Learning theory from first principles

Lecture 8: Neural networks

Francis Bach

November 13, 2020

Class summary

- Single hidden layer neural networks
- Estimation error
- Approximation properties and universality

1 Introduction

In this course, the main focus has been on methods to learn from n observations $(x_i, y_i), i = 1, \dots, n$, with $x_i \in \mathcal{X}$ (input space) and $y_i \in \mathcal{Y}$ (output/label space).

As presented in Lecture 3, a large class of methods relies on minimizing a regularized empirical risk with respect to a function $f : \mathcal{X} \to \mathbb{R}$ where the following cost function is minimized:

$$\frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) + \Omega(f),$$

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$ is a loss function, and $\Omega(f)$ is a regularization term. Typical examples were:

- **Regression:** $\mathcal{Y} = \mathbb{R}$ and $\ell(y_i, f(x_i)) = \frac{1}{2}(y_i - f(x_i))^2$.
- **Classification:** $\mathcal{Y} = \{-1, 1\}$ and $\ell(y_i, f(x_i)) = \Phi(y_i f(x_i)))$ where Φ is convex, e.g., $\Phi(u) = \max\{1 - u, 0\}$ (hinge loss leading to the support vector machine) or $\Phi(u) = \log(1 + \exp(-u))$ (leading to logistic regression).

The class of functions we have considered so far were:

- **Linear functions in some explicit features:** given a feature map $\varphi : \mathcal{X} \to \mathbb{R}^d$, we consider $f(x) = \theta^\top \varphi(x)$, with parameters $\theta \in \mathbb{R}^d$, as analyzed in Lecture 2 (for least-squares) and Lecture 3.

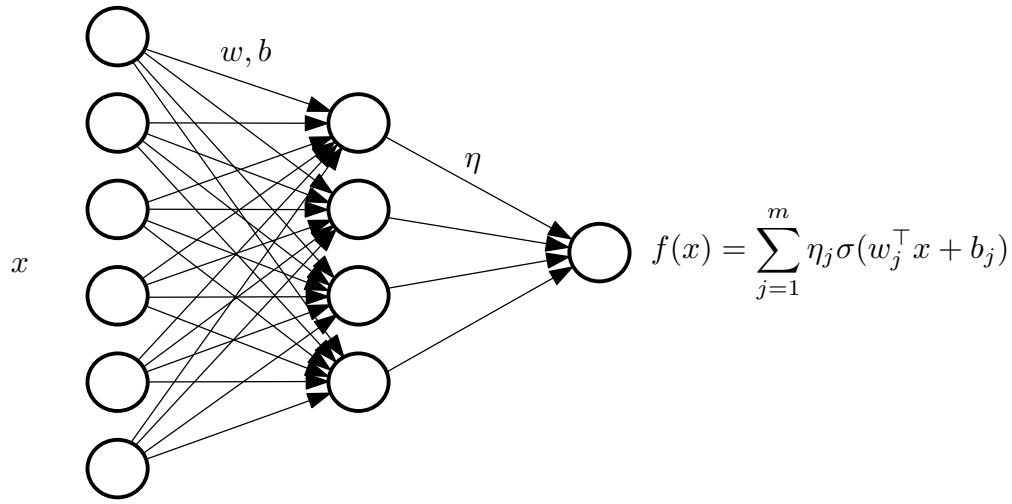
  Pros: simple to implement, convex optimization (gradient descent). Complexity in $O(nd)$.
  Cons: only applies to linear functions on explicit (and fixed feature spaces).

- **Linear functions in some implicit features through kernel methods:** the feature map can have arbitrarily large dimension, that is, $\varphi(x) \in \mathcal{H}$ where $\mathcal{H}$ is a Hilbert space, accessed through a kernel $k(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathcal{H}}$, as presented in Lecture 7.

  Pros: non-linear flexible predictions, simple to implement, convex optimization.
  Cons: complexity in $O(n^2)$.

The goal of this lecture is to explore another class of functions for non-linear predictions, namely neural networks, that come with additional benefits (such as more adaptivity), but comes with some potential drawbacks, such as a harder optimization problem.

2 Single hidden layer neural network

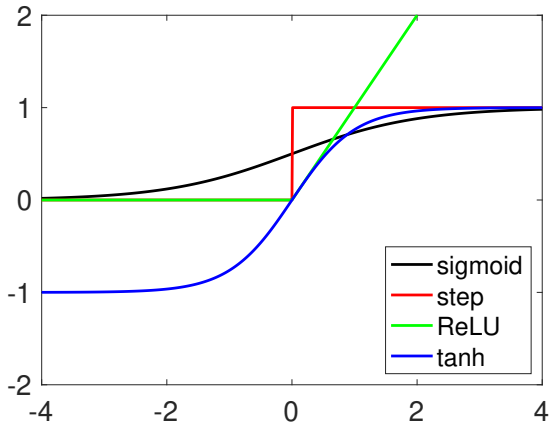


We consider $\mathcal{X} = \mathbb{R}^d$ and the set of functions that can be written as

$$f(x) = \sum_{j=1}^{m} \eta_j \sigma(w_j^\top x + b_j), \tag{1}$$

where $w_j \in \mathbb{R}$, $b_j \in \mathbb{R}$ and $\eta_j \in \mathbb{R}$, $j = 1, \ldots, m$, and σ is an activation function, typically from one of the following examples (see plot below):

- sigmoid $\sigma(u) = \frac{1}{1+e^{-u}}$,
- step $\sigma(u) = 1_{u>0}$,
- rectified linear unit (ReLU) $\sigma(u) = (u)_+ = \max\{u, 0\}$,
- hyperbolic tangent $\sigma(u) = \tanh(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}$.



The function f is defined as the linear combination of m functions $x \mapsto \sigma(w_j^\top x + b_j)$, which are the "hidden neurons".

The constant terms b_j are sometimes referred to as "biases", which is unfortunate in a statistical context.

Do not get confused by the name "neural network" and its biological inspiration. This inspiration is not a proper justification of its behavior on machine learning problems.

Following standard practice, we are not adding a non-linearity for the last layer; note that if we were to use an additional sigmoid activation and using the cross-entropy loss for binary classification, we would exactly be using the logistic loss on the output without extra activation function.

2.1 Optimization

In order to find parameters $\theta = \{(\eta_j), (w_j), (b_j)\} \in \mathbb{R}^{m(d+2)}$, the following optimization problem has to be solved:

$$\min_{\theta \in \mathbb{R}^{m(d+2)}} \frac{1}{n} \sum_{i=1}^{n} \ell\Big(y_i, \sum_{j=1}^{m} \eta_j \sigma(w_j^\top x_i + b_j)\Big).$$

⚠ Note that in the true objective is to perform well on unseen data, and the optimization problem is just a mean to an end. See Lecture 3 and 4.

This is a non-convex problem where the gradient descent algorithms from Lecture 4 can be applied without guarantees (see Lecture 9 for recent results on providing some qualitative global convergence guarantees when m is large). Sometimes regularization is added on the parameters.

While stochastic gradient descent remains an algorithm of choice, several tricks have been observed to lead to better stability and performance: specific step-size decay schedules, momentum, batch-normalization,

etc. But overall, the objective function is non-convex, and it remains difficult to understand why gradient-based methods perform well in practice (some elements in Lecture 9).

See `https://playground.tensorflow.org/` for a nice interactive illustration.

2.2 Estimation error

In order to study the estimation error, we will consider that the parameters of the network are constrained, that is, $\Omega(\theta) \leqslant D$ for a certain norm Ω that we will define below. We can then compute the Rademacher complexity of the class of function $\mathcal{F}$ we just defined, using tools from Lecture 3.

We consider an ℓ_1-bound $\|\eta\|_1 \leqslant D_\eta$, as this will be our main tool for approximation theory in later sections.

We have, by definition, and taking expectation with respect to the data (x_i, y_i), $i = 1, \dots, n$ (which is assumed i.i.d.) and the independent Rademacher random variables $\varepsilon_i \in \{-1, 1\}$:

$$R_n(\mathcal{F}) = \mathbb{E}\Big[\sup_{\theta \in \mathbb{R}^{m(d+2)}} \frac{1}{n} \sum_{i=1}^n \varepsilon_i \ell(y_i, f_\theta(x_i))\Big].$$

Assuming the loss is almost surely G_ℓ-Lipschitz-continuous with respect to the second variable, using Proposition 3 from Lecture 3 that allows to get rid of the loss, we get the bound:

$$R_n(\mathcal{F}) \leqslant G_\ell \, \mathbb{E}\Big[\sup_{\theta \in \mathbb{R}^{m(d+2)}} \frac{1}{n} \sum_{i=1}^n \varepsilon_i f_\theta(x_i)\Big] = G_\ell \, \mathbb{E}\Big[\sup_{\theta \in \mathbb{R}^{m(d+2)}} \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^m \eta_j \varepsilon_i \sigma(w_j^\top x_i + b_j)\Big].$$

Using the ℓ_1-constraint on η and using $\sup_{\|\eta\|_1 \leqslant D_\eta} z^\top \eta = \|z\|_\infty$, we can directly maximize with respect to η, leading to (note that another ℓ_p constraint on η would be harder to deal with):

$$R_n(\mathcal{F}) \leqslant G_\ell \, \mathbb{E}\Big[\sup_{(w,b) \in \mathbb{R}^{m(d+1)}} \sup_{s \in \{-1,1\}} \sup_{j \in \{1,\dots,m\}} D_\eta s \frac{1}{n} \sum_{i=1}^n \varepsilon_i \sigma(w_j^\top x_i + b_j)\Big].$$

Assuming the activation function σ is G_σ-Lipschitz continuous, we get, again using Proposition 3 from Lecture 3:

$$R_n(\mathcal{F}) \leqslant G_\ell D_\eta G_\sigma \, \mathbb{E}\Big[\sup_{(w,b) \in \mathbb{R}^{m(d+1)}} \sup_{j \in \{1,\dots,m\}} \sup_{s \in \{-1,1\}} s\Big\{w_j^\top \Big(\frac{1}{n} \sum_{i=1}^n \varepsilon_i x_i\Big) + b_j \Big(\frac{1}{n} \sum_{i=1}^n \varepsilon_i\Big)\Big\}\Big].$$

If we assume that we bound $\Theta(w_j, b_j) \leqslant D_{w,b}$, for each $j \in \{1, \dots, m\}$, we get, with the usual definition of the dual norm $\Theta^*(u, v) = \sup_{\Theta(w,b) \leqslant 1} \binom{w}{b}^\top \binom{u}{v}$:

$$R_n(\mathcal{F}) \leqslant G_\ell D_\eta G_\sigma D_{w,b} \, \mathbb{E}\Big[\Theta^*\Big(\frac{1}{n} \sum_{i=1}^n \varepsilon_i x_i, \frac{1}{n} \sum_{i=1}^n \varepsilon_i\Big)\Big].$$

Using $\Theta(w,b) = \max\{\|w\|_2, |b|/\sqrt{\mathbb{E}\|x\|_2^2}\}$, we get, using Jensen's inequality:

$$\begin{aligned}\mathbb{E}\Big[\Theta^*\Big(\frac{1}{n}\sum_{i=1}^n \varepsilon_i x_i, \frac{1}{n}\sum_{i=1}^n \varepsilon_i\Big)\Big] &= \mathbb{E}\Big[\Big\|\frac{1}{n}\sum_{i=1}^n \varepsilon_i x_i\Big\|_2\Big] + \sqrt{\mathbb{E}\|x\|_2^2}\,\mathbb{E}\Big[\Big|\frac{1}{n}\sum_{i=1}^n \varepsilon_i\Big|\Big] \\ &\leqslant \sqrt{\mathbb{E}\Big[\Big\|\frac{1}{n}\sum_{i=1}^n \varepsilon_i x_i\Big\|_2^2\Big]} + \sqrt{\mathbb{E}\|x\|_2^2}\,\sqrt{\mathbb{E}\Big[\Big|\frac{1}{n}\sum_{i=1}^n \varepsilon_i\Big|^2\Big]} \\ &= 2\sqrt{\frac{\mathbb{E}\|x\|_2^2}{n}}.\end{aligned}$$

Thus, we get the following proposition, with a bound proportional to $1/\sqrt{n}$ with no explicit dependence in the number of parameters.

Proposition 1 *Let $\mathcal{F}$ be the class of functions $(y,x) \mapsto \ell(y, f(x))$ where f is a neural network defined in Eq. (1), with the constraint that $\|\eta\|_1 \leqslant D_\eta$, $\max\{\|w_j\|_2, |b_j|/\sqrt{\mathbb{E}\|x\|_2^2}\} \leqslant D_{w,b}$ for all $j \in \{1,\dots,m\}$. If the loss function is G_ℓ-Lipschitz-continuous and the activation function σ is G_σ-Lipschitz-continuous, the Rademacher complexity is upperbounded as*

$$R_n(\mathcal{F}) \leqslant 2G_\ell G_\sigma D_{w,b} D_\eta \frac{\sqrt{\mathbb{E}\|x\|_2^2}}{\sqrt{n}}.$$

The proposition above allows to bound the estimation error for neural networks, as the maximal deviation between expected risk and empirical risk over all potential networks with bounded parameters, is bounded in expectation by twice the Rademacher complexity above.

For the ReLU activation function, where $G_\sigma = 1$, this will be combined with a study of the approximation properties in Section 3.

The number of parameters is irrelevant!!!!!!
What counts is the overall norm of the weights.

Check homogeneity.

When the norm of weights is not explicitly penalized or contained, we will see in Lecture 9 some recent results showing how optimization algorithms add an implicit regularization that leads to provable generalization in over-parameterized neural networks (that is, networks with many hidden units).

- **Exercise (♦)**: Provide the bound for $\Omega(w,b) = \max\{\|w\|_1, |b|/\sup\|x\|_\infty\}$, where $\sup\|x\|_\infty$ denotes the supremum of $\|x\|_\infty$ over all x in the support of its distribution.

3 Approximation properties of single-hidden layer neural networks

As seen above, the estimation error grows as $\frac{\|\eta\|_1}{\sqrt{n}}$, and is independent of the number m of neurons. Two important questions will be tackled in this section:

- What is the associated approximation error so that we can derive generalization bounds?
- What will be the number of neurons required to reach such a behavior?

For this, we need to understand the space of functions that neural networks span, and how they relate to smoothness properties of the function.

3.1 Link with kernel methods

- A one-hidden layer neural network corresponds to a linear classifier with feature vector of dimension m

$$\varphi(x)_j = \frac{1}{\sqrt{m}}\sigma(w_j^\top x + b_j)$$

  parameterized by all weights w_j, b_j, with kernel

$$\hat{k}(x, x') = \frac{1}{m}\sum_{j=1}^{m} \sigma(w_j^\top x + b_j)\sigma(w_j^\top x' + b_j).$$

  This corresponds to penalizing the output weights η_j, $j \in \{1, \dots, m\}$, by $m\sum_{j=1}^m \eta_j^2$, and keeping the input weights (w_j, b_j) fixed, for $j = 1, \dots, m$.

- With random independent and identically distributed weights $w_j \in \mathbb{R}^d$ and $b_j \in \mathbb{R}$, when m tends to infinity (a set-up often referred to as the "over-parameterized" set-up), by the law of large numbers, we get

$$\hat{k}(x, x') \to k(x, x') = \mathbb{E}\Big[\sigma(w^\top x + b)\sigma(w^\top x' + b)\Big].$$

  Therefore, infinite width networks where input weights are random and only output weights are learned are in fact kernel methods in disguise [1, 2].

- This kernel can be computed in closed form for simple activations and distributions of weights [3, 4], and thus the same regularization properties may be achieved with algorithms from Lecture 6 (which are based on convex optimization, and thus come with guarantees). Note that a common strategy for kernels defined as expectations is to use the a *random feature* approximation $\hat{k}(x, x')$, that is, here, use explicitly the neural network representation.

  The kernel approximation corresponds to input weights w_j, b_j sampled randomly and *held fixed*. Only the output weights η_j are optimized.

  Exercise: for $\binom{w}{b/R}$ uniform on the sphere, and for the ReLU activation, compute the associated kernel as a function of the cosine between the vectors $\binom{x}{R}$ and $\binom{x'}{R}$.

Integral representations of functions in the RKHS. When using a slightly different normalization and writing instead $f(x) = \frac{1}{m}\sum_{i=1}^m \tilde{\eta}_j \sigma(w_j^\top x + b_j)$, with $\tilde{\eta}_j = m\eta_j$, the penalty becomes $\frac{1}{m}\sum_{j=1}^m \tilde{\eta}_j^2$, expressions of the form

$$\frac{1}{m}\sum_{j=1}^m \tilde{\eta}_j F(w_j, b_j)$$

can be seen as the integral

$$\int_{\mathbb{R}^{d+1}} F(w,b)\eta(w,b)d\tau(w,b)$$

where $\tilde{\eta}_j = \eta(w_j, b_j)$, and $d\tau(w,b)$ is the probability measure on $\mathbb{R}^{d+1}$ generating the weights (w_j, b_j).

Thus, when m tends to infinify, we can represent the function f as

$$f(x) = \int_{\mathbb{R}^{d+1}} \eta(w,b)\sigma(w^\top x + b)d\tau(w,b),$$

where $\eta : \mathbb{R}^{d+1} \to \mathbb{R}$ is chosen as to minimize

$$\int_{\mathbb{R}^{d+1}} |\eta(w,b)|^2 d\tau(w,b).$$

We assume the support of $d\tau$ is compact (bounded and closed). Then the minimum achievable norm is exactly the squared RKHS norm of f, which we denote as $\gamma_2(f)^2$. We denote by $\mathcal{H}_2$ this RKHS, that is, the set of functions f such that $\gamma_2(f)$ is finite. See [4, Section 2.3] for more details.

Because Dirac measures are not squared integrable, the function $x \mapsto \sigma(w^\top x + b)$, that is, a single neuron, is typically not in the RKHS, which is typically composed of smooth functions. See examples below.

3.2 From L_2-norms to L_1-norms

Another function space can be defined, where

$$f(x) = \int_{\mathbb{R}^{d+1}} \eta(w,b)\sigma(w^\top x + b)d\tau(w,b),$$

where η is chosen as to minimize

$$\int_{\mathbb{R}^{d+1}} |\eta(w,b)| d\tau(w,b),$$

and $d\tau(w,b)$ is a probability measure on $\mathbb{R}^{d+1}$. The only difference with the squared RKHS norm above is that we consider the L_1-norm instead of the squared L_2-norm of η (with respect to the probability measure $d\tau$). The minimum achievable norm is a specific norm of f, which we denote as $\gamma_1(f)$.

Note that typically, the infimum over all η is not achieved, as, because we use L_1-norms and the measures $d\mu(w,b) = \eta(w,b)d\tau(w,b)$ can span all measures $d\mu(w,b)$ with finite total variation $\int_{\mathbb{R}^{d+1}} |d\mu(\eta,b)| =$

$\int_{\mathbb{R}^{d+1}} |\eta(w,b)| d\tau(w,b)$, we can reformulate the integral representation of f as

$$f(x) = \int_{\mathbb{R}^{d+1}} \sigma(w^\top x + b) d\mu(w,b),$$

with $d\mu$ a non-negative measure such that the *total variation* $\int_{\mathbb{R}^{d+1}} |d\mu(\eta,b)|$ is minimized. The norm γ_1 is often referred to as the variation norm (see [4] and references therein). We denote by $\mathcal{H}_1$ the set of functions f such that $\gamma_1(f)$ is finite. We have the following properties:

- Because of Jensen's inequality, we have $\gamma_1(f) \leqslant \gamma_2(f)$, and thus $\mathcal{H}_2 \subset \mathcal{H}_1$, that is the space $\mathcal{H}_1$ contains many more functions.
- ⚠ A single neuron is in $\mathcal{H}_1$ with γ_1-norm less than one, as the mass of a Dirac is equal to one.

In this lecture, to describe more precisely the spaces of functions $\mathcal{H}_1$ and $\mathcal{H}_2$, we will consider measures supported on the set $\big\{(w,b),\ \|w\|_2 = 1,\ |b| \leqslant R\big\}$ for R such that almost surely $\|x\|_2 \leqslant R$, and $\sigma(u) = \max\{u,0\} = (u)_+$ the ReLU activation function, which leads to a reasonably simple analysis.

- With the assumptions above, if $f(x) = \sum_{j=1}^m \eta_j (w_j^\top x + b_j)_+$, for $(w_j,b_j) \in \big\{(w,b),\ \|w\|_2 = 1,\ |b| \leqslant R\big\}$ for all $j \in \{1,\dots,m\}$, then $\gamma_1(f) \leqslant \|\eta\|_1$.

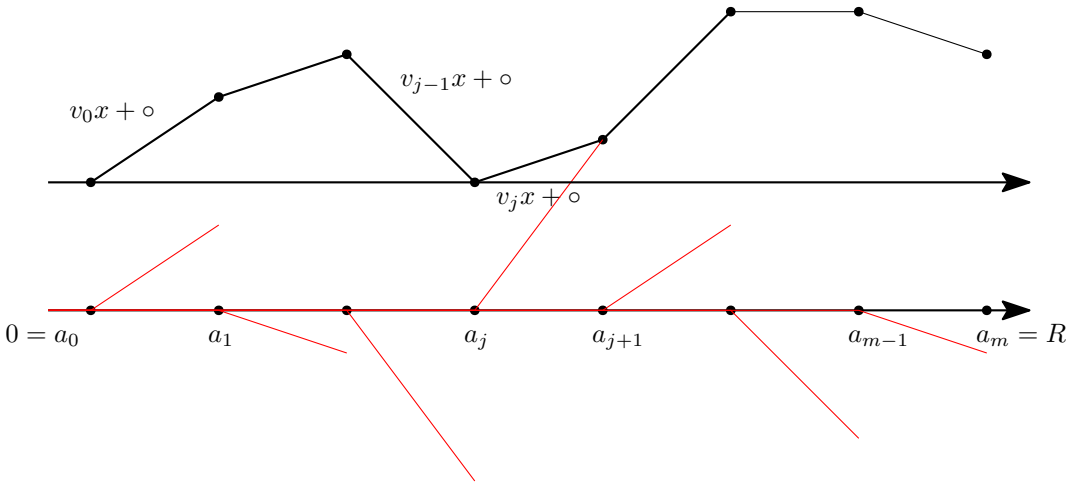
We will show in Section 3.5 how the norm γ_1 controls the number of neurons needed to approximate a function from $\mathcal{H}_1$, but we now study which functions have finite γ_1-norm and how functions outside of $\mathcal{H}_1$ can be approximated by functions in $\mathcal{H}_1$.

3.3 Variation norm in one dimension

The ReLU activation function is specific and leads to simple approximation properties in the interval $[-R,R]$ for functions $g : [-R,R] \to \mathbb{R}$. We start by piecewise affine functions, which, given the shape of the ReLU activation should be easy to approximate.

Piecewise affine functions. We first assume that $g(0) = 0$.

We consider a continuous piecewise affine function on $[-R,R]$ with knots at each $a_j = \frac{j}{m} R$ for $j \in [-m,m] \cap \mathbb{Z}$, so that on $[a_j, a_{j+1}]$, g is affine with slope v_j, for $j \in \{-m, m+1\}$.



if $g(0) = 0$, we can directly approximate on $[0, R]$, by first starting to fit the function on $[a_0, a_1] = [0, \frac{1}{m}]$, as $\hat{g}_0(x) = v_0(x - a_0)_+$. For $x > a_0$, this approximation has slope v_0. In order to be correct on $[a_1, a_2]$ (while not modifying the function on $[a_0, a_1]$, we consider $\hat{g}_1(x) = \hat{g}_0(x) + (v_1 - v_0)(x - a_1)_+$, which is now exact on $[a_0, a_2]$, we can pursue recursively by considering, for $j \in \{1, \dots, m-1\}$

$$\hat{g}_j(x) = \hat{g}_{j-1}(x) + (v_j - v_{j-1})(x - a_j)_+,$$

which is equal to $g(x)$ for $x \in [a_0, a_{j+1}]$. We can thus represent $g(x)$ on $[0, R]$ exactly with $\hat{g}_{m-1}(x)$, which itself is zero on $[-R, 0]$. We have by construction $\gamma_1(\hat{g}_{m-1}) \leqslant |v_0| + \sum_{j=1}^{m-1} |v_j - v_{j-1}|$. On the set $[-R, 0]$, we can obtain the same type of approximation with γ_1-norm less than $|v_{-1}| + \sum_{j=2}^{m} |v_{-j} - v_{-j+1}|$.

Therefore by summing these two approximation and by the triangular inequality, overall,

$$\gamma_1(g) \leqslant |v_0| + \sum_{j=1}^{m-1} |v_j - v_{j-1}| + |v_{-1}| + \sum_{j=2}^{m} |v_{-j} - v_{-j+1}|.$$

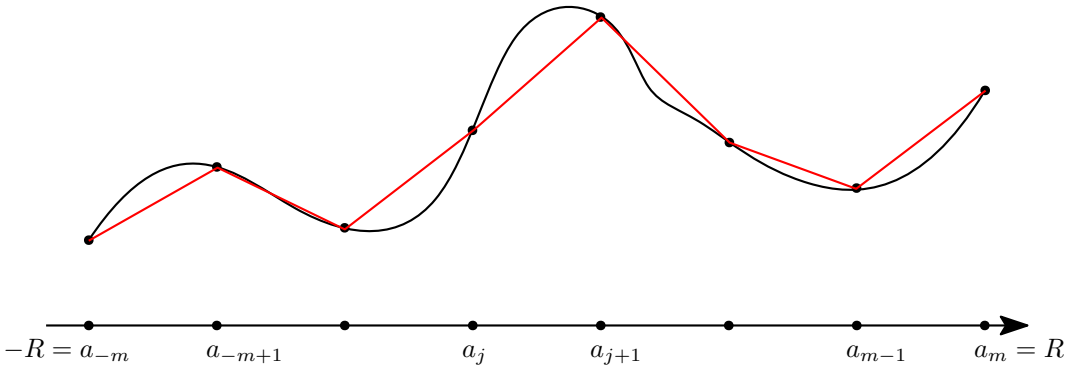
In order to consider functions g without the constraint $g(0) = 0$, we notice that the constant function has norm $\gamma(1) \leqslant \frac{1}{R}$, by using, for $x \in [-R, R]$, $2R = (x + R)_+ + (-x + R)_+$, and apply the result above to $g(x) - g(0)$ (which is zero at zero), thus leading to

$$\begin{aligned}
\gamma_1(g) &\leqslant \frac{|g(0)|}{R} + |v_0| + \sum_{j=1}^{m-1} |v_j - v_{j-1}| + |v_{-1}| + \sum_{j=2}^{m} |v_{-j} - v_{-j+1}| \\
&\leqslant \frac{|g(0)|}{R} + |v_0 + v_{-1}| + \sum_{j=-m+1}^{m-1} |v_j - v_{j-1}|, \text{ using } |v_0| + |v_{-1}| \leqslant |v_0 + v_{-1}| + |v_0 - v_{-1}|.
\end{aligned}$$

We can then use that $v_j = \frac{m}{R}(g(\frac{j+1}{m}R) - g(\frac{j}{m})R)$ to get:

$$\gamma_1(g) \leqslant \frac{|g(0)|}{R} + \frac{m}{R}|g(\frac{R}{m}) - g(-\frac{R}{m})| + \frac{m}{R} \sum_{j=-m+1}^{m-1} \left|g(\frac{j+1}{m}R) - 2g(\frac{j}{m}R) + g(\frac{j-1}{m}R)\right|.$$

Twice continuously differentiable functions. We consider a twice differentiable function g on $[-R,R]$, it is then the limit of its piecewise interpolation.



Thus, when m tends to infinity, $\frac{m}{R}|g(\frac{R}{m}) - g(-\frac{R}{m})|$ tends to $2|g'(0)|$ while $\big|g(\frac{j+1}{m}R) - 2g(\frac{j}{m}R) + |g(\frac{j-1}{m}R)\big|$ is asymptotically equivalent to

$$\big|g(\tfrac{j}{m}R) + \tfrac{R}{m}g'(\tfrac{j}{m}R) + \tfrac{1}{2}\tfrac{R^2}{m^2}g''(\tfrac{j}{m}R) - 2g(\tfrac{j}{m}R) + |g(\tfrac{j}{m}R) - \tfrac{R}{m}g'(\tfrac{j}{m}R) + \tfrac{1}{2}\tfrac{R^2}{m^2}g''(\tfrac{j}{m}R)\big| \sim |\tfrac{R^2}{m^2}g''(\tfrac{j}{m}R)|,$$

and thus we get:

$$\gamma_1(g) \leqslant \lim \sup_{m \to +\infty} \frac{|g(0)|}{R} + 2|g'(0)| + \frac{R}{m} \sum_{j=-m+1}^{m-1} |g''(\tfrac{j}{m}R)|,$$

which thus leads to using approximations of integral by Riemannian sums:

$$\gamma_1(g) \leqslant \frac{|g(0)|}{R} + 2|g'(0)| + \int_{-R}^{R} |g''(x)|dx.$$

In order to allow an extension for non-continuously differentiable functions at 0, we can further use that

$$\begin{aligned}
|g'(0)| &\leqslant |g'(y)| + \int_0^y |g''(x)|dx \leqslant |g'(y)| + \int_0^R |g''(x)|dx \text{ for any } y \in [0,R], \\
\text{leading to } |g'(0)| &\leqslant \frac{1}{R}\int_0^R |g'(x)|dx + \int_0^R |g''(x)|dx \text{ by integration}, \\
\text{and } |g'(0)| &\leqslant \frac{1}{2R}\int_{-R}^R |g'(x)|dx + \frac{1}{2}\int_{-R}^R |g''(x)|dx \text{ by symmetry}.
\end{aligned}$$

Overall, we get the expression

$$\gamma_1(g) \leqslant \tilde{\gamma}_1(g) = \frac{|g(0)|}{R} + \frac{1}{R}\int_{-R}^R |g'(x)|dx + 2\int_{-R}^R |g''(x)|dx, \tag{2}$$

which shows that if the number of neurons is allowed to grow then the ℓ_1-norm of the weights remain bounded by the quantity above to exactly represent the function g.

This can be extended to continuous functions which are only twice differentiable almost everywhere with integrable first and second-order derivatives; thus $\mathcal{H}_1 \subset \tilde{\mathcal{H}}_1$ (which corresponds to the norm $\tilde{\gamma}_1$ defined above). Since this space is dense in L_2 (see more general argument below in higher dimension), we obtain that neural networks are universal approximators.

RKHS norm γ_2 in one dimension (♦♦). In one dimension, with w uniform on the unit sphere, that is, $w \in \{-1, 1\}$, and with b uniform on $[-R, R]$, we have the following kernel

$$\hat{k}(x, x') = \frac{1}{4R} \int_{-R}^{R} \Big((x-b)_+ (x'-b)_+ + (-x-b)_+ (-x'-b)_+ \Big) db$$

Using the same reasoning as the end of Section 3.1, we can get an upper-bound on $\gamma_2(f)$ by decomposing f as

$$f(x) = \int_{-R}^{R} \eta_+(b)(x-b)_+ \frac{db}{4R} + \int_{-R}^{R} \eta_-(b)(-x-b)_+ \frac{db}{4R},$$

with $\gamma_2(f)^2 \leqslant \displaystyle\int_{-R}^{R} \eta_+(b)^2 \frac{db}{4R} + \int_{-R}^{R} \eta_-(b)^2 \frac{db}{4R}$.

By using Taylor expansion with integral remainder, we get, for any twice differentiable function f on $[-R, R]$, such that $f(0) = f'(0) = 0$,

$$f(x) = \int_{0}^{R} f''(b)(x-b)_+ db + \int_{0}^{R} f''(-b)(-x-b)_+ db.$$

Thus, for this function, $\gamma_2(f)^2 \leqslant 4R \displaystyle\int_{-R}^{R} f''(b)^2 db$. We can now use

$$\int_{-R}^{R} \frac{(x-b)_+ - (-x-b)_+}{2R} db = \int_{-R}^{R} \frac{(x-b)_+ - (b-x)_+}{2R} db = \int_{-R}^{R} \frac{x}{2R} db = x$$

to get that that $\gamma_2(x \mapsto x)^2 \leqslant 4$, and use

$$\int_{-R}^{R} \big[(x-b)_+ + (-x-b)_+ \big] db = \int_{-R}^{x} (x-b) db + \int_{-R}^{-x} (-x-b) db = \frac{(x-R)^2}{2} + \frac{(x+R)^2}{2} = x^2 + R^2,$$

to get that $\gamma_2(x \mapsto x^2 + R^2)^2 \leqslant 16R^2$

Thus by considering $\tilde{f}(x) = f(x) - f'(0)x - \frac{f(0)}{R^2}(x^2 + R^2)$, we have:

$$\begin{aligned}
\gamma_2(f) &\leqslant \sqrt{4R \int_{-R}^{R} \tilde{f}''(b)^2 db} + 2|f'(0)| + \frac{|f(0)|}{R} \\
&= \sqrt{4R \int_{-R}^{R} |f''(b) - 2f(0)/R^2|^2 db} + 2|f'(0)| + \frac{|f(0)|}{R} \\
&\leqslant \sqrt{4R \int_{-R}^{R} |f''(b)|^2 db} + \sqrt{4R \int_{-R}^{R} |2f(0)/R^2|^2 db} + 2|f'(0)| + \frac{|f(0)|}{R} \\
&= \sqrt{4R \int_{-R}^{R} |f''(b)|^2 db} + 4\sqrt{2} \frac{|f(0)|}{R} + 2|f'(0)| + \frac{|f(0)|}{R}
\end{aligned}$$

leading to the upper-bound

$$\gamma_2(g)^2 \leqslant \tilde{\gamma}_2(g)^2 = 36 \frac{f(0)^2}{R^2} + 16 f'(0)^2 + 16R \int_{-R}^{R} f''(x)^2 dx. \tag{3}$$

The main different with $\tilde{\gamma}_1$ is that the second-derivative is penalized by an L_2-norm and not by and L_1-norm, and that this L_2-norm can be infinite when the L_1-norm is finite, the classical example being for the hidden neuron functions $(x-b)_+$.

⚠ The RKHS is combining infinitely many hidden neuron functions $(x-b)_+$, none of them are inside the RKKHS,

⚠ This smoothness penalty does not allow the ReLU to be part of the RKHS. However, this is still an universal penalty.

3.4 Variation norm in arbitrary dimension

If we assume that f is continuous on the ball of center zero and radius R, then the Fourier transform $\hat{f}(\omega) = \int_{\mathbb{R}^d} f(x) e^{-i\omega^\top x} dx$ is defined everywhere, and we can write

$$f(x) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{f}(\omega) e^{i\omega^\top x} d\omega.$$

In order to compute an upper-bound on $\gamma_1(f)$, it suffices to upper-bound for each $\omega \in \mathbb{R}^d$, $\gamma_1(x \mapsto e^{i\omega^\top x})$, which is easy because we have the representation from Section 3.3 and Eq. (2): for $u \in [-R, R]$,

$$e^{iu\|\omega\|_2} = \int_{-R}^{R} \eta_+(b)(u-b)_+ db + \int_{-R}^{R} \eta_-(b)(-u-b)_+ db,$$

with $\displaystyle\int_{-R}^{R} |\eta_+(b)| db + \int_{-R}^{R} |\eta_-(b)| db \leqslant \frac{1}{R} + 2\|\omega\|_2 + 4R\|\omega\|_2^2$ (which is the norm defined in Eq. (2)). We can thefore decompose

$$e^{i\omega^\top x} = \int_{-R}^{R} \eta_+(b)(x^\top(\omega/\|\omega\|_2) - b)_+ db + \int_{-R}^{R} \eta_-(b)(x^\top(-\omega/\|\omega\|_2) - b)_+ db,$$

with weights being in the correct constraint set (unit norm for w's and $|b| \leqslant R$, leading to

$$\gamma_1(x \mapsto e^{i\omega^\top x}) \leqslant \tilde{\gamma}_1(x \mapsto e^{i\omega^\top x}) \leqslant \frac{1}{R} + 2\|\omega\|_2 + 4R\|\omega\|_2^2 = \frac{1}{R}(1 + 2R\|\omega\|_2)^2.$$

Thus, we obtain

$$\gamma_1(f) \leqslant \frac{1}{(2\pi)^d} \frac{1}{R} \int_{\mathbb{R}^d} |\hat{f}(\omega)|(1 + 2R^2\|\omega\|_2^2) d\omega.$$

Given a function f, $\displaystyle\int_{\mathbb{R}^d} |\hat{f}(\omega)| d\omega$ is a measure of smoothness of f, and so $\gamma_1(f)$ being finite imposes that f and all second-order derivatives of f have this form of smoothness. See [5] for more details and below for a relationship with Sobolev spaces.

Precise rates of approximation (⧫). In this section, we will relate the space $\mathcal{H}_1$ to Sobolev spaces, by considering $s > d/2$ (to make sure the integral below exists), and bounding using Cauchy-Schwarz

inequality:

$$\begin{aligned}\gamma_1(f) &\leqslant \frac{1}{(2\pi)^d}\frac{1}{R}\int_{\mathbb{R}^d}|\hat{f}(\omega)|(1+2R^2\|\omega\|_2^2)d\omega = \frac{1}{(2\pi)^d}\frac{1}{R}\int_{\mathbb{R}^d}|\hat{f}(\omega)|(1+2R^2\|\omega\|_2^2)^{1+s/2}\frac{d\omega}{(1+2R^2\|\omega\|_2^2)^{s/2}} \\ &\leqslant \frac{1}{(2\pi)^d}\frac{1}{R}\sqrt{\int_{\mathbb{R}^d}|\hat{f}(\omega)|^2(1+2R^2\|\omega\|_2^2)^{2+s}d\omega}\sqrt{\int_{\mathbb{R}^d}\frac{d\omega}{(1+2R^2\|\omega\|_2^2)^s}},\end{aligned}$$

which is a constant times $\sqrt{\int_{\mathbb{R}^d}|\hat{f}(\omega)|^2(1+2R^2\|\omega\|_2^2)^{2+s}d\omega}$, which is exactly the Sobolev norm from Lecture 6, with $s+2$ derivatives (which is an RKHS).

Thus, all approximation properties from Lecture 6 apply. See Lecture 6 for precise rates. Note however, that, *using this reasoning*, if we start from a Lipschitz-continuous function then to approximate it up to L_2-norm ε requires a γ_1-norm exploding as $\varepsilon^{-(s+1)} \geqslant \varepsilon^{-(d/2+1)}$ (as obtained at the end of Section 5.2 of Lecture 6).

Adaptivity to linear structures (⧫). if the target function f depends only a r-dimensional projection of the data, that is, f is of the form $f(x) = g(V^\top x)$, where $V \in \mathbb{R}^{d\times r}$ has all singular values less than 1, and $g : \mathbb{R}^r \to \mathbb{R}$, then if $\gamma_1(g)$ is finite, it can be written as

$$g(z) = \int_{\mathbb{R}^{r+1}}(w^\top z + b)_+ d\mu(w,b),$$

with $d\mu$ supported on $\{(w,b) \in \mathbb{R}^{r+1},\ \|w\|_2 = 1, |b| \leqslant R\}$, and $\gamma_1(g) = \displaystyle\int_{\mathbb{R}^{r+1}}|d\mu(w,b)|$. We then have:

$$f(x) = g(W^\top x) = \int_{\mathbb{R}^{r+1}}((Vw)^\top x + b)_+ d\mu(w,b) = \int_{\mathbb{R}^{r+1}}\Big(\Big(\frac{Vw}{\|Vw\|_2}\Big)^\top x + b\Big)_+ \|Vw\|_2 d\mu(w,b),$$

leading to $\gamma_1(f) \leqslant \displaystyle\int_{\mathbb{R}^{r+1}}\|Vw\|_2|d\mu(w,b)| \leqslant \int_{\mathbb{R}^{r+1}}|d\mu(w,b)| = \gamma_1(g)$. Thus the approximation properties of g translate to g, and thus we pay only the price of these r dimensions and not of all d variables, *without* the need to know V in advance. See [4] for more details.

Kernel methods do not have such adaptivity. In other words, using the ℓ_2-norm instead of the ℓ_1-norm on the output weights, leads to worse performance.

3.5 From the variation norm to a finite number of neurons

Given a measure $d\mu$ on $\mathbb{R}^d$, and a function $g : \mathbb{R}^d \to \mathbb{R}$ such that $\gamma_1(g)$ is finite, we would like to find a set of m neurons $(w_j, b_j) \in \mathcal{V} \subset \mathbb{R}^{d+1}$ (which is the compact support of all measures that we consider), such that the associated function defined through

$$f(x) = \sum_{j=1}^m \eta_j \sigma(w_j^\top x + b_j)$$

is close to g.

If the input weights are fixed, then the bound on $\gamma_1(g)$ translates to a bound $\|\eta\|_1 \leqslant \gamma_1(g)$. The set of such functions f is the convex hull of functions $s_j\gamma_1(g)\sigma(w_j^\top x + b_j)$, for $s_j \in \{-1, \}$. Thus, we are faced with the problem of approximating an elements of a convex hull as an explicit linear combination of extreme points, if possible with as few extreme points as possible.

In finite dimension, Carathéodory's theorem tells that the number of such extreme points can be taken to be equal to the dimension, to get an exact representation. In our case of infinite dimensions, we need an approximate version of Carathéodory's theorem. It turns out that we can create a the "fake" optimization problem of minimizing $\min_{g\in\mathcal{H}_1} \|f-g\|^2$ such that $\gamma_1(f) \leqslant \gamma_1(g)$, whose solution is $f = g$, with an algorithm that constructs an approximate solution from extreme points. This will be achieved by the Frank-Wolfe algorithm (a.k.a. conditional gradient algorithm). This algorithm is applicable more generally, for more details, see [6, 7].

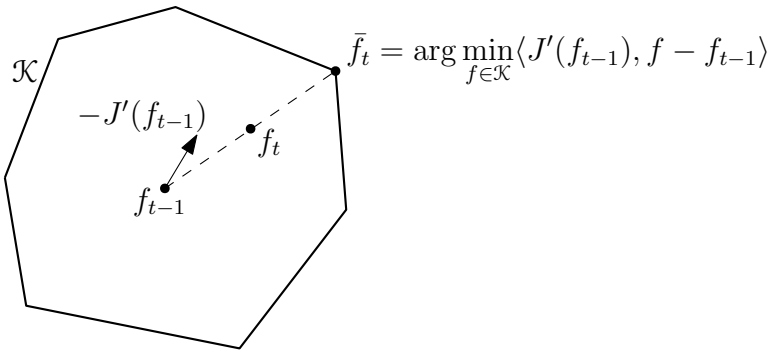
Frank-Wolfe algorithm. We thus make a detour by considering an algorithm defined in a Hilbert space $\mathcal{H}$, such that $\mathcal{K}$ is a bounded convex set, and J a convex smooth function from $\mathcal{H}$ to $\mathbb{R}$, that is such that there exists a gradient function $J' : \mathcal{H} \to \mathcal{H}$ such that for all elements f, g of $\mathcal{H}$:

$$J(g) + \langle J'(g), h - g\rangle_{\mathcal{H}} \leqslant J(f) \leqslant J(g) + \langle J'(g), h - g\rangle_{\mathcal{H}} + \frac{L}{2}\|h - g\|^2_{\mathcal{H}}.$$

The goal is to minimize J on the bounded convex set $\mathcal{K}$, without a particular algorithm that only requires to access the set $\mathcal{K}$ through a "linear minimization" oracle (i.e., through maximizing linear functions), as opposed to the projection oracle that we required in Lecture 4.

We consider the following recursive algorithm, started from a vector $f_0 \in \mathcal{K}$:

$$\bar{f}_t \in \arg\min_{f\in\mathcal{K}} \ \langle J'(f_{t-1}), f - f_{t-1}\rangle_{\mathcal{H}},$$
$$f_t = \frac{t-1}{t+1} f_{t-1} + \frac{2}{t+1}\bar{f}_t = f_{t-1} + \frac{2}{t+1}(\bar{f}_t - f_{t-1}).$$



Because $\bar{f}_t$ is obtained by mininimizing a linear function on a bounded convex set, we can restrict the minimizer $\bar{f}_t$ to be extreme points of $\mathcal{K}$, so that, f_t is the convex combination of t such extreme points $\bar{f}_1, \dots, \bar{f}_t$ (note that the first point f_0 disappears). We now show that

$$J(f_t) - \inf_{f\in\mathcal{K}} J(f) \leqslant \frac{2L}{t+1}\mathrm{diam}_{\mathcal{H}}(\mathcal{K})^2.$$

Proof of convergence rate (⧫). This is simply obtained by using smoothness:

$$\begin{aligned} J(f_t) &\leqslant J(f_{t-1}) + \langle J'(f_{t-1}), f_t - f_{t-1}\rangle_{\mathcal{H}} + \frac{L}{2}\|f_t - f_{t-1}\|_{\mathcal{H}}^2 \\ &= J(f_{t-1}) + \frac{2}{t+1}\langle J'(f_{t-1}), \bar{f}_t - f_{t-1}\rangle_{\mathcal{H}} + \frac{4}{(t+1)^2}\frac{L}{2}\|\bar{f}_t - f_{t-1}\|_{\mathcal{H}}^2 \\ &\leqslant J(f_{t-1}) + \frac{2}{t+1}\min_{f\in\mathcal{K}} \ \langle J'(f_{t-1}), f - f_{t-1}\rangle_{\mathcal{H}} + \frac{4}{(t+1)^2}\frac{L}{2}\mathrm{diam}_{\mathcal{H}}(\mathcal{K})^2. \end{aligned}$$

By convexity of J, we have for all $f \in \mathcal{K}$, $J(f) \geqslant J(f_{t-1}) + \langle J'(f_{t-1}), f - f_{t-1}\rangle_{\mathcal{H}}$, leading to $\inf_{f\in\mathcal{K}} J(f) \geqslant J(f_{t-1}) + \inf_{f\in\mathcal{K}}\langle J'(f_{t-1}), f - f_{t-1}\rangle_{\mathcal{H}}$. Thus, we get

$$\begin{aligned} J(f_t) - \inf_{f\in\mathcal{K}} J(f) &\leqslant \big[J(f_{t-1}) - \inf_{f\in\mathcal{K}} J(f)\big]\frac{t-1}{t+1} + \frac{4}{(t+1)^2}\frac{L}{2}\mathrm{diam}_{\mathcal{H}}(\mathcal{K})^2, \text{ leading to} \\ t(t+1)\big[J(f_t) - \inf_{f\in\mathcal{K}} J(f)\big] &\leqslant (t-1)t\big[J(f_{t-1}) - \inf_{f\in\mathcal{K}} J(f)\big] + 2L\mathrm{diam}_{\mathcal{H}}(\mathcal{K})^2 \\ &\leqslant 2Lt\mathrm{diam}_{\mathcal{H}}(\mathcal{K})^2 \text{ by using a telescoping sum,} \end{aligned}$$

and thus $J(f_t) - \inf_{f\in\mathcal{K}} J(f) \leqslant \frac{2L}{t+1}\mathrm{diam}_{\mathcal{H}}(\mathcal{K})^2$, as claimed earlier.

Application to approximate representations with a finite number of neurons. We can apply this to $\mathcal{H} = L_2(d\mu)$ and $J(f) = \|f - g\|_{L_2(d\mu)}^2$, leading to $L = 2$, with $\mathcal{K} = \{f \in L_2(d\mu),\ \gamma_1(f) \leqslant \gamma_1(g)\}$ for which the set of extreme points are exactly single neurons $s\sigma(w^\top \cdot + b)$ scaled by $\gamma_1(g)$, and with an extra sign $s \in \{-1, 1\}$.

We thus obtain after t steps a representation of f with t neurons for which

$$\|f - g\|_{L_2(d\mu)}^2 \leqslant \frac{4L\gamma_1(g)^2}{t+1}\sup_{(w,b)\in\mathcal{K}} \|\sigma(w^\top \cdot + b)\|_{L_2(d\mu)}^2.$$

Thus, it is sufficient to have t of order $O(\gamma_1(g)^2/\varepsilon^2)$ to achieve $\|f - g\|_{L_2(d\mu)} \leqslant \varepsilon$. Therefore the norm $\gamma_1(g)$ directly controls the approximability of the function g by a finite number of neurons, and tell us how many neurons should be used for a given target function.

4 Extensions

The fully-connected single-hidden layer neural networks is far from what is being used in practice. Indeed, state-of-the-art performance is typically achieved with the following extensions:

- Going deep with multiple layers: The most simple form of deep neural networks is a multilayer fully-connected neural network. Ignoring the constant terms for simplicity, it is of the form $f(x^{(0)}) = y^{(L)}$ with input $x^{(0)}$ and output $y^{(L)}$ given:

$$\begin{aligned} y^{(k)} &= (W^{(k)})^\top x^{(k-1)} \\ x^{(k)} &= \sigma(y^{(k)}), \end{aligned}$$

where $W^{(\ell)}$ is the matrix of weights for layer k.

For these models, obtaining simple and powerful theoretical results is still an active area of research. See, e.g., [8, 9].

- Convolutional neural networks: In order to be able to tackle data of large size and to improve preformances, it is important to leverage the prior knowledge about the structure of the typical data to process. For instance, for signal, images or videos, it is important to take into account the translation invariance (up to boundary issues) of the domain. This is done by constraining the linear operators involved in the linear part of neural networks to respect some form of translation invariance, and thus to use convolutions. See [10] for details.

Acknowledgements

These class notes have been adapted from the notes of many colleagues I have the pleasure to work with, in particular Lénaïc Chizat, Pierre Gaillard, Alessandro Rudi and Simon Lacoste-Julien. Special thanks to Lénaïc Chizat for his help for these notes.

References

[1] R. M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

[2] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2008.

[3] Youngmin Cho and Lawrence K. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems*, 2009.

[4] Francis Bach. Breaking the curse of dimensionality with convex neural networks. *The Journal of Machine Learning Research*, 18(1):629–681, 2017.

[5] Jason M. Klusowski and Andrew R. Barron. Approximation by combinations of relu and squared relu ridge functions with ℓ^1 and ℓ^0 controls. *IEEE Transactions on Information Theory*, 64(12):7649–7656, 2018.

[6] Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning*, 2013.

[7] Francis Bach. Duality between subgradient and conditional gradient methods. *SIAM Journal on Optimization*, 25(1):115–129, 2015.

[8] Jianfeng Lu, Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation for smooth functions. Technical Report 2001.03040, arXiv, 2020.

[9] Chao Ma, Stephan Wojtowytsch, and Lei Wu. Towards a mathematical understanding of neural network-based machine learning: what we know and what we don't. Technical Report 2009.10713, arXiv, 2020.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.