Mathematical Introduction to Deep Learning: Methods, Implementations, and Theory

Arnulf Jentzen
Benno Kuckuck
Philippe von Wurstemberger

Arnulf Jentzen

School of Data Science and Shenzhen Research Institute of Big Data
The Chinese University of Hong Kong, Shenzhen (CUHK-Shenzhen)
Shenzhen, China
email: ajentzen@cuhk.edu.cn

Applied Mathematics: Institute for Analysis and Numerics
University of Münster
Münster, Germany
email: ajentzen@uni-muenster.de

Benno Kuckuck

School of Data Science and Shenzhen Research Institute of Big Data
The Chinese University of Hong Kong Shenzhen (CUHK-Shenzhen)
Shenzhen, China
email: bkuckuck@cuhk.edu.cn

Applied Mathematics: Institute for Analysis and Numerics
University of Münster
Münster, Germany
email: bkuckuck@uni-muenster.de

Philippe von Wurstemberger

School of Data Science
The Chinese University of Hong Kong, Shenzhen (CUHK-Shenzhen)
Shenzhen, China
email: philippevw@cuhk.edu.cn

Risklab, Department of Mathematics
ETH Zurich
Zurich, Switzerland
email: philippe.vonwurstemberger@math.ethz.ch

Keywords: deep learning, artificial neural network, stochastic gradient descent, optimization
Mathematics Subject Classification (2020): 68T07

Version of November 1, 2023

All Python source codes in this book can be downloaded from https://github.com/introdeeplearning/book or from the arXiv page of this book (by clicking on “Other formats” and then “Download source”).

Preface

This book aims to provide an introduction to the topic of deep learning algorithms. Very roughly speaking, when we speak of a *deep learning algorithm* we think of a computational scheme which aims to approximate certain relations, functions, or quantities by means of so-called deep *artificial neural networks* (ANNs) and the iterated use of some kind of data. ANNs, in turn, can be thought of as classes of functions that consist of multiple compositions of certain nonlinear functions, which are referred to as *activation functions*, and certain affine functions. Loosely speaking, the depth of such ANNs corresponds to the number of involved iterated compositions in the ANN and one starts to speak of *deep* ANNs when the number of involved compositions of nonlinear and affine functions is larger than two.

We hope that this book will be useful for students and scientists who do not yet have any background in deep learning at all and would like to gain a solid foundation as well as for practitioners who would like to obtain a firmer mathematical understanding of the objects and methods considered in deep learning.

After a brief introduction, this book is divided into six parts (see Parts I, II, III, IV, V, and VI). In Part I we introduce in Chapter 1 different types of ANNs including *fully-connected feedforward ANNs*, *convolutional ANNs* (CNNs), *recurrent ANNs* (RNNs), and *residual ANNs* (ResNets) in all mathematical details and in Chapter 2 we present a certain calculus for fully-connected feedforward ANNs.

In Part II we present several mathematical results that analyze how well ANNs can approximate given functions. To make this part more accessible, we first restrict ourselves in Chapter 3 to one-dimensional functions from the reals to the reals and, thereafter, we study ANN approximation results for multivariate functions in Chapter 4.

A key aspect of deep learning algorithms is usually to model or reformulate the problem under consideration as a suitable optimization problem involving deep ANNs. It is precisely the subject of Part III to study such and related optimization problems and the corresponding optimization algorithms to approximately solve such problems in detail. In particular, in the context of deep learning methods such optimization problems – typically given in the form of a minimization problem – are usually solved by means of appropriate *gradient based* optimization methods. Roughly speaking, we think of a gradient based optimization method as a computational scheme which aims to solve the considered optimization problem by performing successive steps based on the direction of the (negative) gradient of the function which one wants to optimize. Deterministic variants of such gradient based optimization methods such as the *gradient descent* (GD) optimization method are reviewed and studied in Chapter 6 and stochastic variants of such gradient based optimization methods such as the *stochastic gradient descent* (SGD) optimization method are reviewed and studied in Chapter 7. GD-type and SGD-type optimization methods can, roughly speaking, be viewed as time-discrete approximations of solutions of suitable *gradient flow* (GF) *ordinary differential equations* (ODEs). To develop intuitions for GD-type and SGD-type optimization methods and for some of the tools which we employ to analyze such methods, we study in Chapter 5 such GF ODEs. In particular, we show in Chapter 5 how such GF ODEs can be used to approximately solve appropriate optimization problems. Implementations of the gradient based methods discussed in Chapters 6 and 7 require efficient computations of gradients. The most popular and in some sense most natural method to explicitly compute such gradients in the case of the training of ANNs is the *backpropagation* method, which we derive and present in detail in Chapter 8. The mathematical analyses for gradient based optimization methods that we present in Chapters 5, 6, and 7 are in almost all cases too restrictive to cover optimization problems associated to the training of ANNs. However, such optimization problems can be covered by the *Kurdyka–Łojasiewicz* (KL) approach which we discuss in detail in Chapter 9. In Chapter 10 we rigorously review *batch normalization* (BN) methods, which are popular methods that aim to accelerate ANN training procedures in data-driven learning problems. In Chapter 11 we review and study the approach to optimize an objective function through different random initializations.

The mathematical analysis of deep learning algorithms does not only consist of error estimates for approximation capacities of ANNs (cf. Part II) and of error estimates for the involved optimization methods (cf. Part III) but also requires estimates for the *generalization error* which, roughly speaking, arises when the probability distribution associated to the learning problem cannot be accessed explicitly but is approximated by a finite number of realizations/data. It is precisely the subject of Part IV to study the generalization error. Specifically, in Chapter 12 we review suitable probabilistic generalization error estimates and in Chapter 13 we review suitable strong L^p-type generalization error estimates.

In Part V we illustrate how to combine parts of the *approximation error* estimates from Part II, parts of the *optimization error* estimates from Part III, and parts of the *generalization error* estimates from Part IV to establish estimates for the overall error in the exemplary situation of the training of ANNs based on SGD-type optimization methods with many independent random initializations. Specifically, in Chapter 14 we present a suitable overall error decomposition for supervised learning problems, which we employ in Chapter 15 together with some of the findings of Parts II, III, and IV to establish the aforementioned illustrative overall error analysis.

Deep learning methods have not only become very popular for data-driven learning problems, but are nowadays also heavily used for approximately solving *partial differential equations* (PDEs). In Part VI we review and implement three popular variants of such deep learning methods for PDEs. Specifically, in Chapter 16 we treat *physics-informed neural networks* (PINNs) and *deep Galerkin methods* (DGMs) and in Chapter 17 we treat *deep Kolmogorov methods* (DKMs).

This book contains a number of Python source codes, which can be downloaded from two sources, namely from the public GitHub repository at https://github.com/introdeeplearning/book and from the arXiv page of this book (by clicking on the link "Other formats" and then on "Download source"). For ease of reference, the caption of each source listing in this book contains the filename of the corresponding source file.

This book grew out of a series of lectures held by the authors at ETH Zurich, University of Münster, and the Chinese University of Hong Kong, Shenzhen. It is in parts based on recent joint articles of Christian Beck, Sebastian Becker, Weinan E, Lukas Gonon, Robin Graeber, Philipp Grohs, Fabian Hornung, Martin Hutzenthaler, Nor Jaafari, Joshua Lee Padgett, Adrian Riekert, Diyora Salimova, Timo Welti, and Philipp Zimmermann with the authors of this book. We thank all of our aforementioned co-authors for very fruitful collaborations. Special thanks are due to Timo Welti for his permission to integrate slightly modified extracts of the article [230] into this book. We also thank Lukas Gonon, Timo Kröger, Siyu Liang, and Joshua Lee Padget for several insightful discussions and useful suggestions. Finally, we thank the students of the courses that we held on the basis of preliminary material of this book for bringing several typos to our notice.

This work was supported by the internal project fund from the Shenzhen Research Institute of Big Data under grant T00120220001. This work has been partially funded by the National Science Foundation of China (NSFC) under grant number 12250610192. The first author gratefully acknowledges the support of the Cluster of Excellence EXC 2044-390685587, Mathematics Münster: Dynamics-Geometry-Structure funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation).

Shenzhen and Münster,
November 2023

Arnulf Jentzen
Benno Kuckuck
Philippe von Wurstemberger

Contents

Introduction

Very roughly speaking, the field *deep learning* can be divided into three subfields, deep *supervised learning*, deep *unsupervised learning*, and deep *reinforcement learning*. Algorithms in deep supervised learning often seem to be most accessible for a mathematical analysis. In the following we briefly sketch in a simplified situation some ideas of deep supervised learning.

Let $d, M \in \mathbb{N} = \{1, 2, 3, \dots\}$, $\mathcal{E} \in C(\mathbb{R}^d, \mathbb{R})$, $x_1, x_2, \dots, x_{M+1} \in \mathbb{R}^d$, $y_1, y_2, \dots, y_M \in \mathbb{R}$ satisfy for all $m \in \{1, 2, \dots, M\}$ that

$$y_m = \mathcal{E}(x_m). \tag{1}$$

In the framework described in the previous sentence we think of $M \in \mathbb{N}$ as the number of available known input-output data pairs, we think of $d \in \mathbb{N}$ as the dimension of the input data, we think of $\mathcal{E} \colon \mathbb{R}^d \to \mathbb{R}$ as an unknown function which relates input and output data through (1), we think of $x_1, x_2, \dots, x_{M+1} \in \mathbb{R}^d$ as the available known input data, and we think of $y_1, y_2, \dots, y_M \in \mathbb{R}$ as the available known output data.

In the context of a learning problem of the type (1) the objective then is to approximately compute the output $\mathcal{E}(x_{M+1})$ of the $(M+1)$-th input data x_{M+1} without using explicit knowledge of the function $\mathcal{E} \colon \mathbb{R}^d \to \mathbb{R}$ but instead by using the knowledge of the M input-output data pairs

$$(x_1, y_1) = (x_1, \mathcal{E}(x_1)),\ (x_2, y_2) = (x_2, \mathcal{E}(x_2)),\ \dots\ , (x_M, y_M) = (x_M, \mathcal{E}(x_M)) \in \mathbb{R}^d \times \mathbb{R}. \tag{2}$$

To accomplish this, one considers the optimization problem of computing approximate minimizers of the function $\mathfrak{L} \colon C(\mathbb{R}^d, \mathbb{R}) \to [0, \infty)$ which satisfies for all $\phi \in C(\mathbb{R}^d, \mathbb{R})$ that

$$\mathfrak{L}(\phi) = \frac{1}{M} \left[\sum_{m=1}^{M} |\phi(x_m) - y_m|^2 \right]. \tag{3}$$

Observe that (1) ensures that $\mathfrak{L}(\mathcal{E}) = 0$ and, in particular, we have that the unknown function $\mathcal{E} \colon \mathbb{R}^d \to \mathbb{R}$ in (1) above is a minimizer of the function

$$\mathfrak{L} \colon C(\mathbb{R}^d, \mathbb{R}) \to [0, \infty). \tag{4}$$

The optimization problem of computing approximate minimizers of the function $\mathfrak{L}$ is not suitable for discrete numerical computations on a computer as the function $\mathfrak{L}$ is defined on the infinite dimensional vector space $C(\mathbb{R}^d, \mathbb{R})$.

To overcome this we introduce a spatially discretized version of this optimization problem. More specifically, let $\mathfrak{d} \in \mathbb{N}$, let $\psi = (\psi_\theta)_{\theta \in \mathbb{R}^{\mathfrak{d}}} : \mathbb{R}^{\mathfrak{d}} \to C(\mathbb{R}^d, \mathbb{R})$ be a function, and let $\mathscr{L} : \mathbb{R}^{\mathfrak{d}} \to [0, \infty)$ satisfy

$$\mathscr{L} = \mathfrak{L} \circ \psi. \tag{5}$$

We think of the set

$$\left\{\psi_\theta : \theta \in \mathbb{R}^{\mathfrak{d}}\right\} \subseteq C(\mathbb{R}^d, \mathbb{R}) \tag{6}$$

as a parametrized set of functions which we employ to approximate the infinite dimensional vector space $C(\mathbb{R}^d, \mathbb{R})$ and we think of the function

$$\mathbb{R}^{\mathfrak{d}} \ni \theta \mapsto \psi_\theta \in C(\mathbb{R}^d, \mathbb{R}) \tag{7}$$

as the parametrization function associated to this set. For example, in the case $d = 1$ one could think of (7) as the parametrization function associated to polynomials in the sense that for all $\theta = (\theta_1, \dots, \theta_{\mathfrak{d}}) \in \mathbb{R}^{\mathfrak{d}}$, $x \in \mathbb{R}$ it holds that

$$\psi_\theta(x) = \sum_{k=0}^{\mathfrak{d}-1} \theta_{k+1} x^k \tag{8}$$

or one could think of (7) as the parametrization associated to trigonometric polynomials. However, in the context of *deep supervised learning* one neither chooses (7) as parametrization of polynomials nor as parametrization of trigonometric polynomials, but instead one chooses (7) as a parametrization associated to *deep* ANNs. In Chapter 1 in Part I we present different types of such deep ANN parametrization functions in all mathematical details.

Taking the set in (6) and its parametrization function in (7) into account, we then intend to compute approximate minimizers of the function $\mathfrak{L}$ restricted to the set $\{\psi_\theta : \theta \in \mathbb{R}^{\mathfrak{d}}\}$, that is, we consider the optimization problem of computing approximate minimizers of the function

$$\left\{\psi_\theta : \theta \in \mathbb{R}^{\mathfrak{d}}\right\} \ni \phi \mapsto \mathfrak{L}(\phi) = \frac{1}{M}\left[\sum_{m=1}^{M} |\phi(\mathscr{x}_m) - \mathscr{y}_m|^2\right] \in [0, \infty). \tag{9}$$

Employing the parametrization function in (7), one can also reformulate the optimization problem in (9) as the optimization problem of computing approximate minimizers of the function

$$\mathbb{R}^{\mathfrak{d}} \ni \theta \mapsto \mathscr{L}(\theta) = \mathfrak{L}(\psi_\theta) = \frac{1}{M}\left[\sum_{m=1}^{M} |\psi_\theta(\mathscr{x}_m) - \mathscr{y}_m|^2\right] \in [0, \infty) \tag{10}$$

and this optimization problem now has the potential to be amenable for discrete numerical computations. In the context of deep supervised learning, where one chooses the parametrization function in (7) as deep ANN parametrizations, one would apply an SGD-type optimization algorithm to the optimization problem in (10) to compute approximate minimizers of (10). In Chapter 7 in Part III we present the most common variants of such SGD-type optimization algorithms. If $\vartheta \in \mathbb{R}^{\mathfrak{d}}$ is an approximate minimizer of (10) in the sense that $\mathscr{L}(\vartheta) \approx \inf_{\theta \in \mathbb{R}^{\mathfrak{d}}} \mathscr{L}(\theta)$, one then considers $\psi_\vartheta(\boldsymbol{x}_{M+1})$ as an approximation

$$\psi_\vartheta(\boldsymbol{x}_{M+1}) \approx \mathcal{E}(\boldsymbol{x}_{M+1}) \tag{11}$$

of the unknown output $\mathcal{E}(\boldsymbol{x}_{M+1})$ of the $(M+1)$-th input data $\boldsymbol{x}_{M+1}$. We note that in deep supervised learning algorithms one typically aims to compute an approximate minimizer $\vartheta \in \mathbb{R}^{\mathfrak{d}}$ of (10) in the sense that $\mathscr{L}(\vartheta) \approx \inf_{\theta \in \mathbb{R}^{\mathfrak{d}}} \mathscr{L}(\theta)$, which is, however, typically not a minimizer of (10) in the sense that $\mathscr{L}(\vartheta) = \inf_{\theta \in \mathbb{R}^{\mathfrak{d}}} \mathscr{L}(\theta)$ (cf. Section 9.14).

In (3) above we have set up an optimization problem for the learning problem by using the standard mean squared error function to measure the loss. This *mean squared error loss function* is just one possible example in the formulation of deep learning optimization problems. In particular, in image classification problems other loss functions such as the *cross-entropy loss function* are often used and we refer to Chapter 5 of Part III for a survey of commonly used loss function in deep learning algorithms (see Section 5.4.2). We also refer to Chapter 9 for convergence results in the above framework where the parametrization function in (7) corresponds to *fully-connected feedforward* ANNs (see Section 9.14).

Part I

Artificial neural networks (ANNs)

Chapter 1

Basics on ANNs

In this chapter we review different types of architectures of ANNs such as fully-connected feedforward ANNs (see Sections 1.1 and 1.3), CNNs (see Section 1.4), ResNets (see Section 1.5), and RNNs (see Section 1.6), we review different types of popular activation functions used in applications such as the *rectified linear unit* (ReLU) activation (see Section 1.2.3), the *Gaussian error linear unit* (GELU) activation (see Section 1.2.6), and the standard logistic activation (see Section 1.2.7) among others, and we review different procedures for how ANNs can be formulated in rigorous mathematical terms (see Section 1.1 for a vectorized description and Section 1.3 for a structured description).

In the literature different types of ANN architectures and activation functions have been reviewed in several excellent works; cf., for example, [4, 9, 39, 60, 63, 97, 164, 182, 189, 367, 373, 389, 431] and the references therein. The specific presentation of Sections 1.1 and 1.3 is based on [19, 20, 25, 159, 180].

1.1 Fully-connected feedforward ANNs (vectorized description)

We start the mathematical content of this book with a review of fully-connected feedforward ANNs, the most basic type of ANNs. Roughly speaking, fully-connected feedforward ANNs can be thought of as parametric functions resulting from successive compositions of affine functions followed by nonlinear functions, where the parameters of a fully-connected feedforward ANN correspond to all the entries of the linear transformation matrices and translation vectors of the involved affine functions (cf. Definition 1.1.3 below for a precise definition of fully-connected feedforward ANNs and Figure 1.2 below for a graphical illustration of fully-connected feedforward ANNs). The linear transformation matrices and translation vectors are sometimes called *weight matrices* and *bias vectors*, respectively, and can be thought of as the *trainable parameters* of fully-connected feedforward ANNs (cf. Remark 1.1.5 below).

In this section we introduce in Definition 1.1.3 below a *vectorized description* of fully-connected feedforward ANNs in the sense that all the trainable parameters of a fully-connected feedforward ANN are represented by the components of a single Euclidean vector. In Section 1.3 below we will discuss an alternative way to describe fully-connected feedforward ANNs in which the trainable parameters of a fully-connected feedforward ANN are represented by a tuple of matrix-vector pairs corresponding to the weight matrices and bias vectors of the fully-connected feedforward ANNs (cf. Definitions 1.3.1 and 1.3.4 below).

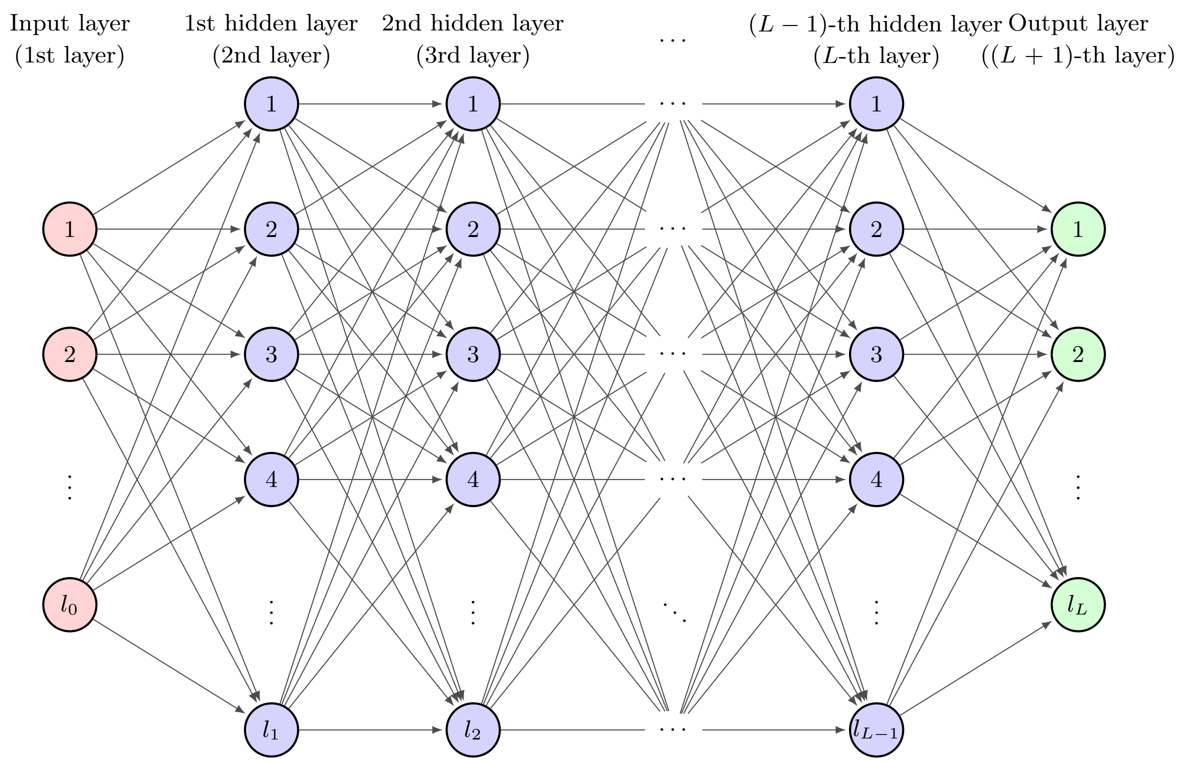

Figure 1.1: Graphical illustration of a fully-connected feedforward ANN consisting of $L \in \mathbb{N}$ affine transformations (i.e., consisting of $L+1$ layers: one input layer, $L-1$ hidden layers, and one output layer) with $l_0 \in \mathbb{N}$ neurons on the input layer (i.e., with l_0-dimensional input layer), with $l_1 \in \mathbb{N}$ neurons on the first hidden layer (i.e., with l_1-dimensional first hidden layer), with $l_2 \in \mathbb{N}$ neurons on the second hidden layer (i.e., with l_2-dimensional second hidden layer), $\dots$, with l_{L-1} neurons on the $(L-1)$-th hidden layer (i.e., with (l_{L-1})-dimensional $(L-1)$-th hidden layer), and with l_L neurons in the output layer (i.e., with l_L-dimensional output layer).

1.1.1 Affine functions

Definition 1.1.1 (Affine functions)**.** *Let $\mathfrak{d}, m, n \in \mathbb{N}$, $s \in \mathbb{N}_0$, $\theta = (\theta_1, \theta_2, \dots, \theta_{\mathfrak{d}}) \in \mathbb{R}^{\mathfrak{d}}$ satisfy $\mathfrak{d} \geq s + mn + m$. Then we denote by $\mathcal{A}^{\theta,s}_{m,n} \colon \mathbb{R}^n \to \mathbb{R}^m$ the function which satisfies for all $x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$ that*

$$\begin{split} \mathcal{A}^{\theta,s}_{m,n}(x) &= \begin{pmatrix} \theta_{s+1} & \theta_{s+2} & \cdots & \theta_{s+n} \\ \theta_{s+n+1} & \theta_{s+n+2} & \cdots & \theta_{s+2n} \\ \theta_{s+2n+1} & \theta_{s+2n+2} & \cdots & \theta_{s+3n} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{s+(m-1)n+1} & \theta_{s+(m-1)n+2} & \cdots & \theta_{s+mn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} \theta_{s+mn+1} \\ \theta_{s+mn+2} \\ \theta_{s+mn+3} \\ \vdots \\ \theta_{s+mn+m} \end{pmatrix} \\ &= \Big(\big[\textstyle\sum_{k=1}^n x_k \theta_{s+k} \big] + \theta_{s+mn+1}, \big[\textstyle\sum_{k=1}^n x_k \theta_{s+n+k} \big] + \theta_{s+mn+2}, \dots, \\ &\qquad\qquad \big[\textstyle\sum_{k=1}^n x_k \theta_{s+(m-1)n+k} \big] + \theta_{s+mn+m} \Big) \end{split} \tag{1.1}$$

and we call $\mathcal{A}^{\theta,s}_{m,n}$ the affine function from $\mathbb{R}^n$ to $\mathbb{R}^m$ associated to (θ, s).

Example 1.1.2 (Example for Definition 1.1.1)**.** *Let $\theta = (0, 1, 2, 0, 3, 3, 0, 1, 7) \in \mathbb{R}^9$. Then*

$$\mathcal{A}^{\theta,1}_{2,2}((1, 2)) = (8, 6) \tag{1.2}$$

(cf. Definition 1.1.1).

Proof for Example 1.1.2. Observe that (1.1) ensures that

$$\mathcal{A}^{\theta,1}_{2,2}((1, 2)) = \begin{pmatrix} 1 & 2 \\ 0 & 3 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} + \begin{pmatrix} 3 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 + 4 \\ 0 + 6 \end{pmatrix} + \begin{pmatrix} 3 \\ 0 \end{pmatrix} = \begin{pmatrix} 8 \\ 6 \end{pmatrix}. \tag{1.3}$$

The proof for Example 1.1.2 is thus complete. $\square$

Exercise 1.1.1. Let $\theta = (3, 1, -2, 1, -3, 0, 5, 4, -1, -1, 0) \in \mathbb{R}^{11}$. Specify $\mathcal{A}^{\theta,2}_{2,3}((-1, 1, -1))$ explicitly and prove that your result is correct (cf. Definition 1.1.1)!

1.1.2 Vectorized description of fully-connected feedforward ANNs

Definition 1.1.3 (Vectorized description of fully-connected feedforward ANNs)**.** *Let $\mathfrak{d}, L \in \mathbb{N}$, $l_0, l_1, \dots, l_L \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ satisfy*

$$\mathfrak{d} \geq \sum_{k=1}^{L} l_k (l_{k-1} + 1) \tag{1.4}$$

and for every $k \in \{1, 2, \dots, L\}$ let $\Psi_k \colon \mathbb{R}^{l_k} \to \mathbb{R}^{l_k}$ be a function. Then we denote by $\mathcal{N}^{\theta, l_0}_{\Psi_1, \Psi_2, \dots, \Psi_L} \colon \mathbb{R}^{l_0} \to \mathbb{R}^{l_L}$ the function which satisfies for all $x \in \mathbb{R}^{l_0}$ that

$$\begin{split}\big(\mathcal{N}^{\theta, l_0}_{\Psi_1, \Psi_2, \dots, \Psi_L}\big)(x) = \big(\Psi_L \circ \mathcal{A}^{\theta, \sum_{k=1}^{L-1} l_k(l_{k-1}+1)}_{l_L, l_{L-1}} \circ \Psi_{L-1} \circ \mathcal{A}^{\theta, \sum_{k=1}^{L-2} l_k(l_{k-1}+1)}_{l_{L-1}, l_{L-2}} \circ \dots \\ \dots \circ \Psi_2 \circ \mathcal{A}^{\theta, l_1(l_0+1)}_{l_2, l_1} \circ \Psi_1 \circ \mathcal{A}^{\theta, 0}_{l_1, l_0}\big)(x)\end{split} \tag{1.5}$$

and we call $\mathcal{N}^{\theta, l_0}_{\Psi_1, \Psi_2, \dots, \Psi_L}$ the realization function of the fully-connected feedforward ANN associated to θ with $L+1$ layers with dimensions $(l_0, l_1, \dots, l_L)$ and activation functions $(\Psi_1, \Psi_2, \dots, \Psi_L)$ (we call $\mathcal{N}^{\theta, l_0}_{\Psi_1, \Psi_2, \dots, \Psi_L}$ the realization of the fully-connected feedforward ANN associated to θ with $L+1$ layers with dimensions $(l_0, l_1, \dots, l_L)$ and activations $(\Psi_1, \Psi_2, \dots, \Psi_L)$) (cf. Definition 1.1.1).

Example 1.1.4 (Example for Definition 1.1.3)**.** *Let $\theta = (1, -1, 2, -2, 3, -3, 0, 0, 1) \in \mathbb{R}^9$ and let $\Psi \colon \mathbb{R}^2 \to \mathbb{R}^2$ satisfy for all $x = (x_1, x_2) \in \mathbb{R}^2$ that*

$$\Psi(x) = (\max\{x_1, 0\}, \max\{x_2, 0\}). \tag{1.6}$$

Then

$$\big(\mathcal{N}^{\theta, 1}_{\Psi, \mathrm{id}_{\mathbb{R}}}\big)(2) = 12 \tag{1.7}$$

(cf. Definition 1.1.3).

Proof for Example 1.1.4. Note that (1.1), (1.5), and (1.6) assure that

$$\begin{split}\big(\mathcal{N}^{\theta, 1}_{\Psi, \mathrm{id}_{\mathbb{R}}}\big)(2) &= \big(\mathrm{id}_{\mathbb{R}} \circ \mathcal{A}^{\theta, 4}_{1,2} \circ \Psi \circ \mathcal{A}^{\theta, 0}_{2,1}\big)(2) = \big(\mathcal{A}^{\theta, 4}_{1,2} \circ \Psi\big)\left(\begin{pmatrix} 1 \\ -1 \end{pmatrix}(2) + \begin{pmatrix} 2 \\ -2 \end{pmatrix}\right) \\ &= \big(\mathcal{A}^{\theta, 4}_{1,2} \circ \Psi\big)\left(\begin{pmatrix} 4 \\ -4 \end{pmatrix}\right) = \mathcal{A}^{\theta, 4}_{1,2}\left(\begin{pmatrix} 4 \\ 0 \end{pmatrix}\right) = \begin{pmatrix} 3 & -3 \end{pmatrix}\begin{pmatrix} 4 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \end{pmatrix} = 12\end{split} \tag{1.8}$$

(cf. Definitions 1.1.1 and 1.1.3). The proof for Example 1.1.4 is thus complete. □

Exercise 1.1.2. Let $\theta = (1, -1, 0, 0, 1, -1, 0) \in \mathbb{R}^7$ and let $\Psi \colon \mathbb{R}^2 \to \mathbb{R}^2$ satisfy for all $x = (x_1, x_2) \in \mathbb{R}^2$ that

$$\Psi(x) = (\max\{x_1, 0\}, \min\{x_2, 0\}). \tag{1.9}$$

Prove or disprove the following statement: It holds that

$$\big(\mathcal{N}^{\theta, 1}_{\Psi, \mathrm{id}_{\mathbb{R}}}\big)(-1) = -1 \tag{1.10}$$

(cf. Definition 1.1.3).

Exercise 1.1.3. Let $\theta = (\theta_1, \theta_2, \dots, \theta_{10}) \in \mathbb{R}^{10}$ satisfy

$$\theta = (\theta_1, \theta_2, \dots, \theta_{10}) = (1, 0, 2, -1, 2, 0, -1, 1, 2, 1)$$

and let $m\colon \mathbb{R} \to \mathbb{R}$ and $q\colon \mathbb{R} \to \mathbb{R}$ satisfy for all $x \in \mathbb{R}$ that

$$m(x) = \max\{-x, 0\} \qquad \text{and} \qquad q(x) = x^2. \tag{1.11}$$

Specify $\big(\mathcal{N}^{\theta,1}_{q,m,q}\big)(0)$, $\big(\mathcal{N}^{\theta,1}_{q,m,q}\big)(1)$, and $\big(\mathcal{N}^{\theta,1}_{q,m,q}\big)(1/2)$ explicitly and prove that your results are correct (cf. Definition 1.1.3)!

Exercise 1.1.4. Let $\theta = (\theta_1, \theta_2, \dots, \theta_{15}) \in \mathbb{R}^{15}$ satisfy

$$(\theta_1, \theta_2, \dots, \theta_{15}) = (1, -2, 0, 3, 2, -1, 0, 3, 1, -1, 1, -1, 2, 0, -1) \tag{1.12}$$

and let $\Phi\colon \mathbb{R}^2 \to \mathbb{R}^2$ and $\Psi\colon \mathbb{R}^2 \to \mathbb{R}^2$ satisfy for all $x, y \in \mathbb{R}$ that $\Phi(x, y) = (y, x)$ and $\Psi(x, y) = (xy, xy)$.

a) Prove or disprove the following statement: It holds that $\big(\mathcal{N}^{\theta,2}_{\Phi,\Psi}\big)(1, -1) = (4, 4)$ (cf. Definition 1.1.3).

b) Prove or disprove the following statement: It holds that $\big(\mathcal{N}^{\theta,2}_{\Phi,\Psi}\big)(-1, 1) = (-4, -4)$ (cf. Definition 1.1.3).

1.1.3 Weight and bias parameters of fully-connected feedforward ANNs

Remark 1.1.5 (Weights and biases for fully-connected feedforward ANNs). Let $L \in \{2, 3, 4, \dots\}$, $v_0, v_1, \dots, v_{L-1} \in \mathbb{N}_0$, $l_0, l_1, \dots, l_L, \mathfrak{d} \in \mathbb{N}$, $\theta = (\theta_1, \theta_2, \dots, \theta_{\mathfrak{d}}) \in \mathbb{R}^{\mathfrak{d}}$ satisfy for all $k \in \{0, 1, \dots, L-1\}$ that

$$\mathfrak{d} \geq \sum_{i=1}^{L} l_i(l_{i-1} + 1) \qquad \text{and} \qquad v_k = \sum_{i=1}^{k} l_i(l_{i-1} + 1), \tag{1.13}$$

let $W_k \in \mathbb{R}^{l_k \times l_{k-1}}$, $k \in \{1, 2, \dots, L\}$, and $b_k \in \mathbb{R}^{l_k}$, $k \in \{1, 2, \dots, L\}$, satisfy for all $k \in \{1, 2, \dots, L\}$ that

$$W_k = \underbrace{\begin{pmatrix} \theta_{v_{k-1}+1} & \theta_{v_{k-1}+2} & \dots & \theta_{v_{k-1}+l_{k-1}} \\ \theta_{v_{k-1}+l_{k-1}+1} & \theta_{v_{k-1}+l_{k-1}+2} & \dots & \theta_{v_{k-1}+2l_{k-1}} \\ \theta_{v_{k-1}+2l_{k-1}+1} & \theta_{v_{k-1}+2l_{k-1}+2} & \dots & \theta_{v_{k-1}+3l_{k-1}} \\ \vdots & \vdots & \vdots & \vdots \\ \theta_{v_{k-1}+(l_k-1)l_{k-1}+1} & \theta_{v_{k-1}+(l_k-1)l_{k-1}+2} & \dots & \theta_{v_{k-1}+l_k l_{k-1}} \end{pmatrix}}_{\text{weight parameters}} \tag{1.14}$$

$$\text{and} \qquad b_k = \underbrace{\big(\theta_{v_{k-1}+l_k l_{k-1}+1}, \theta_{v_{k-1}+l_k l_{k-1}+2}, \dots, \theta_{v_{k-1}+l_k l_{k-1}+l_k}\big)}_{\text{bias parameters}}, \tag{1.15}$$

and let $\Psi_k\colon \mathbb{R}^{l_k} \to \mathbb{R}^{l_k}$, $k \in \{1, 2, \dots, L\}$, be functions. Then

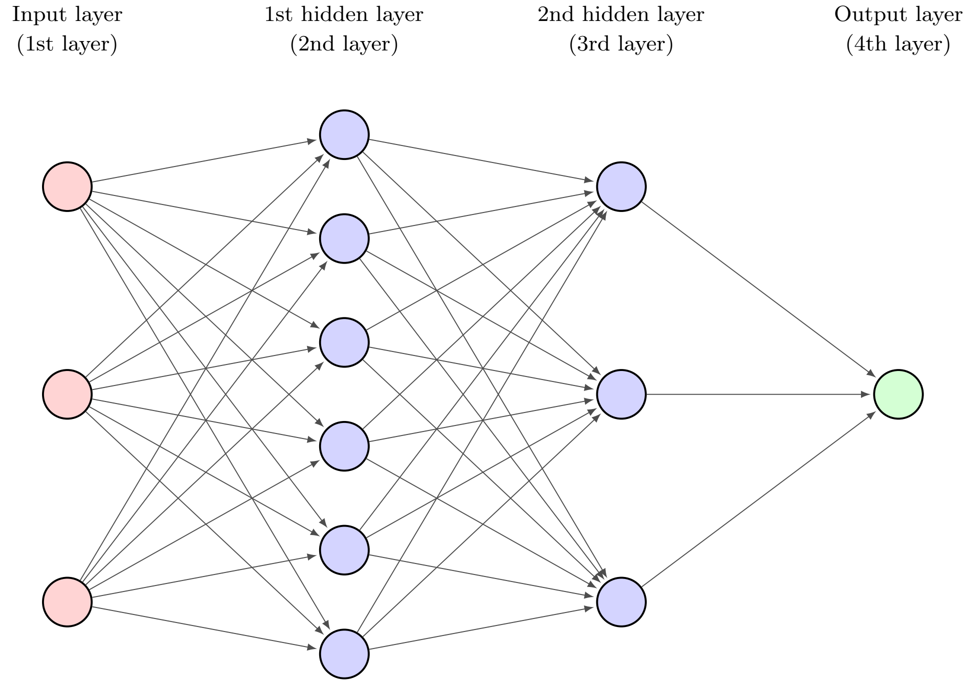

Figure 1.2: Graphical illustration of an ANN. The ANN has 2 hidden layers and length $L = 3$ with 3 neurons in the input layer (corresponding to $l_0 = 3$), 6 neurons in the first hidden layer (corresponding to $l_1 = 6$), 3 neurons in the second hidden layer (corresponding to $l_2 = 3$), and one neuron in the output layer (corresponding to $l_3 = 1$). In this situation we have an ANN with 39 weight parameters and 10 bias parameters adding up to 49 parameters overall. The realization of this ANN is a function from $\mathbb{R}^3$ to $\mathbb{R}$.

(i) it holds that

$$\mathcal{N}^{\theta,l_0}_{\Psi_1,\Psi_2,\dots,\Psi_L} = \Psi_L \circ \mathcal{A}^{\theta,v_{L-1}}_{l_L,l_{L-1}} \circ \Psi_{L-1} \circ \mathcal{A}^{\theta,v_{L-2}}_{l_{L-1},l_{L-2}} \circ \Psi_{L-2} \circ \ldots \circ \mathcal{A}^{\theta,v_1}_{l_2,l_1} \circ \Psi_1 \circ \mathcal{A}^{\theta,v_0}_{l_1,l_0} \tag{1.16}$$

and

(ii) it holds for all $k \in \{1, 2, \dots, L\}$, $x \in \mathbb{R}^{l_{k-1}}$ that $\mathcal{A}^{\theta,v_{k-1}}_{l_k,l_{k-1}}(x) = W_k x + b_k$

(cf. Definitions 1.1.1 and 1.1.3).

1.2 Activation functions

In this section we review a few popular activation functions from the literature (cf. Definition 1.1.3 above and Definition 1.3.4 below for the use of activation functions in the context of fully-connected feedforward ANNs, cf. Definition 1.4.5 below for the use of activation functions in the context of CNNs, cf. Definition 1.5.4 below for the use of activation functions in the context of ResNets, and cf. Definitions 1.6.3 and 1.6.4 below for the use of activation functions in the context of RNNs).

1.2.1 Multidimensional versions

To describe multidimensional activation functions, we frequently employ the concept of the multidimensional version of a function. This concept is the subject of the next notion.

Definition 1.2.1 (Multidimensional versions of one-dimensional functions)**.** *Let $T \in \mathbb{N}$, $d_1, d_2, \ldots, d_T \in \mathbb{N}$ and let $\psi\colon \mathbb{R} \to \mathbb{R}$ be a function. Then we denote by*

$$\mathfrak{M}_{\psi,d_1,d_2,\ldots,d_T}\colon \mathbb{R}^{d_1\times d_2\times\ldots\times d_T} \to \mathbb{R}^{d_1\times d_2\times\ldots\times d_T} \tag{1.17}$$

the function which satisfies for all $x = (x_{k_1,k_2,\ldots,k_T})_{(k_1,k_2,\ldots,k_T)\in(\times_{t=1}^{T}\{1,2,\ldots,d_t\})} \in \mathbb{R}^{d_1\times d_2\times\ldots\times d_T}$, $y = (y_{k_1,k_2,\ldots,k_T})_{(k_1,k_2,\ldots,k_T)\in(\times_{t=1}^{T}\{1,2,\ldots,d_t\})} \in \mathbb{R}^{d_1\times d_2\times\ldots\times d_T}$ with $\forall\, k_1 \in \{1,2,\ldots,d_1\}$, $k_2 \in \{1,2,\ldots,d_2\}$, $\ldots$, $k_T \in \{1,2,\ldots,d_T\}\colon y_{k_1,k_2,\ldots,k_T} = \psi(x_{k_1,k_2,\ldots,k_T})$ that

$$\mathfrak{M}_{\psi,d_1,d_2,\ldots,d_T}(x) = y \tag{1.18}$$

and we call $\mathfrak{M}_{\psi,d_1,d_2,\ldots,d_T}$ the $d_1 \times d_2 \times \ldots \times d_T$-dimensional version of ψ.

Example 1.2.2 (Example for Definition 1.2.1)**.** *Let $A \in \mathbb{R}^{3\times 1\times 2}$ satisfy*

$$A = \big(\begin{pmatrix}1 & -1\end{pmatrix}, \begin{pmatrix}-2 & 2\end{pmatrix}, \begin{pmatrix}3 & -3\end{pmatrix}\big) \tag{1.19}$$

and let $\psi\colon \mathbb{R} \to \mathbb{R}$ satisfy for all $x \in \mathbb{R}$ that $\psi(x) = x^2$. Then

$$\mathfrak{M}_{\psi,3,1,3}(A) = \big(\begin{pmatrix}1 & 1\end{pmatrix}, \begin{pmatrix}4 & 4\end{pmatrix}, \begin{pmatrix}9 & 9\end{pmatrix}\big) \tag{1.20}$$

Proof for Example 1.2.2. Note that (1.18) establishes (1.20). The proof for Example 1.2.2 is thus complete. □

Exercise 1.2.1. Let $A \in \mathbb{R}^{2\times 3}$, $B \in \mathbb{R}^{2\times 2\times 2}$ satisfy

$$A = \begin{pmatrix}3 & -2 & 5\\ 1 & 0 & -2\end{pmatrix} \qquad \text{and} \qquad B = \left(\begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix}, \begin{pmatrix}-3 & -4\\ 5 & 2\end{pmatrix}\right) \tag{1.21}$$

and let $\psi\colon \mathbb{R} \to \mathbb{R}$ satisfy for all $x \in \mathbb{R}$ that $\psi(x) = |x|$. Specify $\mathfrak{M}_{\psi,2,3}(A)$ and $\mathfrak{M}_{\psi,2,2,2}(B)$ explicitly and prove that your results are correct (cf. Definition 1.2.1)!

Exercise 1.2.2. Let $\theta = (\theta_1, \theta_2, \dots, \theta_{14}) \in \mathbb{R}^{14}$ satisfy

$$(\theta_1, \theta_2, \dots, \theta_{14}) = (0, 1, 2, 2, 1, 0, 1, 1, 1, -3, -1, 4, 0, 1) \tag{1.22}$$

and let $f\colon \mathbb{R} \to \mathbb{R}$ and $g\colon \mathbb{R} \to \mathbb{R}$ satisfy for all $x \in \mathbb{R}$ that

$$f(x) = \frac{1}{1 + |x|} \qquad \text{and} \qquad g(x) = x^2. \tag{1.23}$$

Specify $\big(\mathcal{N}^{\theta,1}_{\mathfrak{M}_{f,3},\mathfrak{M}_{g,2}}\big)(1)$ and $\big(\mathcal{N}^{\theta,1}_{\mathfrak{M}_{g,2},\mathfrak{M}_{f,3}}\big)(1)$ explicitly and prove that your results are correct (cf. Definitions 1.1.3 and 1.2.1)!

1.2.2 Single hidden layer fully-connected feedforward ANNs

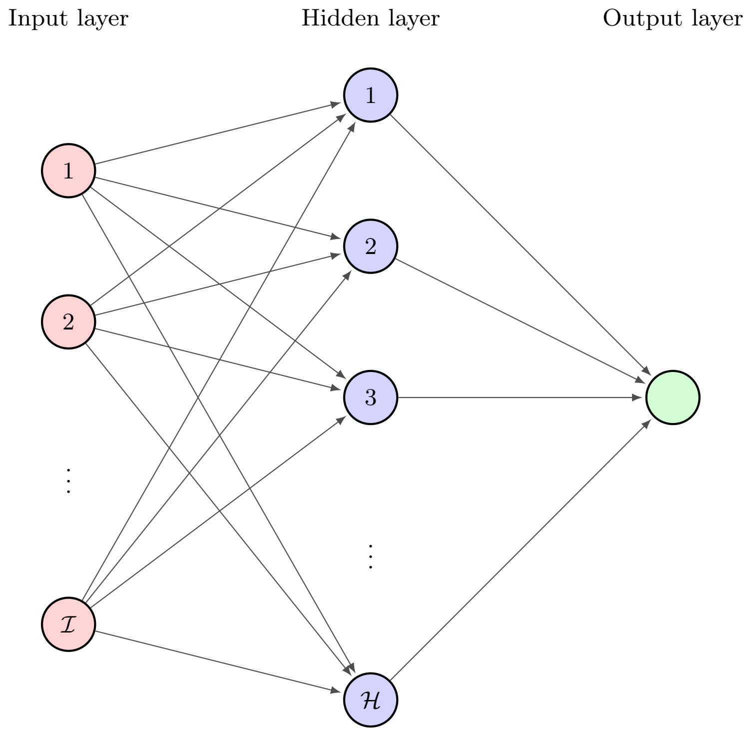

Figure 1.3: Graphical illustration of a fully-connected feedforward ANN consisting of two affine transformations (i.e., consisting of 3 layers: one input layer, one hidden layer, and one output layer) with $\mathcal{I} \in \mathbb{N}$ neurons on the input layer (i.e., with $\mathcal{I}$-dimensional input layer), with $\mathcal{H} \in \mathbb{N}$ neurons on the hidden layer (i.e., with $\mathcal{H}$-dimensional hidden layer), and with one neuron in the output layer (i.e., with 1-dimensional output layer).

Lemma 1.2.3 (Fully-connected feedforward ANN with one hidden layer)**.** *Let $\mathcal{I}, \mathcal{H} \in \mathbb{N}$, $\theta = (\theta_1, \theta_2, \dots, \theta_{\mathcal{H}\mathcal{I}+2\mathcal{H}+1}) \in \mathbb{R}^{\mathcal{H}\mathcal{I}+2\mathcal{H}+1}$, $x = (x_1, x_2, \dots, x_{\mathcal{I}}) \in \mathbb{R}^{\mathcal{I}}$ and let $\psi \colon \mathbb{R} \to \mathbb{R}$ be a function. Then*

$$\mathcal{N}^{\theta,\mathcal{I}}_{\mathfrak{M}_{\psi,\mathcal{H}},\mathrm{id}_{\mathbb{R}}}(x) = \left[\sum_{k=1}^{\mathcal{H}} \theta_{\mathcal{H}\mathcal{I}+\mathcal{H}+k}\, \psi\!\left(\left[\sum_{i=1}^{\mathcal{I}} x_i \theta_{(k-1)\mathcal{I}+i}\right] + \theta_{\mathcal{H}\mathcal{I}+k}\right)\right] + \theta_{\mathcal{H}\mathcal{I}+2\mathcal{H}+1}. \tag{1.24}$$

(cf. Definitions 1.1.1, 1.1.3, and 1.2.1).

Proof of Lemma 1.2.3. Observe that (1.5) and (1.18) show that

$$\begin{aligned}
&\mathcal{N}^{\theta,\mathcal{I}}_{\mathfrak{M}_{\psi,\mathcal{H}},\mathrm{id}_{\mathbb{R}}}(x) \\
&= \left(\mathrm{id}_{\mathbb{R}} \circ \mathcal{A}^{\theta,\mathcal{H}\mathcal{I}+\mathcal{H}}_{1,\mathcal{H}} \circ \mathfrak{M}_{\psi,\mathcal{H}} \circ \mathcal{A}^{\theta,0}_{\mathcal{H},\mathcal{I}}\right)(x) \\
&= \mathcal{A}^{\theta,\mathcal{H}\mathcal{I}+\mathcal{H}}_{1,\mathcal{H}}\big(\mathfrak{M}_{\psi,\mathcal{H}}\big(\mathcal{A}^{\theta,0}_{\mathcal{H},\mathcal{I}}(x)\big)\big) \\
&= \left[\sum_{k=1}^{\mathcal{H}} \theta_{\mathcal{H}\mathcal{I}+\mathcal{H}+k}\, \psi\!\left(\left[\sum_{i=1}^{\mathcal{I}} x_i \theta_{(k-1)\mathcal{I}+i}\right] + \theta_{\mathcal{H}\mathcal{I}+k}\right)\right] + \theta_{\mathcal{H}\mathcal{I}+2\mathcal{H}+1}.
\end{aligned} \tag{1.25}$$

The proof of Lemma 1.2.3 is thus complete. □

1.2.3 Rectified linear unit (ReLU) activation

In this subsection we formulate the ReLU function which is one of the most frequently used activation functions in deep learning applications (cf., for example, LeCun et al. [263]).

Definition 1.2.4 (ReLU activation function)**.** *We denote by $\mathfrak{r} \colon \mathbb{R} \to \mathbb{R}$ the function which satisfies for all $x \in \mathbb{R}$ that*

$$\mathfrak{r}(x) = \max\{x, 0\} \tag{1.26}$$

and we call $\mathfrak{r}$ the ReLU activation function (we call $\mathfrak{r}$ the rectifier function).

```
import matplotlib.pyplot as plt

def setup_axis(xlim, ylim):
    _, ax = plt.subplots()

    ax.set_aspect("equal")
    ax.set_xlim(xlim)
    ax.set_ylim(ylim)
    ax.spines["left"].set_position("zero")
    ax.spines["bottom"].set_position("zero")
    ax.spines["right"].set_color("none")
    ax.spines["top"].set_color("none")
    for s in ax.spines.values():
```

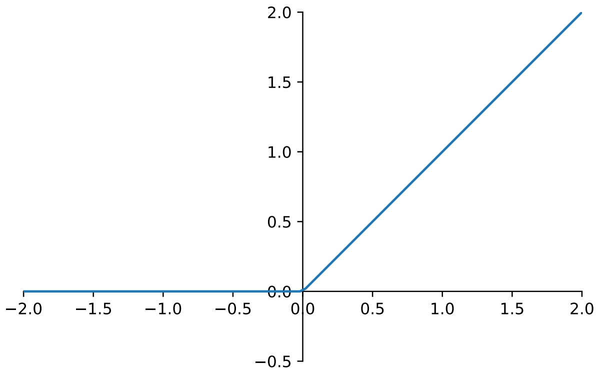

Figure 1.4 (plots/relu.pdf): A plot of the ReLU activation function

```
        s.set_zorder(0)

    return ax
```

Source code 1.1 (code/activation_functions/plot_util.py): Python code for the plot_util module used in the code listings throughout this subsection

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-2,2), (-.5,2))

x = np.linspace(-2, 2, 100)

ax.plot(x, tf.keras.activations.relu(x))

plt.savefig("../../plots/relu.pdf", bbox_inches='tight')
```

Source code 1.2 (code/activation_functions/relu_plot.py): Python code used to create Figure 1.4

Definition 1.2.5 (Multidimensional ReLU activation functions)**.** *Let $d \in \mathbb{N}$. Then we denote by $\mathfrak{R}_d \colon \mathbb{R}^d \to \mathbb{R}^d$ the function given by*

$$\mathfrak{R}_d = \mathfrak{M}_{\mathfrak{r},d} \tag{1.27}$$

and we call $\mathfrak{R}_d$ the d-dimensional ReLU activation function (we call $\mathfrak{R}_d$ the d-dimensional rectifier function) (cf. Definitions 1.2.1 and 1.2.4).

Lemma 1.2.6 (An ANN with the ReLU activation function as the activation function)**.** *Let $W_1 = w_1 = 1$, $W_2 = w_2 = -1$, $b_1 = b_2 = B = 0$. Then it holds for all $x \in \mathbb{R}$ that*

$$x = W_1 \max\{w_1 x + b_1, 0\} + W_2 \max\{w_2 x + b_2, 0\} + B. \tag{1.28}$$

Proof of Lemma 1.2.6. Observe that for all $x \in \mathbb{R}$ it holds that

$$\begin{aligned} &W_1 \max\{w_1 x + b_1, 0\} + W_2 \max\{w_2 x + b_2, 0\} + B \\ &= \max\{w_1 x + b_1, 0\} - \max\{w_2 x + b_2, 0\} = \max\{x, 0\} - \max\{-x, 0\} \\ &= \max\{x, 0\} + \min\{x, 0\} = x. \end{aligned} \tag{1.29}$$

The proof of Lemma 1.2.6 is thus complete. □

Exercise 1.2.3 (Real identity). Prove or disprove the following statement: There exist $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 2l_1 + \big[\sum_{k=2}^{H} l_k(l_{k-1}+1)\big] + l_H + 1$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_{l_1}, \mathfrak{R}_{l_2}, \dots, \mathfrak{R}_{l_H}, \mathrm{id}_{\mathbb{R}}}\big)(x) = x \tag{1.30}$$

(cf. Definitions 1.1.3 and 1.2.5).

The statement of the next lemma, Lemma 1.2.7, provides a partial answer to Exercise 1.2.3. Lemma 1.2.7 follows from an application of Lemma 1.2.6 and the detailed proof of Lemma 1.2.7 is left as an exercise.

Lemma 1.2.7 (Real identity)**.** *Let $\theta = (1, -1, 0, 0, 1, -1, 0) \in \mathbb{R}^7$. Then it holds for all $x \in \mathbb{R}$ that*

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_2, \mathrm{id}_{\mathbb{R}}}\big)(x) = x \tag{1.31}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.4 (Absolute value). Prove or disprove the following statement: There exist $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 2l_1 + \big[\sum_{k=2}^{H} l_k(l_{k-1}+1)\big] + l_H + 1$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_{l_1}, \mathfrak{R}_{l_2}, \dots, \mathfrak{R}_{l_H}, \mathrm{id}_{\mathbb{R}}}\big)(x) = |x| \tag{1.32}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.5 (Exponential). Prove or disprove the following statement: There exist $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 2l_1 + \big[\sum_{k=2}^{H} l_k(l_{k-1}+1)\big] + l_H + 1$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_{l_1}, \mathfrak{R}_{l_2}, \dots, \mathfrak{R}_{l_H}, \mathrm{id}_{\mathbb{R}}}\big)(x) = e^x \tag{1.33}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.6 (Two-dimensional maximum). Prove or disprove the following statement: There exist $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 3l_1 + \big[\sum_{k=2}^{H} l_k(l_{k-1}+1)\big] + l_H + 1$ such that for all $x, y \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,2}_{\mathfrak{R}_{l_1},\mathfrak{R}_{l_2},\dots,\mathfrak{R}_{l_H},\mathrm{id}_{\mathbb{R}}}\big)(x,y) = \max\{x,y\} \tag{1.34}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.7 (Real identity with two hidden layers). Prove or disprove the following statement: There exist $\mathfrak{d}, l_1, l_2 \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 2l_1 + l_1 l_2 + 2l_2 + 1$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_{l_1},\mathfrak{R}_{l_2},\mathrm{id}_{\mathbb{R}}}\big)(x) = x \tag{1.35}$$

(cf. Definitions 1.1.3 and 1.2.5).

The statement of the next lemma, Lemma 1.2.8, provides a partial answer to Exercise 1.2.7. The proof of Lemma 1.2.8 is left as an exercise.

Lemma 1.2.8 (Real identity with two hidden layers). *Let $\theta = (1,\, -1,\, 0,\, 0,\, 1,\, -1,\, -1,\, 1,\, 0,\, 0,\, 1,\, -1,\, 0) \in \mathbb{R}^{13}$. Then it holds for all $x \in \mathbb{R}$ that*

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_2,\mathfrak{R}_2,\mathrm{id}_{\mathbb{R}}}\big)(x) = x \tag{1.36}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.8 (Three-dimensional maximum). Prove or disprove the following statement: There exist $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 4l_1 + \big[\sum_{k=2}^{H} l_k(l_{k-1}+1)\big] + l_H + 1$ such that for all $x, y, z \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,3}_{\mathfrak{R}_{l_1},\mathfrak{R}_{l_2},\dots,\mathfrak{R}_{l_H},\mathrm{id}_{\mathbb{R}}}\big)(x,y,z) = \max\{x,y,z\} \tag{1.37}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.9 (Multidimensional maxima). Prove or disprove the following statement: For every $k \in \mathbb{N}$ there exist $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq (k+1)l_1 + \big[\sum_{k=2}^{H} l_k(l_{k-1}+1)\big] + l_H + 1$ such that for all $x_1, x_2, \dots, x_k \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,k}_{\mathfrak{R}_{l_1},\mathfrak{R}_{l_2},\dots,\mathfrak{R}_{l_H},\mathrm{id}_{\mathbb{R}}}\big)(x_1,x_2,\dots,x_k) = \max\{x_1,x_2,\dots,x_k\} \tag{1.38}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.10. Prove or disprove the following statement: There exist $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 2\,l_1 + \big[\,\sum_{k=2}^{H} l_k(l_{k-1}+1)\big] + (l_H+1)$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_{l_1},\mathfrak{R}_{l_2},\dots,\mathfrak{R}_{l_H},\mathrm{id}_{\mathbb{R}}}\big)(x) = \max\{x, \tfrac{x}{2}\} \tag{1.39}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.11 (Hat function). Prove or disprove the following statement: There exist $\mathfrak{d}, l \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 3l + 1$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_l,\mathrm{id}_{\mathbb{R}}}\big)(x) = \begin{cases} 1 & : x \leq 2 \\ x - 1 & : 2 < x \leq 3 \\ 5 - x & : 3 < x \leq 4 \\ 1 & : x > 4 \end{cases} \tag{1.40}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.12. Prove or disprove the following statement: There exist $\mathfrak{d}, l \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 3l + 1$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_l,\mathrm{id}_{\mathbb{R}}}\big)(x) = \begin{cases} -2 & : x \leq 1 \\ 2x - 4 & : 1 < x \leq 3 \\ 2 & : x > 3 \end{cases} \tag{1.41}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.13. Prove or disprove the following statement: There exists $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 2\,l_1 + \big[\sum_{k=2}^{H} l_k(l_{k-1} + 1)\big] + (l_H + 1)$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_{l_1},\mathfrak{R}_{l_2},\dots,\mathfrak{R}_{l_H},\mathrm{id}_{\mathbb{R}}}\big)(x) = \begin{cases} 0 & : x \leq 1 \\ x - 1 & : 1 \leq x \leq 2 \\ 1 & : x \geq 2 \end{cases} \tag{1.42}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.14. Prove or disprove the following statement: There exist $\mathfrak{d}, l \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 3l + 1$ such that for all $x \in [0, 1]$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_l,\mathrm{id}_{\mathbb{R}}}\big)(x) = x^2 \tag{1.43}$$

(cf. Definitions 1.1.3 and 1.2.5).

Exercise 1.2.15. Prove or disprove the following statement: There exists $\mathfrak{d}, H \in \mathbb{N}$, $l_1, l_2, \dots, l_H \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 2\,l_1 + \big[\sum_{k=2}^{H} l_k(l_{k-1} + 1)\big] + (l_H + 1)$ such that

$$\sup\nolimits_{x \in [-3,-2]} \big|\big(\mathcal{N}^{\theta,1}_{\mathfrak{R}_{l_1},\mathfrak{R}_{l_2},\dots,\mathfrak{R}_{l_H},\mathrm{id}_{\mathbb{R}}}\big)(x) - (x + 2)^2\big| \leq \tfrac{1}{4} \tag{1.44}$$

(cf. Definitions 1.1.3 and 1.2.5).

1.2.4 Clipping activation

Definition 1.2.9 (Clipping activation function)**.** *Let* $u \in [-\infty, \infty)$, $v \in (u, \infty]$. *Then we denote by* $\mathfrak{c}_{u,v} \colon \mathbb{R} \to \mathbb{R}$ *the function which satisfies for all* $x \in \mathbb{R}$ *that*

$$\mathfrak{c}_{u,v}(x) = \max\{u, \min\{x, v\}\}. \tag{1.45}$$

and we call $\mathfrak{c}_{u,v}$ *the* (u, v)*-clipping activation function.*

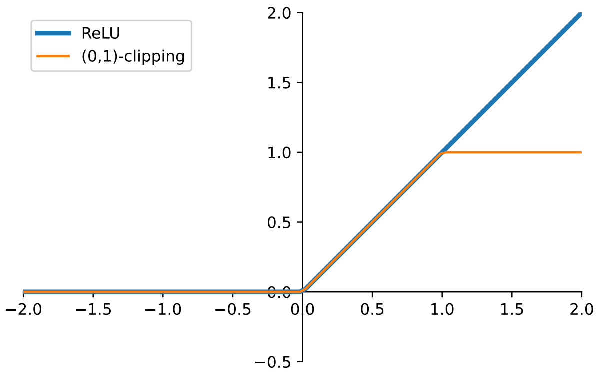

Figure 1.5 (plots/clipping.pdf): A plot of the $(0, 1)$-clipping activation function and the ReLU activation function

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-2,2), (-.5,2))

x = np.linspace(-2, 2, 100)

ax.plot(x, tf.keras.activations.relu(x), linewidth=3, label='ReLU')
ax.plot(x, tf.keras.activations.relu(x, max_value=1),
        label='(0,1)-clipping')
ax.legend()

plt.savefig("../../plots/clipping.pdf", bbox_inches='tight')
```

Source code 1.3 (code/activation_functions/clipping_plot.py): Python code used to create Figure 1.5

Definition 1.2.10 (Multidimensional clipping activation functions). *Let $d \in \mathbb{N}$, $u \in [-\infty, \infty)$, $v \in (u, \infty]$. Then we denote by $\mathfrak{C}_{u,v,d} \colon \mathbb{R}^d \to \mathbb{R}^d$ the function given by*

$$\mathfrak{C}_{u,v,d} = \mathfrak{M}_{\mathfrak{c}_{u,v},d} \tag{1.46}$$

and we call $\mathfrak{C}_{u,v,d}$ the d-dimensional (u, v)-clipping activation function (cf. Definitions 1.2.1 and 1.2.9).

1.2.5 Softplus activation

Definition 1.2.11 (Softplus activation function). *We say that a is the softplus activation function if and only if it holds that $a \colon \mathbb{R} \to \mathbb{R}$ is the function from $\mathbb{R}$ to $\mathbb{R}$ which satisfies for all $x \in \mathbb{R}$ that*

$$a(x) = \ln(1 + \exp(x)). \tag{1.47}$$

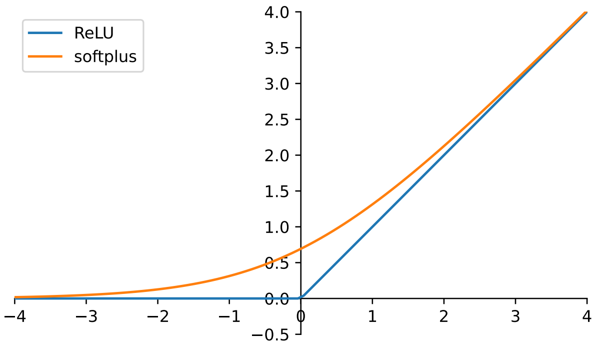

Figure 1.6 (plots/softplus.pdf): A plot of the softplus activation function and the ReLU activation function

```python
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-4,4), (-.5,4))

x = np.linspace(-4, 4, 100)

ax.plot(x, tf.keras.activations.relu(x), label='ReLU')
ax.plot(x, tf.keras.activations.softplus(x), label='softplus')
ax.legend()

plt.savefig("../../plots/softplus.pdf", bbox_inches='tight')
```

Source code 1.4 (`code/activation_functions/softplus_plot.py`): PYTHON code used to create Figure 1.6

The next result, Lemma 1.2.12 below, presents a few elementary properties of the softplus function.

Lemma 1.2.12 (Properties of the softplus function)**.** *Let a be the softplus activation function (cf. Definition 1.2.11). Then*

(i) it holds for all $x \in [0, \infty)$ that $x \leq a(x) \leq x + 1$,

(ii) it holds that $\lim_{x\to-\infty} a(x) = 0$,

(iii) it holds that $\lim_{x\to\infty} a(x) = \infty$, and

(iv) it holds that $a(0) = \ln(2)$

(cf. Definition 1.2.11).

Proof of Lemma 1.2.12. Observe that the fact that $2 \leq \exp(1)$ ensures that for all $x \in [0, \infty)$ it holds that

$$\begin{aligned} x &= \ln(\exp(x)) \leq \ln(1 + \exp(x)) = \ln(\exp(0) + \exp(x)) \\ &\leq \ln(\exp(x) + \exp(x)) = \ln(2 \exp(x)) \leq \ln(\exp(1) \exp(x)) \\ &= \ln(\exp(x + 1)) = x + 1. \end{aligned} \tag{1.48}$$

The proof of Lemma 1.2.12 is thus complete. □

Note that Lemma 1.2.12 ensures that $\mathfrak{s}(0) = \ln(2) = 0.693\ldots$ (cf. Definition 1.2.11). In the next step we introduce the multidimensional version of the softplus function (cf. Definitions 1.2.1 and 1.2.11 above).

Definition 1.2.13 (Multidimensional softplus activation functions)**.** *Let $d \in \mathbb{N}$ and let a be the softplus activation function (cf. Definition 1.2.11). Then we say that A is the d-dimensional softplus activation function if and only if $A = \mathfrak{M}_{a,d}$ (cf. Definition 1.2.1).*

Lemma 1.2.14. *Let $d \in \mathbb{N}$ and let $A\colon \mathbb{R}^d \to \mathbb{R}^d$ be a function. Then A is the d-dimensional softplus activation function if and only if it holds for all $x = (x_1, \ldots, x_d) \in \mathbb{R}^d$ that*

$$A(x) = (\ln(1 + \exp(x_1)), \ln(1 + \exp(x_2)), \ldots, \ln(1 + \exp(x_d))) \tag{1.49}$$

(cf. Definition 1.2.13).

Proof of Lemma 1.2.14. Throughout this proof, let a be the softplus activation function (cf. Definition 1.2.11). Note that (1.18) and (1.47) ensure that for all $x = (x_1, \ldots, x_d) \in \mathbb{R}^d$ it holds that

$$\mathfrak{M}_{a,d}(x) = (\ln(1 + \exp(x_1)), \ln(1 + \exp(x_2)), \ldots, \ln(1 + \exp(x_d))) \tag{1.50}$$

(cf. Definition 1.2.1). The fact that A is the d-dimensional softplus activation function (cf. Definition 1.2.13) if and only if $A = \mathfrak{M}_{a,d}$ hence implies (1.49). The proof of Lemma 1.2.14 is thus complete. □

1.2.6 Gaussian error linear unit (GELU) activation

Another popular activation function is the GELU activation function first introduced in Hendrycks & Gimpel [193]. This activation function is the subject of the next definition.

Definition 1.2.15 (GELU activation function)**.** *We say that a is the GELU unit activation function (we say that a is the GELU activation function) if and only if it holds that $a\colon \mathbb{R} \to \mathbb{R}$ is the function from $\mathbb{R}$ to $\mathbb{R}$ which satisfies for all $x \in \mathbb{R}$ that*

$$a(x) = \frac{x}{\sqrt{2\pi}} \left[\int_{-\infty}^{x} \exp(-\tfrac{z^2}{2}) \, \mathrm{d}z \right]. \tag{1.51}$$

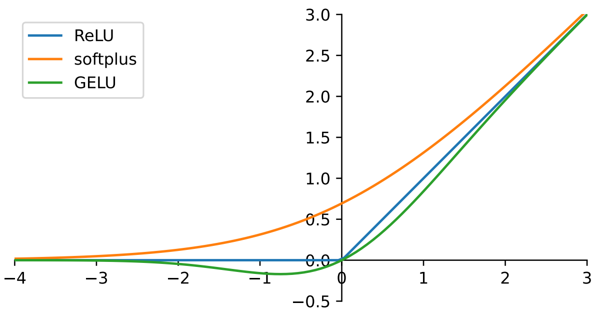

Figure 1.7 (plots/gelu.pdf): A plot of the GELU activation function, the ReLU activation function, and the softplus activation function

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-4,3), (-.5,3))

x = np.linspace(-4, 3, 100)
```

```
ax.plot(x, tf.keras.activations.relu(x), label='ReLU')
ax.plot(x, tf.keras.activations.softplus(x), label='softplus')
ax.plot(x, tf.keras.activations.gelu(x), label='GELU')
ax.legend()

plt.savefig("../../plots/gelu.pdf", bbox_inches='tight')
```

Source code 1.5 (code/activation_functions/gelu_plot.py): PYTHON code used to create Figure 1.7

Lemma 1.2.16. *Let $x \in \mathbb{R}$ and let a be the GELU activation function (cf. Definition 1.2.15). Then the following two statements are equivalent:*

(i) It holds that $a(x) > 0$.

(ii) It holds that $\mathfrak{r}(x) > 0$ (cf. Definition 1.2.4).

Proof of Lemma 1.2.16. Note that (1.26) and (1.51) establish that ((i) $\leftrightarrow$ (ii)). The proof of Lemma 1.2.16 is thus complete. □

Definition 1.2.17 (Multidimensional GELU unit activation function)**.** *Let $d \in \mathbb{N}$ and let a be the GELU activation function (cf. Definition 1.2.15). we say that A is the d-dimensional GELU activation function if and only if $A = \mathfrak{M}_{a,d}$ (cf. Definition 1.2.1).*

1.2.7 Standard logistic activation

Definition 1.2.18 (Standard logistic activation function)**.** *We say that a is the standard logistic activation function if and only if it holds that $a\colon \mathbb{R} \to \mathbb{R}$ is the function from $\mathbb{R}$ to $\mathbb{R}$ which satisfies for all $x \in \mathbb{R}$ that*

$$a(x) = \frac{1}{1 + \exp(-x)} = \frac{\exp(x)}{\exp(x) + 1}. \tag{1.52}$$

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-3,3), (-.5,1.5))

x = np.linspace(-3, 3, 100)

ax.plot(x, tf.keras.activations.relu(x, max_value=1),
        label='(0,1)-clipping')
```

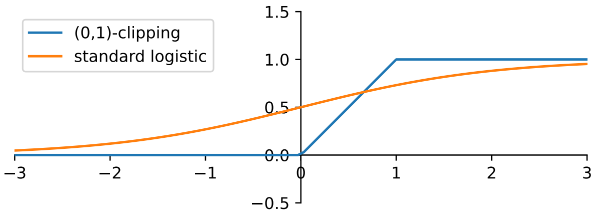

Figure 1.8 (plots/logistic.pdf): A plot of the standard logistic activation function and the $(0, 1)$-clipping activation function

```
ax.plot(x, tf.keras.activations.sigmoid(x),
        label='standard logistic')
ax.legend()

plt.savefig("../../plots/logistic.pdf", bbox_inches='tight')
```

Source code 1.6 (code/activation_functions/logistic_plot.py): Python code used to create Figure 1.8

Definition 1.2.19 (Multidimensional standard logistic activation functions). *Let $d \in \mathbb{N}$ and let a be the standard logistic activation function (cf. Definition 1.2.18). Then we say that A is the d-dimensional standard logistic activation function if and only if $A = \mathfrak{M}_{a,d}$ (cf. Definition 1.2.1).*

1.2.7.1 Derivative of the standard logistic activation function

Proposition 1.2.20 (Logistic ODE). *Let a be the standard logistic activation function (cf. Definition 1.2.18). Then*

(i) it holds that $a\colon \mathbb{R} \to \mathbb{R}$ is infinitely often differentiable and

(ii) it holds for all $x \in \mathbb{R}$ that

$$a(0) = 1/2, \qquad a'(x) = a(x)(1 - a(x)) = a(x) - [a(x)]^2, \qquad \text{and} \tag{1.53}$$

$$a''(x) = a(x)(1 - a(x))(1 - 2\,a(x)) = 2[a(x)]^3 - 3[a(x)]^2 + a(x). \tag{1.54}$$

Proof of Proposition 1.2.20. Note that (1.52) implies item (i). Next observe that (1.52) ensures that for all $x \in \mathbb{R}$ it holds that

$$\begin{aligned} a'(x) &= \frac{\exp(-x)}{(1 + \exp(-x))^2} = a(x)\left(\frac{\exp(-x)}{1 + \exp(-x)}\right) \\ &= a(x)\left(\frac{1 + \exp(-x) - 1}{1 + \exp(-x)}\right) = a(x)\left(1 - \frac{1}{1 + \exp(-x)}\right) \\ &= a(x)(1 - a(x)). \end{aligned} \tag{1.55}$$

Hence, we obtain that for all $x \in \mathbb{R}$ it holds that

$$\begin{aligned}
a''(x) &= \big[a(x)(1-a(x))\big]' = a'(x)(1-a(x)) + a(x)(1-a(x))' \\
&= a'(x)(1-a(x)) - a(x)\,a'(x) = a'(x)(1-2\,a(x)) \\
&= a(x)(1-a(x))(1-2\,a(x)) \\
&= \big(a(x) - [a(x)]^2\big)(1-2\,a(x)) = a(x) - [a(x)]^2 - 2[a(x)]^2 + 2[a(x)]^3 \\
&= 2[a(x)]^3 - 3[a(x)]^2 + a(x).
\end{aligned} \tag{1.56}$$

This establishes item (ii). The proof of Proposition 1.2.20 is thus complete. □

1.2.7.2 Integral of the standard logistic activation function

Lemma 1.2.21 (Primitive of the standard logistic activation function)**.** *Let $\mathfrak{s}$ be the softplus activation function and let $\mathfrak{l}$ be the standard logistic activation function (cf. Definitions 1.2.11 and 1.2.18). Then it holds for all $x \in \mathbb{R}$ that*

$$\int_{-\infty}^{x} \mathfrak{l}(y)\,\mathrm{d}y = \int_{-\infty}^{x} \left(\frac{1}{1+e^{-y}}\right) \mathrm{d}y = \ln(1+\exp(x)) = \mathfrak{s}(x). \tag{1.57}$$

Proof of Lemma 1.2.21. Observe that (1.47) implies that for all $x \in \mathbb{R}$ it holds that

$$\mathfrak{s}'(x) = \left[\frac{1}{1+\exp(x)}\right] \exp(x) = \mathfrak{l}(x). \tag{1.58}$$

The fundamental theorem of calculus hence shows that for all $w, x \in \mathbb{R}$ with $w \leq x$ it holds that

$$\int_{w}^{x} \underbrace{\mathfrak{l}(y)}_{\geq 0}\,\mathrm{d}y = \mathfrak{s}(x) - \mathfrak{s}(w). \tag{1.59}$$

Combining this with the fact that $\lim_{w \to -\infty} \mathfrak{s}(w) = 0$ establishes (1.57). The proof of Lemma 1.2.21 is thus complete. □

1.2.8 Swish activation

Definition 1.2.22 (Swish activation function)**.** *Let $\beta \in \mathbb{R}$. Then we say that a is the swish activation function with parameter β if and only if it holds that $a \colon \mathbb{R} \to \mathbb{R}$ is the function from $\mathbb{R}$ to $\mathbb{R}$ which satisfies for all $x \in \mathbb{R}$ that*

$$a(x) = \frac{x}{1+\exp(-\beta x)}. \tag{1.60}$$

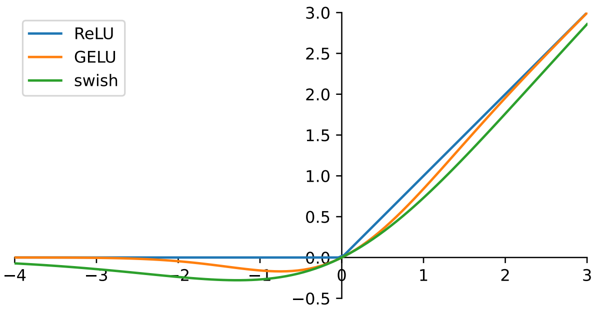

Figure 1.9 (plots/swish.pdf): A plot of the swish activation function, the GELU activation function, and the ReLU activation function

```python
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-4,3), (-.5,3))

x = np.linspace(-4, 3, 100)

ax.plot(x, tf.keras.activations.relu(x), label='ReLU')
ax.plot(x, tf.keras.activations.gelu(x), label='GELU')
ax.plot(x, tf.keras.activations.swish(x), label='swish')
ax.legend()

plt.savefig("../../plots/swish.pdf", bbox_inches='tight')
```

Source code 1.7 (code/activation_functions/swish_plot.py): PYTHON code used to create Figure 1.9

Lemma 1.2.23 (Relation between the swish activation function and the logistic activation function)**.** *Let $\beta \in \mathbb{R}$, let $\mathfrak{s}$ be the swish activation function with parameter* 1*, and let $\mathfrak{l}$ be the standard logistic activation function (cf. Definitions 1.2.18 and 1.2.22). Then it holds for all $x \in \mathbb{R}$ that*

$$\mathfrak{s}(x) = x\mathfrak{l}(\beta x). \tag{1.61}$$

Proof of Lemma 1.2.23. Observe that (1.60) and (1.52) establish (1.61). The proof of Lemma 1.2.23 is thus complete. $\square$

Definition 1.2.24 (Multidimensional swish activation functions)**.** *Let $d \in \mathbb{N}$ and let a be the swish activation function with parameter* 1 *(cf. Definition 1.2.22). Then we say that A is the d-dimensional swish activation function if and only if $A = \mathfrak{M}_{a,d}$ (cf. Definition 1.2.1).*

1.2.9 Hyperbolic tangent activation

Definition 1.2.25 (Hyperbolic tangent activation function)**.** *We denote by* tanh: $\mathbb{R} \to \mathbb{R}$ *the function which satisfies for all* $x \in \mathbb{R}$ *that*

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \tag{1.62}$$

and we call tanh *the hyperbolic tangent activation function (we call* tanh *the hyperbolic tangent).*

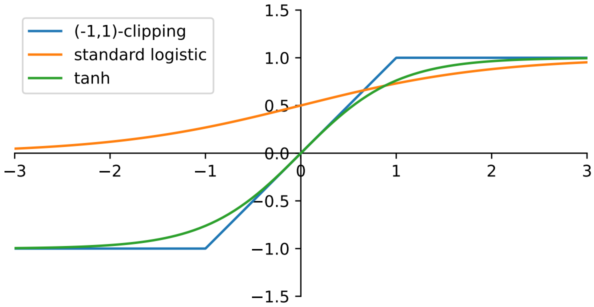

Figure 1.10 (plots/tanh.pdf): A plot of the hyperbolic tangent, the $(-1, 1)$-clipping activation function, and the standard logistic activation function

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-3,3), (-1.5,1.5))

x = np.linspace(-3, 3, 100)

ax.plot(x, tf.keras.activations.relu(x+1, max_value=2)-1,
        label='(-1,1)-clipping')
ax.plot(x, tf.keras.activations.sigmoid(x),
        label='standard logistic')
ax.plot(x, tf.keras.activations.tanh(x), label='tanh')
ax.legend()

plt.savefig("../../plots/tanh.pdf", bbox_inches='tight')
```

Source code 1.8 (code/activation_functions/tanh_plot.py): Python code used to create Figure 1.10

Definition 1.2.26 (Multidimensional hyperbolic tangent activation functions)**.** *Let $d \in \mathbb{N}$. Then we say that A is the d-dimensional hyperbolic tangent activation function if and only if $A = \mathfrak{M}_{\tanh,d}$ (cf. Definitions 1.2.1 and 1.2.25).*

Lemma 1.2.27. *Let a be the standard logistic activation function (cf. Definition 1.2.18). Then it holds for all $x \in \mathbb{R}$ that*

$$\tanh(x) = 2\,a(2x) - 1 \tag{1.63}$$

(cf. Definitions 1.2.18 and 1.2.25).

Proof of Lemma 1.2.27. Observe that (1.52) and (1.62) ensure that for all $x \in \mathbb{R}$ it holds that

$$\begin{aligned} 2\,a(2x) - 1 &= 2\left(\frac{\exp(2x)}{\exp(2x)+1}\right) - 1 = \frac{2\exp(2x) - (\exp(2x)+1)}{\exp(2x)+1} \\ &= \frac{\exp(2x)-1}{\exp(2x)+1} = \frac{\exp(x)(\exp(x)-\exp(-x))}{\exp(x)(\exp(x)+\exp(-x))} \\ &= \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)} = \tanh(x). \end{aligned} \tag{1.64}$$

The proof of Lemma 1.2.27 is thus complete. □

Exercise 1.2.16. Let a be the standard logistic activation function (cf. Definition 1.2.18). Prove or disprove the following statement: There exists $L \in \{2, 3, \ldots\}$, $\mathfrak{d}, l_1, l_2, \ldots, l_{L-1} \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$ with $\mathfrak{d} \geq 2\,l_1 + \left[\sum_{k=2}^{L-1} l_k(l_{k-1}+1)\right] + (l_{L-1}+1)$ such that for all $x \in \mathbb{R}$ it holds that

$$\big(\mathcal{N}^{\theta,1}_{\mathfrak{M}_{a,l_1},\mathfrak{M}_{a,l_2},\ldots,\mathfrak{M}_{a,l_{L-1}},\mathrm{id}_{\mathbb{R}}}\big)(x) = \tanh(x) \tag{1.65}$$

(cf. Definitions 1.1.3, 1.2.1, and 1.2.25).

1.2.10 Softsign activation

Definition 1.2.28 (Softsign activation function)**.** *We say that a is the softsign activation function if and only if it holds that $a\colon \mathbb{R} \to \mathbb{R}$ is the function from $\mathbb{R}$ to $\mathbb{R}$ which satisfies for all $x \in \mathbb{R}$ that*

$$a(x) = \frac{x}{|x|+1}. \tag{1.66}$$

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

```

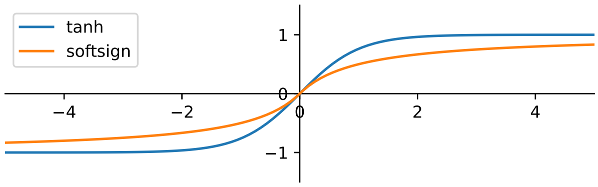

Figure 1.11 (plots/softsign.pdf): A plot of the softsign activation function and the hyperbolic tangent

```
ax = plot_util.setup_axis((-5,5), (-1.5,1.5))

x = np.linspace(-5, 5, 100)

ax.plot(x, tf.keras.activations.tanh(x), label='tanh')
ax.plot(x, tf.keras.activations.softsign(x), label='softsign')
ax.legend()

plt.savefig("../../plots/softsign.pdf", bbox_inches='tight')
```

Source code 1.9 (code/activation_functions/softsign_plot.py): PYTHON code used to create Figure 1.11

Definition 1.2.29 (Multidimensional softsign activation functions)**.** *Let $d \in \mathbb{N}$ and let a be the softsign activation function (cf. Definition 1.2.28). Then we say that A is the d-dimensional softsign activation function if and only if $A = \mathfrak{M}_{a,d}$ (cf. Definition 1.2.1).*

1.2.11 Leaky rectified linear unit (leaky ReLU) activation

Definition 1.2.30 (Leaky ReLU activation function)**.** *Let $\gamma \in [0, \infty)$. Then we say that a is the leaky ReLU activation function with leak factor γ if and only if it holds that $a \colon \mathbb{R} \to \mathbb{R}$ is the function from $\mathbb{R}$ to $\mathbb{R}$ which satisfies for all $x \in \mathbb{R}$ that*

$$a(x) = \begin{cases} x & : x > 0 \\ \gamma x & : x \leq 0. \end{cases} \tag{1.67}$$

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-2,2), (-.5,2))

x = np.linspace(-2, 2, 100)
```

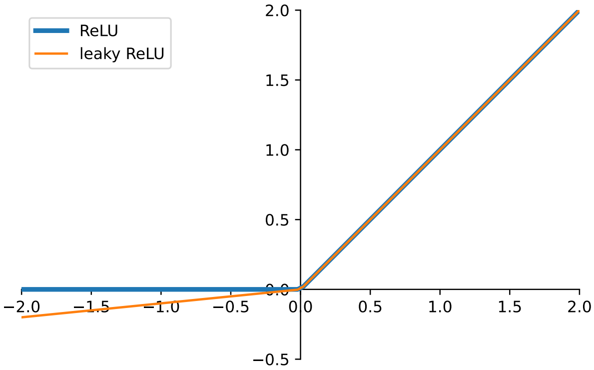

Figure 1.12 (plots/leaky_relu.pdf): A plot of the leaky ReLU activation function with leak factor $1/10$ and the ReLU activation function

```

ax.plot(x, tf.keras.activations.relu(x), linewidth=3, label='ReLU')
ax.plot(x, tf.keras.activations.relu(x, alpha=0.1),
        label='leaky ReLU')
ax.legend()

plt.savefig("../../plots/leaky_relu.pdf", bbox_inches='tight')
```

Source code 1.10 (code/activation_functions/leaky_relu_plot.py): Python code used to create Figure 1.12

Lemma 1.2.31. *Let $\gamma \in [0, 1]$ and let $a : \mathbb{R} \to \mathbb{R}$ be a function. Then a is the leaky ReLU activation function with leak factor γ if and only if it holds for all $x \in \mathbb{R}$ that*

$$a(x) = \max\{x, \gamma x\} \tag{1.68}$$

(cf. Definition 1.2.30).

Proof of Lemma 1.2.31. Note that the fact that $\gamma \leq 1$ and (1.67) establish (1.68). The proof of Lemma 1.2.31 is thus complete. □

Lemma 1.2.32. *Let $u, \beta \in \mathbb{R}$, $v \in (u, \infty)$, $\alpha \in (-\infty, 0]$, let a_1 be the softplus activation function, let a_2 be the GELU activation function, let a_3 be the standard logistic activation function, let a_4 be the swish activation function with parameter β, let a_5 be the softsign activation function, and let l be the leaky ReLU activation function with leaky parameter γ (cf. Definitions 1.2.11, 1.2.15, 1.2.18, 1.2.22, 1.2.28, and 1.2.30). Then*

(i) *it holds for all $f \in \{\mathfrak{r}, \mathfrak{c}_{u,v}, \tanh, a_1, a_2, \dots, a_5\}$ that $\limsup_{x \to -\infty} |f'(x)| = 0$ and* *(ii) it holds that* $\lim_{x\to-\infty} l'(x) = \gamma$

(cf. Definitions 1.2.4, 1.2.9, and 1.2.25).

Proof of Lemma 1.2.32. Note that (1.26), (1.45), (1.47), (1.51), (1.52), (1.60), (1.62), and (1.66) prove item (i). Observe that (1.67) establishes item (ii). The proof of Lemma 1.2.32 is thus complete. □

Definition 1.2.33 (Multidimensional leaky ReLU activation function)**.** *Let* $d \in \mathbb{N}$*,* $\gamma \in [0, \infty)$ *and let* a *be the leaky ReLU activation function with leak factor* γ *(cf. Definition 1.2.30). Then we say that* A *is the* d*-dimensional leaky ReLU activation function with leak factor* γ *if and only if* $A = \mathfrak{M}_{a,d}$ *(cf. Definition 1.2.1).*

1.2.12 Exponential linear unit (ELU) activation

Another popular activation function is the so-called *exponential linear unit* (ELU) activation function which has been introduced in Clevert et al. [83]. This activation function is the subject of the next notion.

Definition 1.2.34 (ELU activation function)**.** *Let* $\gamma \in (-\infty, 0]$*. Then we say that* a *is the ELU activation function with asymptotic* γ *if and only if it holds that* $a\colon \mathbb{R} \to \mathbb{R}$ *is the function from* $\mathbb{R}$ *to* $\mathbb{R}$ *which satisfies for all* $x \in \mathbb{R}$ *that*

$$a(x) = \begin{cases} x & : x > 0 \\ \gamma(1 - \exp(x)) & : x \leq 0. \end{cases} \tag{1.69}$$

```python
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-2,2), (-1,2))

x = np.linspace(-2, 2, 100)

ax.plot(x, tf.keras.activations.relu(x), linewidth=3, label='ReLU')
ax.plot(x, tf.keras.activations.relu(x, alpha=0.1), linewidth=2,
    label='leaky ReLU')
ax.plot(x, tf.keras.activations.elu(x), linewidth=0.9, label='ELU')
ax.legend()

plt.savefig("../../plots/elu.pdf", bbox_inches='tight')
```

Source code 1.11 (code/activation_functions/elu_plot.py): Python code used to create Figure 1.13

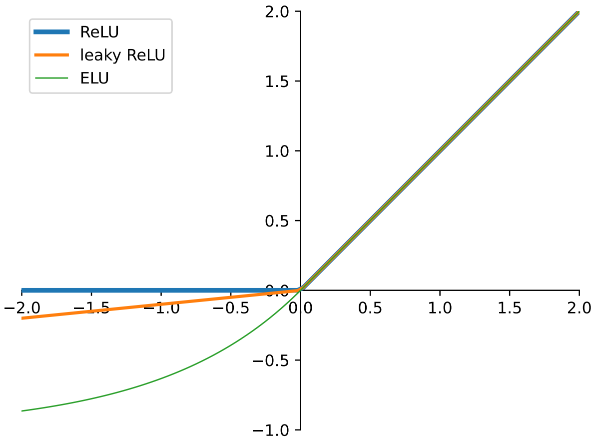

Figure 1.13 (plots/elu.pdf): A plot of the ELU activation function with asymptotic -1, the leaky ReLU activation function with leak factor $1/10$, and the ReLU activation function

Lemma 1.2.35. *Let $\gamma \in (-\infty, 0]$ and let a be the ELU activation function with asymptotic γ (cf. Definition 1.2.34). Then*

$$\limsup_{x \to -\infty} a(x) = \liminf_{x \to -\infty} a(x) = \gamma. \tag{1.70}$$

Proof of Lemma 1.2.35. Observe that (1.69) establishes (1.70). The proof of Lemma 1.2.35 is thus complete. □

Definition 1.2.36 (Multidimensional ELU activation function)**.** *Let $d \in \mathbb{N}$, $\gamma \in (-\infty, 0]$ and let a be the ELU activation function with asymptotic γ (cf. Definition 1.2.34). Then we say that A is the d-dimensional ELU activation function with asymptotic γ if and only if $A = \mathfrak{M}_{a,d}$ (cf. Definition 1.2.1).*

1.2.13 Rectified power unit (RePU) activation

Another popular activation function is the so-called *rectified power unit* (RePU) activation function. This concept is the subject of the next notion.

Definition 1.2.37 (RePU activation function)**.** *Let $p \in \mathbb{N}$. Then we say that a is the RePU activation function with power p if and only if it holds that $a\colon \mathbb{R} \to \mathbb{R}$ is the function from $\mathbb{R}$ to $\mathbb{R}$ which satisfies for all $x \in \mathbb{R}$ that*

$$a(x) = (\max\{x, 0\})^p. \tag{1.71}$$

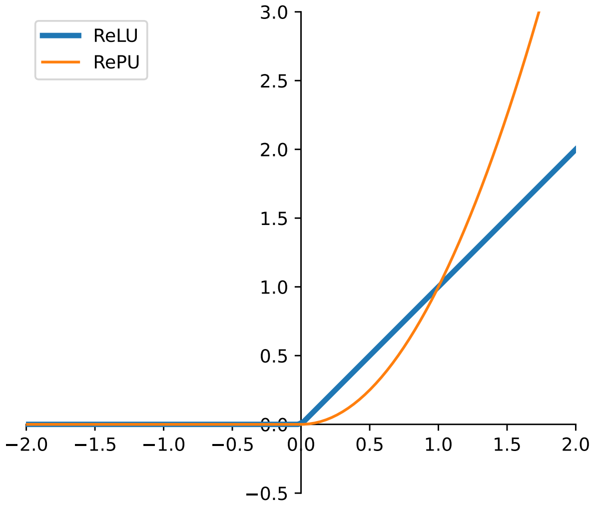

Figure 1.14 (plots/repu.pdf): A plot of the RePU activation function with power 2 and the ReLU activation function

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-2,2), (-.5,3))
ax.set_ylim(-.5, 3)

x = np.linspace(-2, 2, 100)

ax.plot(x, tf.keras.activations.relu(x), linewidth=3, label='ReLU')
ax.plot(x, tf.keras.activations.relu(x)**2, label='RePU')
ax.legend()

plt.savefig("../../plots/repu.pdf", bbox_inches='tight')
```

Source code 1.12 (code/activation_functions/repu_plot.py): Python code used to create Figure 1.14

Definition 1.2.38 (Multidimensional RePU activation function)**.** *Let $d, p \in \mathbb{N}$ and let a be the RePU activation function with power p (cf. Definition 1.2.37). Then we say that A is the d-dimensional RePU activation function with power p if and only if $A = \mathfrak{M}_{a,d}$ (cf. Definition 1.2.1).*

1.2.14 Sine activation

The sine function has been proposed as activation function in Sitzmann et al. [380]. This is formulated in the next notion.

Definition 1.2.39 (Sine activation function)**.** *We say that a is the sine activation function if and only if it holds that $a\colon \mathbb{R} \to \mathbb{R}$ is the function from $\mathbb{R}$ to $\mathbb{R}$ which satisfies for all $x \in \mathbb{R}$ that*

$$a(x) = \sin(x). \tag{1.72}$$

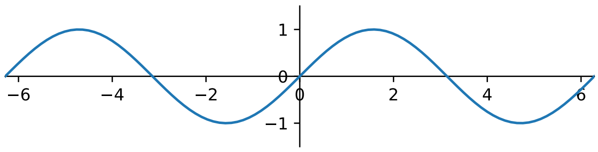

Figure 1.15 (plots/sine.pdf): A plot of the sine activation function

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-2*np.pi,2*np.pi), (-1.5,1.5))

x = np.linspace(-2*np.pi, 2*np.pi, 100)

ax.plot(x, np.sin(x))

plt.savefig("../../plots/sine.pdf", bbox_inches='tight')
```

Source code 1.13 (code/activation_functions/sine_plot.py): Python code used to create Figure 1.15

Definition 1.2.40 (Multidimensional sine activation functions)**.** *Let $d \in \mathbb{N}$ and let a be the sine activation function (cf. Definition 1.2.39). Then we say that A is the d-dimensional sine activation function if and only if $A = \mathfrak{M}_{a,d}$ (cf. Definition 1.2.1).*

1.2.15 Heaviside activation

Definition 1.2.41 (Heaviside activation function)**.** *We say that a is the Heaviside activation function (we say that a is the Heaviside step function, we say that a is the unit step function)* *if and only if it holds that* $a\colon \mathbb{R} \to \mathbb{R}$ *is the function from* $\mathbb{R}$ *to* $\mathbb{R}$ *which satisfies for all* $x \in \mathbb{R}$ *that*

$$a(x) = \mathbb{1}_{[0,\infty)}(x) = \begin{cases} 1 & : x \geq 0 \\ 0 & : x < 0. \end{cases} \tag{1.73}$$

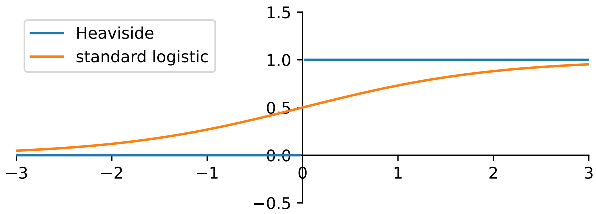

Figure 1.16 (plots/heaviside.pdf): A plot of the Heaviside activation function and the standard logistic activation function

```
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import plot_util

ax = plot_util.setup_axis((-3,3), (-.5,1.5))

x = np.linspace(-3, 3, 100)

ax.plot(x[0:50], [0]*50, 'C0')
ax.plot(x[50:100], [1]*50, 'C0', label='Heaviside')
ax.plot(x, tf.keras.activations.sigmoid(x), 'C1',
        label='standard logistic')
ax.legend()

plt.savefig("../../plots/heaviside.pdf", bbox_inches='tight')
```

Source code 1.14 (code/activation_functions/heaviside_plot.py): PYTHON code used to create Figure 1.16

Definition 1.2.42 (Multidimensional Heaviside activation functions). *Let* $d \in \mathbb{N}$ *and let* a *be the Heaviside activation function (cf. Definition 1.2.41). Then we say that* A *is the* d*-dimensional Heaviside activation function (we say that* A *is the* d*-dimensional Heaviside step function, we say that* A *is the* d*-dimensional unit step function) if and only if* $A = \mathfrak{M}_{a,d}$ *(cf. Definition 1.2.1).*

1.2.16 Softmax activation

Definition 1.2.43 (Softmax activation function)**.** *Let $d \in \mathbb{N}$. Then we say that A is the d-dimensional softmax activation function if and only if it holds that $A\colon \mathbb{R}^d \to \mathbb{R}^d$ is the function from $\mathbb{R}^d$ to $\mathbb{R}^d$ which satisfies for all $x = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$ that*

$$A(x) = \left(\frac{\exp(x_1)}{\left(\sum_{i=1}^d \exp(x_i)\right)}, \frac{\exp(x_2)}{\left(\sum_{i=1}^d \exp(x_i)\right)}, \ldots, \frac{\exp(x_d)}{\left(\sum_{i=1}^d \exp(x_i)\right)} \right). \tag{1.74}$$

Lemma 1.2.44. *Let $d \in \mathbb{N}$ and let $A = (A_1, A_2, \ldots, A_d)$ be the d-dimensional softmax activation function (cf. Definition 1.2.43). Then*

(i) it holds for all $x \in \mathbb{R}^d$, $k \in \{1, 2, \ldots, d\}$ that $A_k(x) \in (0, 1]$ and

(ii) it holds for all $x \in \mathbb{R}^d$ that

$$\sum_{k=1}^d A_k(x) = 1. \tag{1.75}$$

tum

(cf. Definition 1.2.43).

Proof of Lemma 1.2.44. Observe that (1.74) demonstrates that for all $x = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$ it holds that

$$\sum_{k=1}^d A_k(x) = \sum_{k=1}^d \frac{\exp(x_k)}{\left(\sum_{i=1}^d \exp(x_i)\right)} = \frac{\sum_{k=1}^d \exp(x_k)}{\sum_{i=1}^d \exp(x_i)} = 1. \tag{1.76}$$

The proof of Lemma 1.2.44 is thus complete. □

1.3 Fully-connected feedforward ANNs (structured description)

In this section we present an alternative way to describe the fully-connected feedforward ANNs introduced in Section 1.1 above. Roughly speaking, in Section 1.1 above we defined a *vectorized description* of fully-connected feedforward ANNs in the sense that the trainable parameters of a fully-connected feedforward ANN are represented by the components of a single Euclidean vector (cf. Definition 1.1.3 above). In this section we introduce a *structured description* of fully-connected feedforward ANNs in which the trainable parameters of a fully-connected feedforward ANN are represented by a tuple of matrix-vector pairs corresponding to the weight matrices and bias vectors of the fully-connected feedforward ANNs (cf. Definitions 1.3.1 and 1.3.4 below).

1.3.1 Structured description of fully-connected feedforward ANNs

Definition 1.3.1 (Structured description of fully-connected feedforward ANNs)**.** *We denote by* $\mathbf{N}$ *the set given by*

$$\mathbf{N} = \bigcup_{L\in\mathbb{N}} \bigcup_{l_0,l_1,\dots,l_L\in\mathbb{N}} \Big(\times_{k=1}^{L} (\mathbb{R}^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k}) \Big), \tag{1.77}$$

for every $L \in \mathbb{N}$*,* $l_0, l_1, \dots, l_L \in \mathbb{N}$*,* $\Phi \in \big(\times_{k=1}^{L} (\mathbb{R}^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k})\big) \subseteq \mathbf{N}$ *we denote by* $\mathcal{P}(\Phi), \mathcal{L}(\Phi), \mathcal{I}(\Phi), \mathcal{O}(\Phi) \in \mathbb{N}$*,* $\mathcal{H}(\Phi) \in \mathbb{N}_0$ *the numbers given by*

$$\mathcal{P}(\Phi) = \textstyle\sum_{k=1}^{L} l_k(l_{k-1}+1), \;\; \mathcal{L}(\Phi) = L, \;\; \mathcal{I}(\Phi) = l_0, \;\; \mathcal{O}(\Phi) = l_L, \;\; \text{and} \;\; \mathcal{H}(\Phi) = L-1, \tag{1.78}$$

for every $n \in \mathbb{N}_0$*,* $L \in \mathbb{N}$*,* $l_0, l_1, \dots, l_L \in \mathbb{N}$*,* $\Phi \in \big(\times_{k=1}^{L} (\mathbb{R}^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k})\big) \subseteq \mathbf{N}$ *we denote by* $\mathbb{D}_n(\Phi) \in \mathbb{N}_0$ *the number given by*

$$\mathbb{D}_n(\Phi) = \begin{cases} l_n & : n \leq L \\ 0 & : n > L, \end{cases} \tag{1.79}$$

for every $\Phi \in \mathbf{N}$ *we denote by* $\mathcal{D}(\Phi) \in \mathbb{N}^{\mathcal{L}(\Phi)+1}$ *the tuple given by*

$$\mathcal{D}(\Phi) = (\mathbb{D}_0(\Phi), \mathbb{D}_1(\Phi), \dots, \mathbb{D}_{\mathcal{L}(\Phi)}(\Phi)), \tag{1.80}$$

and for every $L \in \mathbb{N}$*,* $l_0, l_1, \dots, l_L \in \mathbb{N}$*,* $\Phi = ((W_1, B_1), \dots, (W_L, B_L)) \in \big(\times_{k=1}^{L} (\mathbb{R}^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k})\big) \subseteq \mathbf{N}$*,* $n \in \{1, 2, \dots, L\}$ *we denote by* $\mathcal{W}_{n,\Phi} \in \mathbb{R}^{l_n\times l_{n-1}}$*,* $\mathcal{B}_{n,\Phi} \in \mathbb{R}^{l_n}$ *the matrix and the vector given by*

$$\mathcal{W}_{n,\Phi} = W_n \qquad \text{and} \qquad \mathcal{B}_{n,\Phi} = B_n. \tag{1.81}$$

Definition 1.3.2 (Fully-connected feedforward ANNs)**.** *We say that* Φ *is a fully-connected feedforward ANN if and only if it holds that*

$$\Phi \in \mathbf{N} \tag{1.82}$$

(cf. Definition 1.3.1).

Lemma 1.3.3. *Let* $\Phi \in \mathbf{N}$ *(cf. Definition 1.3.1). Then*

(i) it holds that $\mathcal{D}(\Phi) \in \mathbb{N}^{\mathcal{L}(\Phi)+1}$*,*

(ii) it holds that

$$\mathcal{I}(\Phi) = \mathbb{D}_0(\Phi) \qquad \text{and} \qquad \mathcal{O}(\Phi) = \mathbb{D}_{\mathcal{L}(\Phi)}(\Phi), \tag{1.83}$$

and

(iii) it holds for all $n \in \{1, 2, \dots, \mathcal{L}(\Phi)\}$ that

$$\mathcal{W}_{n,\Phi} \in \mathbb{R}^{\mathbb{D}_n(\Phi) \times \mathbb{D}_{n-1}(\Phi)} \qquad \text{and} \qquad \mathcal{B}_{n,\Phi} \in \mathbb{R}^{\mathbb{D}_n(\Phi)}. \tag{1.84}$$

.

Proof of Lemma 1.3.3. Note that the assumption that

$$\Phi \in \mathbf{N} = \textstyle\bigcup_{L \in \mathbb{N}} \bigcup_{(l_0, l_1, \dots, l_L) \in \mathbb{N}^{L+1}} \big(\bigtimes_{k=1}^{L} (\mathbb{R}^{l_k \times l_{k-1}} \times \mathbb{R}^{l_k}) \big)$$

ensures that there exist $L \in \mathbb{N}$, $l_0, l_1, \dots, l_L \in \mathbb{N}$ which satisfy that

$$\Phi \in \big(\textstyle\bigtimes_{k=1}^{L} (\mathbb{R}^{l_k \times l_{k-1}} \times \mathbb{R}^{l_k}) \big). \tag{1.85}$$

Observe that (1.85), (1.78), and (1.79) imply that

$$\mathcal{L}(\Phi) = L, \qquad \mathcal{I}(\Phi) = l_0 = \mathbb{D}_0(\Phi), \qquad \text{and} \qquad \mathcal{O}(\Phi) = l_L = \mathbb{D}_L(\Phi). \tag{1.86}$$

This shows that

$$\mathcal{D}(\Phi) = (l_0, l_1, \dots, l_L) \in \mathbb{N}^{L+1} = \mathbb{N}^{\mathcal{L}(\Phi)+1}. \tag{1.87}$$

Next note that (1.85), (1.79), and (1.81) ensure that for all $n \in \{1, 2, \dots, \mathcal{L}(\Phi)\}$ it holds that

$$\mathcal{W}_{n,\Phi} \in \mathbb{R}^{l_n \times l_{n-1}} = \mathbb{R}^{\mathbb{D}_n(\Phi) \times \mathbb{D}_{n-1}(\Phi)} \qquad \text{and} \qquad \mathcal{B}_{n,\Phi} \in \mathbb{R}^{l_n} = \mathbb{R}^{\mathbb{D}_n(\Phi)}. \tag{1.88}$$

The proof of Lemma 1.3.3 is thus complete. □

1.3.2 Realizations of fully-connected feedforward ANNs

Definition 1.3.4 (Realizations of fully-connected feedforward ANNs)**.** *Let $\Phi \in \mathbf{N}$ and let $a\colon \mathbb{R} \to \mathbb{R}$ be a function (cf. Definition 1.3.1). Then we denote by*

$$\mathcal{R}_a^{\mathbf{N}}(\Phi)\colon \mathbb{R}^{\mathcal{I}(\Phi)} \to \mathbb{R}^{\mathcal{O}(\Phi)} \tag{1.89}$$

the function which satisfies for all $x_0 \in \mathbb{R}^{\mathbb{D}_0(\Phi)}$, $x_1 \in \mathbb{R}^{\mathbb{D}_1(\Phi)}$, $\dots$, $x_{\mathcal{L}(\Phi)} \in \mathbb{R}^{\mathbb{D}_{\mathcal{L}(\Phi)}(\Phi)}$ with

$$\forall\, k \in \{1, 2, \dots, \mathcal{L}(\Phi)\}\colon x_k = \mathfrak{M}_{a \mathbb{1}_{(0,\mathcal{L}(\Phi))}(k) + \mathrm{id}_{\mathbb{R}} \mathbb{1}_{\{\mathcal{L}(\Phi)\}}(k), \mathbb{D}_k(\Phi)}(\mathcal{W}_{k,\Phi} x_{k-1} + \mathcal{B}_{k,\Phi}) \tag{1.90}$$

that

$$(\mathcal{R}_a^{\mathbf{N}}(\Phi))(x_0) = x_{\mathcal{L}(\Phi)} \tag{1.91}$$

and we call $\mathcal{R}_a^{\mathbf{N}}(\Phi)$ the realization function of the fully-connected feedforward ANN Φ with activation function a (we call $\mathcal{R}_a^{\mathbf{N}}(\Phi)$ the realization of the fully-connected feedforward ANN Φ with activation a) (cf. Definition 1.2.1).

Exercise 1.3.1. Let

$$\Phi = ((W_1, B_1), (W_2, B_2), (W_3, B_3)) \in (\mathbb{R}^{2\times 1} \times \mathbb{R}^2) \times (\mathbb{R}^{3\times 2} \times \mathbb{R}^3) \times (\mathbb{R}^{1\times 3} \times \mathbb{R}^1) \tag{1.92}$$

satisfy

$$W_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \qquad B_1 = \begin{pmatrix} 3 \\ 4 \end{pmatrix}, \qquad W_2 = \begin{pmatrix} -1 & 2 \\ 3 & -4 \\ -5 & 6 \end{pmatrix}, \qquad B_2 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \tag{1.93}$$

$$W_3 = \begin{pmatrix} -1 & 1 & -1 \end{pmatrix}, \qquad \text{and} \qquad B_3 = \begin{pmatrix} -4 \end{pmatrix}. \tag{1.94}$$

Prove or disprove the following statement: It holds that

$$(\mathcal{R}^{\mathbf{N}}_{\mathfrak{r}}(\Phi))(-1) = 0 \tag{1.95}$$

(cf. Definitions 1.2.4 and 1.3.4).

Exercise 1.3.2. Let a be the standard logistic activation function (cf. Definition 1.2.18). Prove or disprove the following statement: There exists $\Phi \in \mathbf{N}$ such that

$$\mathcal{R}^{\mathbf{N}}_{\tanh}(\Phi) = a \tag{1.96}$$

(cf. Definitions 1.2.25, 1.3.1, and 1.3.4).

```python
import torch
import torch.nn as nn
import torch.nn.functional as F


# To define a neural network, we define a class that inherits from
# torch.nn.Module
class FullyConnectedANN(nn.Module):
    def __init__(self):
        super().__init__()
        # In the constructor, we define the weights and biases.
        # Wrapping the tensors in torch.nn.Parameter objects tells
        # PyTorch that these are parameters that should be
        # optimized during training.
        self.W1 = nn.Parameter(
            torch.Tensor([[1, 0], [0, -1], [-2, 2]])
        )
        self.B1 = nn.Parameter(torch.Tensor([0, 2, -1]))
        self.W2 = nn.Parameter(torch.Tensor([[1, -2, 3]]))
        self.B2 = nn.Parameter(torch.Tensor([1]))

    # The realization function of the network
```

```
    def forward(self, x0):
        x1 = F.relu(self.W1 @ x0 + self.B1)
        x2 = self.W2 @ x1 + self.B2
        return x2


model = FullyConnectedANN()

x0 = torch.Tensor([1, 2])
# Print the output of the realization function for input x0
print(model.forward(x0))

# As a consequence of inheriting from torch.nn.Module we can just
# "call" the model itself (which will call the forward method
# implicitly)
print(model(x0))

# Wrapping a tensor in a Parameter object and assigning it to an
# instance variable of the Module makes PyTorch register it as a
# parameter. We can access all parameters via the parameters
# method.
for p in model.parameters():
    print(p)
```

Source code 1.15 (code/fc-ann-manual.py): PYTHON code for implementing a fully-connected feedforward ANN in PYTORCH. The model created here represents the fully-connected feedforward ANN $\left(\left(\left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \\ -2 & 2 \end{smallmatrix}\right), \left(\begin{smallmatrix} 0 \\ 2 \\ -1 \end{smallmatrix}\right)\right), \left(\left(\begin{smallmatrix} 1 & -2 & 3 \end{smallmatrix}\right), \left(\begin{smallmatrix} 1 \end{smallmatrix}\right)\right)\right) \in (\mathbb{R}^{3\times 2} \times \mathbb{R}^3) \times (\mathbb{R}^{1\times 3} \times \mathbb{R}^1) \subseteq \mathbf{N}$ using the ReLU activation function after the hidden layer.

```
import torch
import torch.nn as nn


class FullyConnectedANN(nn.Module):
    def __init__(self):
        super().__init__()
        # Define the layers of the network in terms of Modules.
        # nn.Linear(3, 20) represents an affine function defined
        # by a 20x3 weight matrix and a 20-dimensional bias vector.
        self.affine1 = nn.Linear(3, 20)
        # The torch.nn.ReLU class simply wraps the
        # torch.nn.functional.relu function as a Module.
        self.activation1 = nn.ReLU()
        self.affine2 = nn.Linear(20, 30)
        self.activation2 = nn.ReLU()
        self.affine3 = nn.Linear(30, 1)

```

```
    def forward(self, x0):
        x1 = self.activation1(self.affine1(x0))
        x2 = self.activation2(self.affine2(x1))
        x3 = self.affine3(x2)
        return x3


model = FullyConnectedANN()

x0 = torch.Tensor([1, 2, 3])
print(model(x0))

# Assigning a Module to an instance variable of a Module registers
# all of the former's parameters as parameters of the latter
for p in model.parameters():
    print(p)
```

Source code 1.16 (code/fc-ann.py): Python code for implementing a fully-connected feedforward ANN in PyTorch. The model implemented here represents a fully-connected feedforward ANN with two hidden layers, 3 neurons in the input layer, 20 neurons in the first hidden layer, 30 neurons in the second hidden layer, and 1 neuron in the output layer. Unlike Source code 1.15, this code uses the `torch.nn.Linear` class to represent the affine transformations.

```
import torch
import torch.nn as nn

# A Module whose forward method is simply a composition of Modules
# can be represented using the torch.nn.Sequential class
model = nn.Sequential(
    nn.Linear(3, 20),
    nn.ReLU(),
    nn.Linear(20, 30),
    nn.ReLU(),
    nn.Linear(30, 1),
)

# Prints a summary of the model architecture
print(model)

x0 = torch.Tensor([1, 2, 3])
print(model(x0))
```

Source code 1.17 (code/fc-ann2.py): Python code for creating a fully-connected feedforward ANN in PyTorch. This creates the same model as Source code 1.16 but uses the `torch.nn.Sequential` class instead of defining a new subclass of `torch.nn.Module`.

1.3.3 On the connection to the vectorized description

Definition 1.3.5 (Transformation from the structured to the vectorized description of fully-connected feedforward ANNs)**.** *We denote by* $\mathcal{T}\colon \mathbf{N} \to \big(\bigcup_{d\in\mathbb{N}} \mathbb{R}^d\big)$ *the function which satisfies for all* $\Phi \in \mathbf{N}$, $k \in \{1,2,\dots,\mathcal{L}(\Phi)\}$, $d \in \mathbb{N}$, $\theta = (\theta_1,\theta_2,\dots,\theta_d) \in \mathbb{R}^d$ *with* $\mathcal{T}(\Phi) = \theta$ *that*

$$d = \mathcal{P}(\Phi), \qquad \mathcal{B}_{k,\Phi} = \begin{pmatrix} \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+l_k l_{k-1}+1} \\ \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+l_k l_{k-1}+2} \\ \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+l_k l_{k-1}+3} \\ \vdots \\ \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+l_k l_{k-1}+l_k} \end{pmatrix}, \qquad \text{and} \qquad \mathcal{W}_{k,\Phi} =$$
$$\begin{pmatrix} \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+1} & \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+2} & \cdots & \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+l_{k-1}} \\ \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+l_{k-1}+1} & \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+l_{k-1}+2} & \cdots & \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+2l_{k-1}} \\ \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+2l_{k-1}+1} & \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+2l_{k-1}+2} & \cdots & \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+3l_{k-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+(l_k-1)l_{k-1}+1} & \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+(l_k-1)l_{k-1}+2} & \cdots & \theta_{(\sum_{i=1}^{k-1} l_i(l_{i-1}+1))+l_k l_{k-1}} \end{pmatrix} \tag{1.97}$$

(cf. Definition 1.3.1).

Lemma 1.3.6. *Let* $\Phi \in (\mathbb{R}^{3\times 3} \times \mathbb{R}^3) \times (\mathbb{R}^{2\times 3} \times \mathbb{R}^2)$ *satisfy*

$$\Phi = \left(\left(\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \begin{pmatrix} 10 \\ 11 \\ 12 \end{pmatrix} \right), \left(\begin{pmatrix} 13 & 14 & 15 \\ 16 & 17 & 18 \end{pmatrix}, \begin{pmatrix} 19 \\ 20 \end{pmatrix} \right) \right). \tag{1.98}$$

Then $\mathcal{T}(\Phi) = (1,2,3,\dots,19,20) \in \mathbb{R}^{20}$.

Proof of Lemma 1.3.6. Observe that (1.97) establishes (1.98). The proof of Lemma 1.3.6 is thus complete. □

Lemma 1.3.7. *Let* $a, b \in \mathbb{N}$, $W = (W_{i,j})_{(i,j)\in\{1,2,\dots,a\}\times\{1,2,\dots,b\}} \in \mathbb{R}^{a\times b}$, $B = (B_1, B_2, \dots, B_a) \in \mathbb{R}^a$. *Then*

$$\begin{aligned} &\mathcal{T}\big(((W,B))\big) \\ &= \big(W_{1,1}, W_{1,2}, \dots, W_{1,b}, W_{2,1}, W_{2,2}, \dots, W_{2,b}, \dots, W_{a,1}, W_{a,2}, \dots, W_{a,b}, B_1, B_2, \dots, B_a\big) \end{aligned} \tag{1.99}$$

(cf. Definition 1.3.5).

Proof of Lemma 1.3.7. Observe that (1.97) establishes (1.99). The proof of Lemma 1.3.7 is thus complete. □

Lemma 1.3.8. *Let $L \in \mathbb{N}$, $l_0, l_1, \ldots, l_L \in \mathbb{N}$ and for every $k \in \{1, 2, \ldots, L\}$ let $W_k = (W_{k,i,j})_{(i,j)\in\{1,2,\ldots,l_k\}\times\{1,2,\ldots,l_{k-1}\}} \in \mathbb{R}^{l_k \times l_{k-1}}$, $B_k = (B_{k,1}, B_{k,2}, \ldots, B_{k,l_k}) \in \mathbb{R}^{l_k}$. Then*

$$\begin{aligned}
&\mathcal{T}\Big(\big((W_1, B_1), (W_2, B_2), \ldots, (W_L, B_L)\big)\Big) \\
&= \Big(W_{1,1,1}, W_{1,1,2}, \ldots, W_{1,1,l_0}, \ldots, W_{1,l_1,1}, W_{1,l_1,2}, \ldots, W_{1,l_1,l_0}, B_{1,1}, B_{1,2}, \ldots, B_{1,l_1}, \\
&\quad W_{2,1,1}, W_{2,1,2}, \ldots, W_{2,1,l_1}, \ldots, W_{2,l_2,1}, W_{2,l_2,2}, \ldots, W_{2,l_2,l_1}, B_{2,1}, B_{2,2}, \ldots, B_{2,l_2}, \\
&\quad \ldots, \\
&\quad W_{L,1,1}, W_{L,1,2}, \ldots, W_{L,1,l_{L-1}}, \ldots, W_{L,l_L,1}, W_{L,l_L,2}, \ldots, W_{L,l_L,l_{L-1}}, B_{L,1}, B_{L,2}, \ldots, B_{L,l_L}\Big)
\end{aligned} \tag{1.100}$$

(cf. Definition 1.3.5).

Proof of Lemma 1.3.8. Note that (1.97) implies (1.100). The proof of Lemma 1.3.8 is thus complete. □

Exercise 1.3.3. Prove or disprove the following statement: The function $\mathcal{T}$ is injective (cf. Definition 1.3.5).

Exercise 1.3.4. Prove or disprove the following statement: The function $\mathcal{T}$ is surjective (cf. Definition 1.3.5).

Exercise 1.3.5. Prove or disprove the following statement: The function $\mathcal{T}$ is bijective (cf. Definition 1.3.5).

Proposition 1.3.9. *Let $a \in C(\mathbb{R}, \mathbb{R})$, $\Phi \in \mathbf{N}$ (cf. Definition 1.3.1). Then*

$$\mathcal{R}_a^{\mathbf{N}}(\Phi) = \begin{cases} \mathcal{N}^{\mathcal{T}(\Phi),\mathcal{I}(\Phi)}_{\mathrm{id}_{\mathbb{R}^{\mathcal{O}(\Phi)}}} & : \mathcal{H}(\Phi) = 0 \\ \mathcal{N}^{\mathcal{T}(\Phi),\mathcal{I}(\Phi)}_{\mathfrak{M}_{a,\mathbb{D}_1(\Phi)}, \mathfrak{M}_{a,\mathbb{D}_2(\Phi)}, \ldots, \mathfrak{M}_{a,\mathbb{D}_{\mathcal{H}(\Phi)}(\Phi)}, \mathrm{id}_{\mathbb{R}^{\mathcal{O}(\Phi)}}} & : \mathcal{H}(\Phi) > 0 \end{cases} \tag{1.101}$$

(cf. Definitions 1.1.3, 1.2.1, 1.3.4, and 1.3.5).

Proof of Proposition 1.3.9. Throughout this proof, let $L \in \mathbb{N}$, $l_0, l_1, \ldots, l_L \in \mathbb{N}$ satisfy that

$$\mathcal{L}(\Phi) = L \qquad \text{and} \qquad \mathcal{D}(\Phi) = (l_0, l_1, \ldots, l_L). \tag{1.102}$$

Note that (1.97) shows that for all $k \in \{1, 2, \ldots, L\}$, $x \in \mathbb{R}^{l_{k-1}}$ it holds that

$$\mathcal{W}_{k,\Phi}x + \mathcal{B}_{k,\Phi} = \big(\mathcal{A}^{\mathcal{T}(\Phi), \sum_{i=1}^{k-1} l_i(l_{i-1}+1)}_{l_k, l_{k-1}}\big)(x) \tag{1.103}$$

(cf. Definitions 1.1.1 and 1.3.5). This demonstrates that for all $x_0 \in \mathbb{R}^{l_0}$, $x_1 \in \mathbb{R}^{l_1}$, $\dots$, $x_{L-1} \in \mathbb{R}^{l_{L-1}}$ with $\forall\, k \in \{1, 2, \dots, L-1\}\colon x_k = \mathfrak{M}_{a,l_k}(\mathcal{W}_{k,\Phi} x_{k-1} + \mathcal{B}_{k,\Phi})$ it holds that

$$x_{L-1} = \begin{cases} x_0 & \colon L = 1 \\ \big(\mathfrak{M}_{a,l_{L-1}} \circ \mathcal{A}^{\mathcal{T}(\Phi), \sum_{i=1}^{L-2} l_i(l_{i-1}+1)}_{l_{L-1},l_{L-2}} & \colon L > 1 \\ \quad \circ \mathfrak{M}_{a,l_{L-2}} \circ \mathcal{A}^{\mathcal{T}(\Phi), \sum_{i=1}^{L-3} l_i(l_{i-1}+1)}_{l_{L-2},l_{L-3}} \circ \ldots \circ \mathfrak{M}_{a,l_1} \circ \mathcal{A}^{\mathcal{T}(\Phi),0}_{l_1,l_0}\big)(x_0) & \end{cases} \tag{1.104}$$

(cf. Definition 1.2.1). This, (1.103), (1.5), and (1.91) show that for all $x_0 \in \mathbb{R}^{l_0}$, $x_1 \in \mathbb{R}^{l_1}, \dots, x_L \in \mathbb{R}^{l_L}$ with $\forall\, k \in \{1, 2, \dots, L\}\colon x_k = \mathfrak{M}_{a\mathbb{1}_{(0,L)}(k) + \mathrm{id}_{\mathbb{R}}\, \mathbb{1}_{\{L\}}(k), l_k}(\mathcal{W}_{k,\Phi} x_{k-1} + \mathcal{B}_{k,\Phi})$ it holds that

$$\begin{split} \big(\mathcal{R}^{\mathbf{N}}_a(\Phi)\big)(x_0) = x_L = \mathcal{W}_{L,\Phi} x_{L-1} + \mathcal{B}_{L,\Phi} = \big(\mathcal{A}^{\mathcal{T}(\Phi), \sum_{i=1}^{L-1} l_i(l_{i-1}+1)}_{l_L,l_{L-1}}\big)(x_{L-1}) \\ = \begin{cases} \big(\mathcal{N}^{\mathcal{T}(\Phi),l_0}_{\mathrm{id}_{\mathbb{R}^{l_L}}}\big)(x_0) & \colon L = 1 \\ \big(\mathcal{N}^{\mathcal{T}(\Phi),l_0}_{\mathfrak{M}_{a,l_1}, \mathfrak{M}_{a,l_2}, \dots, \mathfrak{M}_{a,l_{L-1}}, \mathrm{id}_{\mathbb{R}^{l_L}}}\big)(x_0) & \colon L > 1 \end{cases} \end{split} \tag{1.105}$$

(cf. Definitions 1.1.3 and 1.3.4). The proof of Proposition 1.3.9 is thus complete. □

1.4 Convolutional ANNs (CNNs)

In this section we review CNNs, which are ANNs designed to process data with a spatial structure. In a broad sense, CNNs can be thought of as any ANNs involving a convolution operation (cf, for instance, Definition 1.4.1 below). Roughly speaking, convolutional operations allow CNNs to exploit spatial invariance of data by performing the same operations across different regions of an input data point. In principle, such convolution operations can be employed in combinations with other ANN architecture elements, such as fully-connected layers (cf., for example, Sections 1.1 and 1.3 above), residual layers (cf., for instance, Section 1.5 below), and recurrent structures (cf., for example, Section 1.6 below). However, for simplicity we introduce in this section in all mathematical details feedforward CNNs only involving convolutional layers based on the discrete convolution operation without *padding* (sometimes called *valid padding*) in Definition 1.4.1 (see Definitions 1.4.2 and 1.4.5 below). We refer, for instance, to [4, Section 12.5], [60, Chapter 16], [63, Section 4.2], [164, Chapter 9], and [36, Sectino 1.6.1] for other introductions on CNNs.

CNNs were introduced in LeCun et al. [262] for *computer vision* (CV) applications. The first successful modern CNN architecture is widely considered to be the *AlexNet* architecture proposed in Krizhevsky et al. [257]. A few other very successful early CNN architecures for CV include [152, 190, 206, 282, 291, 371, 378, 390]. While CV is by far the most popular domain of application for CNNs, CNNs have also been employed successfully in several other areas. In particular, we refer, for example, to [110, 143, 245, 430, 434, 437] for applications of CNNs to *natural language processing* (NLP), we refer, for instance, to [1, 59, 78, 359, 396] for applications of CNNs to audio processing, and we refer, for example, to [46, 105, 236, 348, 408, 440] for applications of CNNs to time series analysis. Finally, for approximation results for feedforward CNNs we refer, for instance, to Petersen & Voigtländer [334] and the references therein.

1.4.1 Discrete convolutions

Definition 1.4.1 (Discrete convolutions). *Let $T \in \mathbb{N}$, $a_1, a_2, \dots, a_T, w_1, w_2, \dots, w_T, \mathfrak{d}_1, \mathfrak{d}_2, \dots, \mathfrak{d}_T \in \mathbb{N}$ and let $A = (A_{i_1,i_2,\dots,i_T})_{(i_1,i_2,\dots,i_T)\in(\times_{t=1}^{T}\{1,2,\dots,a_t\})} \in \mathbb{R}^{a_1\times a_2\times\dots\times a_T}$, $W = (W_{i_1,i_2,\dots,i_T})_{(i_1,i_2,\dots,i_T)\in(\times_{t=1}^{T}\{1,2,\dots,w_t\})} \in \mathbb{R}^{w_1\times w_2\times\dots\times w_T}$ satisfy for all $t \in \{1, 2, \dots, T\}$ that*

$$\mathfrak{d}_t = a_t - w_t + 1. \tag{1.106}$$

Then we denote by $A * W = ((A * W)_{i_1,i_2,\dots,i_T})_{(i_1,i_2,\dots,i_T)\in(\times_{t=1}^{T}\{1,2,\dots,\mathfrak{d}_t\})} \in \mathbb{R}^{\mathfrak{d}_1\times\mathfrak{d}_2\times\dots\times\mathfrak{d}_T}$ the tensor which satisfies for all $i_1 \in \{1, 2, \dots, \mathfrak{d}_1\}$, $i_2 \in \{1, 2, \dots, \mathfrak{d}_2\}$, $\dots$, $i_T \in \{1, 2, \dots, \mathfrak{d}_T\}$ that

$$(A * W)_{i_1,i_2,\dots,i_T} = \sum_{r_1=1}^{w_1}\sum_{r_2=1}^{w_2}\cdots\sum_{r_T=1}^{w_T} A_{i_1-1+r_1,i_2-1+r_2,\dots,i_T-1+r_T} W_{r_1,r_2,\dots,r_T}. \tag{1.107}$$

1.4.2 Structured description of feedforward CNNs

Definition 1.4.2 (Structured description of feedforward CNNs). *We denote by $\mathbf{C}$ the set given by*

$$\mathbf{C} = \bigcup_{T,L\in\mathbb{N}} \bigcup_{l_0,l_1,\dots,l_L\in\mathbb{N}} \bigcup_{(c_{k,t})_{(k,t)\in\{1,2,\dots,L\}\times\{1,2,\dots,T\}}\subseteq\mathbb{N}} \left(\bigtimes_{k=1}^{L} \left((\mathbb{R}^{c_{k,1}\times c_{k,2}\times\dots\times c_{k,T}})^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k} \right) \right). \tag{1.108}$$

Definition 1.4.3 (Feedforward CNNs). *We say that Φ is a feedforward CNN if and only if it holds that*

$$\Phi \in \mathbf{C} \tag{1.109}$$

(cf. Definition 1.4.2).

1.4.3 Realizations of feedforward CNNs

Definition 1.4.4 (One tensor). *Let $T \in \mathbb{N}$, $d_1, d_2, \dots, d_T \in \mathbb{N}$. Then we denote by $\mathbf{I}^{d_1,d_2,\dots,d_T} = (\mathbf{I}^{d_1,d_2,\dots,d_T}_{i_1,i_2,\dots,i_T})_{(i_1,i_2,\dots,i_T)\in(\times_{t=1}^{T}\{1,2,\dots,d_t\})} \in \mathbb{R}^{d_1\times d_2\times\dots\times d_T}$ the tensor which satisfies for all $i_1 \in \{1, 2, \dots, d_1\}$, $i_2 \in \{1, 2, \dots, d_2\}$, $\dots$, $i_T \in \{1, 2, \dots, d_T\}$ that*

$$\mathbf{I}^{d_1,d_2,\dots,d_T}_{i_1,i_2,\dots,i_T} = 1. \tag{1.110}$$

Definition 1.4.5 (Realizations associated to feedforward CNNs)**.** *Let $T, L \in \mathbb{N}$, $l_0, l_1, \dots, l_L \in \mathbb{N}$, let $(c_{k,t})_{(k,t)\in\{1,2,\dots,L\}\times\{1,2,\dots,T\}} \subseteq \mathbb{N}$, let $\Phi = (((W_{k,n,m})_{(n,m)\in\{1,2,\dots,l_k\}\times\{1,2,\dots,l_{k-1}\}}, (B_{k,n})_{n\in\{1,2,\dots,l_k\}}))_{k\in\{1,2,\dots,L\}} \in \bigtimes_{k=1}^L ((\mathbb{R}^{c_{k,1}\times c_{k,2}\times\dots\times c_{k,T}})^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k}) \subseteq \mathbf{C}$, and let $a\colon \mathbb{R}\to\mathbb{R}$ be a function. Then we denote by*

$$\mathcal{R}_a^{\mathbf{C}}(\Phi)\colon \left(\bigcup_{\substack{d_1,d_2,\dots,d_T\in\mathbb{N} \\ \forall\, t\in\{1,2,\dots,T\}\colon d_t - \sum_{k=1}^L (c_{k,t}-1)\geq 1}} (\mathbb{R}^{d_1\times d_2\times\dots\times d_T})^{l_0} \right) \to \left(\bigcup_{d_1,d_2,\dots,d_T\in\mathbb{N}} (\mathbb{R}^{d_1\times d_2\times\dots\times d_T})^{l_L} \right) \tag{1.111}$$

the function which satisfies for all $(\mathfrak{d}_{k,t})_{(k,t)\in\{0,1,\dots,L\}\times\{1,2,\dots,T\}} \subseteq \mathbb{N}$, $x_0 = (x_{0,1},\dots,x_{0,l_0}) \in (\mathbb{R}^{\mathfrak{d}_{0,1}\times\mathfrak{d}_{0,2}\times\dots\times\mathfrak{d}_{0,T}})^{l_0}$, $x_1 = (x_{1,1},\dots,x_{1,l_1}) \in (\mathbb{R}^{\mathfrak{d}_{1,1}\times\mathfrak{d}_{1,2}\times\dots\times\mathfrak{d}_{1,T}})^{l_1}$, $\dots$, $x_L = (x_{L,1},\dots,x_{L,l_L}) \in (\mathbb{R}^{\mathfrak{d}_{L,1}\times\mathfrak{d}_{L,2}\times\dots\times\mathfrak{d}_{L,T}})^{l_L}$ with

$$\forall\, k\in\{1,2,\dots,L\},\ t\in\{1,2,\dots,T\}\colon \mathfrak{d}_{k,t} = \mathfrak{d}_{k-1,t} - c_{k,t} + 1 \tag{1.112}$$

and

$$\forall\, k\in\{1,2,\dots,L\},\ n\in\{1,2,\dots,l_k\}\colon \\ x_{k,n} = \mathfrak{M}_{a\mathbb{1}_{(0,L)}(k)+\mathrm{id}_{\mathbb{R}}\mathbb{1}_{\{L\}}(k),\mathfrak{d}_{k,1},\mathfrak{d}_{k,2},\dots,\mathfrak{d}_{k,T}}\big(B_{k,n}\mathbf{I}^{\mathfrak{d}_{k,1},\mathfrak{d}_{k,2},\dots,\mathfrak{d}_{k,T}} + \textstyle\sum_{m=1}^{l_{k-1}} x_{k-1,m} * W_{k,n,m}\big) \tag{1.113}$$

that

$$(\mathcal{R}_a^{\mathbf{C}}(\Phi))(x_0) = x_L \tag{1.114}$$

and we call $\mathcal{R}_a^{\mathbf{C}}(\Phi)$ the realization function of the feedforward CNN Φ with activation function a (we call $\mathcal{R}_a^{\mathbf{C}}(\Phi)$ the realization of the feedforward CNN Φ with activation a) (cf. Definitions 1.2.1, 1.4.1, 1.4.2, and 1.4.4).

```python
import torch
import torch.nn as nn


class ConvolutionalANN(nn.Module):
    def __init__(self):
        super().__init__()
        # The convolutional layer defined here takes any tensor of
        # shape (1, n, m) [a single input] or (N, 1, n, m) [a batch
        # of N inputs] where N, n, m are natural numbers satisfying
        # n >= 3 and m >= 3.
        self.conv1 = nn.Conv2d(
            in_channels=1, out_channels=5, kernel_size=(3, 3)
        )
```

```
        self.activation1 = nn.ReLU()
        self.conv2 = nn.Conv2d(
            in_channels=5, out_channels=5, kernel_size=(5, 3)
        )

    def forward(self, x0):
        x1 = self.activation1(self.conv1(x0))
        print(x1.shape)
        x2 = self.conv2(x1)
        print(x2.shape)
        return x2


model = ConvolutionalANN()
x0 = torch.rand(1, 20, 20)
# This will print the shapes of the outputs of the two layers of
# the model, in this case:
# torch.Size([5, 18, 18])
# torch.Size([5, 14, 16])
model(x0)
```

Source code 1.18 (code/conv-ann.py): PYTHON code implementing a feedforward CNN in PYTORCH. The implemented model here corresponds to a feedforward CNN $\Phi \in \mathbf{C}$ where $T = 2$, $L = 2$, $l_0 = 1$, $l_1 = 5$, $l_2 = 5$, $(c_{1,1}, c_{1,2}) = (3, 3)$, $(c_{2,1}, c_{2,2}) = (5, 3)$, and $\Phi \in \big(\bigtimes_{k=1}^{L}\big((\mathbb{R}^{c_{k,1}\times c_{k,2}\times\ldots\times c_{k,T}})^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k}\big)\big) = ((\mathbb{R}^{3\times 3})^{5\times 1} \times \mathbb{R}^5) \times ((\mathbb{R}^{3\times 5})^{5\times 5} \times \mathbb{R}^5)$. The model, given an input of shape $(1, d_1, d_2)$ with $d_1 \in \mathbb{N} \cap [7, \infty)$, $d_2 \in \mathbb{N} \cap [5, \infty)$, produces an output of shape $(5, d_1 - 6, d_2 - 4)$, (corresponding to the realization function $\mathcal{R}_a^{\mathbf{C}}(\Phi)$ for $a \in C(\mathbb{R}, \mathbb{R})$ having domain $\bigcup_{d_1,d_2\in\mathbb{N},\, d_1\geq 7,\, d_2\geq 5}(\mathbb{R}^{d_1\times d_2})^1$ and satisfying for all $d_1 \in \mathbb{N} \cap [7, \infty)$, $d_2 \in \mathbb{N} \cap [5, \infty)$, $x_0 \in (\mathbb{R}^{d_1\times d_2})^1$ that $(\mathcal{R}_a^{\mathbf{C}}(\Phi))(x_0) \in (\mathbb{R}^{d_1-6, d_2-4})^5)$.

Example 1.4.6 (Example for Definition 1.4.5)**.** *Let* $T = 2$*,* $L = 2$*,* $l_0 = 1$*,* $l_1 = 2$*,* $l_2 = 1$*,* $c_{1,1} = 2$*,* $c_{1,2} = 2$*,* $c_{2,1} = 1$*,* $c_{2,2} = 1$ *and let*

$$\Phi \in \left(\bigtimes_{k=1}^{L}\big((\mathbb{R}^{c_{k,1}\times c_{k,2}\times\ldots\times c_{k,T}})^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k}\big)\right) = \big((\mathbb{R}^{2\times 2})^{2\times 1} \times \mathbb{R}^2\big) \times \big((\mathbb{R}^{1\times 1})^{1\times 2} \times \mathbb{R}^1\big) \tag{1.115}$$

satisfy

$$\Phi = \left(\left(\left(\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right), \begin{pmatrix} 1 \\ -1 \end{pmatrix}\right), \big(\big((-2) \quad (2)\big), (3)\big)\right). \tag{1.116}$$

Then

$$\big(\mathcal{R}_{\mathfrak{r}}^{\mathbf{C}}(\Phi)\big)\left(\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}\right) = \begin{pmatrix} 11 & 15 \\ 23 & 27 \end{pmatrix} \tag{1.117}$$

(cf. Definitions 1.2.4 and 1.4.5).

Proof for Example 1.4.6. Throughout this proof, let $x_0 \in \mathbb{R}^{3\times 3}$, $x_1 = (x_{1,1}, x_{1,2}) \in (\mathbb{R}^{2\times 2})^2$, $x_2 \in \mathbb{R}^{2\times 2}$ with satisfy that

$$x_0 = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \qquad x_{1,1} = \mathfrak{M}_{\mathfrak{r},2\times 2}\left(\mathbf{I}^{2,2} + x_0 * \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}\right), \tag{1.118}$$

$$x_{1,2} = \mathfrak{M}_{\mathfrak{r},2\times 2}\left((-1)\mathbf{I}^{2,2} + x_0 * \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right), \tag{1.119}$$

$$\text{and} \qquad x_2 = \mathfrak{M}_{\mathrm{id}_{\mathbb{R}},2\times 2}\big(3\mathbf{I}^{2,2} + x_{1,1} * \big(-2\big) + x_{1,2} * \big(2\big)\big). \tag{1.120}$$

Note that (1.114), (1.116), (1.118), (1.119), and (1.120) imply that

$$\big(\mathcal{R}_{\mathfrak{r}}^{\mathbf{C}}(\Phi)\big)\left(\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}\right) = \big(\mathcal{R}_{\mathfrak{r}}^{\mathbf{C}}(\Phi)\big)(x_0) = x_2. \tag{1.121}$$

Next observe that (1.118) ensures that

$$\begin{split} x_{1,1} &= \mathfrak{M}_{\mathfrak{r},2\times 2}\left(\mathbf{I}^{2,2} + x_0 * \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}\right) = \mathfrak{M}_{\mathfrak{r},2\times 2}\left(\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}\right) \\ &= \mathfrak{M}_{\mathfrak{r},2\times 2}\left(\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}\right) = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}. \end{split} \tag{1.122}$$

Furthermore, note that (1.119) assures that

$$\begin{split} x_{1,2} &= \mathfrak{M}_{\mathfrak{r},2\times 2}\left((-1)\mathbf{I}^{2,2} + x_0 * \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right) = \mathfrak{M}_{\mathfrak{r},2\times 2}\left(\begin{pmatrix} -1 & -1 \\ -1 & -1 \end{pmatrix} + \begin{pmatrix} 6 & 8 \\ 12 & 14 \end{pmatrix}\right) \\ &= \mathfrak{M}_{\mathfrak{r},2\times 2}\left(\begin{pmatrix} 5 & 7 \\ 11 & 13 \end{pmatrix}\right) = \begin{pmatrix} 5 & 7 \\ 11 & 13 \end{pmatrix}. \end{split} \tag{1.123}$$

Moreover, observe that this, (1.122), and (1.120) demonstrate that

$$\begin{split} x_2 &= \mathfrak{M}_{\mathrm{id}_{\mathbb{R}},2\times 2}\big(3\mathbf{I}^{2,2} + x_{1,1} * \big(-2\big) + x_{1,2} * \big(2\big)\big) \\ &= \mathfrak{M}_{\mathrm{id}_{\mathbb{R}},2\times 2}\left(3\mathbf{I}^{2,2} + \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} * \big(-2\big) + \begin{pmatrix} 5 & 7 \\ 11 & 13 \end{pmatrix} * \big(2\big)\right) \\ &= \mathfrak{M}_{\mathrm{id}_{\mathbb{R}},2\times 2}\left(\begin{pmatrix} 3 & 3 \\ 3 & 3 \end{pmatrix} + \begin{pmatrix} -2 & -2 \\ -2 & -2 \end{pmatrix} + \begin{pmatrix} 10 & 14 \\ 22 & 26 \end{pmatrix}\right) \\ &= \mathfrak{M}_{\mathrm{id}_{\mathbb{R}},2\times 2}\left(\begin{pmatrix} 11 & 15 \\ 23 & 27 \end{pmatrix}\right) = \begin{pmatrix} 11 & 15 \\ 23 & 27 \end{pmatrix}. \end{split} \tag{1.124}$$

This and (1.121) establish (1.117). The proof for Example 1.4.6 is thus complete. □

```python
import torch
import torch.nn as nn


model = nn.Sequential(
    nn.Conv2d(in_channels=1, out_channels=2, kernel_size=(2, 2)),
    nn.ReLU(),
    nn.Conv2d(in_channels=2, out_channels=1, kernel_size=(1, 1)),
)

with torch.no_grad():
    model[0].weight.set_(
        torch.Tensor([[[[0, 0], [0, 0]]], [[[1, 0], [0, 1]]]])
    )
    model[0].bias.set_(torch.Tensor([1, -1]))
    model[2].weight.set_(torch.Tensor([[[[-2]], [[2]]]]))
    model[2].bias.set_(torch.Tensor([3]))

x0 = torch.Tensor([[[1, 2, 3], [4, 5, 6], [7, 8, 9]]])
print(model(x0))
```

Source code 1.19 (code/conv-ann-ex.py): PYTHON code implementing the feedforward CNN Φ from Example 1.4.6 (see (1.116)) in PYTORCH and verifying (1.117).

Exercise 1.4.1. Let

$$\begin{aligned}\Phi = \big(&((W_{1,n,m})_{(n,m)\in\{1,2,3\}\times\{1\}}, (B_{1,n})_{n\in\{1,2,3\}}),\\ &((W_{2,n,m})_{(n,m)\in\{1\}\times\{1,2,3\}}, (B_{2,n})_{n\in\{1\}})\big) \in ((\mathbb{R}^2)^{3\times 1}\times\mathbb{R}^3)\times((\mathbb{R}^3)^{1\times 3}\times\mathbb{R}^1)\end{aligned} \tag{1.125}$$

satisfy

$$W_{1,1,1} = (1,-1),\quad W_{1,2,1} = (2,-2),\quad W_{1,3,1} = (-3,3),\quad (B_{1,n})_{n\in\{1,2,3\}} = (1,2,3), \tag{1.126}$$

$$W_{2,1,1} = (1,-1,1),\quad W_{2,1,2} = (2,-2,2),\quad W_{2,1,3} = (-3,3,-3),\quad \text{and}\quad B_{2,1} = -2 \tag{1.127}$$

and let $v \in \mathbb{R}^9$ satisfy $v = (1,2,3,4,5,4,3,2,1)$. Specify

$$(\mathcal{R}^{\mathbf{C}}_{\mathfrak{r}}(\Phi))(v) \tag{1.128}$$

explicitly and prove that your result is correct (cf. Definitions 1.2.4 and 1.4.5)!

Exercise 1.4.2. Let

$$\Phi = \big(((W_{1,n,m})_{(n,m)\in\{1,2,3\}\times\{1\}}, (B_{1,n})_{n\in\{1,2,3\}}),\\ ((W_{2,n,m})_{(n,m)\in\{1\}\times\{1,2,3\}}, (B_{2,n})_{n\in\{1\}})\big) \in ((\mathbb{R}^3)^{3\times 1} \times \mathbb{R}^3) \times ((\mathbb{R}^2)^{1\times 3} \times \mathbb{R}^1) \tag{1.129}$$

satisfy

$$W_{1,1,1} = (1,1,1), \qquad W_{1,2,1} = (2,-2,-2), \tag{1.130}$$

$$W_{1,3,1} = (-3,-3,3), \qquad (B_{1,n})_{n\in\{1,2,3\}} = (3,-2,-1), \tag{1.131}$$

$$W_{2,1,1} = (2,-1), \quad W_{2,1,2} = (-1,2), \quad W_{2,1,3} = (-1,0), \quad \text{and} \quad B_{2,1} = -2 \tag{1.132}$$

and let $v \in \mathbb{R}^9$ satisfy $v = (1,-1,1,-1,1,-1,1,-1,1)$. Specify

$$(\mathcal{R}^{\mathbf{C}}_{\mathfrak{r}}(\Phi))(v) \tag{1.133}$$

explicitly and prove that your result is correct (cf. Definitions 1.2.4 and 1.4.5)!

Exercise 1.4.3. Prove or disprove the following statement: For every $a \in C(\mathbb{R},\mathbb{R})$, $\Phi \in \mathbf{N}$ there exists $\Psi \in \mathbf{C}$ such that for all $x \in \mathbb{R}^{\mathcal{I}(\Phi)}$ it holds that $\mathbb{R}^{\mathcal{I}(\Phi)} \subseteq \operatorname{Domain}(\mathcal{R}^{\mathbf{C}}_a(\Psi))$ and

$$(\mathcal{R}^{\mathbf{C}}_a(\Psi))(x) = (\mathcal{R}^{\mathbf{N}}_a(\Phi))(x) \tag{1.134}$$

(cf. Definitions 1.3.1, 1.3.4, 1.4.2, and 1.4.5).

Definition 1.4.7 (Standard scalar products)**.** *We denote by* $\langle\cdot,\cdot\rangle\colon \big[\bigcup_{d\in\mathbb{N}}(\mathbb{R}^d\times\mathbb{R}^d)\big] \to \mathbb{R}$ *the function which satisfies for all* $d \in \mathbb{N}$, $x = (x_1, x_2, \ldots, x_d)$, $y = (y_1, y_2, \ldots, y_d) \in \mathbb{R}^d$ *that*

$$\langle x, y\rangle = \sum_{i=1}^d x_i y_i. \tag{1.135}$$

Exercise 1.4.4. For every $d \in \mathbb{N}$ let $\mathbf{e}^{(d)}_1, \mathbf{e}^{(d)}_2, \ldots, \mathbf{e}^{(d)}_d \in \mathbb{R}^d$ satisfy $\mathbf{e}^{(d)}_1 = (1,0,\ldots,0)$, $\mathbf{e}^{(d)}_2 = (0,1,0,\ldots,0)$, $\ldots$, $\mathbf{e}^{(d)}_d = (0,\ldots,0,1)$. Prove or disprove the following statement: For all $a \in C(\mathbb{R},\mathbb{R})$, $\Phi \in \mathbf{N}$, $D \in \mathbb{N}$, $x = ((x_{i,j})_{j\in\{1,2,\ldots,D\}})_{i\in\{1,2,\ldots,\mathcal{I}(\Phi)\}} \in (\mathbb{R}^D)^{\mathcal{I}(\Phi)}$ it holds that

$$(\mathcal{R}^{\mathbf{C}}_a(\Phi))(x) = \Big(\big(\langle \mathbf{e}^{(\mathcal{O}(\Phi))}_k, (\mathcal{R}^{\mathbf{N}}_a(\Phi))((x_{i,j})_{i\in\{1,2,\ldots,\mathcal{I}(\Phi)\}})\rangle\big)_{j\in\{1,2,\ldots,D\}}\Big)_{k\in\{1,2,\ldots,\mathcal{O}(\Phi)\}} \tag{1.136}$$

(cf. Definitions 1.3.1, 1.3.4, 1.4.5, and 1.4.7).

1.5 Residual ANNs (ResNets)

In this section we review ResNets. Roughly speaking, plain-vanilla feedforward ANNs can be seen as having a computational structure consisting of sequentially chained layers in which each layer feeds information forward to the next layer (cf., for example, Definitions 1.1.3 and 1.3.4 above). ResNets, in turn, are ANNs involving so-called *skip connections* in their computational structure, which allow information from one layer to be fed not only to the next layer, but also to other layers further down the computational structure. In principle, such skip connections can be employed in combinations with other ANN architecture elements, such as fully-connected layers (cf., for instance, Sections 1.1 and 1.3 above), convolutional layers (cf., for example, Section 1.4 above), and recurrent structures (cf., for instance, Section 1.6 below). However, for simplicity we introduce in this section in all mathematical details feedforward fully-connected ResNets in which the skip connection is a learnable linear map (see Definitions 1.5.1 and 1.5.4 below).

ResNets were introduced in He et al. [190] as an attempt to improve the performance of deep ANNs which typically are much harder to train than shallow ANNs (cf., for example, [30, 153, 328]). The ResNets in He et al. [190] only involve skip connections that are identity mappings without trainable parameters, and are thus a special case of the definition of ResNets provided in this section (see Definitions 1.5.1 and 1.5.4 below). The idea of skip connection (sometimes also called *shortcut connections*) has already been introduced before ResNets and has been used in earlier ANN architecture such as the *highway nets* in Srivastava et al. [384, 385] (cf. also [264, 293, 345, 390, 398]). In addition, we refer to [191, 206, 404, 417, 427] for a few successful ANN architecures building on the ResNets in He et al. [190].

1.5.1 Structured description of fully-connected ResNets

Definition 1.5.1 (Structured description of fully-connected ResNets)**.** *We denote by* $\mathbf{R}$ *the set given by*

$$\mathbf{R} = \bigcup_{L\in\mathbb{N}} \bigcup_{l_0,l_1,\dots,l_L\in\mathbb{N}} \bigcup_{S\subseteq\{(r,k)\in(\mathbb{N}_0)^2\colon r<k\leq L\}} \Big(\big(\times_{k=1}^{L} (\mathbb{R}^{l_k\times l_{k-1}} \times \mathbb{R}^{l_k})\big) \times \big(\times_{(r,k)\in S} \mathbb{R}^{l_k\times l_r}\big)\Big). \tag{1.137}$$

Definition 1.5.2 (Fully-connected ResNets)**.** *We say that* Φ *is a fully-connected ResNet if and only if it holds that*

$$\Phi \in \mathbf{R} \tag{1.138}$$

(cf. Definition 1.5.1).

Lemma 1.5.3 (On an empty set of skip connections)**.** *Let $L \in \mathbb{N}$, $l_0, l_1, \dots, l_L \in \mathbb{N}$, $S \subseteq \{(r,k) \in (\mathbb{N}_0)^2 \colon r < k \leq L\}$. Then*

$$\#\big(\times_{(r,k)\in S} \mathbb{R}^{l_k \times l_r}\big) = \begin{cases} 1 & : S = \emptyset \\ \infty & : S \neq \emptyset. \end{cases} \tag{1.139}$$

Proof of Lemma 1.5.3. Throughout this proof, for all sets A and B let $F(A,B)$ be the set of all function from A to B. Note that

$$\#\big(\times_{(r,k)\in S} \mathbb{R}^{l_k \times l_r}\big) = \#\big\{f \in F\big(S, \cup_{(r,k)\in S} \mathbb{R}^{l_k \times l_r}\big) \colon (\forall\, (r,k) \in S \colon f(r,k) \in \mathbb{R}^{l_k \times l_r})\big\}. \tag{1.140}$$

This and the fact that for all sets B it holds that $\#(F(\emptyset, B)) = 1$ ensure that

$$\#\big(\times_{(r,k)\in \emptyset} \mathbb{R}^{l_k \times l_r}\big) = \#(F(\emptyset,\emptyset)) = 1. \tag{1.141}$$

Next note that (1.140) assures that for all $(R,K) \in S$ it holds that

$$\#\big(\times_{(r,k)\in S} \mathbb{R}^{l_k \times l_r}\big) \geq \#\big(F\big(\{(R,K)\}, \mathbb{R}^{l_K \times l_R}\big)\big) = \infty. \tag{1.142}$$

Combining this and (1.141) establishes (1.139). The proof of Lemma 1.5.3 is thus complete. □

1.5.2 Realizations of fully-connected ResNets

Definition 1.5.4 (Realizations associated to fully-connected ResNets)**.** *Let $L \in \mathbb{N}$, $l_0, l_1, \dots, l_L \in \mathbb{N}$, $S \subseteq \{(r,k) \in (\mathbb{N}_0)^2 \colon r < k \leq L\}$, $\Phi = ((W_k, B_k)_{k\in\{1,2,\dots,L\}}, (V_{r,k})_{(r,k)\in S}) \in \big(\big(\times_{k=1}^{L} (\mathbb{R}^{l_k \times l_{k-1}} \times \mathbb{R}^{l_k})\big) \times \big(\times_{(r,k)\in S} \mathbb{R}^{l_k \times l_r}\big)\big) \subseteq \mathbf{R}$ and let $a \colon \mathbb{R} \to \mathbb{R}$ be a function. Then we denote by*

$$\mathcal{R}^{\mathbf{R}}_a(\Phi) \colon \mathbb{R}^{l_0} \to \mathbb{R}^{l_L} \tag{1.143}$$

the function which satisfies for all $x_0 \in \mathbb{R}^{l_0}, x_1 \in \mathbb{R}^{l_1}, \dots, x_L \in \mathbb{R}^{l_L}$ with

$$\forall\, k \in \{1,2,\dots,L\} \colon \quad x_k = \mathfrak{M}_{a\mathbb{1}_{(0,L)}(k) + \mathrm{id}_{\mathbb{R}}\, \mathbb{1}_{\{L\}}(k), l_k}\big(W_k x_{k-1} + B_k + \textstyle\sum_{r\in\mathbb{N}_0, (r,k)\in S} V_{r,k} x_r\big) \tag{1.144}$$

that

$$(\mathcal{R}^{\mathbf{R}}_a(\Phi))(x_0) = x_L \tag{1.145}$$

and we call $\mathcal{R}^{\mathbf{R}}_a(\Phi)$ the realization function of the fully-connected ResNet Φ with activation function a (we call $\mathcal{R}^{\mathbf{R}}_a(\Phi)$ the realization of the fully-connected ResNet Φ with activation a) (cf. Definitions 1.2.1 and 1.5.1).

Definition 1.5.5 (Identity matrices)**.** *Let $d \in \mathbb{N}$. Then we denote by $\mathrm{I}_d \in \mathbb{R}^{d\times d}$ the identity matrix in $\mathbb{R}^{d\times d}$.*

```python
import torch
import torch.nn as nn

class ResidualANN(nn.Module):
    def __init__(self):
        super().__init__()
        self.affine1 = nn.Linear(3, 10)
        self.activation1 = nn.ReLU()
        self.affine2 = nn.Linear(10, 20)
        self.activation2 = nn.ReLU()
        self.affine3 = nn.Linear(20, 10)
        self.activation3 = nn.ReLU()
        self.affine4 = nn.Linear(10, 1)

    def forward(self, x0):
        x1 = self.activation1(self.affine1(x0))
        x2 = self.activation2(self.affine2(x1))
        x3 = self.activation3(x1 + self.affine3(x2))
        x4 = self.affine4(x3)
        return x4
```

Source code 1.20 (`code/res-ann.py`): PYTHON code implementing a fully-connected ResNet in PYTORCH. The implemented model here corresponds to a fully-connected ResNet (Φ, V) where $l_0 = 3$, $l_1 = 10$, $l_2 = 20$, $l_3 = 10$, $l_4 = 1$, $\Phi = ((W_1, B_1), (W_2, B_2), (W_3, B_3), (W_4, B_4)) \in \big(\bigtimes_{k=1}^{4} (\mathbb{R}^{l_k \times l_{k-1}} \times \mathbb{R}^{l_k})\big)$, $S = \{(1, 3)\}$, $V = (V_{r,k})_{(r,k)\in S} \in \big(\bigtimes_{(r,k)\in S} \mathbb{R}^{l_k\times l_r}\big)$, and $V_{1,3} = \mathrm{I}_{10}$ (cf. Definition 1.5.5).

Example 1.5.6 (Example for Definition 1.5.2)**.** *Let $l_0 = 1$, $l_1 = 1$, $l_2 = 2$, $l_3 = 2$, $l_4 = 1$, $S = \{(0, 4)\}$, let*

$$\Phi = ((W_1, B_1), (W_2, B_2), (W_3, B_3), (W_4, B_4)) \in \big(\bigtimes_{k=1}^{4} (\mathbb{R}^{l_k \times l_{k-1}} \times \mathbb{R}^{l_k})\big) \tag{1.146}$$

satisfy

$$W_1 = \begin{pmatrix} 1 \end{pmatrix}, \qquad B_1 = \begin{pmatrix} 0 \end{pmatrix}, \qquad W_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \qquad B_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{1.147}$$

$$W_3 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad B_3 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad W_4 = \begin{pmatrix} 2 & 2 \end{pmatrix}, \qquad \text{and} \qquad B_4 = \begin{pmatrix} 1 \end{pmatrix}, \tag{1.148}$$

and let $V = (V_{r,k})_{(r,k)\in S} \in \bigtimes_{(r,k)\in S} \mathbb{R}^{l_k\times l_r}$ satisfy

$$V_{0,4} = \begin{pmatrix} -1 \end{pmatrix}. \tag{1.149}$$

Then

$$(\mathcal{R}_{\mathfrak{r}}^{\mathbf{R}}(\Phi, V))(5) = 28 \tag{1.150}$$

(cf. Definitions 1.2.4 and 1.5.4).

Proof for Example 1.5.6. Throughout this proof, let $x_0 \in \mathbb{R}^1$, $x_1 \in \mathbb{R}^1$, $x_2 \in \mathbb{R}^2$, $x_3 \in \mathbb{R}^2$, $x_4 \in \mathbb{R}^1$ satisfy for all $k \in \{1, 2, 3, 4\}$ that $x_0 = 5$ and

$$x_k = \mathfrak{M}_{\mathfrak{r}\mathbb{1}_{(0,4)}(k)+\mathrm{id}_{\mathbb{R}}\mathbb{1}_{\{4\}}(k),l_k}(W_k x_{k-1} + B_k + \textstyle\sum_{r\in\mathbb{N}_0,(r,k)\in S} V_{r,k} x_r). \tag{1.151}$$

Observe that (1.151) assures that

$$(\mathcal{R}_{\mathfrak{r}}^{\mathbf{R}}(\Phi, V))(5) = x_4. \tag{1.152}$$

Next note that (1.151) ensures that

$$x_1 = \mathfrak{M}_{\mathfrak{r},1}(W_1 x_0 + B_1) = \mathfrak{M}_{\mathfrak{r},1}(5), \tag{1.153}$$

$$x_2 = \mathfrak{M}_{\mathfrak{r},2}(W_2 x_1 + B_2) = \mathfrak{M}_{\mathfrak{r},1}\left(\begin{pmatrix}1\\2\end{pmatrix}(5) + \begin{pmatrix}0\\1\end{pmatrix}\right) = \mathfrak{M}_{\mathfrak{r},1}\left(\begin{pmatrix}5\\11\end{pmatrix}\right) = \begin{pmatrix}5\\11\end{pmatrix}, \tag{1.154}$$

$$x_3 = \mathfrak{M}_{\mathfrak{r},2}(W_3 x_2 + B_3) = \mathfrak{M}_{\mathfrak{r},1}\left(\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\begin{pmatrix}5\\11\end{pmatrix} + \begin{pmatrix}0\\0\end{pmatrix}\right) = \mathfrak{M}_{\mathfrak{r},1}\left(\begin{pmatrix}5\\11\end{pmatrix}\right) = \begin{pmatrix}5\\11\end{pmatrix}, \tag{1.155}$$

$$\begin{split}\text{and} \qquad x_4 &= \mathfrak{M}_{\mathfrak{r},1}(W_4 x_3 + B_4 + V_{0,4} x_0) \\ &= \mathfrak{M}_{\mathfrak{r},1}\left(\begin{pmatrix}2 & 2\end{pmatrix}\begin{pmatrix}5\\11\end{pmatrix} + \begin{pmatrix}1\end{pmatrix} + \begin{pmatrix}-1\end{pmatrix}(5)\right) = \mathfrak{M}_{\mathfrak{r},1}(28) = 28.\end{split} \tag{1.156}$$

This and (1.152) establish (1.150). The proof for Example 1.5.6 is thus complete. □

Exercise 1.5.1. Let $l_0 = 1$, $l_1 = 2$, $l_2 = 3$, $l_3 = 1$, $S = \{(0, 3), (1, 3)\}$, let

$$\Phi = ((W_1, B_1), (W_2, B_2), (W_3, B_3)) \in \left(\times_{k=1}^{3}(\mathbb{R}^{l_k \times l_{k-1}} \times \mathbb{R}^{l_k})\right) \tag{1.157}$$

satisfy

$$W_1 = \begin{pmatrix}1\\2\end{pmatrix}, \qquad B_1 = \begin{pmatrix}3\\4\end{pmatrix}, \qquad W_2 = \begin{pmatrix}-1 & 2\\3 & -4\\-5 & 6\end{pmatrix}, \qquad B_2 = \begin{pmatrix}0\\0\\0\end{pmatrix}, \tag{1.158}$$

$$W_3 = \begin{pmatrix}-1 & 1 & -1\end{pmatrix}, \qquad \text{and} \qquad B_3 = \begin{pmatrix}-4\end{pmatrix}, \tag{1.159}$$

and let $V = (V_{r,k})_{(r,k)\in S} \in \times_{(r,k)\in S} \mathbb{R}^{l_k \times l_r}$ satisfy

$$V_{0,3} = \begin{pmatrix}1\end{pmatrix} \qquad \text{and} \qquad V_{1,3} = \begin{pmatrix}3 & -2\end{pmatrix}. \tag{1.160}$$

Prove or disprove the following statement: It holds that

$$(\mathcal{R}_{\mathfrak{r}}^{\mathbf{R}}(\Phi, V))(-1) = 0 \tag{1.161}$$

(cf. Definitions 1.2.4 and 1.5.4).

1.6 Recurrent ANNs (RNNs)

In this section we review RNNs, a type of ANNs designed to take sequences of data points as inputs. Roughly speaking, unlike in feedforward ANNs where an input is processed by a successive application of series of *different* parametric functions (cf. Definitions 1.1.3, 1.3.4, 1.4.5, and 1.5.4 above), in RNNs an input sequence is processed by a repeated application of the *same* parametric function whereby after the first application, each subsequent application of the parametric function takes as input a new element of the input sequence and a partial output from the previous application of the parametric function. The output of an RNN is then given by a sequence of partial outputs coming from the repeated applications of the parametric function (see Definition 1.6.2 below for a precise description of RNNs and cf., for instance, [4, Section 12.7], [60, Chapter 17] [63, Chapter 5], and [164, Chapter 10] for other introductions to RNNs).

The repeatedly applied parametric function in an RNN is typically called an *RNN node* and any RNN architecture is determined by specifying the architecture of the corresponding RNN node. We review a simple variant of such RNN nodes and the corresponding RNNs in Section 1.6.2 in detail and we briefly address one of the most commonly used RNN nodes, the so-called *long short-term memory* (LSTM) node, in Section 1.6.3.

There is a wide range of application areas where sequential data are considered and RNN based deep learning methods are being employed and developed. Examples of such applications areas are NLP including language translation (cf., for example, [11, 76, 77, 388] and the references therein), language generation (cf., for instance, [51, 169, 238, 340] and the references therein), and speech recognition (cf., for example, [6, 81, 170, 172, 360] and the references therein), time series prediction analysis including stock market prediction (cf., for instance, [130, 133, 372, 376] and the references therein) and weather prediction (cf., for example, [352, 375, 407] and the references therein) and video analysis (cf., for instance, [108, 235, 307, 401] and the references therein).

1.6.1 Description of RNNs

Definition 1.6.1 (Function unrolling). *Let X, Y, I be sets, let $f\colon X \times I \to Y \times I$ be a function, and let $T \in \mathbb{N}$, $\mathbb{I} \in I$. Then we denote by $\mathfrak{R}_{f,T,\mathbb{I}}\colon X^T \to Y^T$ the function which satisfies for all $x_1, x_2, \dots, x_T \in X$, $y_1, y_2, \dots, y_T \in Y$, $i_0, i_1, \dots, i_T \in I$ with $i_0 = \mathbb{I}$ and $\forall\, t \in \{1, 2, \dots, T\}\colon (y_t, i_t) = f(x_t, i_{t-1})$ that*

$$\mathfrak{R}_{f,T,\mathbb{I}}(x_1, x_2, \dots, x_T) = (y_1, y_2, \dots, y_T) \tag{1.162}$$

and we call $\mathfrak{R}_{f,T,\mathfrak{i}}$ the T-times unrolled function f with initial information $\mathbb{I}$.

Definition 1.6.2 (Description of RNNs). *Let X, Y, I be sets, let $\mathfrak{d}, T \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$, $\mathbb{I} \in I$, and let $\mathfrak{N} = (\mathfrak{N}_\vartheta)_{\vartheta \in \mathbb{R}^{\mathfrak{d}}}\colon \mathbb{R}^{\mathfrak{d}} \times X \times I \to Y \times I$ be a function. Then we call R the realization function of the T-step unrolled RNN with RNN node $\mathfrak{N}$, parameter vector θ, and initial* *information $\mathbb{I}$ (we call R the realization of the T-step unrolled RNN with RNN node $\mathfrak{N}$, parameter vector θ, and initial information $\mathbb{I}$) if and only if*

$$R = \mathfrak{R}_{\mathfrak{N}_\theta, T, \mathbb{I}} \tag{1.163}$$

(cf. Definition 1.6.1).

1.6.2 Vectorized description of simple fully-connected RNNs

Definition 1.6.3 (Vectorized description of simple fully-connected RNN nodes)**.** *Let $\mathfrak{x}, \mathfrak{y}, \mathfrak{i} \in \mathbb{N}$, $\theta \in \mathbb{R}^{(\mathfrak{x}+\mathfrak{i}+1)\mathfrak{i}+(\mathfrak{i}+1)\mathfrak{y}}$ and let $\Psi_1 \colon \mathbb{R}^{\mathfrak{i}} \to \mathbb{R}^{\mathfrak{i}}$ and $\Psi_2 \colon \mathbb{R}^{\mathfrak{y}} \to \mathbb{R}^{\mathfrak{y}}$ be functions. Then we call r the realization function of the simple fully-connected RNN node with parameter vector θ and activation functions Ψ_1 and Ψ_2 (we call r the realization of the simple fully-connected RNN node with parameter vector θ and activations Ψ_1 and Ψ_2) if and only if it holds that $r \colon \mathbb{R}^{\mathfrak{x}} \times \mathbb{R}^{\mathfrak{i}} \to \mathbb{R}^{\mathfrak{y}} \times \mathbb{R}^{\mathfrak{i}}$ is the function from $\mathbb{R}^{\mathfrak{x}} \times \mathbb{R}^{\mathfrak{i}}$ to $\mathbb{R}^{\mathfrak{y}} \times \mathbb{R}^{\mathfrak{i}}$ which satisfies for all $x \in \mathbb{R}^{\mathfrak{x}}$, $i \in \mathbb{R}^{\mathfrak{i}}$ that*

$$r(x, i) = \Big(\big(\Psi_2 \circ \mathcal{A}^{\theta,(\mathfrak{x}+\mathfrak{i}+1)\mathfrak{i}}_{\mathfrak{y},\mathfrak{i}} \circ \Psi_1 \circ \mathcal{A}^{\theta,0}_{\mathfrak{i},\mathfrak{x}+\mathfrak{i}}\big)(x, i), \big(\Psi_1 \circ \mathcal{A}^{\theta,0}_{\mathfrak{i},\mathfrak{x}+\mathfrak{i}}\big)(x, i) \Big) \tag{1.164}$$

(cf. Definition 1.1.1).

Definition 1.6.4 (Vectorized description of simple fully-connected RNNs)**.** *Let $\mathfrak{x}, \mathfrak{y}, \mathfrak{i}, T \in \mathbb{N}$, $\theta \in \mathbb{R}^{(\mathfrak{x}+\mathfrak{i}+1)\mathfrak{i}+(\mathfrak{i}+1)\mathfrak{y}}$, $\mathbb{I} \in \mathbb{R}^{\mathfrak{i}}$ and let $\Psi_1 \colon \mathbb{R}^{\mathfrak{i}} \to \mathbb{R}^{\mathfrak{i}}$ and $\Psi_2 \colon \mathbb{R}^{\mathfrak{y}} \to \mathbb{R}^{\mathfrak{y}}$ be functions. Then we call R the realization function of the T-step unrolled simple fully-connected RNN with parameter vector θ, activation functions Ψ_1 and Ψ_2, and initial information $\mathbb{I}$ (we call R the realization of the T-step unrolled simple fully-connected RNN with parameter vector θ, activations Ψ_1 and Ψ_2, and initial information $\mathbb{I}$) if and only if there exists $r \colon \mathbb{R}^{\mathfrak{x}} \times \mathbb{R}^{\mathfrak{i}} \to \mathbb{R}^{\mathfrak{y}} \times \mathbb{R}^{\mathfrak{i}}$ such that*

(i) it holds that r is the realization of the simple fully-connected RNN node with parameters θ and activations Ψ_1 and Ψ_2 and

(ii) it holds that

$$R = \mathfrak{R}_{r, T, \mathbb{I}} \tag{1.165}$$

(cf. Definitions 1.6.1 and 1.6.3).

Lemma 1.6.5. *Let $\mathfrak{x}, \mathfrak{y}, \mathfrak{i}, \mathfrak{d}, T \in \mathbb{N}$, $\theta \in \mathbb{R}^{\mathfrak{d}}$, $\mathbb{I} \in \mathbb{R}^{\mathfrak{i}}$ satisfy $\mathfrak{d} = (\mathfrak{x} + \mathfrak{i} + 1)\mathfrak{i} + (\mathfrak{i} + 1)\mathfrak{y}$, let $\Psi_1 \colon \mathbb{R}^{\mathfrak{i}} \to \mathbb{R}^{\mathfrak{i}}$ and $\Psi_2 \colon \mathbb{R}^{\mathfrak{y}} \to \mathbb{R}^{\mathfrak{y}}$ be functions, and let $\mathfrak{N} = (\mathfrak{N}_\vartheta)_{\vartheta \in \mathbb{R}^{\mathfrak{d}}} \colon \mathbb{R}^{\mathfrak{d}} \times \mathbb{R}^{\mathfrak{x}} \times \mathbb{R}^{\mathfrak{i}} \to \mathbb{R}^{\mathfrak{y}} \times \mathbb{R}^{\mathfrak{i}}$ satisfy for all $\vartheta \in \mathbb{R}^{\mathfrak{d}}$ that $\mathfrak{N}_\vartheta$ is the realization of the simple fully-connected RNN node with parameter vector ϑ and activations Ψ_1 and Ψ_2 (cf. Definition 1.6.3). Then the following two statements are equivalent:*

(i) It holds that R is the realization of the T-step unrolled simple fully-connected RNN with parameter vector θ, activations Ψ_1 and Ψ_2, and initial information $\mathbb{I}$ (cf. Definition 1.6.4).

(ii) It holds that R is the realization of the T-step unrolled RNN with RNN node $\mathfrak{N}$, parameter vector θ, and initial information $\mathbb{I}$ (cf. Definition 1.6.2).

Proof of Lemma 1.6.5. Observe that (1.163) and (1.165) ensure that ((i) $\leftrightarrow$ (ii)). The proof of Lemma 1.6.5 is thus complete. □

Exercise 1.6.1. For every $T \in \mathbb{N}$, $\alpha \in (0,1)$ let $R_{T,\alpha}$ be the realization of the T-step unrolled simple fully-connected RNN with parameter vector $(1, 0, 0, \alpha, 0, 1-\alpha, 0, 0, -1, 1, 0)$, activations $\mathfrak{M}_{\mathfrak{r},2}$ and $\mathrm{id}_{\mathbb{R}}$, and initial information $(0, 0)$ (cf. Definitions 1.2.1, 1.2.4, and 1.6.4). For every $T \in \mathbb{N}$, $\alpha \in (0,1)$ specify $R_{T,\alpha}(1, 1, \ldots, 1)$ explicitly and prove that your result is correct!

1.6.3 Long short-term memory (LSTM) RNNs

In this section we briefly discuss a very popular type of RNN nodes called *LSTM nodes* and the corresponding RNNs called *LSTM networks* which were introduced in Hochreiter & Schmidhuber [201]. Loosely speaking, LSTM nodes were invented to attempt to the tackle the issue that most RNNs based on simple RNN nodes, such as the simple fully-connected RNN nodes in Section 1.6.2 above, struggle to learn to understand long-term dependencies in sequences of data (cf., for example, [30, 328]). Roughly speaking, an RNN processes an input sequence by repeatedly applying an RNN node to a tuple consisting of a new element of the input sequence and a partial output of the previous application of the RNN node (see Definition 1.6.2 above for a precise description of RNNs). Therefore, the only information on previously processed elements of the input sequence that any application of an RNN node has access to, is the information encoded in the output produced by the last application of the RNN node. For this reason, RNNs can be seen as only having a *short-term memory*. The LSTM architecture, however is designed with the aim to facilitate the transmission of long-term information within this short-term memory. LSTM networks can thus be seen as having a sort of *long short-term memory*.

For a precise definition of LSTM networks we refer to the original article Hochreiter & Schmidhuber [201] and, for instance, to the excellent explanations in [133, 169, 319]. For a few selected references on LSTM networks in the literature we refer, for example, to [11, 77, 133, 147, 148, 169, 171–174, 288, 330, 360, 367, 388, 425] and the references therein.

1.7 Further types of ANNs

In this section we present a selection of references and some rough comments on a couple of further popular types of ANNs in the literature which were not discussed in the previous sections of this chapter above.

1.7.1 ANNs with encoder-decoder architectures: autoencoders

In this section we discuss the idea of autoencoders which are based on encoder-decoder ANN architectures. Roughly speaking, the goal of autoencoders is to learn a simplified representation of data points and a way to closely reconstruct the original data points from the simplified representation. The simplified representation of data points is usually called the *encoding* and is obtained by applying an *encoder ANN* to the data points. The approximate reconstruction of the original data points from the encoded representations is, in turn, called the *decoding* and is obtained by applying a *decoder ANN* to the encoded representations. The composition of the encoder ANN with the decoder ANN is called the *autoencoder*. In the simplest situations the encoder ANN and decoder ANN are trained to perform their respective desired functions by training the full autoencoder to be as close to the identity mapping on the data points as possible.

A large number of different architectures and training procedures for autoencoders have been proposed in the literature. In the following we list a selection of a few popular ideas from the scientific literature.

- We refer, for instance, to [49, 198, 200, 253, 356] for foundational references introducing and refining the idea of autoencoders,
- we refer, for example, to [402, 403, 416] for so-called *denoising autoencoders* which add random pertubation to the input data in the training of autoencoders,
- we refer, for instance, to [51, 107, 246] for so-called *variational autoencoders* which use techniques from bayesian statistics in the training of autoencoders,
- we refer, for example, [294, 349] for autoencoders involving convolutions, and
- we refer, for instance, [118, 292] for *adversarial autoencoders* which combine the principles of autoencoders with the paradigm of generative adversarial networks (see Goodfellow et al. [165]).

1.7.2 Transformers and the attention mechanism

In Section 1.6 we reviewed RNNs which are a type of ANNs designed to take sequences of data points as inputs. Very roughly speaking, RNNs process a sequence of data points by sequentially processing one data point of the sequence after the other and thereby constantly updating an information state encoding previously processed information (see Section 1.6.1 above for a precise description of RNNs). When processing a data point of the sequence, any information coming from earlier data points is thus only available to the RNN through the information state passed on from the previous processing step of the RNN. Consequently, it can be hard for RNNs to learn to understand long-term dependencies in the input sequence. In Section 1.6.3 above, we briefly discussed the LSTM architecture for RNNs which is an architecture for RNNs aimed at giving such RNNs the capacity to indeed learn to understand such long-term dependencies.

Another approach in the literature to design ANN architectures which process sequential data and are capable to efficiently learn to understand long-term dependencies in data sequences is called the *attention mechanism*. Very roughly speaking, in the context of sequences of the data, the attention mechanism aims to give ANNs the capacity to "*pay attention*" to selected parts of the entire input sequence when they are processing a data point of the sequence. The idea for using attention mechanisms in ANNs was first introduced in Bahdanau et al. [11] in the context of RNNs trained for machine translation. In this context the proposed ANN architecture still processes the input sequence sequentially, however past information is not only available through the information state from the previous processing step, but also through the attention mechanism, which can directly extract information from data points far away from the data point being processed.

Likely the most famous ANNs based on the attention mechanism do however not involve any recurrent elements and have been named *Transfomer ANNs* by the authors of the seminal paper Vaswani et al. [397] called "Attention is all you need". Roughly speaking, Transfomer ANNs are designed to process sequences of data by considering the entire input sequence at once and relying only on the attention mechanism to understand dependencies between the data points in the sequence. Transfomer ANNs are the basis for many recently very successful *large language models* (LLMs), such as, *generative pre-trained transformers* (GPTs) in [54, 320, 341, 342] which are the models behind the famous *ChatGPT* application, *Bidirectional Encoder Representations from Transformers* (BERT) models in Devlin et al. [104], and many others (cf., for example, [91, 267, 343, 418, 422] and the references therein).

Beyond the NLP applications for which Transformers and attention mechanisms have been introduced, similar ideas have been employed in several other areas, such as, computer vision (cf., for instance, [109, 240, 278, 404]), protein structure prediction (cf., for example, [232]), multimodal learning (cf., for instance, [283]), and long sequence time-series forecasting (cf., for example, [441]). Moreover, we refer, for instance, to [81, 288], [157, Chapter 17], and [164, Section 12.4.5.1] for explorations and explanations of the attention mechanism in the literature.

1.7.3 Graph neural networks (GNNs)

All ANNs reviewed in the previous sections of this book are designed to take real-valued vectors or sequences of real-valued vectors as inputs. However, there are several learning problems based on data, such as social network data or molecular data, that are not optimally represented by real-valued vectors but are better represented by graphs (see, for example, West [411] for an introduction on graphs). As a consequence, many ANN architectures which can process graphs as inputs, so-called *graph neural networks* (GNNs), have been introduced in the literature.

- We refer, for instance, to [362, 415, 439, 442] for overview articles on GNNs,
- we refer, for example, to [166, 366] for foundational articles for GNNs,
- we refer, for instance, to [399, 426] for applications of attention mechanisms (cf. Section 1.7.2 above) to GNNs,
- we refer, for example, to [55, 95, 412, 424] for GNNs involving convolutions on graphs, and
- we refer, for instance, to [16, 151, 361, 368, 414] for applications of GNNs to problems from the natural sciences.

1.7.4 Neural operators

In this section we review a few popular ANN-type architectures employed in *operator learning*. Roughly speaking, in operator learning one is not interested in learning a map between finite dimensional euclidean spaces, but in learning a map from a space of functions to a space of functions. Such a map between (typically infinite-dimensional) vector spaces is usually called an *operator*. An example of such a map is the solution operator of an evolutionary PDE which maps the initial condition of the PDE to the corresponding terminal value of the PDE. To approximate/learn operators it is necessary to develop parametrized families of operators, objects which we refer to as *neural operators*. Many different architectures for such neural operators have been proposed in the literature, some of which we now list in the next paragraphs.

One of the most successful neural operator architectures are so-called *Fourier neural operators* (FNOs) introduced in Li et al. [271] (cf. also Kovachki et al. [252]). Very roughly speaking, FNOs are parametric maps on function spaces, which involve transformations on function values as well as on Fourier coefficients. FNOs have been derived based on the neural operators introduced in Li et al. [270, 272] which are based on integral transformations with parametric integration kernels. We refer, for example, to [53, 251, 269, 410] and the references therein for extensions and theoretical results on FNOs.

A simple and successful architecture for neural operators, which is based on a universal approximation theorem for neural operators, are the *deep operator networks* (deepONets) introduced in Lu et al. [284]. Roughly speaking, a deepONet consists of two ANNs that take as input the evaluation point of the output space and input function values at predetermined "sensor" points respectively, and that are joined together by a scalar product to produce the output of the deepONet. We refer, for instance, to [115, 167, 249, 261, 276, 297, 335, 392, 406, 413, 432] for extensions and theoretical results on deepONets. For a comparison between deepONets and FNOs we refer, for example, to Lu et al. [285].

A further natural approach is to employ CNNs (see Section 1.4) to develop neural operator architectures. We refer, for instance, to [185, 192, 244, 350, 443] for such CNN-based neural operators. Finally, we refer, for example, to [67, 94, 98, 135, 136, 227, 273, 277, 301, 344, 369, 419] for further neural operator architectures and theoretical results for neural operators.

Chapter 2

ANN calculus

In this chapter we review certain operations that can be performed on the set of fully-connected feedforward ANNs such as compositions (see Section 2.1), paralellizations (see Section 2.2), scalar multiplications (see Section 2.3), and sums (see Section 2.4) and thereby review an appropriate calculus for fully-connected feedforward ANNs. The operations and the calculus for fully-connected feedforward ANNs presented in this chapter will be used in Chapters 3 and 4 to establish certain ANN approximation results.

In the literature such operations on ANNs and such kind of calculus on ANNs has been used in many research articles such as [128, 159, 180, 181, 184, 228, 321, 329, 333] and the references therein. The specific presentation of this chapter is based on Grohs et al. [180, 181].

2.1 Compositions of fully-connected feedforward ANNs

2.1.1 Compositions of fully-connected feedforward ANNs

Definition 2.1.1 (Composition of ANNs). *We denote by*

$$(\cdot) \bullet (\cdot) \colon \{(\Phi, \Psi) \in \mathbf{N} \times \mathbf{N} \colon \mathcal{I}(\Phi) = \mathcal{O}(\Psi)\} \to \mathbf{N} \tag{2.1}$$

the function which satisfies for all $\Phi, \Psi \in \mathbf{N}$, $k \in \{1, 2, \ldots, \mathcal{L}(\Phi) + \mathcal{L}(\Psi) - 1\}$ with $\mathcal{I}(\Phi) = \mathcal{O}(\Psi)$ that $\mathcal{L}(\Phi \bullet \Psi) = \mathcal{L}(\Phi) + \mathcal{L}(\Psi) - 1$ and

$$(\mathcal{W}_{k,\Phi\bullet\Psi}, \mathcal{B}_{k,\Phi\bullet\Psi}) = \begin{cases} (\mathcal{W}_{k,\Psi}, \mathcal{B}_{k,\Psi}) & : k < \mathcal{L}(\Psi) \\ (\mathcal{W}_{1,\Phi}\mathcal{W}_{\mathcal{L}(\Psi),\Psi}, \mathcal{W}_{1,\Phi}\mathcal{B}_{\mathcal{L}(\Psi),\Psi} + \mathcal{B}_{1,\Phi}) & : k = \mathcal{L}(\Psi) \\ (\mathcal{W}_{k-\mathcal{L}(\Psi)+1,\Phi}, \mathcal{B}_{k-\mathcal{L}(\Psi)+1,\Phi}) & : k > \mathcal{L}(\Psi) \end{cases} \tag{2.2}$$

(cf. Definition 1.3.1).

2.1.2 Elementary properties of compositions of fully-connected feedforward ANNs

Proposition 2.1.2 (Properties of standard compositions of fully-connected feedforward ANNs)**.** *Let $\Phi, \Psi \in \mathbf{N}$ satisfy $\mathcal{I}(\Phi) = \mathcal{O}(\Psi)$ (cf. Definition 1.3.1). Then*

(i) it holds that

$$\mathcal{D}(\Phi \bullet \Psi) = (\mathbb{D}_0(\Psi), \mathbb{D}_1(\Psi), \dots, \mathbb{D}_{\mathcal{H}(\Psi)}(\Psi), \mathbb{D}_1(\Phi), \mathbb{D}_2(\Phi), \dots, \mathbb{D}_{\mathcal{L}(\Phi)}(\Phi)), \tag{2.3}$$

(ii) it holds that

$$[\mathcal{L}(\Phi \bullet \Psi) - 1] = [\mathcal{L}(\Phi) - 1] + [\mathcal{L}(\Psi) - 1], \tag{2.4}$$

(iii) it holds that

$$\mathcal{H}(\Phi \bullet \Psi) = \mathcal{H}(\Phi) + \mathcal{H}(\Psi), \tag{2.5}$$

(iv) it holds that

$$\begin{aligned} \mathcal{P}(\Phi \bullet \Psi) &= \mathcal{P}(\Phi) + \mathcal{P}(\Psi) + \mathbb{D}_1(\Phi)(\mathbb{D}_{\mathcal{L}(\Psi)-1}(\Psi) + 1) \\ &\quad - \mathbb{D}_1(\Phi)(\mathbb{D}_0(\Phi) + 1) - \mathbb{D}_{\mathcal{L}(\Psi)}(\Psi)(\mathbb{D}_{\mathcal{L}(\Psi)-1}(\Psi) + 1) \\ &\leq \mathcal{P}(\Phi) + \mathcal{P}(\Psi) + \mathbb{D}_1(\Phi)\mathbb{D}_{\mathcal{H}(\Psi)}(\Psi), \end{aligned} \tag{2.6}$$

and

(v) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$ that $\mathcal{R}_a^{\mathbf{N}}(\Phi \bullet \Psi) \in C(\mathbb{R}^{\mathcal{I}(\Psi)}, \mathbb{R}^{\mathcal{O}(\Phi)})$ and

$$\mathcal{R}_a^{\mathbf{N}}(\Phi \bullet \Psi) = [\mathcal{R}_a^{\mathbf{N}}(\Phi)] \circ [\mathcal{R}_a^{\mathbf{N}}(\Psi)] \tag{2.7}$$

(cf. Definitions 1.3.4 and 2.1.1).

Proof of Proposition 2.1.2. Throughout this proof, let $L = \mathcal{L}(\Phi \bullet \Psi)$ and for every $a \in C(\mathbb{R}, \mathbb{R})$ let

$$\begin{aligned} X_a = \big\{ &x = (x_0, x_1, \dots, x_L) \in \mathbb{R}^{\mathbb{D}_0(\Phi\bullet\Psi)} \times \mathbb{R}^{\mathbb{D}_1(\Phi\bullet\Psi)} \times \dots \times \mathbb{R}^{\mathbb{D}_L(\Phi\bullet\Psi)} \colon \\ &\big(\forall\, k \in \{1, 2, \dots, L\} \colon x_k = \mathfrak{M}_{a\mathbb{1}_{(0,L)}(k) + \mathrm{id}_{\mathbb{R}}\mathbb{1}_{\{L\}}(k), \mathbb{D}_k(\Phi\bullet\Psi)}(\mathcal{W}_{k,\Phi\bullet\Psi} x_{k-1} + \mathcal{B}_{k,\Phi\bullet\Psi})\big)\big\}. \end{aligned} \tag{2.8}$$

Note that the fact that $\mathcal{L}(\Phi \bullet \Psi) = \mathcal{L}(\Phi) + \mathcal{L}(\Psi) - 1$ and the fact that for all $\Theta \in \mathbf{N}$ it holds that $\mathcal{H}(\Theta) = \mathcal{L}(\Theta) - 1$ establish items (ii) and (iii). Observe that item (iii) in Lemma 1.3.3 and (2.2) show that for all $k \in \{1, 2, \dots, L\}$ it holds that

$$\mathcal{W}_{k,\Phi\bullet\Psi} \in \begin{cases} \mathbb{R}^{\mathbb{D}_k(\Psi) \times \mathbb{D}_{k-1}(\Psi)} & : k < \mathcal{L}(\Psi) \\ \mathbb{R}^{\mathbb{D}_1(\Phi) \times \mathbb{D}_{\mathcal{L}(\Psi)-1}(\Psi)} & : k = \mathcal{L}(\Psi) \\ \mathbb{R}^{\mathbb{D}_{k-\mathcal{L}(\Psi)+1}(\Phi) \times \mathbb{D}_{k-\mathcal{L}(\Psi)}(\Phi)} & : k > \mathcal{L}(\Psi). \end{cases} \tag{2.9}$$

This, item (iii) in Lemma 1.3.3, and the fact that $\mathcal{H}(\Psi) = \mathcal{L}(\Psi) - 1$ ensure that for all $k \in \{0, 1, \dots, L\}$ it holds that

$$\mathbb{D}_k(\Phi \bullet \Psi) = \begin{cases} \mathbb{D}_k(\Psi) & : k \leq \mathcal{H}(\Psi) \\ \mathbb{D}_{k-\mathcal{L}(\Psi)+1}(\Phi) & : k > \mathcal{H}(\Psi). \end{cases} \tag{2.10}$$

This establishes item (i). Note that (2.10) implies that

$$\begin{split} \mathcal{P}(\Phi_1 \bullet \Phi_2) &= \sum_{j=1}^{L} \mathbb{D}_j(\Phi \bullet \Psi)(\mathbb{D}_{j-1}(\Phi \bullet \Psi) + 1) \\ &= \left[\sum_{j=1}^{\mathcal{H}(\Psi)} \mathbb{D}_j(\Psi)(\mathbb{D}_{j-1}(\Psi) + 1) \right] + \mathbb{D}_1(\Phi)(\mathbb{D}_{\mathcal{H}(\Psi)}(\Psi) + 1) \\ &\quad + \left[\sum_{j=\mathcal{L}(\Psi)+1}^{L} \mathbb{D}_{j-\mathcal{L}(\Psi)+1}(\Phi)(\mathbb{D}_{j-\mathcal{L}(\Psi)}(\Phi) + 1) \right] \\ &= \left[\sum_{j=1}^{\mathcal{L}(\Psi)-1} \mathbb{D}_j(\Psi)(\mathbb{D}_{j-1}(\Psi) + 1) \right] + \mathbb{D}_1(\Phi)(\mathbb{D}_{\mathcal{H}(\Psi)}(\Psi) + 1) \\ &\quad + \left[\sum_{j=2}^{\mathcal{L}(\Phi)} \mathbb{D}_j(\Phi)(\mathbb{D}_{j-1}(\Phi) + 1) \right] \\ &= \big[\mathcal{P}(\Psi) - \mathbb{D}_{\mathcal{L}(\Psi)}(\Psi)(\mathbb{D}_{\mathcal{L}(\Psi)-1}(\Psi) + 1)\big] + \mathbb{D}_1(\Phi)(\mathbb{D}_{\mathcal{H}(\Psi)}(\Psi) + 1) \\ &\quad + \big[\mathcal{P}(\Phi) - \mathbb{D}_1(\Phi)(\mathbb{D}_0(\Phi) + 1)\big]. \end{split} \tag{2.11}$$

This proves item (iv). Observe that (2.10) and item (ii) in Lemma 1.3.3 ensure that

$$\begin{split} &\mathcal{I}(\Phi \bullet \Psi) = \mathbb{D}_0(\Phi \bullet \Psi) = \mathbb{D}_0(\Psi) = \mathcal{I}(\Psi) \\ \text{and} \quad &\mathcal{O}(\Phi \bullet \Psi) = \mathbb{D}_{\mathcal{L}(\Phi\bullet\Psi)}(\Phi \bullet \Psi) = \mathbb{D}_{\mathcal{L}(\Phi\bullet\Psi)-\mathcal{L}(\Psi)+1}(\Phi) = \mathbb{D}_{\mathcal{L}(\Phi)}(\Phi) = \mathcal{O}(\Phi). \end{split} \tag{2.12}$$

This demonstrates that for all $a \in C(\mathbb{R}, \mathbb{R})$ it holds that

$$\mathcal{R}_a^{\mathbf{N}}(\Phi \bullet \Psi) \in C(\mathbb{R}^{\mathcal{I}(\Phi\bullet\Psi)}, \mathbb{R}^{\mathcal{O}(\Phi\bullet\Psi)}) = C(\mathbb{R}^{\mathcal{I}(\Psi)}, \mathbb{R}^{\mathcal{O}(\Phi)}). \tag{2.13}$$

Next note that (2.2) implies that for all $k \in \mathbb{N} \cap (1, \mathcal{L}(\Phi) + 1)$ it holds that

$$(\mathcal{W}_{\mathcal{L}(\Psi)+k-1,\Phi\bullet\Psi}, \mathcal{B}_{\mathcal{L}(\Psi)+k-1,\Phi\bullet\Psi}) = (\mathcal{W}_{k,\Phi}, \mathcal{B}_{k,\Phi}). \tag{2.14}$$

This and (2.10) ensure that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x = (x_0, x_1, \dots, x_L) \in X_a$, $k \in \mathbb{N} \cap (1, \mathcal{L}(\Phi) + 1)$ it holds that

$$\begin{split} x_{\mathcal{L}(\Psi)+k-1} &= \mathfrak{M}_{a\mathbb{1}_{(0,L)}(\mathcal{L}(\Psi)+k-1)+\mathrm{id}_{\mathbb{R}}\,\mathbb{1}_{\{L\}}(\mathcal{L}(\Psi)+k-1),\mathbb{D}_k(\Phi)}(\mathcal{W}_{k,\Phi}x_{\mathcal{L}(\Psi)+k-2} + \mathcal{B}_{k,\Phi}) \\ &= \mathfrak{M}_{a\mathbb{1}_{(0,\mathcal{L}(\Phi))}(k)+\mathrm{id}_{\mathbb{R}}\,\mathbb{1}_{\{\mathcal{L}(\Phi)\}}(k),\mathbb{D}_k(\Phi)}(\mathcal{W}_{k,\Phi}x_{\mathcal{L}(\Psi)+k-2} + \mathcal{B}_{k,\Phi}). \end{split} \tag{2.15}$$

Furthermore, observe that (2.2) and (2.10) show that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x = (x_0, x_1, \ldots, x_L) \in X_a$ it holds that

$$\begin{aligned} x_{\mathcal{L}(\Psi)} &= \mathfrak{M}_{a\mathbb{1}_{(0,L)}(\mathcal{L}(\Psi))+\mathrm{id}_{\mathbb{R}}\,\mathbb{1}_{\{L\}}(\mathcal{L}(\Psi)),\mathbb{D}_{\mathcal{L}(\Psi)}(\Phi\bullet\Psi)}(\mathcal{W}_{\mathcal{L}(\Psi),\Phi\bullet\Psi}x_{\mathcal{L}(\Psi)-1} + \mathcal{B}_{\mathcal{L}(\Psi),\Phi\bullet\Psi}) \\ &= \mathfrak{M}_{a\mathbb{1}_{(0,\mathcal{L}(\Phi))}(1)+\mathrm{id}_{\mathbb{R}}\,\mathbb{1}_{\{\mathcal{L}(\Phi)\}}(1),\mathbb{D}_1(\Phi)}(\mathcal{W}_{1,\Phi}\mathcal{W}_{\mathcal{L}(\Psi),\Psi}x_{\mathcal{L}(\Psi)-1} + \mathcal{W}_{1,\Phi}\mathcal{B}_{\mathcal{L}(\Psi),\Psi} + \mathcal{B}_{1,\Phi}) \\ &= \mathfrak{M}_{a\mathbb{1}_{(0,\mathcal{L}(\Phi))}(1)+\mathrm{id}_{\mathbb{R}}\,\mathbb{1}_{\{\mathcal{L}(\Phi)\}}(1),\mathbb{D}_1(\Phi)}(\mathcal{W}_{1,\Phi}(\mathcal{W}_{\mathcal{L}(\Psi),\Psi}x_{\mathcal{L}(\Psi)-1} + \mathcal{B}_{\mathcal{L}(\Psi),\Psi}) + \mathcal{B}_{1,\Phi}). \end{aligned} \tag{2.16}$$

Combining this and (2.15) proves that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x = (x_0, x_1, \ldots, x_L) \in X_a$ it holds that

$$(\mathcal{R}_a^{\mathbf{N}}(\Phi))(\mathcal{W}_{\mathcal{L}(\Psi),\Psi}x_{\mathcal{L}(\Psi)-1} + \mathcal{B}_{\mathcal{L}(\Psi),\Psi}) = x_L. \tag{2.17}$$

Moreover, note that (2.2) and (2.10) imply that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x = (x_0, x_1, \ldots, x_L) \in X_a$, $k \in \mathbb{N} \cap (0, \mathcal{L}(\Psi))$ it holds that

$$x_k = \mathfrak{M}_{a,\mathbb{D}_k(\Psi)}(\mathcal{W}_{k,\Psi}x_{k-1} + \mathcal{B}_{k,\Psi}) \tag{2.18}$$

This proves that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x = (x_0, x_1, \ldots, x_L) \in X_a$ it holds that

$$(\mathcal{R}_a^{\mathbf{N}}(\Psi))(x_0) = \mathcal{W}_{\mathcal{L}(\Psi),\Psi}x_{\mathcal{L}(\Psi)-1} + \mathcal{B}_{\mathcal{L}(\Psi),\Psi}. \tag{2.19}$$

Combining this with (2.17) demonstrates that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x = (x_0, x_1, \ldots, x_L) \in X_a$ it holds that

$$(\mathcal{R}_a^{\mathbf{N}}(\Phi))\big((\mathcal{R}_a^{\mathbf{N}}(\Psi))(x_0)\big) = x_L = \big(\mathcal{R}_a^{\mathbf{N}}(\Phi \bullet \Psi)\big)(x_0). \tag{2.20}$$

This and (2.13) prove item (v). The proof of Proposition 2.1.2 is thus complete. □

2.1.3 Associativity of compositions of fully-connected feedforward ANNs

Lemma 2.1.3. *Let $\Phi_1, \Phi_2, \Phi_3 \in \mathbf{N}$ satisfy $\mathcal{I}(\Phi_1) = \mathcal{O}(\Phi_2)$, $\mathcal{I}(\Phi_2) = \mathcal{O}(\Phi_3)$, and $\mathcal{L}(\Phi_2) = 1$ (cf. Definition 1.3.1). Then*

$$(\Phi_1 \bullet \Phi_2) \bullet \Phi_3 = \Phi_1 \bullet (\Phi_2 \bullet \Phi_3) \tag{2.21}$$

(cf. Definition 2.1.1).

Proof of Lemma 2.1.3. Observe that the fact that for all $\Psi_1, \Psi_2 \in \mathbf{N}$ with $\mathcal{I}(\Psi_1) = \mathcal{O}(\Psi_2)$ it holds that $\mathcal{L}(\Psi_1 \bullet \Psi_2) = \mathcal{L}(\Psi_1) + \mathcal{L}(\Psi_2) - 1$ and the assumption that $\mathcal{L}(\Phi_2) = 1$ ensure that

$$\mathcal{L}(\Phi_1 \bullet \Phi_2) = \mathcal{L}(\Phi_1) \qquad \text{and} \qquad \mathcal{L}(\Phi_2 \bullet \Phi_3) = \mathcal{L}(\Phi_3) \tag{2.22}$$

(cf. Definition 2.1.1). Therefore, we obtain that

$$\mathcal{L}((\Phi_1 \bullet \Phi_2) \bullet \Phi_3) = \mathcal{L}(\Phi_1) + \mathcal{L}(\Phi_3) = \mathcal{L}(\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)). \tag{2.23}$$

Next note that (2.22), (2.2), and the assumption that $\mathcal{L}(\Phi_2) = 1$ imply that for all $k \in \{1, 2, \dots, \mathcal{L}(\Phi_1)\}$ it holds that

$$(\mathcal{W}_{k,\Phi_1 \bullet \Phi_2}, \mathcal{B}_{k,\Phi_1 \bullet \Phi_2}) = \begin{cases} (\mathcal{W}_{1,\Phi_1}\mathcal{W}_{1,\Phi_2}, \mathcal{W}_{1,\Phi_1}\mathcal{B}_{1,\Phi_2} + \mathcal{B}_{1,\Phi_1}) & : k = 1 \\ (\mathcal{W}_{k,\Phi_1}, \mathcal{B}_{k,\Phi_1}) & : k > 1. \end{cases} \tag{2.24}$$

This, (2.2), and (2.23) prove that for all $k \in \{1, 2, \dots, \mathcal{L}(\Phi_1) + \mathcal{L}(\Phi_3) - 1\}$ it holds that

$$\begin{aligned} &(\mathcal{W}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}, \mathcal{B}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}) \\ &= \begin{cases} (\mathcal{W}_{k,\Phi_3}, \mathcal{B}_{k,\Phi_3}) & : k < \mathcal{L}(\Phi_3) \\ (\mathcal{W}_{1,\Phi_1 \bullet \Phi_2}\mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_3}, \mathcal{W}_{1,\Phi_1 \bullet \Phi_2}\mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_3} + \mathcal{B}_{1,\Phi_1 \bullet \Phi_2}) & : k = \mathcal{L}(\Phi_3) \\ (\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1 \bullet \Phi_2}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1 \bullet \Phi_2}) & : k > \mathcal{L}(\Phi_3) \end{cases} \\ &= \begin{cases} (\mathcal{W}_{k,\Phi_3}, \mathcal{B}_{k,\Phi_3}) & : k < \mathcal{L}(\Phi_3) \\ (\mathcal{W}_{1,\Phi_1 \bullet \Phi_2}\mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_3}, \mathcal{W}_{1,\Phi_1 \bullet \Phi_2}\mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_3} + \mathcal{B}_{1,\Phi_1 \bullet \Phi_2}) & : k = \mathcal{L}(\Phi_3) \\ (\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1}) & : k > \mathcal{L}(\Phi_3). \end{cases} \end{aligned} \tag{2.25}$$

Furthermore, observe that (2.2), (2.22), and (2.23) show that for all $k \in \{1, 2, \dots, \mathcal{L}(\Phi_1) + \mathcal{L}(\Phi_3) - 1\}$ it holds that

$$\begin{aligned} &(\mathcal{W}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}, \mathcal{B}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}) \\ &= \begin{cases} (\mathcal{W}_{k,\Phi_2 \bullet \Phi_3}, \mathcal{B}_{k,\Phi_2 \bullet \Phi_3}) & : k < \mathcal{L}(\Phi_2 \bullet \Phi_3) \\ (\mathcal{W}_{1,\Phi_1}\mathcal{W}_{\mathcal{L}(\Phi_2 \bullet \Phi_3),\Phi_2 \bullet \Phi_3}, \mathcal{W}_{1,\Phi}\mathcal{B}_{\mathcal{L}(\Phi_2 \bullet \Phi_3),\Phi_2 \bullet \Phi_3} + \mathcal{B}_{1,\Phi_1}) & : k = \mathcal{L}(\Phi_2 \bullet \Phi_3) \\ (\mathcal{W}_{k-\mathcal{L}(\Phi_2 \bullet \Phi_3)+1,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_2 \bullet \Phi_3)+1,\Phi_1}) & : k > \mathcal{L}(\Phi_2 \bullet \Phi_3) \end{cases} \\ &= \begin{cases} (\mathcal{W}_{k,\Phi_3}, \mathcal{B}_{k,\Phi_3}) & : k < \mathcal{L}(\Phi_3) \\ (\mathcal{W}_{1,\Phi_1}\mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_2 \bullet \Phi_3}, \mathcal{W}_{1,\Phi}\mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_2 \bullet \Phi_3} + \mathcal{B}_{1,\Phi_1}) & : k = \mathcal{L}(\Phi_3) \\ (\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1}) & : k > \mathcal{L}(\Phi_3). \end{cases} \end{aligned} \tag{2.26}$$

Combining this with (2.25) establishes that for all $k \in \{1, 2, \dots, \mathcal{L}(\Phi_1) + \mathcal{L}(\Phi_3) - 1\} \backslash \{\mathcal{L}(\Phi_3)\}$ it holds that

$$(\mathcal{W}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}, \mathcal{B}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}) = (\mathcal{W}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}, \mathcal{B}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}). \tag{2.27}$$

Moreover, note that (2.24) and (2.2) ensure that

$$\mathcal{W}_{1,\Phi_1 \bullet \Phi_2}\mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_3} = \mathcal{W}_{1,\Phi_1}\mathcal{W}_{1,\Phi_2}\mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_3} = \mathcal{W}_{1,\Phi_1}\mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_2 \bullet \Phi_3}. \tag{2.28}$$

In addition, observe that (2.24) and (2.2) demonstrate that

$$\begin{aligned} \mathcal{W}_{1,\Phi_1 \bullet \Phi_2}\mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_3} + \mathcal{B}_{1,\Phi_1 \bullet \Phi_2} &= \mathcal{W}_{1,\Phi_1}\mathcal{W}_{1,\Phi_2}\mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_3} + \mathcal{W}_{1,\Phi_1}\mathcal{B}_{1,\Phi_2} + \mathcal{B}_{1,\Phi_1} \\ &= \mathcal{W}_{1,\Phi_1}(\mathcal{W}_{1,\Phi_2}\mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_3} + \mathcal{B}_{1,\Phi_2}) + \mathcal{B}_{1,\Phi_1} \\ &= \mathcal{W}_{1,\Phi}\mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_2 \bullet \Phi_3} + \mathcal{B}_{1,\Phi_1}. \end{aligned} \tag{2.29}$$

Combining this and (2.28) with (2.27) proves that for all $k \in \{1, 2, \ldots, \mathcal{L}(\Phi_1) + \mathcal{L}(\Phi_3) - 1\}$ it holds that

$$(\mathcal{W}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}, \mathcal{B}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}) = (\mathcal{W}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}, \mathcal{B}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}). \tag{2.30}$$

This and (2.23) imply that

$$(\Phi_1 \bullet \Phi_2) \bullet \Phi_3 = \Phi_1 \bullet (\Phi_2 \bullet \Phi_3). \tag{2.31}$$

The proof of Lemma 2.1.3 is thus complete. □

Lemma 2.1.4. *Let $\Phi_1, \Phi_2, \Phi_3 \in \mathbf{N}$ satisfy $\mathcal{I}(\Phi_1) = \mathcal{O}(\Phi_2)$, $\mathcal{I}(\Phi_2) = \mathcal{O}(\Phi_3)$, and $\mathcal{L}(\Phi_2) > 1$ (cf. Definition 1.3.1). Then*

$$(\Phi_1 \bullet \Phi_2) \bullet \Phi_3 = \Phi_1 \bullet (\Phi_2 \bullet \Phi_3) \tag{2.32}$$

(cf. Definition 2.1.1).

Proof of Lemma 2.1.4. Note that the fact that for all $\Psi, \Theta \in \mathbf{N}$ it holds that $\mathcal{L}(\Psi \bullet \Theta) = \mathcal{L}(\Psi) + \mathcal{L}(\Theta) - 1$ ensures that

$$\begin{aligned} \mathcal{L}((\Phi_1 \bullet \Phi_2) \bullet \Phi_3) &= \mathcal{L}(\Phi_1 \bullet \Phi_2) + \mathcal{L}(\Phi_3) - 1 \\ &= \mathcal{L}(\Phi_1) + \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 2 \\ &= \mathcal{L}(\Phi_1) + \mathcal{L}(\Phi_2 \bullet \Phi_3) - 1 \\ &= \mathcal{L}(\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)) \end{aligned} \tag{2.33}$$

(cf. Definition 2.1.1). Furthermore, observe that (2.2) shows that for all $k \in \{1, 2, \ldots, \mathcal{L}((\Phi_1 \bullet \Phi_2) \bullet \Phi_3)\}$ it holds that

$$\begin{aligned} &(\mathcal{W}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}, \mathcal{B}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}) \\ &= \begin{cases} (\mathcal{W}_{k,\Phi_3}, \mathcal{B}_{k,\Phi_3}) & : k < \mathcal{L}(\Phi_3) \\ (\mathcal{W}_{1,\Phi_1 \bullet \Phi_2} \mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_3}, \mathcal{W}_{1,\Phi_1 \bullet \Phi_2} \mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_3} + \mathcal{B}_{1,\Phi_1 \bullet \Phi_2}) & : k = \mathcal{L}(\Phi_3) \\ (\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1 \bullet \Phi_2}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1 \bullet \Phi_2}) & : k > \mathcal{L}(\Phi_3). \end{cases} \end{aligned} \tag{2.34}$$

Moreover, note that (2.2) and the assumption that $\mathcal{L}(\Phi_2) > 1$ ensure that for all $k \in \mathbb{N} \cap (\mathcal{L}(\Phi_3), \mathcal{L}((\Phi_1 \bullet \Phi_2) \bullet \Phi_3)]$ it holds that

$$\begin{aligned} &(\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1 \bullet \Phi_2}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_1 \bullet \Phi_2}) \\ &= \begin{cases} (\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_2}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_2}) & : k - \mathcal{L}(\Phi_3) + 1 < \mathcal{L}(\Phi_2) \\ (\mathcal{W}_{1,\Phi_1} \mathcal{W}_{\mathcal{L}(\Phi_2),\Phi_2}, \mathcal{W}_{1,\Phi_1} \mathcal{B}_{\mathcal{L}(\Phi_2),\Phi_2} + \mathcal{B}_{1,\Phi_1}) & : k - \mathcal{L}(\Phi_3) + 1 = \mathcal{L}(\Phi_2) \\ (\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1-\mathcal{L}(\Phi_2)+1,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1-\mathcal{L}(\Phi_2)+1,\Phi_1}) & : k - \mathcal{L}(\Phi_3) + 1 > \mathcal{L}(\Phi_2) \end{cases} \\ &= \begin{cases} (\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_2}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_2}) & : k < \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1 \\ (\mathcal{W}_{1,\Phi_1} \mathcal{W}_{\mathcal{L}(\Phi_2),\Phi_2}, \mathcal{W}_{1,\Phi_1} \mathcal{B}_{\mathcal{L}(\Phi_2),\Phi_2} + \mathcal{B}_{1,\Phi_1}) & : k = \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1 \\ (\mathcal{W}_{k-\mathcal{L}(\Phi_3)-\mathcal{L}(\Phi_2)+2,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)-\mathcal{L}(\Phi_2)+2,\Phi_1}) & : k > \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1. \end{cases} \end{aligned} \tag{2.35}$$

Combining this with (2.34) proves that for all $k \in \{1, 2, \dots, \mathcal{L}((\Phi_1 \bullet \Phi_2) \bullet \Phi_3)\}$ it holds that

$$
\begin{split}
&(\mathcal{W}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}, \mathcal{B}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}) \\
&= \begin{cases}
(\mathcal{W}_{k,\Phi_3}, \mathcal{B}_{k,\Phi_3}) & : k < \mathcal{L}(\Phi_3) \\
(\mathcal{W}_{1,\Phi_2} \mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_3}, \mathcal{W}_{1,\Phi_2} \mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_3} + \mathcal{B}_{1,\Phi_2}) & : k = \mathcal{L}(\Phi_3) \\
(\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_2}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_2}) & : \mathcal{L}(\Phi_3) < k < \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1 \\
(\mathcal{W}_{1,\Phi_1} \mathcal{W}_{\mathcal{L}(\Phi_2),\Phi_2}, \mathcal{W}_{1,\Phi_1} \mathcal{B}_{\mathcal{L}(\Phi_2),\Phi_2} + \mathcal{B}_{1,\Phi_1}) & : k = \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1 \\
(\mathcal{W}_{k-\mathcal{L}(\Phi_3)-\mathcal{L}(\Phi_2)+2,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)-\mathcal{L}(\Phi_2)+2,\Phi_1}) & : k > \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1.
\end{cases}
\end{split}
\tag{2.36}
$$

In addition, observe that (2.2), the fact that $\mathcal{L}(\Phi_2 \bullet \Phi_3) = \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1$, and the assumption that $\mathcal{L}(\Phi_2) > 1$ demonstrate that for all $k \in \{1, 2, \dots, \mathcal{L}(\Phi_1 \bullet (\Phi_2 \bullet \Phi_3))\}$ it holds that

$$
\begin{split}
&(\mathcal{W}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}, \mathcal{B}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}) \\
&= \begin{cases}
(\mathcal{W}_{k,\Phi_2 \bullet \Phi_3}, \mathcal{B}_{k,\Phi_2 \bullet \Phi_3}) & : k < \mathcal{L}(\Phi_2 \bullet \Phi_3) \\
(\mathcal{W}_{1,\Phi_1} \mathcal{W}_{\mathcal{L}(\Phi_2 \bullet \Phi_3),\Phi_2 \bullet \Phi_3}, \mathcal{W}_{1,\Phi} \mathcal{B}_{\mathcal{L}(\Phi_2 \bullet \Phi_3),\Phi_2 \bullet \Phi_3} + \mathcal{B}_{1,\Phi_1}) & : k = \mathcal{L}(\Phi_2 \bullet \Phi_3) \\
(\mathcal{W}_{k-\mathcal{L}(\Phi_2 \bullet \Phi_3)+1,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_2 \bullet \Phi_3)+1,\Phi_1}) & : k > \mathcal{L}(\Phi_2 \bullet \Phi_3)
\end{cases} \\
&= \begin{cases}
(\mathcal{W}_{k,\Phi_2 \bullet \Phi_3}, \mathcal{B}_{k,\Phi_2 \bullet \Phi_3}) & : k < \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1 \\
\begin{aligned}(&\mathcal{W}_{1,\Phi_1} \mathcal{W}_{\mathcal{L}(\Phi_2)+\mathcal{L}(\Phi_3)-1,\Phi_2 \bullet \Phi_3}, \\ &\quad \mathcal{W}_{1,\Phi} \mathcal{B}_{\mathcal{L}(\Phi_2)+\mathcal{L}(\Phi_3)-1,\Phi_2 \bullet \Phi_3} + \mathcal{B}_{1,\Phi_1})\end{aligned} & : k = \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1 \\
(\mathcal{W}_{k-\mathcal{L}(\Phi_2)-\mathcal{L}(\Phi_3)+2,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_2)-\mathcal{L}(\Phi_3)+2,\Phi_1}) & : k > \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1
\end{cases} \\
&= \begin{cases}
(\mathcal{W}_{k,\Phi_3}, \mathcal{B}_{k,\Phi_3}) & : k < \mathcal{L}(\Phi_3) \\
(\mathcal{W}_{1,\Phi_2} \mathcal{W}_{\mathcal{L}(\Phi_3),\Phi_3}, \mathcal{W}_{1,\Phi_2} \mathcal{B}_{\mathcal{L}(\Phi_3),\Phi_3} + \mathcal{B}_{1,\Phi_2}) & : k = \mathcal{L}(\Phi_3) \\
(\mathcal{W}_{k-\mathcal{L}(\Phi_3)+1,\Phi_2}, \mathcal{B}_{k-\mathcal{L}(\Phi_3)+1,\Phi_2}) & : \mathcal{L}(\Phi_3) < k < \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1 \\
(\mathcal{W}_{1,\Phi_1} \mathcal{W}_{\mathcal{L}(\Phi_2),\Phi_2}, \mathcal{W}_{1,\Phi} \mathcal{B}_{\mathcal{L}(\Phi_2),\Phi_2} + \mathcal{B}_{1,\Phi_1}) & : k = \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1 \\
(\mathcal{W}_{k-\mathcal{L}(\Phi_2)-\mathcal{L}(\Phi_3)+2,\Phi_1}, \mathcal{B}_{k-\mathcal{L}(\Phi_2)-\mathcal{L}(\Phi_3)+2,\Phi_1}) & : k > \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 1.
\end{cases}
\end{split}
\tag{2.37}
$$

This, (2.36), and (2.33) establish that for all $k \in \{1, 2, \dots, \mathcal{L}(\Phi_1) + \mathcal{L}(\Phi_2) + \mathcal{L}(\Phi_3) - 2\}$ it holds that

$$
(\mathcal{W}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}, \mathcal{B}_{k,(\Phi_1 \bullet \Phi_2) \bullet \Phi_3}) = (\mathcal{W}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}, \mathcal{B}_{k,\Phi_1 \bullet (\Phi_2 \bullet \Phi_3)}).
\tag{2.38}
$$

Hence, we obtain that

$$
(\Phi_1 \bullet \Phi_2) \bullet \Phi_3 = \Phi_1 \bullet (\Phi_2 \bullet \Phi_3).
\tag{2.39}
$$

The proof of Lemma 2.1.4 is thus complete. □

Corollary 2.1.5. *Let $\Phi_1, \Phi_2, \Phi_3 \in \mathbf{N}$ satisfy $\mathcal{I}(\Phi_1) = \mathcal{O}(\Phi_2)$ and $\mathcal{I}(\Phi_2) = \mathcal{O}(\Phi_3)$ (cf. Definition 1.3.1). Then*

$$(\Phi_1 \bullet \Phi_2) \bullet \Phi_3 = \Phi_1 \bullet (\Phi_2 \bullet \Phi_3) \tag{2.40}$$

(cf. Definition 2.1.1).

Proof of Corollary 2.1.5. Note that Lemma 2.1.3 and Lemma 2.1.4 establish (2.40). The proof of Corollary 2.1.5 is thus complete. □

2.1.4 Powers of fully-connected feedforward ANNs

Definition 2.1.6 (Powers of fully-connected feedforward ANNs)**.** *We denote by $(\cdot)^{\bullet n} \colon \{\Phi \in \mathbf{N} \colon \mathcal{I}(\Phi) = \mathcal{O}(\Phi)\} \to \mathbf{N}$, $n \in \mathbb{N}_0$, the functions which satisfy for all $n \in \mathbb{N}_0$, $\Phi \in \mathbf{N}$ with $\mathcal{I}(\Phi) = \mathcal{O}(\Phi)$ that*

$$\Phi^{\bullet n} = \begin{cases} \big(\mathrm{I}_{\mathcal{O}(\Phi)}, (0,0,\dots,0)\big) \in \mathbb{R}^{\mathcal{O}(\Phi)\times\mathcal{O}(\Phi)} \times \mathbb{R}^{\mathcal{O}(\Phi)} & : n = 0 \\ \Phi \bullet (\Phi^{\bullet(n-1)}) & : n \in \mathbb{N} \end{cases} \tag{2.41}$$

(cf. Definitions 1.3.1, 1.5.5, and 2.1.1).

Lemma 2.1.7 (Number of hidden layers of powers of ANNs)**.** *Let $n \in \mathbb{N}_0$, $\Phi \in \mathbf{N}$ satisfy $\mathcal{I}(\Phi) = \mathcal{O}(\Phi)$ (cf. Definition 1.3.1). Then*

$$\mathcal{H}(\Phi^{\bullet n}) = n\mathcal{H}(\Phi) \tag{2.42}$$

(cf. Definition 2.1.6).

Proof of Lemma 2.1.7. Observe that Proposition 2.1.2, (2.41), and induction establish (2.42). The proof of Lemma 2.1.7 is thus complete. □

2.2 Parallelizations of fully-connected feedforward ANNs

2.2.1 Parallelizations of fully-connected feedforward ANNs with the same length

Definition 2.2.1 (Parallelization of fully-connected feedforward ANNs)**.** *Let $n \in \mathbb{N}$. Then we denote by*

$$\mathbf{P}_n \colon \big\{\Phi = (\Phi_1, \dots, \Phi_n) \in \mathbf{N}^n \colon \mathcal{L}(\Phi_1) = \mathcal{L}(\Phi_2) = \ldots = \mathcal{L}(\Phi_n)\big\} \to \mathbf{N} \tag{2.43}$$

the function which satisfies for all $\Phi = (\Phi_1, \dots, \Phi_n) \in \mathbf{N}^n$, $k \in \{1, 2, \dots, \mathcal{L}(\Phi_1)\}$ *with* $\mathcal{L}(\Phi_1) = \mathcal{L}(\Phi_2) = \dots = \mathcal{L}(\Phi_n)$ *that*

$$\mathcal{L}(\mathbf{P}_n(\Phi)) = \mathcal{L}(\Phi_1), \qquad \mathcal{W}_{k,\mathbf{P}_n(\Phi)} = \begin{pmatrix} \mathcal{W}_{k,\Phi_1} & 0 & 0 & \cdots & 0 \\ 0 & \mathcal{W}_{k,\Phi_2} & 0 & \cdots & 0 \\ 0 & 0 & \mathcal{W}_{k,\Phi_3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \mathcal{W}_{k,\Phi_n} \end{pmatrix},$$

$$\text{and} \qquad \mathcal{B}_{k,\mathbf{P}_n(\Phi)} = \begin{pmatrix} \mathcal{B}_{k,\Phi_1} \\ \mathcal{B}_{k,\Phi_2} \\ \vdots \\ \mathcal{B}_{k,\Phi_n} \end{pmatrix} \tag{2.44}$$

(cf. Definition 1.3.1).

Lemma 2.2.2 (Architectures of parallelizations of fully-connected feedforward ANNs)**.** *Let* $n, L \in \mathbb{N}$, $\Phi = (\Phi_1, \dots, \Phi_n) \in \mathbf{N}^n$ *satisfy* $L = \mathcal{L}(\Phi_1) = \mathcal{L}(\Phi_2) = \ldots = \mathcal{L}(\Phi_n)$ *(cf. Definition 1.3.1). Then*

(i) it holds that

$$\mathbf{P}_n(\Phi) \in \left(\bigtimes_{k=1}^{L} \left(\mathbb{R}^{(\sum_{j=1}^n \mathbb{D}_k(\Phi_j)) \times (\sum_{j=1}^n \mathbb{D}_{k-1}(\Phi_j))} \times \mathbb{R}^{(\sum_{j=1}^n \mathbb{D}_k(\Phi_j))} \right) \right), \tag{2.45}$$

(ii) it holds for all $k \in \mathbb{N}_0$ *that*

$$\mathbb{D}_k(\mathbf{P}_n(\Phi)) = \mathbb{D}_k(\Phi_1) + \mathbb{D}_k(\Phi_2) + \ldots + \mathbb{D}_k(\Phi_n), \tag{2.46}$$

and

(iii) it holds that

$$\mathcal{D}\big(\mathbf{P}_n(\Phi)\big) = \mathcal{D}(\Phi_1) + \mathcal{D}(\Phi_2) + \ldots + \mathcal{D}(\Phi_n) \tag{2.47}$$

(cf. Definition 2.2.1).

Proof of Lemma 2.2.2. Note that item (iii) in Lemma 1.3.3 and (2.44) imply that for all $k \in \{1, 2, \dots, L\}$ it holds that

$$\mathcal{W}_{k,\mathbf{P}_n(\Phi)} \in \mathbb{R}^{(\sum_{j=1}^n \mathbb{D}_k(\Phi_j)) \times (\sum_{j=1}^n \mathbb{D}_{k-1}(\Phi_j))} \qquad \text{and} \qquad \mathcal{B}_{k,\mathbf{P}_n(\Phi)} \in \mathbb{R}^{(\sum_{j=1}^n \mathbb{D}_{k-1}(\Phi_j))} \tag{2.48}$$

(cf. Definition 2.2.1). Item (iii) in Lemma 1.3.3 therefore establishes items (i) and (ii). Note that item (ii) implies item (iii). The proof of Lemma 2.2.2 is thus complete. $\square$

Proposition 2.2.3 (Realizations of parallelizations of fully-connected feedforward ANNs)**.** *Let $a \in C(\mathbb{R}, \mathbb{R})$, $n \in \mathbb{N}$, $\Phi = (\Phi_1, \ldots, \Phi_n) \in \mathbf{N}^n$ satisfy $\mathcal{L}(\Phi_1) = \mathcal{L}(\Phi_2) = \cdots = \mathcal{L}(\Phi_n)$ (cf. Definition 1.3.1). Then*

(i) it holds that

$$\mathcal{R}_a^{\mathbf{N}}(\mathbf{P}_n(\Phi)) \in C\big(\mathbb{R}^{[\sum_{j=1}^n \mathcal{I}(\Phi_j)]}, \mathbb{R}^{[\sum_{j=1}^n \mathcal{O}(\Phi_j)]}\big) \tag{2.49}$$

and

(ii) it holds for all $x_1 \in \mathbb{R}^{\mathcal{I}(\Phi_1)}, x_2 \in \mathbb{R}^{\mathcal{I}(\Phi_2)}, \ldots, x_n \in \mathbb{R}^{\mathcal{I}(\Phi_n)}$ that

$$\begin{split} &\big(\mathcal{R}_a^{\mathbf{N}}\big(\mathbf{P}_n(\Phi)\big)\big)(x_1, x_2, \ldots, x_n) \\ &= \big((\mathcal{R}_a^{\mathbf{N}}(\Phi_1))(x_1), (\mathcal{R}_a^{\mathbf{N}}(\Phi_2))(x_2), \ldots, (\mathcal{R}_a^{\mathbf{N}}(\Phi_n))(x_n)\big) \in \mathbb{R}^{[\sum_{j=1}^n \mathcal{O}(\Phi_j)]} \end{split} \tag{2.50}$$

(cf. Definitions 1.3.4 and 2.2.1).

Proof of Proposition 2.2.3. Throughout this proof, let $L = \mathcal{L}(\Phi_1)$, for every $j \in \{1, 2, \ldots, n\}$ let

$$\begin{split} X^j = \big\{&x = (x_0, x_1, \ldots, x_L) \in \mathbb{R}^{\mathbb{D}_0(\Phi_j)} \times \mathbb{R}^{\mathbb{D}_1(\Phi_j)} \times \cdots \times \mathbb{R}^{\mathbb{D}_L(\Phi_j)} \colon \\ &\big(\forall\, k \in \{1, 2, \ldots, L\} \colon x_k = \mathfrak{M}_{a\mathbb{1}_{(0,L)}(k) + \mathrm{id}_{\mathbb{R}} \mathbb{1}_{\{L\}}(k), \mathbb{D}_k(\Phi_j)}(\mathcal{W}_{k,\Phi_j} x_{k-1} + \mathcal{B}_{k,\Phi_j})\big)\big\}, \end{split} \tag{2.51}$$

and let

$$\begin{split} \mathfrak{X} = \big\{&\mathfrak{x} = (\mathfrak{x}_0, \mathfrak{x}_1, \ldots, \mathfrak{x}_L) \in \mathbb{R}^{\mathbb{D}_0(\mathbf{P}_n(\Phi))} \times \mathbb{R}^{\mathbb{D}_1(\mathbf{P}_n(\Phi))} \times \cdots \times \mathbb{R}^{\mathbb{D}_L(\mathbf{P}_n(\Phi))} \colon \\ &\big(\forall\, k \in \{1, 2, \ldots, L\} \colon \mathfrak{x}_k = \mathfrak{M}_{a\mathbb{1}_{(0,L)}(k) + \mathrm{id}_{\mathbb{R}} \mathbb{1}_{\{L\}}(k), \mathbb{D}_k(\mathbf{P}_n(\Phi))}(\mathcal{W}_{k,\mathbf{P}_n(\Phi)} \mathfrak{x}_{k-1} + \mathcal{B}_{k,\mathbf{P}_n(\Phi)})\big)\big\}. \end{split} \tag{2.52}$$

Observe that item (ii) in Lemma 2.2.2 and item (ii) in Lemma 1.3.3 imply that

$$\mathcal{I}(\mathbf{P}_n(\Phi)) = \mathbb{D}_0(\mathbf{P}_n(\Phi)) = \sum_{j=1}^n \mathbb{D}_0(\Phi_n) = \sum_{j=1}^n \mathcal{I}(\Phi_n). \tag{2.53}$$

Furthermore, note that item (ii) in Lemma 2.2.2 and item (ii) in Lemma 1.3.3 ensure that

$$\mathcal{O}(\mathbf{P}_n(\Phi)) = \mathbb{D}_{\mathcal{L}(\mathbf{P}_n(\Phi))}(\mathbf{P}_n(\Phi)) = \sum_{j=1}^n \mathbb{D}_{\mathcal{L}(\Phi_n)}(\Phi_n) = \sum_{j=1}^n \mathcal{O}(\Phi_n). \tag{2.54}$$

Observe that (2.44) and item (ii) in Lemma 2.2.2 show that for all $\mathscr{a} \in C(\mathbb{R}, \mathbb{R})$, $k \in \{1, 2, \ldots, L\}$, $x^1 \in \mathbb{R}^{\mathbb{D}_k(\Phi_1)}$, $x^2 \in \mathbb{R}^{\mathbb{D}_k(\Phi_2)}$, $\ldots$, $x^n \in \mathbb{R}^{\mathbb{D}_k(\Phi_n)}$, $\mathfrak{x} \in \mathbb{R}^{[\sum_{j=1}^n \mathbb{D}_k(\Phi_j)]}$ with $\mathfrak{x} =$

$(x^1, x^2, \ldots, x^n)$ it holds that

$$
\begin{split}
&\mathfrak{M}_{a,\mathbb{D}_k(\mathbf{P}_n(\Phi))}(\mathcal{W}_{k,\mathbf{P}_n(\Phi)}\mathfrak{x} + \mathcal{B}_{k,\mathbf{P}_n(\Phi)}) \\
&= \mathfrak{M}_{a,\mathbb{D}_k(\mathbf{P}_n(\Phi))}\left(\begin{pmatrix} \mathcal{W}_{k,\Phi_1} & 0 & 0 & \cdots & 0 \\ 0 & \mathcal{W}_{k,\Phi_2} & 0 & \cdots & 0 \\ 0 & 0 & \mathcal{W}_{k,\Phi_3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \mathcal{W}_{k,\Phi_n} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} \mathcal{B}_{k,\Phi_1} \\ \mathcal{B}_{k,\Phi_2} \\ \mathcal{B}_{k,\Phi_3} \\ \vdots \\ \mathcal{B}_{k,\Phi_n} \end{pmatrix}\right) \\
&= \mathfrak{M}_{a,\mathbb{D}_k(\mathbf{P}_n(\Phi))}\left(\begin{pmatrix} \mathcal{W}_{k,\Phi_1}x_1 + \mathcal{B}_{k,\Phi_1} \\ \mathcal{W}_{k,\Phi_2}x_2 + \mathcal{B}_{k,\Phi_2} \\ \mathcal{W}_{k,\Phi_3}x_3 + \mathcal{B}_{k,\Phi_3} \\ \vdots \\ \mathcal{W}_{k,\Phi_n}x_n + \mathcal{B}_{k,\Phi_n} \end{pmatrix}\right) = \begin{pmatrix} \mathfrak{M}_{a,\mathbb{D}_k(\Phi_1)}(\mathcal{W}_{k,\Phi_1}x_1 + \mathcal{B}_{k,\Phi_1}) \\ \mathfrak{M}_{a,\mathbb{D}_k(\Phi_2)}(\mathcal{W}_{k,\Phi_2}x_2 + \mathcal{B}_{k,\Phi_2}) \\ \mathfrak{M}_{a,\mathbb{D}_k(\Phi_3)}(\mathcal{W}_{k,\Phi_3}x_3 + \mathcal{B}_{k,\Phi_3}) \\ \vdots \\ \mathfrak{M}_{a,\mathbb{D}_k(\Phi_n)}(\mathcal{W}_{k,\Phi_n}x_n + \mathcal{B}_{k,\Phi_n}) \end{pmatrix}.
\end{split}
\tag{2.55}
$$

This proves that for all $k \in \{1, 2, \ldots, L\}$, $\mathfrak{x} = (\mathfrak{x}_0, \mathfrak{x}_1, \ldots, \mathfrak{x}_L) \in \mathfrak{X}$, $x^1 = (x^1_0, x^1_1, \ldots, x^1_L) \in X^1$, $x^2 = (x^2_0, x^2_1, \ldots, x^2_L) \in X^2$, $\ldots$, $x^n = (x^n_0, x^n_1, \ldots, x^n_L) \in X^n$ with $\mathfrak{x}_{k-1} = (x^1_{k-1}, x^2_{k-1}, \ldots, x^n_{k-1})$ it holds that

$$
\mathfrak{x}_k = (x^1_k, x^2_k, \ldots, x^n_k). \tag{2.56}
$$

Induction, and (1.91) hence demonstrate that for all $k \in \{1, 2, \ldots, L\}$, $\mathfrak{x} = (\mathfrak{x}_0, \mathfrak{x}_1, \ldots, \mathfrak{x}_L) \in \mathfrak{X}$, $x^1 = (x^1_0, x^1_1, \ldots, x^1_L) \in X^1$, $x^2 = (x^2_0, x^2_1, \ldots, x^2_L) \in X^2$, $\ldots$, $x^n = (x^n_0, x^n_1, \ldots, x^n_L) \in X^n$ with $\mathfrak{x}_0 = (x^1_0, x^2_0, \ldots, x^n_0)$ it holds that

$$
\begin{split}
\big(\mathcal{R}^{\mathbf{N}}_a(\mathbf{P}_n(\Phi))\big)(\mathfrak{x}_0) &= \mathfrak{x}_L = (x^1_L, x^2_L, \ldots, x^n_L) \\
&= \big((\mathcal{R}^{\mathbf{N}}_a(\Phi_1))(x^1_0), (\mathcal{R}^{\mathbf{N}}_a(\Phi_2))(x^2_0), \ldots, (\mathcal{R}^{\mathbf{N}}_a(\Phi_n))(x^n_0)\big).
\end{split}
\tag{2.57}
$$

This establishes item (ii). The proof of Proposition 2.2.3 is thus complete. □

Proposition 2.2.4 (Upper bounds for the numbers of parameters of parallelizations of fully-connected feedforward ANNs)**.** *Let $n, L \in \mathbb{N}$, $\Phi_1, \Phi_2, \ldots, \Phi_n \in \mathbf{N}$ satisfy $L = \mathcal{L}(\Phi_1) = \mathcal{L}(\Phi_2) = \ldots = \mathcal{L}(\Phi_n)$ (cf. Definition 1.3.1). Then*

$$
\mathcal{P}\big(\mathbf{P}_n(\Phi_1, \Phi_2, \ldots, \Phi_n)\big) \le \tfrac{1}{2}\big[\textstyle\sum_{j=1}^{n} \mathcal{P}(\Phi_j)\big]^2 \tag{2.58}
$$

(cf. Definition 2.2.1).

Proof of Proposition 2.2.4. Throughout this proof, for every $j \in \{1, 2, \ldots, n\}$, $k \in \{0, 1,$

$\ldots, L\}$ let $l_{j,k} = \mathbb{D}_k(\Phi_j)$. Note that item (ii) in Lemma 2.2.2 demonstrates that

$$\begin{aligned}
&\mathcal{P}(\mathbf{P}_n(\Phi_1, \Phi_2, \ldots, \Phi_n)) = \sum_{k=1}^{L} \Big[\textstyle\sum_{i=1}^{n} l_{i,k}\Big]\Big[\big(\textstyle\sum_{i=1}^{n} l_{i,k-1}\big) + 1\Big] \\
&= \sum_{k=1}^{L} \Big[\textstyle\sum_{i=1}^{n} l_{i,k}\Big]\Big[\big(\textstyle\sum_{j=1}^{n} l_{j,k-1}\big) + 1\Big] \\
&\leq \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{L} l_{i,k}(l_{j,k-1}+1) \leq \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k,\ell=1}^{L} l_{i,k}(l_{j,\ell-1}+1) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} \Big[\textstyle\sum_{k=1}^{L} l_{i,k}\Big]\Big[\textstyle\sum_{\ell=1}^{L} (l_{j,\ell-1}+1)\Big] \\
&\leq \sum_{i=1}^{n}\sum_{j=1}^{n} \Big[\textstyle\sum_{k=1}^{L} \frac{1}{2} l_{i,k}(l_{i,k-1}+1)\Big]\Big[\textstyle\sum_{\ell=1}^{L} l_{j,\ell}(l_{j,\ell-1}+1)\Big] \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} \tfrac{1}{2}\mathcal{P}(\Phi_i)\mathcal{P}(\Phi_j) = \tfrac{1}{2}\Big[\textstyle\sum_{i=1}^{n} \mathcal{P}(\Phi_i)\Big]^2.
\end{aligned} \tag{2.59}$$

The proof of Proposition 2.2.4 is thus complete. □

Corollary 2.2.5 (Lower and upper bounds for the numbers of parameters of parallelizations of fully-connected feedforward ANNs)**.** *Let $n \in \mathbb{N}$, $\Phi = (\Phi_1, \ldots, \Phi_n) \in \mathbf{N}^n$ satisfy $\mathcal{D}(\Phi_1) = \mathcal{D}(\Phi_2) = \ldots = \mathcal{D}(\Phi_n)$ (cf. Definition 1.3.1). Then*

$$\big[\tfrac{n^2}{2}\big]\mathcal{P}(\Phi_1) \leq \big[\tfrac{n^2+n}{2}\big]\mathcal{P}(\Phi_1) \leq \mathcal{P}(\mathbf{P}_n(\Phi)) \leq n^2\mathcal{P}(\Phi_1) \leq \tfrac{1}{2}\big[\textstyle\sum_{i=1}^{n} \mathcal{P}(\Phi_i)\big]^2 \tag{2.60}$$

(cf. Definition 2.2.1).

Proof of Corollary 2.2.5. Throughout this proof, let $L \in \mathbb{N}$, $l_0, l_1, \ldots, l_L \in \mathbb{N}$ satisfy

$$\mathcal{D}(\Phi_1) = (l_0, l_1, \ldots, l_L). \tag{2.61}$$

Observe that (2.61) and the assumption that $\mathcal{D}(\Phi_1) = \mathcal{D}(\Phi_2) = \ldots = \mathcal{D}(\Phi_n)$ imply that for all $j \in \{1, 2, \ldots, n\}$ it holds that

$$\mathcal{D}(\Phi_j) = (l_0, l_1, \ldots, l_L). \tag{2.62}$$

Combining this with item (iii) in Lemma 2.2.2 demonstrates that

$$\mathcal{P}(\mathbf{P}_n(\Phi)) = \sum_{j=1}^{L} (nl_j)\big((nl_{j-1}) + 1\big). \tag{2.63}$$

Hence, we obtain that

$$\mathcal{P}(\mathbf{P}_n(\Phi)) \leq \sum_{j=1}^{L}(nl_j)\big((nl_{j-1})+n\big) = n^2\left[\sum_{j=1}^{L} l_j(l_{j-1}+1)\right] = n^2\mathcal{P}(\Phi_1). \tag{2.64}$$

Furthermore, note that the assumption that $\mathcal{D}(\Phi_1)=\mathcal{D}(\Phi_2)=\ldots=\mathcal{D}(\Phi_n)$ and the fact that $\mathcal{P}(\Phi_1)\geq l_1(l_0+1)\geq 2$ ensure that

$$n^2\mathcal{P}(\Phi_1) \leq \tfrac{n^2}{2}[\mathcal{P}(\Phi_1)]^2 = \tfrac{1}{2}[n\mathcal{P}(\Phi_1)]^2 = \tfrac{1}{2}\left[\sum_{i=1}^{n}\mathcal{P}(\Phi_1)\right]^2 = \tfrac{1}{2}\left[\sum_{i=1}^{n}\mathcal{P}(\Phi_i)\right]^2. \tag{2.65}$$

Moreover, observe that (2.63) and the fact that for all $a,b\in\mathbb{N}$ it holds that

$$2(ab+1) = ab+1+(a-1)(b-1)+a+b \geq ab+a+b+1 = (a+1)(b+1) \tag{2.66}$$

show that

$$\begin{split}\mathcal{P}(\mathbf{P}_n(\Phi)) &\geq \tfrac{1}{2}\left[\sum_{j=1}^{L}(nl_j)(n+1)(l_{j-1}+1)\right]\\ &= \tfrac{n(n+1)}{2}\left[\sum_{j=1}^{L} l_j(l_{j-1}+1)\right] = \left[\tfrac{n^2+n}{2}\right]\mathcal{P}(\Phi_1).\end{split} \tag{2.67}$$

This, (2.64), and (2.65) establish (2.60). The proof of Corollary 2.2.5 is thus complete. □

Exercise 2.2.1. Prove or disprove the following statement: For every $n\in\mathbb{N}$, $\Phi=(\Phi_1,\ldots,\Phi_n)\in\mathbf{N}^n$ with $\mathcal{L}(\Phi_1)=\mathcal{L}(\Phi_2)=\ldots=\mathcal{L}(\Phi_n)$ it holds that

$$\mathcal{P}(\mathbf{P}_n(\Phi_1,\Phi_2,\ldots,\Phi_n)) \leq n\big[\textstyle\sum_{i=1}^{n}\mathcal{P}(\Phi_i)\big]. \tag{2.68}$$

Exercise 2.2.2. Prove or disprove the following statement: For every $n\in\mathbb{N}$, $\Phi=(\Phi_1,\ldots,\Phi_n)\in\mathbf{N}^n$ with $\mathcal{P}(\Phi_1)=\mathcal{P}(\Phi_2)=\ldots=\mathcal{P}(\Phi_n)$ it holds that

$$\mathcal{P}(\mathbf{P}_n(\Phi_1,\Phi_2,\ldots,\Phi_n)) \leq n^2\mathcal{P}(\Phi_1). \tag{2.69}$$

2.2.2 Representations of the identities with ReLU activation functions

Definition 2.2.6 (Fully-connected feedforward ReLU identity ANNs)**.** *We denote by $\mathfrak{I}_d\in\mathbf{N}$, $d\in\mathbb{N}$, the fully-connected feedforward ANNs which satisfy for all $d\in\mathbb{N}$ that*

$$\mathfrak{I}_1 = \left(\left(\begin{pmatrix}1\\-1\end{pmatrix},\begin{pmatrix}0\\0\end{pmatrix}\right),\left(\begin{pmatrix}1 & -1\end{pmatrix},0\right)\right) \in \big((\mathbb{R}^{2\times1}\times\mathbb{R}^2)\times(\mathbb{R}^{1\times2}\times\mathbb{R}^1)\big) \tag{2.70}$$

and

$$\mathfrak{I}_d = \mathbf{P}_d(\mathfrak{I}_1,\mathfrak{I}_1,\ldots,\mathfrak{I}_1) \tag{2.71}$$

(cf. Definitions 1.3.1 and 2.2.1).

Lemma 2.2.7 (Properties of fully-connected feedforward ReLU identity ANNs)**.** *Let $d \in \mathbb{N}$. Then*

(i) it holds that
$$\mathcal{D}(\mathfrak{I}_d) = (d, 2d, d) \in \mathbb{N}^3 \tag{2.72}$$
and

(ii) it holds that
$$\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}(\mathfrak{I}_d) = \mathrm{id}_{\mathbb{R}^d} \tag{2.73}$$

(cf. Definitions 1.3.1, 1.3.4, and 2.2.6).

Proof of Lemma 2.2.7. Throughout this proof, let $L = 2$, $l_0 = 1$, $l_1 = 2$, $l_2 = 1$. Note that (2.70) establishes that
$$\mathcal{D}(\mathfrak{I}_1) = (1, 2, 1) = (l_0, l_1, l_2). \tag{2.74}$$
This, (2.71), and Proposition 2.2.4 prove that
$$\mathcal{D}(\mathfrak{I}_d) = (d, 2d, d) \in \mathbb{N}^3. \tag{2.75}$$
This establishes item (i). Next note that (2.70) assures that for all $x \in \mathbb{R}$ it holds that
$$(\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}(\mathfrak{I}_1))(x) = \mathfrak{r}(x) - \mathfrak{r}(-x) = \max\{x, 0\} - \max\{-x, 0\} = x. \tag{2.76}$$
Combining this and Proposition 2.2.3 demonstrates that for all $x = (x_1, \ldots, x_d) \in \mathbb{R}^d$ it holds that $\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}(\mathfrak{I}_d) \in C(\mathbb{R}^d, \mathbb{R}^d)$ and
$$\begin{aligned}
(\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}(\mathfrak{I}_d))(x) &= \big(\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}\big(\mathbf{P}_d(\mathfrak{I}_1, \mathfrak{I}_1, \ldots, \mathfrak{I}_1)\big)\big)(x_1, x_2, \ldots, x_d) \\
&= \big((\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}(\mathfrak{I}_1))(x_1), (\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}(\mathfrak{I}_1))(x_2), \ldots, (\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}(\mathfrak{I}_1))(x_d)\big) \\
&= (x_1, x_2, \ldots, x_d) = x
\end{aligned} \tag{2.77}$$
(cf. Definition 2.2.1). This establishes item (ii). The proof of Lemma 2.2.7 is thus complete. □

2.2.3 Extensions of fully-connected feedforward ANNs

Definition 2.2.8 (Extensions of fully-connected feedforward ANNs)**.** *Let $L \in \mathbb{N}$, $\mathbb{I} \in \mathbf{N}$ satisfy $\mathcal{I}(\mathbb{I}) = \mathcal{O}(\mathbb{I})$. Then we denote by*
$$\mathcal{E}_{L,\mathbb{I}} \colon \big\{\Phi \in \mathbf{N} \colon \big(\mathcal{L}(\Phi) \leq L \text{ and } \mathcal{O}(\Phi) = \mathcal{I}(\mathbb{I})\big)\big\} \to \mathbf{N} \tag{2.78}$$
the function which satisfies for all $\Phi \in \mathbf{N}$ with $\mathcal{L}(\Phi) \leq L$ and $\mathcal{O}(\Phi) = \mathcal{I}(\mathbb{I})$ that
$$\mathcal{E}_{L,\mathbb{I}}(\Phi) = \big(\mathbb{I}^{\bullet(L - \mathcal{L}(\Phi))}\big) \bullet \Phi \tag{2.79}$$
(cf. Definitions 1.3.1, 2.1.1, and 2.1.6).

Lemma 2.2.9 (Length of extensions of fully-connected feedforward ANNs)**.** *Let $d, \mathfrak{i} \in \mathbb{N}$, $\Psi \in \mathbf{N}$ satisfy $\mathcal{D}(\Psi) = (d, \mathfrak{i}, d)$ (cf. Definition 1.3.1). Then*

(i) it holds for all $n \in \mathbb{N}_0$ that $\mathcal{H}(\Psi^{\bullet n}) = n$, $\mathcal{L}(\Psi^{\bullet n}) = n + 1$, $\mathcal{D}(\Psi^{\bullet n}) \in \mathbb{N}^{n+2}$, and

$$\mathcal{D}(\Psi^{\bullet n}) = \begin{cases} (d, d) & : n = 0 \\ (d, \mathfrak{i}, \mathfrak{i}, \dots, \mathfrak{i}, d) & : n \in \mathbb{N} \end{cases} \tag{2.80}$$

and

(ii) it holds for all $\Phi \in \mathbf{N}$, $L \in \mathbb{N} \cap [\mathcal{L}(\Phi), \infty)$ with $\mathcal{O}(\Phi) = d$ that

$$\mathcal{L}(\mathcal{E}_{L,\Psi}(\Phi)) = L \tag{2.81}$$

(cf. Definitions 2.1.6 and 2.2.8).

Proof of Lemma 2.2.9. Throughout this proof, let $\Phi \in \mathbf{N}$ satisfy $\mathcal{O}(\Phi) = d$. Observe that Lemma 2.1.7 and the fact that $\mathcal{H}(\Psi) = 1$ show that for all $n \in \mathbb{N}_0$ it holds that

$$\mathcal{H}(\Psi^{\bullet n}) = n\mathcal{H}(\Psi) = n \tag{2.82}$$

(cf. Definition 2.1.6). Combining this with (1.78) and Lemma 1.3.3 ensures that

$$\mathcal{H}(\Psi^{\bullet n}) = n, \qquad \mathcal{L}(\Psi^{\bullet n}) = n + 1, \qquad \text{and} \qquad \mathcal{D}(\Psi^{\bullet n}) \in \mathbb{N}^{n+2}. \tag{2.83}$$

Next we claim that for all $n \in \mathbb{N}_0$ it holds that

$$\mathbb{N}^{n+2} \ni \mathcal{D}(\Psi^{\bullet n}) = \begin{cases} (d, d) & : n = 0 \\ (d, \mathfrak{i}, \mathfrak{i}, \dots, \mathfrak{i}, d) & : n \in \mathbb{N}. \end{cases} \tag{2.84}$$

We now prove (2.84) by induction on $n \in \mathbb{N}_0$. Note that the fact that

$$\Psi^{\bullet 0} = (\mathrm{I}_d, 0) \in \mathbb{R}^{d \times d} \times \mathbb{R}^d \tag{2.85}$$

establishes (2.84) in the base case $n = 0$ (cf. Definition 1.5.5). For the induction step assume that there exists $n \in \mathbb{N}_0$ which satisfies

$$\mathbb{N}^{n+2} \ni \mathcal{D}(\Psi^{\bullet n}) = \begin{cases} (d, d) & : n = 0 \\ (d, \mathfrak{i}, \mathfrak{i}, \dots, \mathfrak{i}, d) & : n \in \mathbb{N}. \end{cases} \tag{2.86}$$

Note that (2.86), (2.41), (2.83), item (i) in Proposition 2.1.2, and the fact that $\mathcal{D}(\Psi) = (d, \mathfrak{i}, d) \in \mathbb{N}^3$ imply that

$$\mathcal{D}(\Psi^{\bullet (n+1)}) = \mathcal{D}(\Psi \bullet (\Psi^{\bullet n})) = (d, \mathfrak{i}, \mathfrak{i}, \dots, \mathfrak{i}, d) \in \mathbb{N}^{n+3} \tag{2.87}$$

(cf. Definition 2.1.1). Induction therefore proves (2.84). This and (2.83) establish item (i). Observe that (2.79), item (iii) in Proposition 2.1.2, (2.82), and the fact that $\mathcal{H}(\Phi) = \mathcal{L}(\Phi) - 1$ imply that for all $L \in \mathbb{N} \cap [\mathcal{L}(\Phi), \infty)$ it holds that

$$\begin{aligned} \mathcal{H}\big(\mathcal{E}_{L,\Psi}(\Phi)\big) &= \mathcal{H}\big((\Psi^{\bullet(L-\mathcal{L}(\Phi))}) \bullet \Phi\big) = \mathcal{H}\big(\Psi^{\bullet(L-\mathcal{L}(\Phi))}\big) + \mathcal{H}(\Phi) \\ &= (L - \mathcal{L}(\Phi)) + \mathcal{H}(\Phi) = L - 1. \end{aligned} \tag{2.88}$$

The fact that $\mathcal{H}\big(\mathcal{E}_{L,\Psi}(\Phi)\big) = \mathcal{L}\big(\mathcal{E}_{L,\Psi}(\Phi)\big) - 1$ hence proves that

$$\mathcal{L}\big(\mathcal{E}_{L,\Psi}(\Phi)\big) = \mathcal{H}\big(\mathcal{E}_{L,\Psi}(\Phi)\big) + 1 = L. \tag{2.89}$$

This establishes item (ii). The proof of Lemma 2.2.9 is thus complete. □

Lemma 2.2.10 (Realizations of extensions of fully-connected feedforward ANNs)**.** *Let $a \in C(\mathbb{R}, \mathbb{R})$, $\mathbb{I} \in \mathbf{N}$ satisfy $\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}) = \mathrm{id}_{\mathbb{R}^{\mathcal{I}(\mathbb{I})}}$ (cf. Definitions 1.3.1 and 1.3.4). Then*

(i) it holds for all $n \in \mathbb{N}_0$ that

$$\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet n}) = \mathrm{id}_{\mathbb{R}^{\mathcal{I}(\mathbb{I})}} \tag{2.90}$$

and

(ii) it holds for all $\Phi \in \mathbf{N}$, $L \in \mathbb{N} \cap [\mathcal{L}(\Phi), \infty)$ with $\mathcal{O}(\Phi) = \mathcal{I}(\mathbb{I})$ that

$$\mathcal{R}^{\mathbf{N}}_a(\mathcal{E}_{L,\mathbb{I}}(\Phi)) = \mathcal{R}^{\mathbf{N}}_a(\Phi) \tag{2.91}$$

(cf. Definitions 2.1.6 and 2.2.8).

Proof of Lemma 2.2.10. Throughout this proof, let $\Phi \in \mathbf{N}$, $L, d \in \mathbb{N}$ satisfy $\mathcal{L}(\Phi) \leq L$ and $\mathcal{I}(\mathbb{I}) = \mathcal{O}(\Phi) = d$. We claim that for all $n \in \mathbb{N}_0$ it holds that

$$\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet n}) \in C(\mathbb{R}^d, \mathbb{R}^d) \qquad \text{and} \qquad \forall\, x \in \mathbb{R}^d \colon\ (\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet n}))(x) = x. \tag{2.92}$$

We now prove (2.92) by induction on $n \in \mathbb{N}_0$. Note that (2.41) and the fact that $\mathcal{O}(\mathbb{I}) = d$ demonstrate that $\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet 0}) \in C(\mathbb{R}^d, \mathbb{R}^d)$ and $\forall\, x \in \mathbb{R}^d \colon (\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet 0}))(x) = x$. This establishes (2.92) in the base case $n = 0$. For the induction step observe that for all $n \in \mathbb{N}_0$ with $\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet n}) \in C(\mathbb{R}^d, \mathbb{R}^d)$ and $\forall\, x \in \mathbb{R}^d \colon (\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet n}))(x) = x$ it holds that

$$\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet(n+1)}) = \mathcal{R}^{\mathbf{N}}_a(\mathbb{I} \bullet (\mathbb{I}^{\bullet n})) = (\mathcal{R}^{\mathbf{N}}_a(\mathbb{I})) \circ (\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet n})) \in C(\mathbb{R}^d, \mathbb{R}^d) \tag{2.93}$$

and

$$\begin{aligned} \forall\, x \in \mathbb{R}^d \colon\ \big(\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet(n+1)})\big)(x) &= \big([\mathcal{R}^{\mathbf{N}}_a(\mathbb{I})] \circ [\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet n})]\big)(x) \\ &= (\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}))\big(\big(\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}^{\bullet n})\big)(x)\big) = (\mathcal{R}^{\mathbf{N}}_a(\mathbb{I}))(x) = x. \end{aligned} \tag{2.94}$$

Induction therefore proves (2.92). This establishes item (i). Note (2.79), item (v) in Proposition 2.1.2, item (i), and the fact that $\mathcal{I}(\mathbb{I}) = \mathcal{O}(\Phi)$ ensure that

$$\begin{aligned} \mathcal{R}_a^{\mathbf{N}}(\mathcal{E}_{L,\mathbb{I}}(\Phi)) &= \mathcal{R}_a^{\mathbf{N}}((\mathbb{I}^{\bullet(L-\mathcal{L}(\Phi))}) \bullet \Phi) \\ &\in C(\mathbb{R}^{\mathcal{I}(\Phi)}, \mathbb{R}^{\mathcal{O}(\mathbb{I})}) = C(\mathbb{R}^{\mathcal{I}(\Phi)}, \mathbb{R}^{\mathcal{I}(\mathbb{I})}) = C(\mathbb{R}^{\mathcal{I}(\Phi)}, \mathbb{R}^{\mathcal{O}(\Phi)}) \end{aligned} \tag{2.95}$$

and

$$\begin{aligned} \forall\, x \in \mathbb{R}^{\mathcal{I}(\Phi)} \colon \big(\mathcal{R}_a^{\mathbf{N}}(\mathcal{E}_{L,\mathbb{I}}(\Phi))\big)(x) &= \big(\mathcal{R}_a^{\mathbf{N}}(\mathbb{I}^{\bullet(L-\mathcal{L}(\Phi))})\big)\big((\mathcal{R}_a^{\mathbf{N}}(\Phi))(x)\big) \\ &= (\mathcal{R}_a^{\mathbf{N}}(\Phi))(x). \end{aligned} \tag{2.96}$$

This establishes item (ii). The proof of Lemma 2.2.10 is thus complete. □

Lemma 2.2.11 (Architectures of extensions of fully-connected feedforward ANNs)**.** *Let* $d, \mathfrak{i}, L, \mathfrak{L} \in \mathbb{N}$, $l_0, l_1, \dots, l_{L-1} \in \mathbb{N}$, $\Phi, \Psi \in \mathbf{N}$ *satisfy*

$$\mathfrak{L} \geq L, \qquad \mathcal{D}(\Phi) = (l_0, l_1, \dots, l_{L-1}, d), \qquad \textit{and} \qquad \mathcal{D}(\Psi) = (d, \mathfrak{i}, d) \tag{2.97}$$

(cf. Definition 1.3.1). Then $\mathcal{D}(\mathcal{E}_{\mathfrak{L},\Psi}(\Phi)) \in \mathbb{N}^{\mathfrak{L}+1}$ *and*

$$\mathcal{D}(\mathcal{E}_{\mathfrak{L},\Psi}(\Phi)) = \begin{cases} (l_0, l_1, \dots, l_{L-1}, d) & : \mathfrak{L} = L \\ (l_0, l_1, \dots, l_{L-1}, \mathfrak{i}, \mathfrak{i}, \dots, \mathfrak{i}, d) & : \mathfrak{L} > L \end{cases} \tag{2.98}$$

(cf. Definition 2.2.8).

Proof of Lemma 2.2.11. Observe that item (i) in Lemma 2.2.9 demonstrates that

$$\mathcal{H}(\Psi^{\bullet(\mathfrak{L}-L)})) = \mathfrak{L} - L, \qquad \mathcal{D}(\Psi^{\bullet(\mathfrak{L}-L)}) \in \mathbb{N}^{\mathfrak{L}-L+2}, \tag{2.99}$$

$$\text{and} \qquad \mathcal{D}(\Psi^{\bullet(\mathfrak{L}-L)}) = \begin{cases} (d, d) & : \mathfrak{L} = L \\ (d, \mathfrak{i}, \mathfrak{i}, \dots, \mathfrak{i}, d) & : \mathfrak{L} > L \end{cases} \tag{2.100}$$

(cf. Definition 2.1.6). Combining this with Proposition 2.1.2 establishes that

$$\mathcal{H}\big((\Psi^{\bullet(\mathfrak{L}-L)}) \bullet \Phi\big) = \mathcal{H}(\Psi^{\bullet(\mathfrak{L}-L)}) + \mathcal{H}(\Phi) = (\mathfrak{L} - L) + L - 1 = \mathfrak{L} - 1, \tag{2.101}$$

$$\mathcal{D}((\Psi^{\bullet(\mathfrak{L}-L)}) \bullet \Phi) \in \mathbb{N}^{\mathfrak{L}+1}, \tag{2.102}$$

$$\text{and} \qquad \mathcal{D}((\Psi^{\bullet(\mathfrak{L}-L)}) \bullet \Phi) = \begin{cases} (l_0, l_1, \dots, l_{L-1}, d) & : \mathfrak{L} = L \\ (l_0, l_1, \dots, l_{L-1}, \mathfrak{i}, \mathfrak{i}, \dots, \mathfrak{i}, d) & : \mathfrak{L} > L. \end{cases} \tag{2.103}$$

This and (2.79) establish (2.98). The proof of Lemma 2.2.11 is thus complete. □

2.2.4 Parallelizations of fully-connected feedforward ANNs with different lengths

Definition 2.2.12 (Parallelization of fully-connected feedforward ANNs with different length)**.** *Let $n \in \mathbb{N}$, $\Psi = (\Psi_1, \ldots, \Psi_n) \in \mathbf{N}^n$ satisfy for all $j \in \{1, 2, \ldots, n\}$ that*

$$\mathcal{H}(\Psi_j) = 1 \qquad \text{and} \qquad \mathcal{I}(\Psi_j) = \mathcal{O}(\Psi_j) \tag{2.104}$$

(cf. Definition 1.3.1). Then we denote by

$$\mathrm{P}_{n,\Psi} \colon \big\{\Phi = (\Phi_1, \ldots, \Phi_n) \in \mathbf{N}^n \colon \big(\forall\, j \in \{1, 2, \ldots, n\} \colon \mathcal{O}(\Phi_j) = \mathcal{I}(\Psi_j)\big)\big\} \to \mathbf{N} \tag{2.105}$$

the function which satisfies for all $\Phi = (\Phi_1, \ldots, \Phi_n) \in \mathbf{N}^n$ with $\forall\, j \in \{1, 2, \ldots, n\} \colon \mathcal{O}(\Phi_j) = \mathcal{I}(\Psi_j)$ that

$$\mathrm{P}_{n,\Psi}(\Phi) = \mathbf{P}_n\big(\mathcal{E}_{\max_{k \in \{1,2,\ldots,n\}} \mathcal{L}(\Phi_k), \Psi_1}(\Phi_1), \ldots, \mathcal{E}_{\max_{k \in \{1,2,\ldots,n\}} \mathcal{L}(\Phi_k), \Psi_n}(\Phi_n)\big) \tag{2.106}$$

(cf. Definitions 2.2.1 and 2.2.8 and Lemma 2.2.9).

Lemma 2.2.13 (Realizations for parallelizations of fully-connected feedforward ANNs with different length)**.** *Let $a \in C(\mathbb{R}, \mathbb{R})$, $n \in \mathbb{N}$, $\mathbb{I} = (\mathbb{I}_1, \ldots, \mathbb{I}_n)$, $\Phi = (\Phi_1, \ldots, \Phi_n) \in \mathbf{N}^n$ satisfy for all $j \in \{1, 2, \ldots, n\}$, $x \in \mathbb{R}^{\mathcal{O}(\Phi_j)}$ that $\mathcal{H}(\mathbb{I}_j) = 1$, $\mathcal{I}(\mathbb{I}_j) = \mathcal{O}(\mathbb{I}_j) = \mathcal{O}(\Phi_j)$, and $(\mathcal{R}_a^{\mathbf{N}}(\mathbb{I}_j))(x) = x$ (cf. Definitions 1.3.1 and 1.3.4). Then*

(i) it holds that

$$\mathcal{R}_a^{\mathbf{N}}\big(\mathrm{P}_{n,\mathbb{I}}(\Phi)\big) \in C\big(\mathbb{R}^{[\sum_{j=1}^n \mathcal{I}(\Phi_j)]}, \mathbb{R}^{[\sum_{j=1}^n \mathcal{O}(\Phi_j)]}\big) \tag{2.107}$$

and

(ii) it holds for all $x_1 \in \mathbb{R}^{\mathcal{I}(\Phi_1)}, x_2 \in \mathbb{R}^{\mathcal{I}(\Phi_2)}, \ldots, x_n \in \mathbb{R}^{\mathcal{I}(\Phi_n)}$ that

$$\begin{aligned} &\big(\mathcal{R}_a^{\mathbf{N}}(\mathrm{P}_{n,\mathbb{I}}(\Phi))\big)(x_1, x_2, \ldots, x_n) \\ &= \big((\mathcal{R}_a^{\mathbf{N}}(\Phi_1))(x_1), (\mathcal{R}_a^{\mathbf{N}}(\Phi_2))(x_2), \ldots, (\mathcal{R}_a^{\mathbf{N}}(\Phi_n))(x_n)\big) \in \mathbb{R}^{[\sum_{j=1}^n \mathcal{O}(\Phi_j)]} \end{aligned} \tag{2.108}$$

(cf. Definition 2.2.12).

Proof of Lemma 2.2.13. Throughout this proof, let $L \in \mathbb{N}$ satisfy $L = \max_{j \in \{1,2,\ldots,n\}} \mathcal{L}(\Phi_j)$. Note that item (ii) in Lemma 2.2.9, the assumption that for all $j \in \{1, 2, \ldots, n\}$ it holds that $\mathcal{H}(\mathbb{I}_j) = 1$, (2.79), (2.4), and item (ii) in Lemma 2.2.10 demonstrate

(I) that for all $j \in \{1, 2, \ldots, n\}$ it holds that $\mathcal{L}(\mathcal{E}_{L,\mathbb{I}_j}(\Phi_j)) = L$ and $\mathcal{R}_a^{\mathbf{N}}(\mathcal{E}_{L,\mathbb{I}_j}(\Phi_j)) \in C(\mathbb{R}^{\mathcal{I}(\Phi_j)}, \mathbb{R}^{\mathcal{O}(\Phi_j)})$ and

(II) that for all $j \in \{1, 2, \ldots, n\}$, $x \in \mathbb{R}^{\mathcal{I}(\Phi_j)}$ it holds that

$$\big(\mathcal{R}_a^{\mathbf{N}}(\mathcal{E}_{L,\mathbb{I}_j}(\Phi_j))\big)(x) = (\mathcal{R}_a^{\mathbf{N}}(\Phi_j))(x) \tag{2.109}$$

(cf. Definition 2.2.8). Items (i) and (ii) in Proposition 2.2.3 therefore imply

(A) that

$$\mathcal{R}_a^{\mathbf{N}}\big(\mathbf{P}_n\big(\mathcal{E}_{L,\mathbb{I}_1}(\Phi_1),\mathcal{E}_{L,\mathbb{I}_2}(\Phi_2),\dots,\mathcal{E}_{L,\mathbb{I}_n}(\Phi_n)\big)\big)\in C\big(\mathbb{R}^{[\sum_{j=1}^n \mathcal{I}(\Phi_j)]},\mathbb{R}^{[\sum_{j=1}^n \mathcal{O}(\Phi_j)]}\big) \tag{2.110}$$

and

(B) that for all $x_1\in\mathbb{R}^{\mathcal{I}(\Phi_1)}, x_2\in\mathbb{R}^{\mathcal{I}(\Phi_2)},\dots,x_n\in\mathbb{R}^{\mathcal{I}(\Phi_n)}$ it holds that

$$\begin{aligned}
&\big(\mathcal{R}_a^{\mathbf{N}}\big(\mathbf{P}_n\big(\mathcal{E}_{L,\mathbb{I}_1}(\Phi_1),\mathcal{E}_{L,\mathbb{I}_2}(\Phi_2),\dots,\mathcal{E}_{L,\mathbb{I}_n}(\Phi_n)\big)\big)\big)(x_1,x_2,\dots,x_n)\\
&=\Big(\big(\mathcal{R}_a^{\mathbf{N}}\big(\mathcal{E}_{L,\mathbb{I}_1}(\Phi_1)\big)\big)(x_1),\big(\mathcal{R}_a^{\mathbf{N}}\big(\mathcal{E}_{L,\mathbb{I}_2}(\Phi_2)\big)\big)(x_2),\dots,\big(\mathcal{R}_a^{\mathbf{N}}\big(\mathcal{E}_{L,\mathbb{I}_n}(\Phi_n)\big)\big)(x_n)\Big)\\
&=\Big((\mathcal{R}_a^{\mathbf{N}}(\Phi_1))(x_1),(\mathcal{R}_a^{\mathbf{N}}(\Phi_2))(x_2),\dots,(\mathcal{R}_a^{\mathbf{N}}(\Phi_n))(x_n)\Big)
\end{aligned} \tag{2.111}$$

(cf. Definition 2.2.1). Combining this with (2.106) and the fact that $L=\max_{j\in\{1,2,\dots,n\}}\mathcal{L}(\Phi_j)$ ensures

(C) that

$$\mathcal{R}_a^{\mathbf{N}}\big(\mathrm{P}_{n,\mathbb{I}}(\Phi)\big)\in C\big(\mathbb{R}^{[\sum_{j=1}^n \mathcal{I}(\Phi_j)]},\mathbb{R}^{[\sum_{j=1}^n \mathcal{O}(\Phi_j)]}\big) \tag{2.112}$$

and

(D) that for all $x_1\in\mathbb{R}^{\mathcal{I}(\Phi_1)}, x_2\in\mathbb{R}^{\mathcal{I}(\Phi_2)},\dots,x_n\in\mathbb{R}^{\mathcal{I}(\Phi_n)}$ it holds that

$$\begin{aligned}
&\big(\mathcal{R}_a^{\mathbf{N}}\big(\mathrm{P}_{n,\mathbb{I}}(\Phi)\big)\big)(x_1,x_2,\dots,x_n)\\
&=\big(\mathcal{R}_a^{\mathbf{N}}\big(\mathbf{P}_n\big(\mathcal{E}_{L,\mathbb{I}_1}(\Phi_1),\mathcal{E}_{L,\mathbb{I}_2}(\Phi_2),\dots,\mathcal{E}_{L,\mathbb{I}_n}(\Phi_n)\big)\big)\big)(x_1,x_2,\dots,x_n)\\
&=\Big((\mathcal{R}_a^{\mathbf{N}}(\Phi_1))(x_1),(\mathcal{R}_a^{\mathbf{N}}(\Phi_2))(x_2),\dots,(\mathcal{R}_a^{\mathbf{N}}(\Phi_n))(x_n)\Big).
\end{aligned} \tag{2.113}$$

This establishes items items (i) and (ii). The proof of Lemma 2.2.13 is thus complete. □

Exercise 2.2.3. For every $d\in\mathbb{N}$ let $F_d\colon\mathbb{R}^d\to\mathbb{R}^d$ satisfy for all $x=(x_1,\dots,x_d)\in\mathbb{R}^d$ that

$$F_d(x)=(\max\{|x_1|\},\max\{|x_1|,|x_2|\},\dots,\max\{|x_1|,|x_2|,\dots,|x_d|\}). \tag{2.114}$$

Prove or disprove the following statement: For all $d\in\mathbb{N}$ there exists $\Phi\in\mathbf{N}$ such that

$$\mathcal{R}_{\mathfrak{r}}^{\mathbf{N}}(\Phi)=F_d \tag{2.115}$$

(cf. Definitions 1.2.4, 1.3.1, and 1.3.4).

2.3 Scalar multiplications of fully-connected feedforward ANNs

2.3.1 Affine transformations as fully-connected feedforward ANNs

Definition 2.3.1 (Fully-connected feedforward affine transformation ANNs). *Let $m, n \in \mathbb{N}$, $W \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^m$. Then we denote by*

$$\mathbf{A}_{W,B} \in (\mathbb{R}^{m \times n} \times \mathbb{R}^m) \subseteq \mathbf{N} \tag{2.116}$$

the fully-connected feedforward ANN given by

$$\mathbf{A}_{W,B} = (W, B) \tag{2.117}$$

(cf. Definitions 1.3.1 and 1.3.2).

Lemma 2.3.2 (Realizations of fully-connected feedforward affine transformation of ANNs). *Let $m, n \in \mathbb{N}$, $W \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^m$. Then*

(i) it holds that $\mathcal{D}(\mathbf{A}_{W,B}) = (n, m) \in \mathbb{N}^2$,

(ii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$ that $\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{W,B}) \in C(\mathbb{R}^n, \mathbb{R}^m)$, and

(iii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^n$ that

$$(\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{W,B}))(x) = Wx + B \tag{2.118}$$

(cf. Definitions 1.3.1, 1.3.4, and 2.3.1).

Proof of Lemma 2.3.2. Note that the fact that $\mathbf{A}_{W,B} \in (\mathbb{R}^{m \times n} \times \mathbb{R}^m) \subseteq \mathbf{N}$ shows that

$$\mathcal{D}(\mathbf{A}_{W,B}) = (n, m) \in \mathbb{N}^2. \tag{2.119}$$

This proves item (i). Furthermore, observe that the fact that

$$\mathbf{A}_{W,B} = (W, B) \in (\mathbb{R}^{m \times n} \times \mathbb{R}^m) \tag{2.120}$$

and (1.91) ensure that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^n$ it holds that $\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{W,B}) \in C(\mathbb{R}^n, \mathbb{R}^m)$ and

$$(\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{W,B}))(x) = Wx + B. \tag{2.121}$$

This establishes items (ii) and (iii). The proof of Lemma 2.3.2 is thus complete. The proof of Lemma 2.3.2 is thus complete. □

Lemma 2.3.3 (Compositions with fully-connected feedforward affine transformation ANNs). *Let $\Phi \in \mathbf{N}$ (cf. Definition 1.3.1). Then* *(i) it holds for all $m \in \mathbb{N}$, $W \in \mathbb{R}^{m \times \mathcal{O}(\Phi)}$, $B \in \mathbb{R}^m$ that*

$$\mathcal{D}(\mathbf{A}_{W,B} \bullet \Phi) = (\mathbb{D}_0(\Phi), \mathbb{D}_1(\Phi), \ldots, \mathbb{D}_{\mathcal{H}(\Phi)}(\Phi), m), \tag{2.122}$$

(ii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$, $m \in \mathbb{N}$, $W \in \mathbb{R}^{m \times \mathcal{O}(\Phi)}$, $B \in \mathbb{R}^m$ that $\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{W,B} \bullet \Phi) \in C(\mathbb{R}^{\mathcal{I}(\Phi)}, \mathbb{R}^m)$,

(iii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$, $m \in \mathbb{N}$, $W \in \mathbb{R}^{m \times \mathcal{O}(\Phi)}$, $B \in \mathbb{R}^m$, $x \in \mathbb{R}^{\mathcal{I}(\Phi)}$ that

$$(\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{W,B} \bullet \Phi))(x) = W\big((\mathcal{R}_a^{\mathbf{N}}(\Phi))(x)\big) + B, \tag{2.123}$$

(iv) it holds for all $n \in \mathbb{N}$, $W \in \mathbb{R}^{\mathcal{I}(\Phi) \times n}$, $B \in \mathbb{R}^{\mathcal{I}(\Phi)}$ that

$$\mathcal{D}(\Phi \bullet \mathbf{A}_{W,B}) = (n, \mathbb{D}_1(\Phi), \mathbb{D}_2(\Phi), \ldots, \mathbb{D}_{\mathcal{L}(\Phi)}(\Phi)), \tag{2.124}$$

(v) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$, $n \in \mathbb{N}$, $W \in \mathbb{R}^{\mathcal{I}(\Phi) \times n}$, $B \in \mathbb{R}^{\mathcal{I}(\Phi)}$ that $\mathcal{R}_a^{\mathbf{N}}(\Phi \bullet \mathbf{A}_{W,B}) \in C(\mathbb{R}^n, \mathbb{R}^{\mathcal{O}(\Phi)})$, and

(vi) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$, $n \in \mathbb{N}$, $W \in \mathbb{R}^{\mathcal{I}(\Phi) \times n}$, $B \in \mathbb{R}^{\mathcal{I}(\Phi)}$, $x \in \mathbb{R}^n$ that

$$(\mathcal{R}_a^{\mathbf{N}}(\Phi \bullet \mathbf{A}_{W,B}))(x) = (\mathcal{R}_a^{\mathbf{N}}(\Phi))(Wx + B) \tag{2.125}$$

(cf. Definitions 1.3.4, 2.1.1, and 2.3.1).

Proof of Lemma 2.3.3. Note that Lemma 2.3.2 implies that for all $m, n \in \mathbb{N}$, $W \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^m$, $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^n$ it holds that $\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{W,B}) \in C(\mathbb{R}^n, \mathbb{R}^m)$ and

$$(\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{W,B}))(x) = Wx + B \tag{2.126}$$

(cf. Definitions 1.3.4 and 2.3.1). Combining this and Proposition 2.1.2 proves items (i), (ii), (iii), (iv), (v), and (vi). The proof of Lemma 2.3.3 is thus complete. □

2.3.2 Scalar multiplications of fully-connected feedforward ANNs

Definition 2.3.4 (Scalar multiplications of ANNs)**.** *We denote by $(\cdot) \circledast (\cdot) \colon \mathbb{R} \times \mathbf{N} \to \mathbf{N}$ the function which satisfies for all $\lambda \in \mathbb{R}$, $\Phi \in \mathbf{N}$ that*

$$\lambda \circledast \Phi = \mathbf{A}_{\lambda \mathrm{I}_{\mathcal{O}(\Phi)}, 0} \bullet \Phi \tag{2.127}$$

(cf. Definitions 1.3.1, 1.5.5, 2.1.1, and 2.3.1).

Lemma 2.3.5. *Let $\lambda \in \mathbb{R}$, $\Phi \in \mathbf{N}$ (cf. Definition 1.3.1). Then*

(i) it holds that $\mathcal{D}(\lambda \circledast \Phi) = \mathcal{D}(\Phi)$,

(ii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$ *that* $\mathcal{R}_a^{\mathbf{N}}(\lambda \circledast \Phi) \in C(\mathbb{R}^{\mathcal{I}(\Phi)}, \mathbb{R}^{\mathcal{O}(\Phi)})$*, and*

(iii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$*,* $x \in \mathbb{R}^{\mathcal{I}(\Phi)}$ *that*

$$\big(\mathcal{R}_a^{\mathbf{N}}(\lambda \circledast \Phi)\big)(x) = \lambda\big((\mathcal{R}_a^{\mathbf{N}}(\Phi))(x)\big) \tag{2.128}$$

(cf. Definitions 1.3.4 and 2.3.4).

Proof of Lemma 2.3.5. Throughout this proof, let $L \in \mathbb{N}$, $l_0, l_1, \dots, l_L \in \mathbb{N}$ satisfy

$$L = \mathcal{L}(\Phi) \qquad \text{and} \qquad (l_0, l_1, \dots, l_L) = \mathcal{D}(\Phi). \tag{2.129}$$

Observe that item (i) in Lemma 2.3.2 demonstrates that

$$\mathcal{D}(\mathbf{A}_{\lambda \mathrm{I}_{\mathcal{O}(\Phi)},0}) = (\mathcal{O}(\Phi), \mathcal{O}(\Phi)) \tag{2.130}$$

(cf. Definitions 1.5.5 and 2.3.1). Combining this and item (i) in Lemma 2.3.3 shows that

$$\mathcal{D}(\lambda \circledast \Phi) = \mathcal{D}(\mathbf{A}_{\lambda \mathrm{I}_{\mathcal{O}(\Phi)},0} \bullet \Phi) = (l_0, l_1, \dots, l_{L-1}, \mathcal{O}(\Phi)) = \mathcal{D}(\Phi) \tag{2.131}$$

(cf. Definitions 2.1.1 and 2.3.4). This establishes item (i). Note that items (ii) and (iii) in Lemma 2.3.3 ensure that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^{\mathcal{I}(\Phi)}$ it holds that $\mathcal{R}_a^{\mathbf{N}}(\lambda \circledast \Phi) \in C(\mathbb{R}^{\mathcal{I}(\Phi)}, \mathbb{R}^{\mathcal{O}(\Phi)})$ and

$$\begin{split} \big(\mathcal{R}_a^{\mathbf{N}}(\lambda \circledast \Phi)\big)(x) &= \big(\mathcal{R}_a^{\mathbf{N}}(\mathbf{A}_{\lambda \mathrm{I}_{\mathcal{O}(\Phi)},0} \bullet \Phi)\big)(x) \\ &= \lambda \mathrm{I}_{\mathcal{O}(\Phi)}\big((\mathcal{R}_a^{\mathbf{N}}(\Phi))(x)\big) \\ &= \lambda\big((\mathcal{R}_a^{\mathbf{N}}(\Phi))(x)\big) \end{split} \tag{2.132}$$

(cf. Definition 1.3.4). This proves items (ii) and (iii). The proof of Lemma 2.3.5 is thus complete. □

2.4 Sums of fully-connected feedforward ANNs with the same length

2.4.1 Sums of vectors as fully-connected feedforward ANNs

Definition 2.4.1 (Sums of vectors as fully-connected feedforward ANNs)**.** *Let* $m, n \in \mathbb{N}$*. Then we denote by*

$$\mathbb{S}_{m,n} \in (\mathbb{R}^{m \times (mn)} \times \mathbb{R}^m) \subseteq \mathbf{N} \tag{2.133}$$

the fully-connected feedforward ANN given by

$$\mathbb{S}_{m,n} = \mathbf{A}_{(\mathrm{I}_m \;\; \mathrm{I}_m \;\; \dots \;\; \mathrm{I}_m),0} \tag{2.134}$$

(cf. Definitions 1.3.1, 1.3.2, 1.5.5, and 2.3.1).

Lemma 2.4.2. *Let $m, n \in \mathbb{N}$. Then*

(i) it holds that $\mathcal{D}(\mathbb{S}_{m,n}) = (mn, m) \in \mathbb{N}^2$,

(ii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$ that $\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n}) \in C(\mathbb{R}^{mn}, \mathbb{R}^m)$, and

(iii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$, $x_1, x_2, \dots, x_n \in \mathbb{R}^m$ that

$$(\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n}))(x_1, x_2, \dots, x_n) = \sum_{k=1}^{n} x_k \tag{2.135}$$

(cf. Definitions 1.3.1, 1.3.4, and 2.4.1).

Proof of Lemma 2.4.2. Observe that the fact that $\mathbb{S}_{m,n} \in (\mathbb{R}^{m \times (mn)} \times \mathbb{R}^m)$ implies that

$$\mathcal{D}(\mathbb{S}_{m,n}) = (mn, m) \in \mathbb{N}^2 \tag{2.136}$$

(cf. Definitions 1.3.1 and 2.4.1). This establishes item (i). Note that items (ii) and (iii) in Lemma 2.3.2 demonstrate that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x_1, x_2, \dots, x_n \in \mathbb{R}^m$ it holds that $\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n}) \in C(\mathbb{R}^{mn}, \mathbb{R}^m)$ and

$$\begin{split} (\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n}))(x_1, x_2, \dots, x_n) &= \big(\mathcal{R}_a^{\mathbf{N}}\big(\mathbf{A}_{(\mathrm{I}_m \;\; \mathrm{I}_m \;\; \dots \;\; \mathrm{I}_m),0}\big)\big)(x_1, x_2, \dots, x_n) \\ &= (\mathrm{I}_m \;\; \mathrm{I}_m \;\; \dots \;\; \mathrm{I}_m)(x_1, x_2, \dots, x_n) = \sum_{k=1}^{n} x_k \end{split} \tag{2.137}$$

(cf. Definitions 1.3.4, 1.5.5, and 2.3.1). This proves items (ii) and (iii). The proof of Lemma 2.4.2 is thus complete. □

Lemma 2.4.3. *Let $m, n \in \mathbb{N}$, $a \in C(\mathbb{R}, \mathbb{R})$, $\Phi \in \mathbf{N}$ satisfy $\mathcal{O}(\Phi) = mn$ (cf. Definition 1.3.1). Then*

(i) it holds that $\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n} \bullet \Phi) \in C(\mathbb{R}^{\mathcal{I}(\Phi)}, \mathbb{R}^m)$ and

(ii) it holds for all $x \in \mathbb{R}^{\mathcal{I}(\Phi)}$, $y_1, y_2, \dots, y_n \in \mathbb{R}^m$ with $(\mathcal{R}_a^{\mathbf{N}}(\Phi))(x) = (y_1, y_2, \dots, y_n)$ that

$$\big(\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n} \bullet \Phi)\big)(x) = \sum_{k=1}^{n} y_k \tag{2.138}$$

(cf. Definitions 1.3.4, 2.1.1, and 2.4.1).

Proof of Lemma 2.4.3. Observe that Lemma 2.4.2 shows that for all $x_1, x_2, \dots, x_n \in \mathbb{R}^m$ it holds that $\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n}) \in C(\mathbb{R}^{mn}, \mathbb{R}^m)$ and

$$(\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n}))(x_1, x_2, \dots, x_n) = \sum_{k=1}^{n} x_k \tag{2.139}$$

(cf. Definitions 1.3.4 and 2.4.1). Combining this and item (v) in Proposition 2.1.2 establishes items (i) and (ii). The proof of Lemma 2.4.3 is thus complete. □

Lemma 2.4.4. *Let $n \in \mathbb{N}$, $a \in C(\mathbb{R}, \mathbb{R})$, $\Phi \in \mathbf{N}$ (cf. Definition 1.3.1). Then*

(i) it holds that $\mathcal{R}_a^{\mathbf{N}}(\Phi \bullet \mathbb{S}_{\mathcal{I}(\Phi),n}) \in C(\mathbb{R}^{n\mathcal{I}(\Phi)}, \mathbb{R}^{\mathcal{O}(\Phi)})$ and

(ii) it holds for all $x_1, x_2, \ldots, x_n \in \mathbb{R}^{\mathcal{I}(\Phi)}$ that

$$\big(\mathcal{R}_a^{\mathbf{N}}(\Phi \bullet \mathbb{S}_{\mathcal{I}(\Phi),n})\big)(x_1, x_2, \ldots, x_n) = (\mathcal{R}_a^{\mathbf{N}}(\Phi))\left(\sum_{k=1}^{n} x_k\right) \tag{2.140}$$

(cf. Definitions 1.3.4, 2.1.1, and 2.4.1).

Proof of Lemma 2.4.4. Note that Lemma 2.4.2 ensures that for all $m \in \mathbb{N}$, $x_1, x_2, \ldots, x_n \in \mathbb{R}^m$ it holds that $\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n}) \in C(\mathbb{R}^{mn}, \mathbb{R}^m)$ and

$$(\mathcal{R}_a^{\mathbf{N}}(\mathbb{S}_{m,n}))(x_1, x_2, \ldots, x_n) = \sum_{k=1}^{n} x_k \tag{2.141}$$

(cf. Definitions 1.3.4 and 2.4.1). Combining this and item (v) in Proposition 2.1.2 proves items (i) and (ii). The proof of Lemma 2.4.4 is thus complete. □

2.4.2 Concatenation of vectors as fully-connected feedforward ANNs

Definition 2.4.5 (Transpose of a matrix)**.** *Let $m, n \in \mathbb{N}$, $A \in \mathbb{R}^{m \times n}$. Then we denote by $A^* \in \mathbb{R}^{n \times m}$ the transpose of A.*

Definition 2.4.6 (Concatenation of vectors as fully-connected feedforward ANNs)**.** *Let $m, n \in \mathbb{N}$. Then we denote by*

$$\mathbb{T}_{m,n} \in (\mathbb{R}^{(mn) \times m} \times \mathbb{R}^{mn}) \subseteq \mathbf{N} \tag{2.142}$$

the fully-connected feedforward ANN given by

$$\mathbb{T}_{m,n} = \mathbf{A}_{(\mathrm{I}_m \;\; \mathrm{I}_m \;\; \ldots \;\; \mathrm{I}_m)^*, 0} \tag{2.143}$$

(cf. Definitions 1.3.1, 1.3.2, 1.5.5, 2.3.1, and 2.4.5).

Lemma 2.4.7. *Let $m, n \in \mathbb{N}$. Then*

(i) it holds that $\mathcal{D}(\mathbb{T}_{m,n}) = (m, mn) \in \mathbb{N}^2$,

(ii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$ that $\mathcal{R}_a^{\mathbf{N}}(\mathbb{T}_{m,n}) \in C(\mathbb{R}^m, \mathbb{R}^{mn})$, and

(iii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^m$ that

$$(\mathcal{R}_a^{\mathbf{N}}(\mathbb{T}_{m,n}))(x) = (x, x, \ldots, x) \tag{2.144}$$

(cf. Definitions 1.3.1, 1.3.4, and 2.4.6).

Proof of Lemma 2.4.7. Observe that the fact that $\mathbb{T}_{m,n} \in (\mathbb{R}^{(mn)\times m} \times \mathbb{R}^{mn})$ implies that

$$\mathcal{D}(\mathbb{T}_{m,n}) = (m, mn) \in \mathbb{N}^2 \tag{2.145}$$

(cf. Definitions 1.3.1 and 2.4.6). This establishes item (i). Note that item (iii) in Lemma 2.3.2 demonstrates that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^m$ it holds that $\mathcal{R}^{\mathbf{N}}_a(\mathbb{T}_{m,n}) \in C(\mathbb{R}^m, \mathbb{R}^{mn})$ and

$$\begin{aligned} (\mathcal{R}^{\mathbf{N}}_a(\mathbb{T}_{m,n}))(x) &= \big(\mathcal{R}^{\mathbf{N}}_a\big(\mathbf{A}_{(\mathrm{I}_m \;\; \mathrm{I}_m \;\; \dots \;\; \mathrm{I}_m)^*,0}\big)\big)(x) \\ &= (\mathrm{I}_m \;\; \mathrm{I}_m \;\; \dots \;\; \mathrm{I}_m)^* x = (x, x, \dots, x) \end{aligned} \tag{2.146}$$

(cf. Definitions 1.3.4, 1.5.5, 2.3.1, and 2.4.5). This proves items (ii) and (iii). The proof of Lemma 2.4.7 is thus complete. □

Lemma 2.4.8. *Let $n \in \mathbb{N}$, $a \in C(\mathbb{R}, \mathbb{R})$, $\Phi \in \mathbf{N}$ (cf. Definition 1.3.1). Then*

(i) it holds that $\mathcal{R}^{\mathbf{N}}_a(\mathbb{T}_{\mathcal{O}(\Phi),n} \bullet \Phi) \in C(\mathbb{R}^{\mathcal{I}(\Phi)}, \mathbb{R}^{n\mathcal{O}(\Phi)})$ and

(ii) it holds for all $x \in \mathbb{R}^{\mathcal{I}(\Phi)}$ that

$$\big(\mathcal{R}^{\mathbf{N}}_a(\mathbb{T}_{\mathcal{O}(\Phi),n} \bullet \Phi)\big)(x) = \big((\mathcal{R}^{\mathbf{N}}_a(\Phi))(x), (\mathcal{R}^{\mathbf{N}}_a(\Phi))(x), \dots, (\mathcal{R}^{\mathbf{N}}_a(\Phi))(x)\big) \tag{2.147}$$

(cf. Definitions 1.3.4, 2.1.1, and 2.4.6).

Proof of Lemma 2.4.8. Observe that Lemma 2.4.7 shows that for all $m \in \mathbb{N}$, $x \in \mathbb{R}^m$ it holds that $\mathcal{R}^{\mathbf{N}}_a(\mathbb{T}_{m,n}) \in C(\mathbb{R}^m, \mathbb{R}^{mn})$ and

$$(\mathcal{R}^{\mathbf{N}}_a(\mathbb{T}_{m,n}))(x) = (x, x, \dots, x) \tag{2.148}$$

(cf. Definitions 1.3.4 and 2.4.6). Combining this and item (v) in Proposition 2.1.2 establishes items (i) and (ii). The proof of Lemma 2.4.8 is thus complete. □

Lemma 2.4.9. *Let $m, n \in \mathbb{N}$, $a \in C(\mathbb{R}, \mathbb{R})$, $\Phi \in \mathbf{N}$ satisfy $\mathcal{I}(\Phi) = mn$ (cf. Definition 1.3.1). Then*

(i) it holds that $\mathcal{R}^{\mathbf{N}}_a(\Phi \bullet \mathbb{T}_{m,n}) \in C(\mathbb{R}^m, \mathbb{R}^{\mathcal{O}(\Phi)})$ and

(ii) it holds for all $x \in \mathbb{R}^m$ that

$$\big(\mathcal{R}^{\mathbf{N}}_a(\Phi \bullet \mathbb{T}_{m,n})\big)(x) = (\mathcal{R}^{\mathbf{N}}_a(\Phi))(x, x, \dots, x) \tag{2.149}$$

(cf. Definitions 1.3.4, 2.1.1, and 2.4.6).

Proof of Lemma 2.4.9. Note that Lemma 2.4.7 ensures that for all $x \in \mathbb{R}^m$ it holds that $\mathcal{R}^{\mathbf{N}}_a(\mathbb{T}_{m,n}) \in C(\mathbb{R}^m, \mathbb{R}^{mn})$ and

$$(\mathcal{R}^{\mathbf{N}}_a(\mathbb{T}_{m,n}))(x) = (x, x, \dots, x) \tag{2.150}$$

(cf. Definitions 1.3.4 and 2.4.6). Combining this and item (v) in Proposition 2.1.2 proves items (i) and (ii). The proof of Lemma 2.4.9 is thus complete. □

2.4.3 Sums of fully-connected feedforward ANNs

Definition 2.4.10 (Sums of fully-connected feedforward ANNs with the same length)**.** *Let $m \in \mathbb{Z}$, $n \in \{m, m+1, \dots\}$, $\Phi_m, \Phi_{m+1}, \dots, \Phi_n \in \mathbf{N}$ satisfy for all $k \in \{m, m+1, \dots, n\}$ that*

$$\mathcal{L}(\Phi_k) = \mathcal{L}(\Phi_m), \qquad \mathcal{I}(\Phi_k) = \mathcal{I}(\Phi_m), \qquad \text{and} \qquad \mathcal{O}(\Phi_k) = \mathcal{O}(\Phi_m) \tag{2.151}$$

(cf. Definition 1.3.1). Then we denote by $\bigoplus_{k=m}^{n} \Phi_k \in \mathbf{N}$ (we denote by $\Phi_m \oplus \Phi_{m+1} \oplus \dots \oplus \Phi_n \in \mathbf{N}$) the fully-connected feedforward ANN given by

$$\bigoplus_{k=m}^{n} \Phi_k = \big(\mathbb{S}_{\mathcal{O}(\Phi_m), n-m+1} \bullet \big[\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)\big] \bullet \mathbb{T}_{\mathcal{I}(\Phi_m), n-m+1}\big) \in \mathbf{N} \tag{2.152}$$

(cf. Definitions 1.3.2, 2.1.1, 2.2.1, 2.4.1, and 2.4.6).

Lemma 2.4.11 (Realizations of sums of fully-connected feedforward ANNs)**.** *Let $m \in \mathbb{Z}$, $n \in \{m, m+1, \dots\}$, $\Phi_m, \Phi_{m+1}, \dots, \Phi_n \in \mathbf{N}$ satisfy for all $k \in \{m, m+1, \dots, n\}$ that*

$$\mathcal{L}(\Phi_k) = \mathcal{L}(\Phi_m), \qquad \mathcal{I}(\Phi_k) = \mathcal{I}(\Phi_m), \qquad \text{and} \qquad \mathcal{O}(\Phi_k) = \mathcal{O}(\Phi_m) \tag{2.153}$$

(cf. Definition 1.3.1). Then

(i) it holds that $\mathcal{L}\big(\bigoplus_{k=m}^{n} \Phi_k\big) = \mathcal{L}(\Phi_m)$,

(ii) it holds that

$$\mathcal{D}\bigg(\bigoplus_{k=m}^{n} \Phi_k\bigg) = \bigg(\mathcal{I}(\Phi_m), \sum_{k=m}^{n} \mathbb{D}_1(\Phi_k), \sum_{k=m}^{n} \mathbb{D}_2(\Phi_k), \dots, \sum_{k=m}^{n} \mathbb{D}_{\mathcal{H}(\Phi_m)}(\Phi_k), \mathcal{O}(\Phi_m)\bigg), \tag{2.154}$$

and

(iii) it holds for all $a \in C(\mathbb{R}, \mathbb{R})$ that

$$\mathcal{R}_a^{\mathbf{N}}\bigg(\bigoplus_{k=m}^{n} \Phi_k\bigg) = \sum_{k=m}^{n} (\mathcal{R}_a^{\mathbf{N}}(\Phi_k)) \tag{2.155}$$

(cf. Definitions 1.3.4 and 2.4.10).

Proof of Lemma 2.4.11. First, observe that Lemma 2.2.2 implies that

$$\begin{aligned}
&\mathcal{D}\big(\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)\big) \\
&= \bigg(\sum_{k=m}^{n} \mathbb{D}_0(\Phi_k), \sum_{k=m}^{n} \mathbb{D}_1(\Phi_k), \dots, \sum_{k=m}^{n} \mathbb{D}_{\mathcal{L}(\Phi_m)-1}(\Phi_k), \sum_{k=m}^{n} \mathbb{D}_{\mathcal{L}(\Phi_m)}(\Phi_k)\bigg) \\
&= \bigg((n-m+1)\mathcal{I}(\Phi_m), \sum_{k=m}^{n} \mathbb{D}_1(\Phi_k), \sum_{k=m}^{n} \mathbb{D}_2(\Phi_k), \dots, \sum_{k=m}^{n} \mathbb{D}_{\mathcal{L}(\Phi_m)-1}(\Phi_k), \\
&\qquad (n-m+1)\mathcal{O}(\Phi_m)\bigg)
\end{aligned} \tag{2.156}$$

(cf. Definition 2.2.1). Furthermore, note that item (i) in Lemma 2.4.2 demonstrates that

$$\mathcal{D}(\mathbb{S}_{\mathcal{O}(\Phi_m),n-m+1}) = ((n-m+1)\mathcal{O}(\Phi_m), \mathcal{O}(\Phi_m)) \tag{2.157}$$

(cf. Definition 2.4.1). This, (2.156), and item (i) in Proposition 2.1.2 show that

$$\begin{split} &\mathcal{D}\big(\mathbb{S}_{\mathcal{O}(\Phi_m),n-m+1} \bullet \big[\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)\big]\big) \\ &= \left((n-m+1)\mathcal{I}(\Phi_m), \sum_{k=m}^{n} \mathbb{D}_1(\Phi_k), \sum_{k=m}^{n} \mathbb{D}_2(\Phi_k), \dots, \sum_{k=m}^{n} \mathbb{D}_{\mathcal{L}(\Phi_m)-1}(\Phi_k), \mathcal{O}(\Phi_m)\right). \end{split} \tag{2.158}$$

Moreover, observe that item (i) in Lemma 2.4.7 establishes that

$$\mathcal{D}\big(\mathbb{T}_{\mathcal{I}(\Phi_m),n-m+1}\big) = (\mathcal{I}(\Phi_m), (n-m+1)\mathcal{I}(\Phi_m)) \tag{2.159}$$

(cf. Definitions 2.1.1 and 2.4.6). Combining this, (2.158), and item (i) in Proposition 2.1.2 ensures that

$$\begin{split} &\mathcal{D}\left(\bigoplus_{k=m}^{n} \Phi_k\right) \\ &= \mathcal{D}\big(\mathbb{S}_{\mathcal{O}(\Phi_m),(n-m+1)} \bullet \big[\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)\big] \bullet \mathbb{T}_{\mathcal{I}(\Phi_m),(n-m+1)}\big) \\ &= \left(\mathcal{I}(\Phi_m), \sum_{k=m}^{n} \mathbb{D}_1(\Phi_k), \sum_{k=m}^{n} \mathbb{D}_2(\Phi_k), \dots, \sum_{k=m}^{n} \mathbb{D}_{\mathcal{L}(\Phi_m)-1}(\Phi_k), \mathcal{O}(\Phi_m)\right) \end{split} \tag{2.160}$$

(cf. Definition 2.4.10). This proves items (i) and (ii). Note that Lemma 2.4.9 and (2.156) imply that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^{\mathcal{I}(\Phi_m)}$ it holds that

$$\mathcal{R}_a^{\mathbf{N}}\big([\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)] \bullet \mathbb{T}_{\mathcal{I}(\Phi_m),n-m+1}\big) \in C(\mathbb{R}^{\mathcal{I}(\Phi_m)}, \mathbb{R}^{(n-m+1)\mathcal{O}(\Phi_m)}) \tag{2.161}$$

and

$$\begin{split} &\big(\mathcal{R}_a^{\mathbf{N}}\big([\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)] \bullet \mathbb{T}_{\mathcal{I}(\Phi_m),n-m+1}\big)\big)(x) \\ &= \big(\mathcal{R}_a^{\mathbf{N}}\big(\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)\big)\big)(x, x, \dots, x) \end{split} \tag{2.162}$$

(cf. Definition 1.3.4). Combining this with item (ii) in Proposition 2.2.3 demonstrates that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^{\mathcal{I}(\Phi_m)}$ it holds that

$$\begin{split} &\big(\mathcal{R}_a^{\mathbf{N}}\big([\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)] \bullet \mathbb{T}_{\mathcal{I}(\Phi_m),n-m+1}\big)\big)(x) \\ &= \big((\mathcal{R}_a^{\mathbf{N}}(\Phi_m))(x), (\mathcal{R}_a^{\mathbf{N}}(\Phi_{m+1}))(x), \dots, (\mathcal{R}_a^{\mathbf{N}}(\Phi_n))(x)\big) \in \mathbb{R}^{(n-m+1)\mathcal{O}(\Phi_m)}. \end{split} \tag{2.163}$$

Lemma 2.4.3, (2.157), and Corollary 2.1.5 hence show that for all $a \in C(\mathbb{R}, \mathbb{R})$, $x \in \mathbb{R}^{\mathcal{I}(\Phi_m)}$ it holds that $\mathcal{R}_a^{\mathbf{N}}\big(\bigoplus_{k=m}^{n} \Phi_k\big) \in C(\mathbb{R}^{\mathcal{I}(\Phi_m)}, \mathbb{R}^{\mathcal{O}(\Phi_m)})$ and

$$\begin{split} &\left(\mathcal{R}_a^{\mathbf{N}}\left(\bigoplus_{k=m}^{n} \Phi_k\right)\right)(x) \\ &= \big(\mathcal{R}_a^{\mathbf{N}}\big(\mathbb{S}_{\mathcal{O}(\Phi_m),n-m+1} \bullet [\mathbf{P}_{n-m+1}(\Phi_m, \Phi_{m+1}, \dots, \Phi_n)] \bullet \mathbb{T}_{\mathcal{I}(\Phi_m),n-m+1}\big)\big)(x) \\ &= \sum_{k=m}^{n} (\mathcal{R}_a^{\mathbf{N}}(\Phi_k))(x). \end{split} \tag{2.164}$$

This establishes item (iii). The proof of Lemma 2.4.11 is thus complete. □