TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG
Die Ressourcenuniversität. Seit 1765.

Prof. Dr. Björn Sprungk
Faculty of Mathematics and Computer Science
Institute of Stochastics

Mathematics of machine learning

6. Neural networks

Winter term 2024/25

Chapter 5: Neural networks

Content

Chapter 5: Neural networks

What's it about?

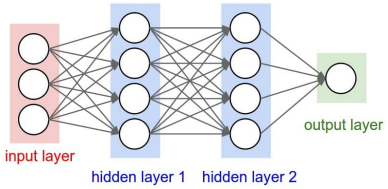
- To learn about the structure of (feedforward) neural networks
- Understand basic advantages of this structure, especially the advantage of the *depth* of neural networks
- Get to know classical results on the learnability and expressivity of neural networks.
- To discuss disadvantages and challenges of neural networks

6.1 Structure and Learnability

- Recall a FNN consists of L layers of n_k neurons processing and passing information from layer to layer
- Each neuron $v_{k,i}$, $k = 1, \dots, L$, $i = 1, \dots, n_k$ is a linear hypothesis

$$y_{k,i} = v_{k,i}(\mathbf{y}_{k-1}) := \phi \left(\sum_{j=1}^{n} w_j \; y_{k-1,j} + b \right)$$

with activation function $\phi \colon \mathbb{R} \to \mathbb{R}$.



- The output of the kth layer $V_k = \{v_{k,1}, \dots, v_{k,n_k}\}$ can then be written by

$$\mathbf{y}_k = \phi \circ f_{\mathbf{W}_k, \mathbf{b}_k}(\mathbf{y}_{k-1}), \qquad f_{\mathbf{W}_k, \mathbf{b}_k}(\mathbf{y}) := \mathbf{W}_k \; \mathbf{y} + \mathbf{b}_k$$

where ϕ is applied componentwise and we introduced the layerwise weight matrices $\mathbf{W}_k \in \mathbb{R}^{n_k \times n_{k-1}}$ and bias vectors $\mathbf{b}_k \in \mathbb{R}^{n_k}$

Formal description of artifical neural networks

A feedforward neural network is a hypothesis $h\colon \mathcal{X} \to \mathcal{Y}$ of the form

$$h(\mathbf{x}) = \rho \circ f_{\mathbf{W}_L,\mathbf{b}_L} \circ \phi \circ f_{\mathbf{W}_{L-1},\mathbf{b}_{L-1}} \circ \phi \circ \cdots \circ \phi \circ f_{\mathbf{W}_1,\mathbf{b}_1}(\mathbf{x}),$$

where

- $\phi\colon \mathbb{R} \to \mathbb{R}$ as well as $\rho\colon \mathbb{R} \to \mathcal{Y}$ are chosen activation functions whose applications are to be understood componentwise,
- given layerwise weight matrices $\mathbf{W}_k \in \mathbb{R}^{n_k \times n_{k-1}}$ and bias vectors $\mathbf{b}_k \in \mathbb{R}^{n_k}$

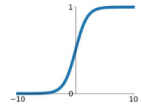
$$f_{\mathbf{W}_k,\mathbf{b}_k}(\mathbf{y}) := \mathbf{W}_k\ \mathbf{y} + \mathbf{b}_k$$

- with $n_k \in \mathbb{N}$ denotes the size of the k-th layer V_k.

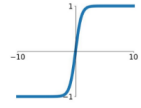
- An FNN with $L = 1$ is nothing but a linear hypothesis.
- Neural networks are thus also called multilayer perceptrons.
- A FNN with $L = 2$ is called a shallow neural network and with $L > 2$ a deep neural network.

- **Typical activation functions:**
  - **Sign:** $\phi(t) = \operatorname{sgn}(t)$ or $\phi(t) = \mathbb{1}_{[0,\infty)}(t)$
  - **Sigmoid:** $\phi(t) = \operatorname{sig}(t) = \frac{1}{1+\exp(-t)} \in [0, 1]$
  - **Identity:** $\phi(t) = \operatorname{id}(t) = t$
  - **ReLU function:** $\sigma(t) = \max\{0, t\}$.

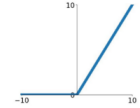
Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$



tanh
$\tanh(x)$



ReLU
$\max(0, x)$



Characteristics of FNN

$$h(\mathbf{x}) = \rho \circ f_{\mathbf{W}_L,\mathbf{b}_L} \circ \sigma \circ f_{\mathbf{W}_{L-1},\mathbf{b}_{L-1}} \circ \sigma \circ \cdots \circ \sigma \circ f_{\mathbf{W}_1,\mathbf{b}_1}(\mathbf{x}),$$

output

input layer

- **Depth:** L
- **Width:** $B := \max_{k=0,\ldots,L} n_k$
- **Size:** $n := n_0 + n_1 + \ldots + n_L$
- **Architecture:** (V, E) with $V = (V_0, \ldots, V_L)$ and

$$E \subseteq \{(v_{k,i}, v_{k+1,j}) \colon v_{k,i} \in V_k \text{ and } v_{k+1,j} \in V_{k+1}\}$$

- **Number of parameters:**

$$p_{V,E} := |E| + |V_1| + \ldots + |V_L| \leq L\,(B^2 + B)$$

Definition 6.1: Class of FNN

For a given architecture (V, E, ϕ, ρ) we denote by $\mathcal{H}_{V,E,\phi,\rho} \subseteq \mathcal{Y}^{\mathcal{X}}$ the set of all FNN $h\colon \mathcal{X} \to \mathcal{Y}$, $\mathcal{X} = \mathbb{R}^d$, $d = n_0 = |V_0|$, with just this architecture. If $\phi = \rho$, we write only $\mathcal{H}_{V,E,\phi}$.

The class $\mathcal{H}_{V,E,\phi,\rho}$ has the special structure (here for $\phi = \rho$)

$$\mathcal{H} = \mathcal{H}_L \circ \cdots \circ \mathcal{H}_1, \qquad \mathcal{H}_k = \mathcal{L}_{\phi,d_{k,1}} \times \cdots \times \mathcal{L}_{\phi,d_{k,n_k}}$$

where $d_{k,j} \leq n_{k-1}$ denotes the number of incoming edges at node $v_{k,j}$

Example 6.2: Shallow FNN ($L = 2$)

Shallow, fully connected FNN with $L = 2$ on $\mathbb{R}^d$ are of the form

$$h(\mathbf{x}) = \rho\left(\beta + \sum_{i=1}^{n} \alpha_i \sigma(\mathbf{w}_i^\top \mathbf{x} + b_i)\right), \qquad \alpha_i, \beta, b_i \in \mathbb{R},\ \mathbf{w}_i \in \mathbb{R}^d,$$

where $|V_1| = n$ and $E = (V_0 \times V_1) \cup (V_1 \times V_2)$.

Learnability

FNN with suitable activation function are **PAC-learnable**, but the learnability decreases with the size of the network:

- we do not use Sgn because it isn't differentiable
- we can use sub-gradient for ReLU to use it.

	$\mathrm{VCD}(\mathcal{H}_{V,E,\phi,\mathrm{sgn}})$	
ϕ	Lower bound	Upper bound
sign	$\Omega(p \ln p)$	$\mathcal{O}(p \ln p)$
sigmoid	$\Omega(\lvert E\rvert^2)$	$\mathcal{O}(p^2)$
ReLU	$\Omega(L\, p \ln(p/L))$	$\mathcal{O}(L\, p \ln p)$

→ faster than linear with params (p)

number of layers (linear)

Similar bounds can be obtained in case of regression for the pseudo or fat-shattering dimension.

Why use Sgn? as ϕ | Sgn is not continuous

6.2 Expressivity

- We will now study the expressive power of neural networks: How small is ε_{app} here.
  1. I.e., which functions/hypotheses can be represented accurately by FNN?
  2. Or: Which functions/hypotheses can be (arbitrarily) well approximated by FNN?
  3. And how large (i.e., how wide or how deep) must a FNN be to achieve this?
- This then allows statements about the approximation error ε_{app} of the FNN classes $\mathcal{H}_{V,E,\phi,\rho}$.
- We will first consider binary FNN and then approximations of real-valued hypotheses by FNN.

Representation of binary hypotheses by FNN

- We focus on shallow FNN at the beginning:

$$h(\mathbf{x}) = \rho\left(\beta + \sum_{i=1}^{n} \alpha_i \, \phi\left(\mathbf{w}_i \cdot \mathbf{x} + b_i\right)\right) \in \mathcal{Y}. \tag{FFNN}$$

- For these there are many results, i.e., for the binary case $\phi = \rho = \operatorname{sgn}$:

Theorem 6.3:

Let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^d$, $|\mathcal{X}| = N$ be finite. Then, any binary hypothesis $h\colon \mathcal{X} \to \{-1, 1\}$ can be represented by a shallow FNN with $\phi = \rho = \operatorname{sgn}$ and $n_1 \leq N$ neurons per hidden layer. In particular,

$$\mathcal{H}_{V,E,\operatorname{sgn}} = \{-1, +1\}^{\mathcal{X}},$$

for the architecture of fully connected shallow FNN with $n = d + N + 1$, i.e., $V = (V_0, V_1, V_2)$, $E = V_0 \times V_1 \cup V_1 \times V_2$ and $|V_0| = d$, $|V_1| = N$, $|V_2| = 1$.

proof Th 6.3:

Let $h \in (\pm 1)^2$ arbitrary & set $X_+ = \{x \in X, h(x) = +1\}$, we assume:

$X_- = \{x \in X; h(x) = -1\}$ $|X_+| \leq |X_-|$

since $x \not\subset \infty$, $\exists \forall x_i$ a hyperplane H_{w_i, b_i} in format: $|x_+| \leq \frac{N}{2}$

$H_{w_i, b_i} = \{x \in \mathbb{R}^d; w_i^T x + b = 0\}$ such that:

$H_{w_i, b_i} \cap X = \{x_i\}$ in particular $\exists H_{w_i, b_i}$ parallel to H_{w_i, b_i}

such that in between these H_{w_i, b_i}, we only points of X is x_i. I.e

$\exists \epsilon_i > 0$ such that $h_i(x_j) = \text{sgn}(w_i^T x_j + b_i + \epsilon_i) + \text{sgn}(-w_i^T x_j - b_i - \epsilon$

$= \begin{cases} 0, & x_j \neq x_i \\ 2, & x_j = x_i \end{cases} \quad \forall x_j \in X.$

We can now combine these h_i for each $x_i \in X_+$:

$h(x) = \mathrm{Sgn}\left(-1 + \sum_{x_i \in X_+} h_{x_i}(x)\right)$. for this $h(x)$, we have for train points $x_j \in X$: $h(x_j) = \begin{cases} \mathrm{Sgn}(+1), & x_j \in X_+ \\ \mathrm{Sgn}(-1), & x_j \in X_- \end{cases}$

By construction, h is a shallow NN with $\rho = \phi = \mathrm{Sgn}$ $\Sigma n_j = 2|X_+|$

$\leq 2 \cdot \frac{N}{2} = N$ ✗ ($\because n = 2|X_+| \leq 2 \cdot \frac{N}{2}$)

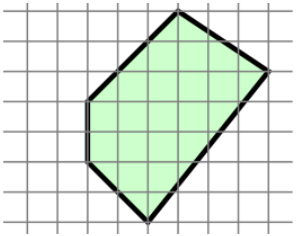
Geometric interpretation

- Linear hypotheses or FNN with $L = 1$ and $\rho = \operatorname{sgn}$, i.e. $h(\mathbf{x}) = \operatorname{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$, $\mathbf{x} \in \mathbb{R}^d$, divide the feature space into two half-spaces and can be viewed as indicator functions of half-spaces.

- For shallow FNN with $L = 2$ and $\phi = \rho = \operatorname{sgn}$,

$$h(\mathbf{x}) = \operatorname{sgn}\Big(\beta + \sum_{i=1}^{n} \alpha_i \operatorname{sgn}(\mathbf{w}_i \cdot \mathbf{x} + b_i)\Big), \qquad \mathbf{x} \in \mathbb{R}^d,$$

each neuron $v_{1,i}$ in the hidden layer V_1 implements a half-space classifier and the output neuron $v_{2,1}$ can thus be seen as indicator function of a convex polytope $A \subseteq \mathbb{R}^d$:

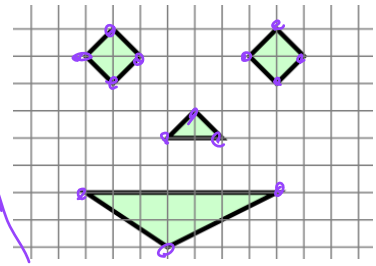


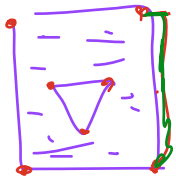
Intersection of $n_1 = 5$ half-spaces whose indicator function can be represented by an FNN with $L = 2$, $n = 2 + n_1 + 1$ neurons and $\phi = \rho = \operatorname{sgn}$.

Source: "Understanding Machine Learning" (2014)

- For deep FNN with $L = 3$ and $\phi = \rho = \operatorname{sgn}$ each neuron $v_{2,i}$ in the 2nd hidden layer V_2 represents now the indicator function of a convex polytope $A_i \subset \mathbb{R}^d$ and, hence, the output neuron $v_{3,1}$ can be seen as the indicator function of the superposition of these $n_1 = |V_2|$ convex polytopes

$$A = \bigcup_{i=1}^{n_2} A_i \subseteq \mathbb{R}^d.$$





Union of $n_2 = 4$ convex polygons whose indicator function can be represented by an FNN with $L = 3$, $n = 2 + 14 + n_2 + 1$ neurons and $\phi = \rho = \operatorname{sgn}$.

Source: "Understanding Machine Learning" (2014)

- Also nonconvex polytopes or regions with "polygonal holes" can be represented by deep binary FNN quite simply: $v_{2,1}$ represents the convex polytope A_1 and $v_{2,2}$ the convex polytope $A_2 \subset A_1$. If the output neuron $v_{3,1}$ combines the signals of both with $w_{3,1} = 1$, $w_{3,2} = -2$ and $\mathbf{b}_3 = 0$ this yields the indicator function of $A_1 \setminus A_2$.

Approximation of real-valued hypotheses

Theorem 6.4: Universal approximation theorem, (Cybenko, 1989)

Let $\phi\colon \mathbb{R} \to \mathbb{R}$ be a continous and sigmoidal activation function, i.e.,

$$\lim_{t\to-\infty} \phi(t) = 0, \qquad \lim_{t\to\infty} \phi(t) = 1.$$

Then, for any continuous hypothesis $g\colon \mathcal{X} \to \mathbb{R}$, $\mathcal{X} \subset \mathbb{R}^d$ being compact, and any $\epsilon > 0$, there exists a shallow FNN of finite width $n = n(\epsilon) \in \mathbb{N}$

$$h(x) = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{w}_i^\top \mathbf{x} + b_i)$$

such that

$$\sup_{\mathbf{x}\in\mathcal{X}} |g(\mathbf{x}) - h(\mathbf{x})| \leq \epsilon.$$

$\Rightarrow$ Already shallow FNN of arbitrary width can approximate any $g \in C(\mathcal{X})$ up to arbitrary precision!

The Universal Apprimxation theorm or Cybenko theorm states that if we do have a NN specifically a Shallow FNN which has L=2, then it learn any kind of pattern or function mostly continous if we could give it enough neurons and a finite width.

here we do have a continous input activation function which is a sigmoid activation function. in the Shallow FFN formula we tend to set our learning NN to learn a continous other actvation function with respect to a threshold.

This threshold mean that the Maximum differences of the Shallow FNN and the learned continous function should not very low and should not be greater that the threshold. this means that the leanred continous function or pattern should be as close as possible to the main Shallow FNN.
We could also tell that bound of differneces, that can be the Approximation error too for any epsilon > 0

The Kigner and Lyons theorm says that even if we could have a deep narro NN, unlike the Cybenko Theorm, we could have a smooth term of the NN too that can approximate the ocntinous g function. again, the same as above it says that learned continous patter should be as close as possible to main NN that have learned. even with having the small fixed with : d+3 neurons per Layer!

so Going deeper is better than going wider, because it gives: Faster improvement in accuracy - Lower VC dimension not to tend to infinity or even Estimation Error. according to Petersen theorm.
This is a lower bound on how close the neural network's output h(x) can get to the target g(x).
The error shrinks (gets better) as:

Depth L increases → very fast (exponentially)

Width B increases → much slower (polynomially)

$$\sup_{x \in [0,1]^d} |g(x) - f(x)| \geq C_g (2B+2)^{-2L-2}$$

Universal approximation by arbitrary depth

Theorem 6.5: (Kidger & Lyons, 2020)

Let $\phi\colon \mathbb{R} \to \mathbb{R}$ be continuous, nonaffine and continuously differentiable at at least one point $t_0 \in \mathbb{R}$ with $\phi'(t_0) \neq 0$, e.g., $\phi = \mathrm{sig}$ or $\phi(t) = \max\{0, t\}$).

Furthermore, let $\mathcal{H}_{B,\phi}(\mathcal{X})$ be the set of all FNN on compact $\mathcal{X} \subset \mathbb{R}^d$ with $\rho = \mathrm{id}$, arbitrary depth $L \in \mathbb{N}$, but maximum width $B := \max_{k=0,\dots,L} |V_k|$.

Then, for every continuous function $g\colon \mathcal{X} \to \mathbb{R}$ and every $\epsilon > 0$ there is an FNN $h \in \mathcal{H}_{d+3,\phi}(\mathcal{X})$ with depth $L = L(\epsilon)$ such that

$$\sup_{\mathbf{x}\in\mathcal{X}} |g(\mathbf{x}) - h(\mathbf{x})| \leq \epsilon.$$

$\Rightarrow$ Also narrow neural networks of fixed width $B = d + 3$ but arbitrary depth are universal approximators in $C(\mathcal{X})$

$\Rightarrow$ What is now better: large depth or large width?

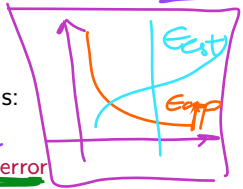
The advantage of depth

Theorem 6.6: (Petersen, 2020)

Let $g \in C^2([0,1]^d)$ be nonaffine. For any FNN h with $\phi(t) = \max\{0, t\}$ and $\rho = \mathrm{id}$ having depth L and width B we have with a $c_f > 0$

$$\sup_{\mathbf{x}\in[0,1]^d} |g(\mathbf{x}) - h(\mathbf{x})| \geq c_g (2B+2)^{-2L-2}.$$

- The lower bound decays exponentially in L but only polynomially in B.
- And the parameters of a FNN can be bounded by $p \leq L(B^2 + B)$. Thus:
  1. Going deep rather than going wide yields faster decaying approximation error
  2. Going deep rather than going wide yields slower growing VC dimension (or estimation error).
- **Important for analysis:** ReLU-FNN are piecewise linear hypotheses!



The power of concatenation

- The true power of FNN lies in their concatenating structure of linear hypotheses $h_k = \phi \circ f_{\mathbf{W}_k, \mathbf{b}_k}$:

$$h = h_L \circ \cdots \circ h_1(\mathbf{x}).$$

- To understand that, let us consider a "shallow" polynomial p of degree 25

$$p(x) = \sum_{i=0}^{25} w_i x^i.$$

where we need $n = 25$ parameters to represent that.

- Consider now a "deep" polynomial of degree 25 given by the composition of two polynomials p_1, p_2 of degree 5 with in total only 12 parameters:

$$p(x) = p_2 \circ p_1(x) = p_2(p_1(x)) = \sum_{i=0}^{5} w_i \left(\sum_{j=0}^{5} v_j x^j \right)^i.$$

- **Vice versa**: Consider the composition of $L = 5$ polynomials p_i each of degree q:

$$p(x) = p_5 \circ p_4 \circ p_3 \circ p_2 \circ p_1(x)$$

  This is a (deep) polynomial of degree q^5 represented by $5 \cdot (q+1)$ parameters.

- By a shallow representation we would require for that $q^5 + 1$ parameters.

- Given $q = 5$ that is the difference between 30 to 3126 parameters!

- Of course, not each polynomial of degree q^5 allows for a deep representation with $5q + 5$ parameters.

- But the set of such deep polynomials might be a dense set (as are FNN).

The shape of neural networks spaces

The set $\mathcal{H}_{V,E,\phi,\mathrm{id}}$ with suitable ϕ is star-shaped and neither convex nor even closed! See here for more details.