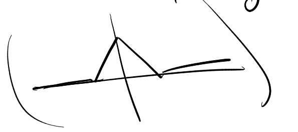
nodes = 4n

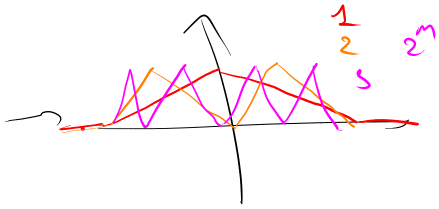
→ Flat network : need 2^n nodes to be carved on binary inputs ⟹ learn by heart

exponential in input dimension

. [Telgarsky 2015] target funct° :







Flat network

2 layers → require $2^{m/2}$ nodes

$\sqrt{m}$ layers → $2^{\sqrt{m}}$ nodes

$x \to o \to o \to o \to o \to$
$\in \mathbb{R}$ $\quad \underbrace{\hphantom{o \to o \to o \to o}}_{\text{depth } m}$

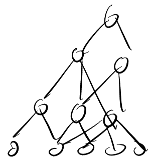
one single node / layer

thin deep network

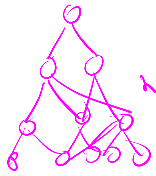
[Mhaskar & co & Poggio] about architecture suitability for f° estimation
2016

Theorem: if target f° = computational tree of input variables
then the # of samples required to train a similar-shaped network : O(d)
vs exp d

target f°



← inputs
end



new network

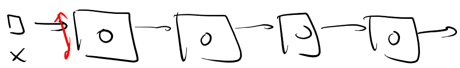
· Depth is sufficient (without width)

previously:

x
N
2 layers
large width ⇒ expressivity

[Lin & Jegelka, 2018]

L → too

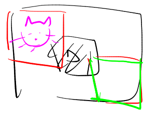


x
1 neuron/each
⇒ universal approxim° power

Does it work? When?

- computer vision & NLP (natural language processing)
  - ↳ big success → designed to handle image properties
    - → invariant to translations
    - ↳ hierarchical models
    - ↳ handle pixel location (geometry)
  - CNN
  - vs random forests (e.g.)

spatially structured data
e.g.: 1D temporal signal
3D videos



Convolution ⇒ any place

- on the opposite:
  if no data structure

ex: medical observations; temperature, coughings?

NS
permut°

→ random forests or SVM
might do the job
will

⇒ if small data :
Neural Network → bad performance

$\boxed{\text{Geometry}}$ is lost → no exploitat°
+ need to learn the appearance of objects at any possible location
⇒ # samples ⇒ huge

- big success in Reinforcement Learning ?
  α-Go, StarCraft, ...
  α-Go/α-0 : game of go : - 44 millions training games (>> one life)
  ≈ total Humanity?
  - trained for only 4 hours on 5000 TPU ≈ GPU graphic card
    very good
    = >2 years of 1 TPU
    ≃ 1000s years of 1 CPU

- computational power : NLP → huge networks
  BERT, XLnet, GPT-3 ... 10^{10} parameters
  ↓
  state of the art for text translation, etc.
  ⇒ re-used through transfer learning

  ⇒ very long time to train on huge GPU clusters
  ⇒ $ electricity
  ⇒ environmental impact

Gap between classical ML & DL

Bayesian view $\Rightarrow$ information theory

$\exp(-\cdot)$

Reminder: classical ML

- samples (x_i, y_i)
- estimate $f: x_i \mapsto \simeq y_i$
- quantify "goodness" with "loss f°" criterion

$$\Rightarrow \inf_{f \in \mathcal{F}} \sum_{\text{examples}} \text{Loss}(f(x_i), y_i) + \text{Regularizer}(f)$$

$\mathcal{F}$: predefined parameterized family

e.g. $\int_\Omega \left\| \frac{\partial f}{\partial x} \right\|^2 dx$

- without regularizer: overfit
- with " $\Rightarrow$ hope for good generalization

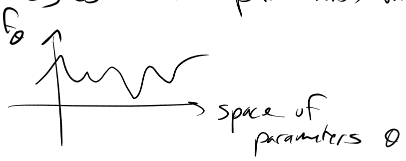
$\Rightarrow$ MDL: minimum description length paradigm

Occam's razor $\rightarrow$ prefer simpler models

↳ prefer models with fewer parameters

Potential issues with DL

- 10^6 parameters ... Occam's razor? MDL?
- models: are able to overfit (easily) + no regularizer!
- possible to train huge models without overfitting (still good generaliz°) ← gap between train error & test error
- " — with fewer samples than parameters
  ↳ estimator (of parameters) convergence?
- highly non-convex optimis° in a high-dimensional space



$\Rightarrow$ supposed to be very hard!
(difficulty : $\sim \exp(\text{dims})$) ← #parameters

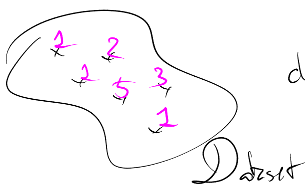
- add noise to optimiz° process $\Rightarrow$ works better
- train to optimize a criterion: cross-entropy
  but evaluate: with accuracy
  ↳ not differential
- common recommendation: new task $\rightarrow$ new architecture $\rightarrow$ check able to overfit! (small part of data)
  $\Rightarrow$ means enough expressive power

A closer look at overfitting

[Zhang & co, 2017]

- huge models can do the job (might not overfit)
- " " can completely overfit also (able to)

Theorem: $\underbrace{n \text{ samples}}_{\text{dataset}} \in \mathbb{R}^{\overbrace{d}^{\text{input dim}}}$ $\exists$ 2-layer network with ReLU activat° f° with $2n+d$ weights that can represent any function on such a dataset



classific° task

random labels $\rightarrow$ perfect fit $\Downarrow$ overfit

$\Rightarrow$ capacity of networks is not the issue

{ Rademacher complexity
Vapnik–Chervonenkis

Palliatives for regularization

→ what about adding a functional regularizer? eg: $\int_{x \in \Omega} \left\| \frac{\partial f}{\partial x} \right\|^2 dx$

⇒ norm of a function

→ first: checking whether $f^\circ = 0$?

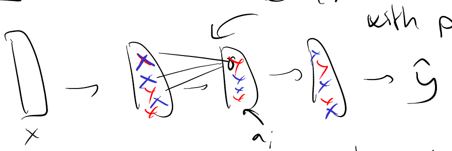
↳ NP-hard (eg: inputs = binary ⇒ SAT pb)

⇒ stochastic approximations of norms = tractable

eg: $\sum_i \left\| \frac{\partial f}{\partial x} \right\|^2_{(x_i)}$

Drop-out

each neuron will be replaced by 0 with probability 1/2



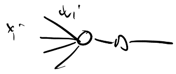
important informat° to the next layer → duplicated

⇒ many ≠ ways to express similar things

⇒ robust

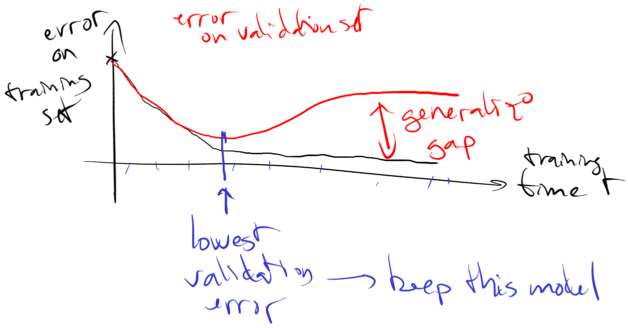
* $\frac{\partial f}{\partial a_i} \simeq 0$ ⇒ make f smooth ⇒ regularizer
* Bayesian point of view → ensemble method → robust

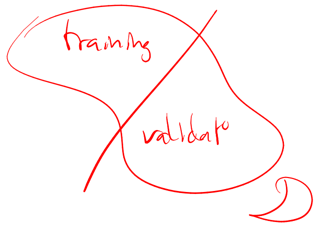
* at test time, use all neurons with activities /2

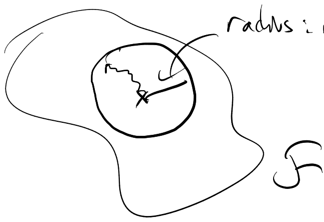


$\sigma\Big(\underbrace{\sum w_i x_i + b}_{\text{pre-activations}}\Big)$ — activ° fo — activity / activat°

Early stopping



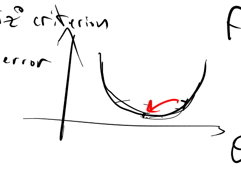


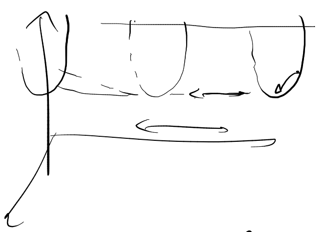


Optimiz° noise acts as a regularizer
(due to SGD, or other)

[Poggio] · around convergence, to a local minimum

Hessian matrix: $\frac{d^2 f}{dw^2}$ → if > 0 : then no pb
→ if not : pb
degenerated





> 0 directions: due to training samples

= 0 directions: include test sample errors

⇒ make the Hessian > 0!

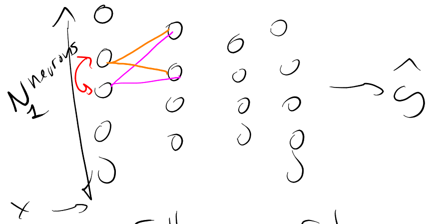
how? → add a weight regularizer: criterion $+ \lambda \sum_i \|w_i\|^2$ works!

"weight decay"

how not? → batch norm → improve optim° but doesn't make H > 0

Optimiz° Landscape

Local minima & saddle points

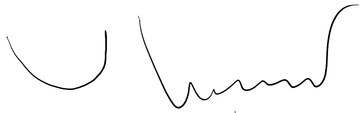


Fully-connected

permutation of neurons in one layer
⇒ get the same function

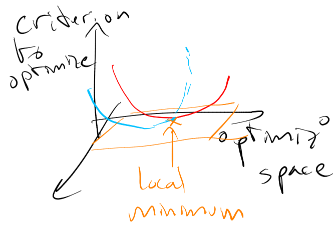
1 local minimum
⇒ duplicates of this minimum

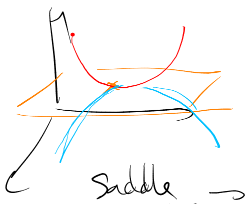
$N_1! \times N_2! \times N_3! \ldots$



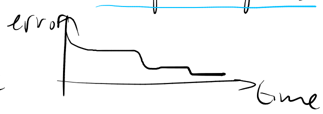
really not convex

- [Poggio] bound on the number of minima
  if activ° funct° σ = polynomial
  $\mathbb{R} \to \mathbb{R}$ ex: $\sigma(x) = x^3$
  then $\left\{\begin{array}{l} \text{output}(x) \\ \text{loss}(x) \end{array}\right. = \text{polynomial}(x)$
  ↳ Bezout's theorem ⇒ bound depending on the degree
- So many parameters: local minimum = very strong notion (huge neighborhood in optimiz° space)
  ⇒ local minima = very good
- many local minima ⇒ even more: saddle points





Saddle point → Issue: slow down the optimiz° process



critical points

[Dauphin, 2014]
Hessian: second derivative
↳ symmetric matrices

standard law on random symmetric matrices:

$P^T D P$

↑ pick a random diagonal matrix

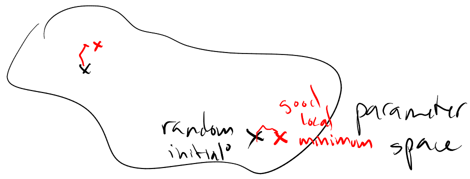
↑ pick an random orthogonal matrix

⇒ chances to have all eigenvalues of the same sign $\sim 2^{\#\text{parameters}}$

Lots of works on convergence:

- always under strong hypotheses
  specific to a particular architecture (eg: 2 layers)

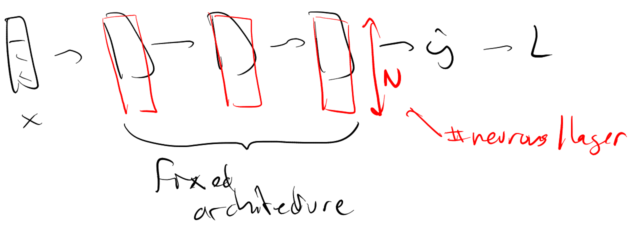
over-parametrized networks

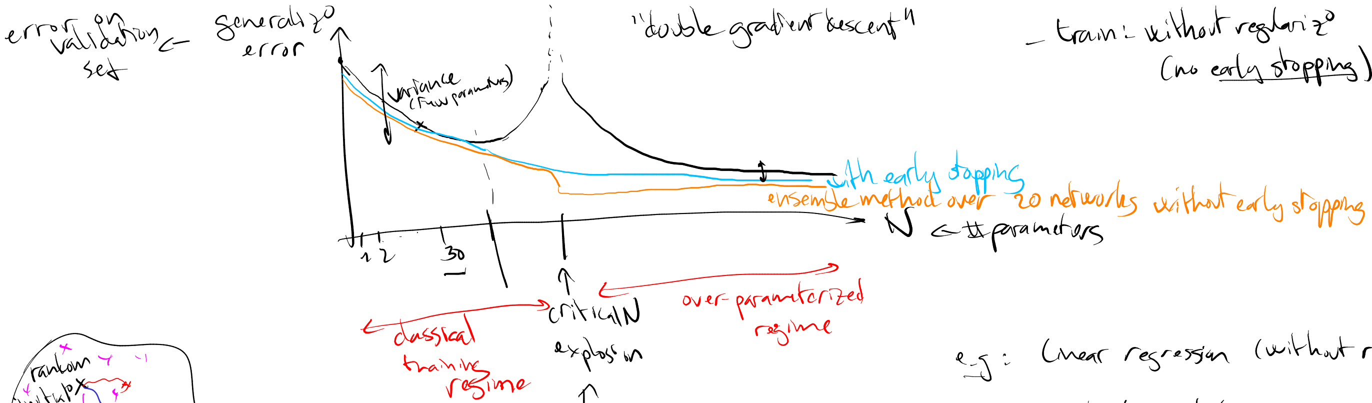


⇒ Francis Bach
↳ "lazy training regime"

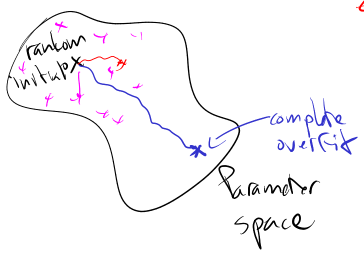
Over-parameterization helps

[Geiger et al, 2019]





– train: without regulariz° (no early stopping)



e.g.: linear regression (without regularizer)

linear system
$$\begin{cases} \alpha_1 x_1 + \alpha_2 x_2 \dots = b_2 \\ \vdots \\ \dots \end{cases}$$

N variables, N equations ⇒ exactly one solution

↳ data: a bit noisy
⇒ overfit exactly the noise

→ prove theoretically: (without early stopping)

denote by
- f_N a trained network with N neurons/layer
- $\bar{f}_N$: average of trained networks with all possible random init°

$\bar{f}_N \to f_\infty$

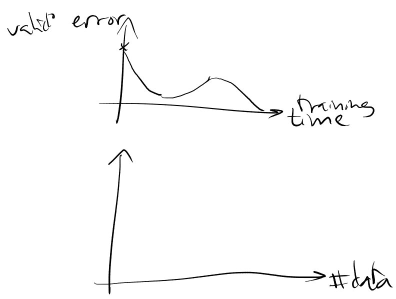
$$\| f_N - \bar{f}_N \| = O\left(\frac{1}{\sqrt{N}}\right)$$

or ∞?

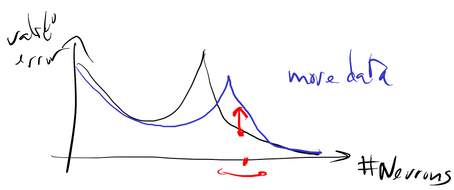
⇒ the training of neural networks gets robust to init° for large N (# neurons)

⇒ don't worry about large # of neurons causing overfit / optimiz° issues
⇒ put as many neurons as you can

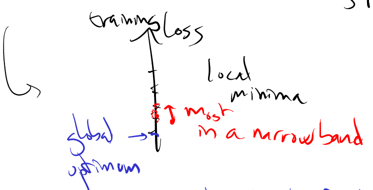
Further: [Nakkiran & co, 2019]



Large architecture without regulariz°

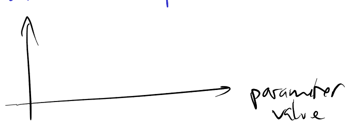


More: [Choromanska & al, 2015]
- optimiz° a network ⟷ Hamiltonian of spherical spin-glass model
↳ results on statistics over critical points



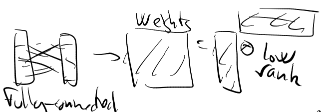
→ # minima outside that band $\propto e^{-\text{Network size}}$

↳ gets harder to find → but: global minimum = prone to overfit

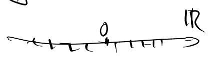


About Minimum Description Length paradigm

- MDL is lost? # parameters = huge, but:
  - no precision needed when encoding parameters
  - high redundancy: different parts of the network may compute similar things
    - ↳ NN compression: prune / quantize weights / ↳ tensor factoriz° } compression factor > 100 → [Louizos & al, 2017]
    - ⇒ run online video object detection on a smartphone



double precision: 8 bytes
float → 4 bytes



8 possible values → 3 bit (< ½ byte)