# Advanced Learning Models

Julien Mairal and Jakob Verbeek

Inria Grenoble

MSIAM/MoSIG, 2018/2019/

## Goal

Introducing two major paradigms in machine learning called kernel methods and neural networks.

## Ressources

- check the website of the course. `http://thoth.inrialpes.fr/people/mairal/teaching/2018-2019/MSIAM/`.

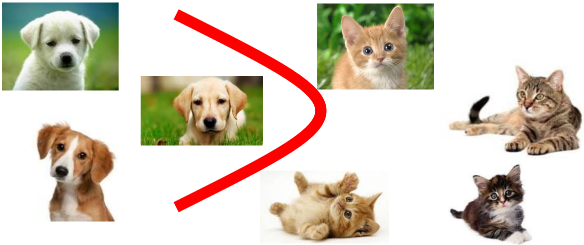
## Grading

- 1 homework (30%), one data challenge (30%) and one exam (40%).
- 1 data challenge; can also be done by teams of two students;

# Common paradigm: optimization for machine learning

Optimization is central to machine learning. For instance, in supervised learning, the goal is to learn a **prediction function** $f : \mathcal{X} \to \mathcal{Y}$ given labeled training data $(x_i, y_i)_{i=1,\ldots,n}$ with $x_i$ in $\mathcal{X}$, and $y_i$ in $\mathcal{Y}$:

$$\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda\Omega(f)}_{\text{regularization}} .$$



[Vapnik, 1995, Bottou, Curtis, and Nocedal, 2016]...

# Common paradigm: optimization for machine learning

Optimization is central to machine learning. For instance, in supervised learning, the goal is to learn a **prediction function** $f : \mathcal{X} \to \mathcal{Y}$ given labeled training data $(x_i, y_i)_{i=1,\ldots,n}$ with $x_i$ in $\mathcal{X}$, and $y_i$ in $\mathcal{Y}$:

$$\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda\Omega(f)}_{\text{regularization}} .$$

The scalars $y_i$ are in

- $\{-1, +1\}$ for **binary** classification problems.
- $\{1, \ldots, K\}$ for **multi-class** classification problems.
- $\mathbb{R}$ for **regression** problems.
- $\mathbb{R}^k$ for **multivariate regression** problems.

# Common paradigm: optimization for machine learning

Optimization is central to machine learning. For instance, in supervised learning, the goal is to learn a **prediction function** $f : \mathcal{X} \to \mathcal{Y}$ given labeled training data $(x_i, y_i)_{i=1,\ldots,n}$ with $x_i$ in $\mathcal{X}$, and $y_i$ in $\mathcal{Y}$:

$$\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda \Omega(f)}_{\text{regularization}}.$$

Example with linear models: logistic regression, SVMs, *etc.*

- assume there exists a linear relation between $y$ and features $x$ in $\mathbb{R}^p$.
- $f(x) = w^\top x + b$ is parametrized by $w, b$ in $\mathbb{R}^{p+1}$;
- $L$ is often a **convex** loss function;
- $\Omega(f)$ is often the squared $\ell_2$-norm $\|w\|^2$.

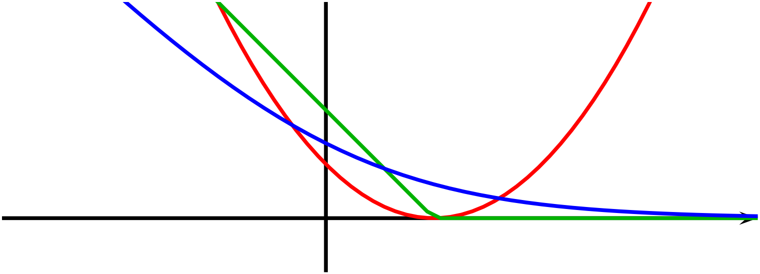
# Common paradigm: optimization for machine learning

A few examples of linear models with no bias $b$:

**Ridge regression:** $$\min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2}(y_i - w^\top x_i)^2 + \lambda \|w\|_2^2.$$

**Linear SVM:** $$\min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} \max(0, 1 - y_i w^\top x_i) + \lambda \|w\|_2^2.$$

**Logistic regression:** $$\min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} \log\left(1 + e^{-y_i w^\top x_i}\right) + \lambda \|w\|_2^2.$$



# Common paradigm: optimization for machine learning

The previous formulation is called *empirical risk minimization*; it follows a classical scientific paradigm:

1. **observe** the world (gather data);
2. **propose models** of the world (design and learn);
3. **test** on new data (estimate the generalization error).

# Common paradigm: optimization for machine learning

The previous formulation is called *empirical risk minimization*; it follows a classical scientific paradigm:

1. **observe** the world (gather data);
2. **propose models** of the world (design and learn);
3. **test** on new data (estimate the generalization error).

## A general principle

It underlies many paradigms:

- **deep neural networks,**
- **kernel methods,**
- sparse estimation.

# Common paradigm: optimization for machine learning

The previous formulation is called *empirical risk minimization*; it follows a classical scientific paradigm:

1. **observe** the world (gather data);
2. **propose models** of the world (design and learn);
3. **test** on new data (estimate the generalization error).

Even with simple linear models, it leads to challenging problems in optimization: develop algorithms that

- **scale** both in the problem size $n$ and dimension $p$;
- are able to **exploit the problem structure** (sum, composite);
- come with **convergence and numerical stability** guarantees;
- come with **statistical guarantees**.

# Common paradigm: optimization for machine learning

The previous formulation is called *empirical risk minimization*; it follows a classical scientific paradigm:

1. **observe** the world (gather data);
2. **propose models** of the world (design and learn);
3. **test** on new data (estimate the generalization error).

## It is not limited to supervised learning

$$\min_{f \in \mathcal{F}} \quad \frac{1}{n} \sum_{i=1}^{n} L(f(x_i)) \; + \; \lambda \Omega(f).$$

- $L$ is not a classification loss any more;
- K-means, PCA, EM with mixture of Gaussian, matrix factorization,... can be expressed that way.

# Paradigm 1: Deep neural networks

$$\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda\Omega(f)}_{\text{regularization}}.$$

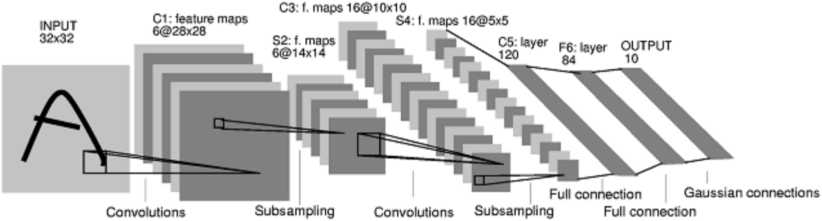
- The "deep learning" space $\mathcal{F}$ is parametrized:

$$f(x) = \sigma_k(A_k\sigma_{k-1}(A_{k-1}\ldots\sigma_2(A_2\sigma_1(A_1x))\ldots)).$$

- Finding the optimal $A_1, A_2, \ldots, A_k$ yields an (intractable) **non-convex** optimization problem in **huge dimension.**
- Linear operations are either unconstrained (fully connected) or involve parameter sharing (e.g., convolutions).

# Paradigm 1: Deep neural networks

## A quick zoom on convolutional neural networks



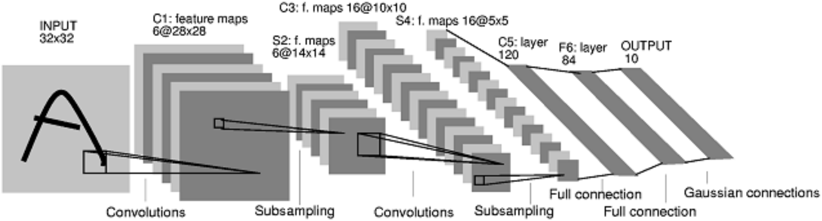

What are the main features of CNNs?

- they capture **compositional** and **multiscale** structures in images;
- they provide some **invariance**;
- they model **local stationarity** of images at several scales.
- **state-of-the-art** in many fields.

[LeCun et al., 1989, 1998, Ciresan et al., 2012, Krizhevsky et al., 2012]...

# Paradigm 1: Deep neural networks

## A quick zoom on convolutional neural networks



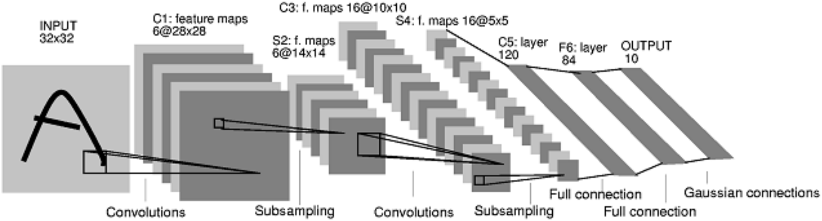

What are the main open problems?

- very little **theoretical understanding**;
- they require **large amounts of labeled data**;
- they require **manual design and parameter tuning**;
- how to **regularize** is unclear;

[LeCun et al., 1989, 1998, Ciresan et al., 2012, Krizhevsky et al., 2012]...

# Paradigm 1: Deep neural networks

## A quick zoom on convolutional neural networks




### How to use them?

- they are the focus of a **huge academic and industrial effort**;
- there is **efficient and well-documented open-source software**;

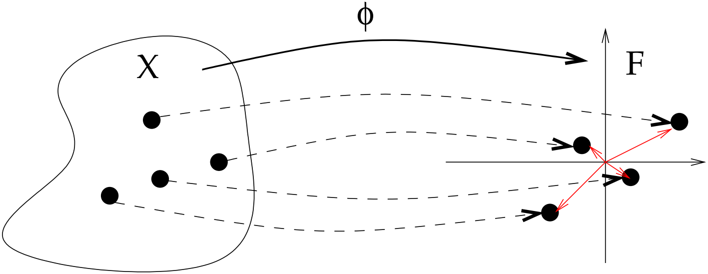
[LeCun et al., 1989, 1998, Ciresan et al., 2012, Krizhevsky et al., 2012]...

# Paradigm 2: Kernel methods

$$\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i)) + \lambda\|f\|_{\mathcal{H}}^2.$$

- **map** data $x$ in $\mathcal{X}$ to a Hilbert space and work with **linear forms**:

$$\varphi : \mathcal{X} \to \mathcal{H} \qquad \text{and} \qquad f(x) = \langle \varphi(x), f\rangle_{\mathcal{H}}.$$




[Shawe-Taylor and Cristianini, 2004, Schölkopf and Smola, 2002]...

# Paradigm 2: Kernel methods

$$\min_{f \in \mathcal{H}} \quad \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) \; + \; \lambda \|f\|_{\mathcal{H}}^2.$$

First purpose: embed data in a vectorial space where

- many **geometrical operations** exist (angle computation, projection on linear subspaces, definition of barycenters....).
- one may learn potentially **rich infinite-dimensional models**.
- **regularization** is natural (see next...)

# Paradigm 2: Kernel methods

$$\min_{f \in \mathcal{H}} \quad \frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i)) \; + \; \lambda \|f\|_{\mathcal{H}}^2.$$

First purpose: embed data in a vectorial space where

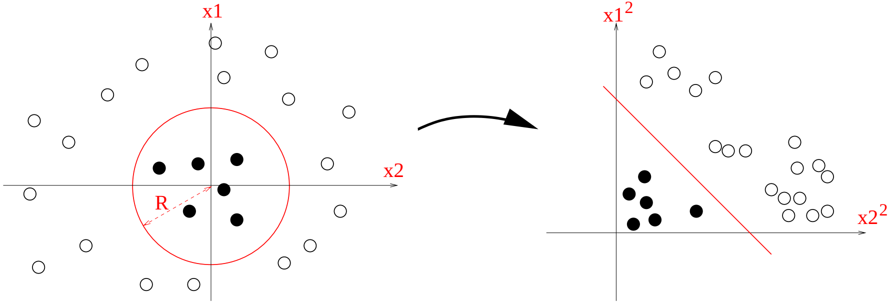
- many **geometrical operations** exist (angle computation, projection on linear subspaces, definition of barycenters....).
- one may learn potentially **rich infinite-dimensional models**.
- **regularization** is natural (see next...)

The principle is **generic** and does not assume anything about the nature of the set $\mathcal{X}$ (vectors, sets, graphs, sequences).

# Paradigm 2: Kernel methods

Second purpose: unhappy with the current Euclidean structure?

- lift data to a higher-dimensional space with **nicer properties** (e.g., linear separability, clustering structure).
- then, the **linear** form $f(x) = \langle \varphi(x), f \rangle_{\mathcal{H}}$ in $\mathcal{H}$ may correspond to a **non-linear** model in $\mathcal{X}$.

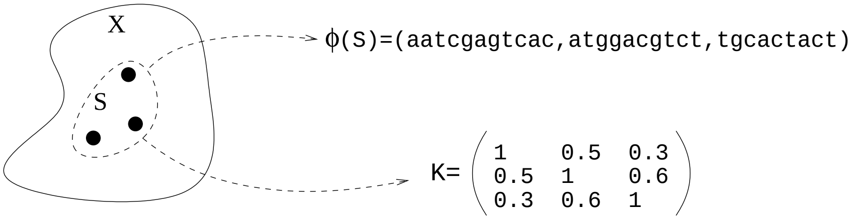


# Paradigm 2: Kernel methods

How does it work? representation by pairwise comparisons

- Define a "comparison function": $K : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$.
- Represent a set of $n$ data points $\mathcal{S} = \{x_1, \ldots, x_n\}$ by the $n \times n$ **matrix**:

$$\mathbf{K}_{ij} := K(x_i, x_j).$$



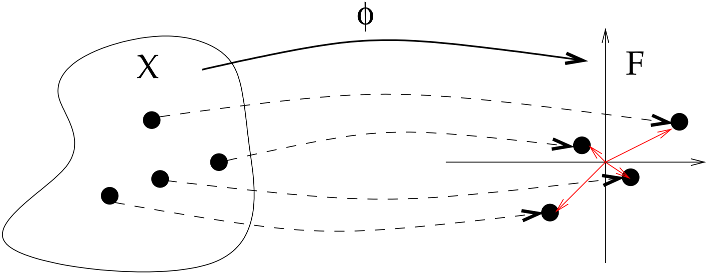

$$K=\begin{pmatrix} 1 & 0.5 & 0.3 \\ 0.5 & 1 & 0.6 \\ 0.3 & 0.6 & 1 \end{pmatrix}$$

# Paradigm 2: Kernel methods

## Theorem (Aronszajn, 1950)

$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive definite kernel if and only if there exists a Hilbert space $\mathcal{H}$ and a mapping $\varphi : \mathcal{X} \to \mathcal{H}$, such that

$$\text{for any } x, x' \text{ in } \mathcal{X}, \qquad K(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathcal{H}}.$$



# Paradigm 2: Kernel methods

## Mathematical details

- the only thing we require about $K$ is **symmetry** and **positive definiteness**

$$\forall x_1, \ldots, x_n \in \mathcal{X}, \alpha_1, \ldots, \alpha_n \in \mathbb{R}, \quad \sum_{ij} \alpha_i \alpha_j K(x_i, x_j) \geq 0.$$

- then, there exists a Hilbert space $\mathcal{H}$ of functions $f : \mathcal{X} \to \mathbb{R}$, called the **reproducing kernel Hilbert space (RKHS)** such that

$$\forall f \in \mathcal{H}, x \in \mathcal{X}, \quad f(x) = \langle \varphi(x), f \rangle_{\mathcal{H}},$$

  and the mapping $\varphi : \mathcal{X} \to \mathcal{H}$ (from Aronszajn's theorem) satisfies

$$\varphi(x) : y \mapsto K(x, y).$$

# Paradigm 2: Kernel methods

Why mapping data in $\mathcal{X}$ to the functional space $\mathcal{H}$?

- it becomes feasible to learn a prediction function $f \in \mathcal{H}$:

$$\min_{f \in \mathcal{H}} \underbrace{\frac{1}{n}\sum_{i=1}^{n} L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda \|f\|_{\mathcal{H}}^2}_{\text{regularization}}.$$

  (why? the solution lives in a finite-dimensional hyperplane).

- **non-linear** operations in $\mathcal{X}$ become **inner-products** in $\mathcal{H}$ since

$$\forall f \in \mathcal{H}, x \in \mathcal{X}, \quad f(x) = \langle \varphi(x), f \rangle_{\mathcal{H}}.$$

- the norm of the RKHS is a **natural regularization function**:

$$|f(x) - f(x')| \leq \|f\|_{\mathcal{H}} \|\varphi(x) - \varphi(x')\|_{\mathcal{H}}.$$

# Paradigm 2: Kernel methods

What are the main features of kernel methods?

- builds **well-studied functional spaces** to do machine learning;
- **decoupling** of data representation and learning algorithm;
- typically, **convex optimization problems** in a supervised context;
- **versatility**: applies to vectors, sequences, graphs, sets,...;
- **natural regularization function** to control the learning capacity;

[Shawe-Taylor and Cristianini, 2004, Schölkopf and Smola, 2002, Müller et al., 2001]

# Paradigm 2: Kernel methods

What are the main features of kernel methods?

- builds **well-studied functional spaces** to do machine learning;
- **decoupling** of data representation and learning algorithm;
- typically, **convex optimization problems** in a supervised context;
- **versatility**: applies to vectors, sequences, graphs, sets,. . . ;
- **natural regularization function** to control the learning capacity;

But...

- **decoupling** of data representation and learning may not be a good thing, according to recent **supervised** deep learning success.
- requires **kernel design**.
- $O(n^2)$ **scalability problems**.

[Shawe-Taylor and Cristianini, 2004, Schölkopf and Smola, 2002, Müller et al., 2001]

# Course Organization

We will alternate "kernel method classes", given by Julien Mairal, and "neural network classes" given by Jakob Verbeek.

Eventually, we may end up showing that the two paradigms are much closer to each other than one may think at first sight.

# References I


Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *arXiv preprint arXiv:1606.04838*, 2016.

D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *P. IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

# References II


K-R Müller, Sebastian Mika, Gunnar Ratsch, Koji Tsuda, and Bernhard Scholkopf. An introduction to kernel-based learning algorithms. *IEEE transactions on neural networks*, 12(2):181–201, 2001.

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

John Shawe-Taylor and Nello Cristianini. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, 2004.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1995.