Announcement

Seminar „Neural Network Approximation“

Course type: Seminar, 2 semester hours per week, 5 ECTS, Summer term 2024

Contact person: PD Dr. Cornelia Schneider *schneider@math.fau.de*

Content: The last decades observed a tremendous success of artificial neural networks in many machine learning tasks, including computer vision, speech recognition, natural language processing, or games solutions to name just a few. Despite their wide use, many of their properties are not fully understood and many aspects of their great practical performance lack a rigorous explanation. In this seminar we look at how neural networks operate from the mathematical perspective, having in mind that the success of the neural networks methods should not be determined by trial-and-error or luck, but by a clear mathematical analysis. The topics therefore cover a mixture of good old classical mathematics and modern concepts of deep learning. In particular, we will be concerned with the approximation and expressive powers of neural networks.

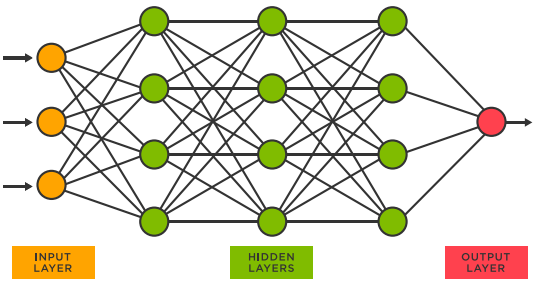
A neural network is a collection of computing units, which are connected together, called neurons, each producing a real-valued outcome, called activation. Input neurons get activated from the sensors that perceive the environment, while the other neurons get activated from the previous neuron activations. This structure allows neurons to send messages among themselves and consequently, to straighten those connections that lead to success in solving a problem and diminishing those which are leading to failure.
The simplest neural network is called perceptron and consists of a single hidden layer having 1 hidden neuron with activation function σ. For example, if σ is the Heaviside function

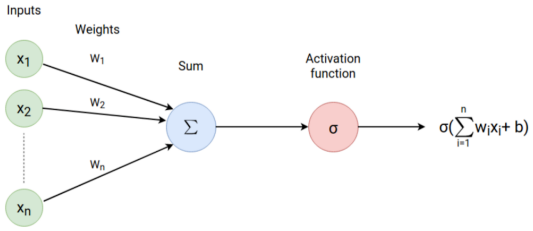
$$\sigma(x) = \begin{cases} 0, & x < 0, \\ 1, & x \geq 0, \end{cases}$$

then the output of the perceptron is a treshold function

$$y = \sigma\left(w \cdot x + b\right) = \begin{cases} 0, & w \cdot x + b < 0, \\ 1, & w \cdot x + b \geq 0. \end{cases}$$

Therefore, using several neurons (instead of 1) in a single hidden layer and choosing the weights and biases appropriately, the output of a neural network can be a simple function.





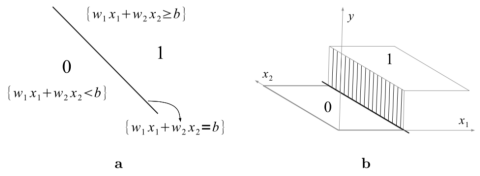


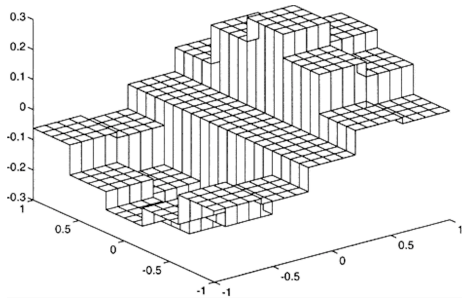
Figure . **a**. *The separation of the plane into half-planes.* **b**. *The graph of the activation function for a perceptron with two inputs, x_1 and x_2.*

The universal approximation theorem states that any continuous function $f : [0,1]^n \to [0,1]$ can be approximated arbitrarily well by a neural network with at least 1 hidden layer using a finite number of weights: More precisely, for adequate activation functions the finite sums of the form

$$G(x) = \sum_{j=1}^{N} a_j \sigma(w_j^T \cdot x + b_j), \qquad w_j \in \mathbb{R}^n,\ a_j, b_j \in \mathbb{R}$$

are dense in $C([0,1]^n)$, i.e., given $f \in C([0,1]^n)$ and $\varepsilon > 0$ there is a sum $G(x)$ of the above form for which

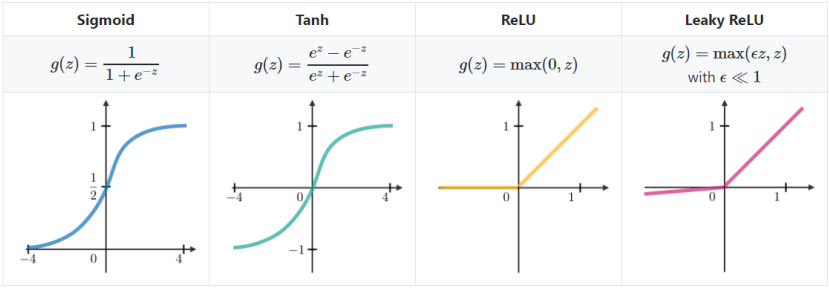
$$|G(x) - f(x)| < \varepsilon \qquad \text{for all } x \in [0,1]^n.$$



The topics we cover within this seminar vary from simple to complex and include:

(1) **Introduction to neural networks**

- *First examples*
- *Activation functions* of sigmoid type (e.g. logistic, hyperbolic tangent, arctangent), hockey-stick type (e.g. ReLU) or bumped type (e.g. Gaussian, double exponential)



- *Cost functions* (error function, loss function): During the learning process a neural network has to adjust parameters such that a certain cost function gets minimized.
- *Neurons*: The simplest building block of a neural network where the computation is done (e.g. perceptron, sigmoid neuron)
- *Architecture of a neural network* as well as the *backpropagation method* used for training

(2) **Approximation theorems and their applications to neural networks**

- Dini's Theorem, Arzela-Ascoli's Theorem, Stone-Weierstrass' Theorem, Wiener's Tauberian Theorems, Contraction Principle

(3) **Neural networks as universal approximators**

- What kind of functions can be learned by one-hidden layer neural networks?

↷ neural network approximation of continuous functions $f \in C([0,1]^n)$, square integrable functions $f \in L^2([0,1]^n)$, integrable functions $f \in L^1([0,1]^n)$, or measurable functions $f \in \mathcal{M}([0,1]^n)$

(4) **Advanced research-related topics (for master students)**

- based on the recent papers [3] and [4]

The topic of the seminar can be the starting point for a bachelor or master thesis.

Prior knowledge: 'Mathematics for Data Science I+II' or the basic 'Analysis' and 'Linear Algebra' modules. For the more advanced topics 'Functional Analysis' might be helpful.

Course achievement/ Academic assessment: Presentation (90min) and handout (approx. 5 pages)

Literature

[1] Ovidiu Calin. *Deep Learning Architectures. A Mathematical Approach*. Springer Series in the Data Sciences, 2020.

[2] I. Daubechies, R. DeVore, S. Foucart, B. Hanin, and G. Petrova. Nonlinear Approximation and (Deep) ReLU Networks. *Constr. Approx*, 55:127–172, 2022.

[3] P. Grohs and F. Voigtländer. *Sobolev-type embeddings for neural network approximation spaces*. *Constr. Approx.*, 57:579–599, 2023, arXiv:2110.15304.

[4] C. Schneider and J. Vybíral. A multivariate Riesz basis of ReLU neural networks. *To appear in ACHA*, 2023, arXiv:2303.00076.