Petite histoire des algorithmes et perspectives de l'Intelligence Artificielle

Nicolas Vayatis

CMLA, ENS Paris-Saclay

Juin 2019

Forum TERATEC

Notre écosystème

Recherche

CMLA
Centre de Mathématiques
et de Leurs Applications

Formation

MATHÉMATIQUES
VISION
APPRENTISSAGE

MASTER M2 MVA
MATHÉMATIQUES, VISION, APPRENTISSAGE

école normale supérieure paris-saclay · CentraleSupélec · École des Ponts · ENSAE · ENSTA · MINES PARIS ParisTech · ÉCOLE POLYTECHNIQUE · TELECOM ParisTech · DAUPHINE UNIVERSITÉ PARIS

Partenariats

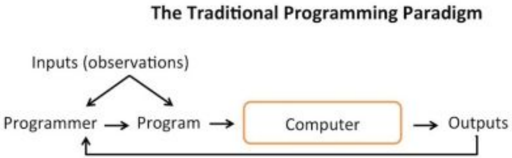
DAP
DATA
ANALYTICS
POST

Sensibiliser, démystifier, co-développer

Contact : `<vayatis@cmla.ens-cachan.fr>`

Machine learning in a nutshell

Symbolic AI vs. Machine Learning

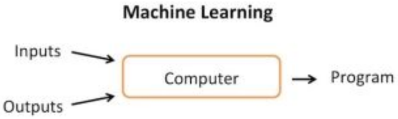


The Traditional Programming Paradigm

Inputs (observations)

Programmer → Program → Computer → Outputs

Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed
– Arthur Samuel (1959)



Machine Learning

Inputs

Outputs

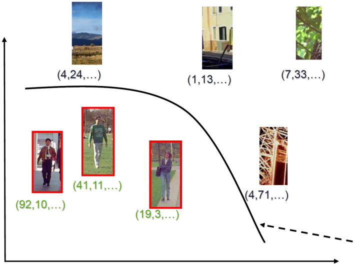
Computer → Program

Sebastian Raschka, 2016

The goal of machine learning

Finding a function

- Example : Pedestrian detection from video cameras

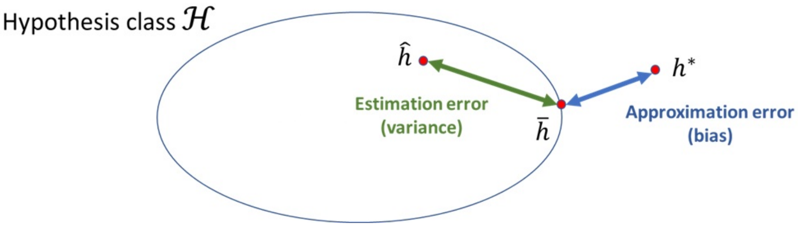


- What is the search space for such a function?

The art of machine learning

Solving the "bias-variance" trade-off

- Distance between solution provided by a learning method and the optimal solution (function): sum of *Approximation error* and *Estimation error*



- Learning a function amounts to:
  (a) chosing a search space (design process),
  (b) estimating the best function in this space (training process).

Mainstream ML methods

The three central paradigms of ML

1. Local methods: based on grouping and local voting (or averaging)
    - k-Nearest-Neighbors
    - Kernel rules
    - Decision trees
2. Global methods: based on functional optimization
    - Regularized regression (Ridge, LASSO...)
    - Support Vector Machines
    - Boosting
    - Feedforward neural networks
3. Ensemble methods: based on resampling and aggregation
    - Bagging
    - Boosting
    - Random forests

Shallow vs. Deep Learning

- Shallow learning: often relates to Tikohnov's regularization

$$\min_{h\in\mathcal{H}}\left(\frac{1}{n}\sum_{i=1}^{n}\ell(h(X_i),Y_i)+\lambda_n\cdot\mathrm{pen}(h,n)\right)$$

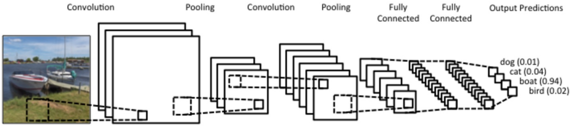
  - The penalty controls the variance term (Occam's razzor)
  - It may also induce a desired structure of the function (e.g. sparsity).

- Deep Learning:

  - Universal approximators (zero bias)
  - No penalty term in the optimization but lots of tricks in the implementation which amount to *implicit regularization*

The case of Deep Learning

Deep Feedforward Network (DFN)



- Search space: functions of the form

$$h(x, \theta) = \sigma_m \circ A_m \circ \sigma_{m-1} \circ \ldots \circ A_2 \circ \sigma_1 \circ A_1 x$$

where $\theta = (A_1, \ldots, A_m)$ sequence of parameters to be estimated through learning, and $\sigma = (\sigma_1, \ldots, \sigma_m)$ are the so-called activation functions.

Learning a DFN
A two-stage procedure

Design process - The architecture of the Deep Network has to be selected. This amounts to chosing *hyperparameters*:

- The number of layers m
- The nature of the activation functions σ (sigmoid, ReLU...)
- The number of units per layer d_j (number of rows of A_j, $j = 1, \ldots, m$)
- Plus various optional operators used at each layer (pooling, convolution...)

Training process (next slide)

Training a DFN

For a *given* set of hyperparameters, find θ from the data

- Optimization *objective* (far from convex!):

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(h(X_i, \theta), Y_i)$$

- Optimization *method* based on stochastic gradient descent (iterates over data points)

$$\theta_{i+1} = \theta_i - \eta \frac{\partial \ell(h(X_i, \theta), Y_i)}{\partial \theta}(\theta_i)$$

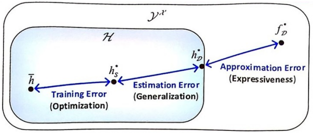
until convergence...

Bias-variance revisited

Approximation, Estimation and Optimization

[The trade-offs of Large Scale Learning, L. Bottou, O. Bousquet, 2011]

$$\begin{aligned}\mathcal{E} &= \mathbb{E}\left[E(f^*_{\mathcal{F}}) - E(f^*)\right] + \mathbb{E}\left[E(f_n) - E(f^*_{\mathcal{F}})\right] + \mathbb{E}\left[E(\tilde{f}_n) - E(f_n)\right] \\ &= \mathcal{E}_{\mathrm{app}} + \mathcal{E}_{\mathrm{est}} + \mathcal{E}_{\mathrm{opt}}.\end{aligned}$$



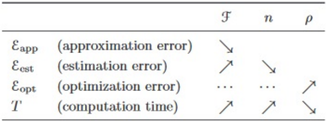
$f^*_{\mathcal{D}}$ – ground truth ($\mathrm{argmin}_{f \in \mathcal{Y}^{\mathcal{X}}} L_{\mathcal{D}}(f)$)

$h^*_{\mathcal{D}}$ – optimal hypothesis ($\mathrm{argmin}_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$)

h^*_S – empirically optimal hypothesis ($\mathrm{argmin}_{h \in \mathcal{H}} L_S(h)$)

$\bar{h}$ – returned hypothesis

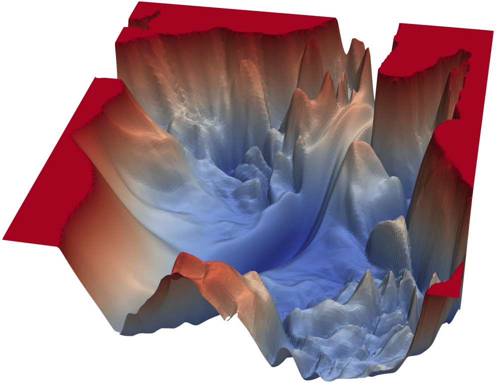
Trade-off wrt: Search space $\mathcal{F}$, sample size n, numerical tolerance ρ



		$\mathcal{F}$	n	ρ
$\mathcal{E}_{\mathrm{app}}$	(approximation error)	↘		
$\mathcal{E}_{\mathrm{est}}$	(estimation error)	↗	↘	
$\mathcal{E}_{\mathrm{opt}}$	(optimization error)	$\cdots$	$\cdots$	↗
T	(computation time)	↗	↗	↘

The loss landscape of Deep Learning

View on a 56-layer neural network without skip-connection



From [Visualizing the Loss Landscape of Neural Nets,
H. Li, Z. Xu1, G. Taylor, C. Studer, T. Goldstein, 2018]

Some hope for Deep Learning

- Under certain conditions, no poor local minima

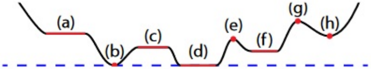


Figure 1. Example critical points of a non-convex function (shown in red). (a,c) Plateaus. (b,d) Global minima. (e,g) Local maxima. (f,h) Local minima.

- SGD avoids bad critical points

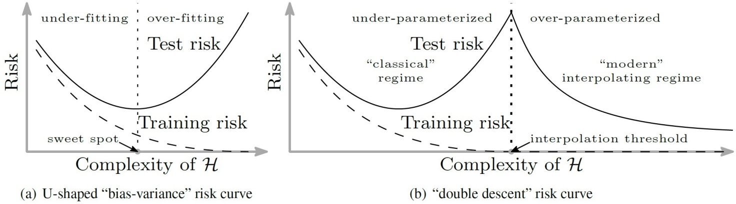
- Larger networks are better behaved (local minima are global)

References:

Soudry and Carmon (2016), "No bad local minima: Data independent training error guarantees for multilayer neural networks".
Kawaguchi (2016), "Deep learning without poor local minima".
Haeffele and Vidal (2017), "Global optimality in neural network training".
Janzamin, Sedghi, and Anandkumar (2015), "Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods".
Panageas and Piliouras (2016), "Gradient descent only converges to minimizers: Non-isolated critical points and invariant regions".
Brutzkus, Alon et al. (2017), "SGD Learns Over-parameterized Networks that Provably Generalize on Linearly Separable Data".

The theory of a double descent risk curve

How Deep Learning (and random forests) avoid overfitting



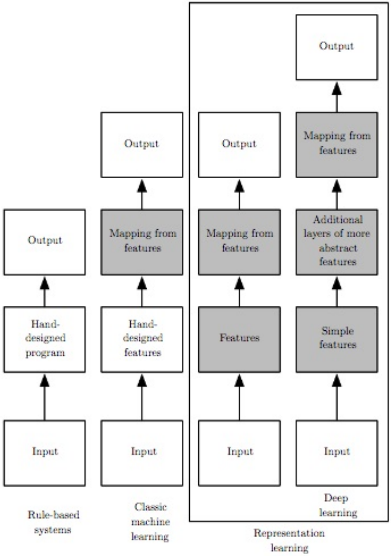
(a) U-shaped "bias-variance" risk curve

(b) "double descent" risk curve

From [Reconciling modern machine learning and the bias-variance trade-off, M. Belkin, D. Hsu, S. Ma, S. Mandal, 2018]

The seven sins of Deep Learning

The big picture



Some facts about Deep Learning

1. Design process involves random search in ”cursed” spaces
2. Training process needs huge amounts of data
3. Training process highly demanding in computing power
4. Failure of reproducibility
5. Representation learning generates ”monsters”
6. Leads to black-box decision systems
7. Performs often worst than shallow learning

Why Deep Learning may not be the cure

- How to bridge prediction with optimization? Towards **risk communication**...
- The **design process** is more complex than the training process and often leads to suboptimal architectures
- **Representation learning** is not magical: the structure of the search space has to reflect the "physics" underlying the data
- The success of *any* Machine Learning method is tied to the **assumption of stationarity**: this is not handled by the ML method itself but requires a monitoring algorithm assessing observed performance and breaks of stationarity

Discussion

The key ingredients of *any* Machine Learning method

- Information (data compression, data representation)
- Design process
- Training process
- Assessment and Monitoring

Research worth being done

- Hybrid modeling

  physics- or simulation-based AND data-driven

- Expert-based and data-driven representation learning

  Plugging prior knowledge into the representation learning step and also in the regularization principles

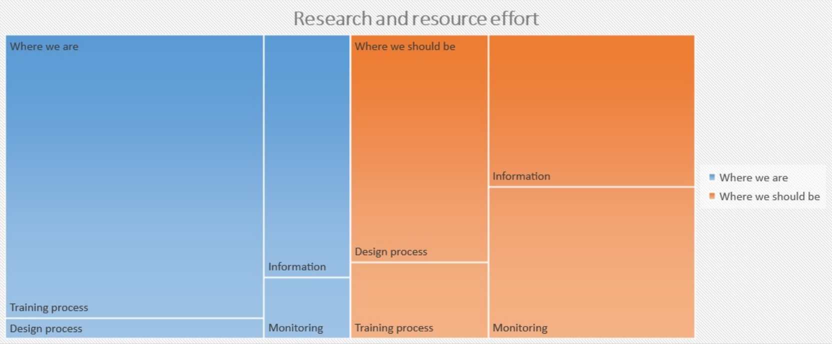
- Global optimization

  Useful for architecture design to refine the design stage

- Statistical procedures and signal processing for monitoring ML methods

  Such as homogeneity tests, confidence bounds, breakpoint detection

For researchers and engineers


Research and resource effort
Where we are
Training process
Design process
Information
Monitoring
Where we should be
Design process
Training process
Information
Monitoring
🟦 Where we are
🟧 Where we should be