Approximation with Neural Networks from a Theoretical and Practical Perspective

vorgelegt von

M. Sc.

Ingo Gühring

an der Fakultät IV – Elektrotechnik und Informatik

der Technischen Universität Berlin

zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften

– Dr. rer. nat. –

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Benjamin Blankertz
Gutachter:	Prof. Dr. Klaus-Robert Müller
Gutachter:	Prof. Dr. Philipp Grohs
Gutachter:	Prof. Dr. Helmut Bölcskei

Tag der wissenschaftlichen Aussprache: 21. November 2022

Berlin 2022

To Ngoc Dung Do

Abstract

Deep learning algorithms are currently revolutionizing the way computers are used to assist in the solution of challenging problems that impact all aspects of daily life. This shift from rule- or model-driven approaches to data-driven methodologies has just started to unfold in scientific computing. In particular, deep neural networks have lately shown promising results in classical mathematical areas, such as solving inverse problems or partial differential equations (PDEs). Despite overwhelming success in applications, the theoretical foundations of deep learning are from a mathematically rigorous point of view not yet sufficiently apprehended. However, an even deeper theoretical understanding of these techniques may be an essential factor for their acceptance in safety-critical applications, such as medical diagnostics.

In the framework of statistical learning theory, the overall error of learning a function from samples can be decomposed into a training, generalization, and approximation error. Analyzing approximation properties of neural networks is, thus, crucial for extending the understanding of deep learning. This dissertation studies approximation properties of feedforward neural networks. It is divided into a theoretical analysis in classical function spaces (Part A) and an empirical case study at the example of solving a computer tomography inverse problem (Part B).

Function spaces are an essential tool in various mathematical fields, such as the theory of inverse problems or PDEs. Fractional Sobolev spaces, for example, have been shown to be a suitable image model for the theory of computed tomography. In Part A, we quantify the optimal complexity of neural networks (measured in the number of weights and layers) to approximate functions from classical function spaces. For sufficiently smooth activation functions, we provide lower bounds for the number of weights for neural network approximations in (fractional) Sobolev spaces, Besov spaces, and more. For our proofs we make use of the concept of metric entropy and the Vapnik–Chervonenkis-dimension. Furthermore, we develop a unifying framework for the construction of approximate partitions of unity by neural networks with fairly general activation functions. Based on our framework, we derive almost optimal upper bounds in higher-order Sobolev norms.

Part B is devoted to an empirical case study of the approximation properties of neural networks. We investigate if deep-learning-based methods can solve noise-free inverse problems to near-perfect accuracy. For this, we focus on a prototypical computed tomography setup. Computed tomography allows, among other things, the non-invasive study of the human body and is one of the most used medical imaging technologies for diagnostics. A strategy to reduce radiation is to only sample a limited number of measurements which turns the reconstruction task into a severely ill-posed inverse problem. We demonstrate that an iterative end-to-end network scheme enables reconstructions close to numerical precision, comparable to compressed sensing strategies. Apart from an in-depth analysis of our methodology, we layout our conceptual findings. Our results confirm the reliability of deep learning based methods for computed tomography and more broadly for solving inverse problems.

Zusammenfassung in deutscher Sprache

Deep-Learning-basierte Algorithmen revolutionieren derzeit die Art und Weise, wie Computer bei der Lösung anspruchsvoller Probleme, die alle Aspekte des täglichen Lebens beeinflussen, eingesetzt werden. Dieser Wandel von regel- oder modellgetriebenen zu datengetriebenen Herangehensweisen beginnt gerade erst seine Wirkung im wissenschaftlichen Rechnen zu entfalten. Immer häufiger werden Deep-Learning-basierte Algorithmen beispielsweise erfolgreich zur Bewältigung klassisch mathematischer Probleme, wie etwa dem Lösen inverser Probleme oder partieller Differentialgleichungen (PDGen), eingesetzt. Trotz bahnbrechender empirischer Ergebnisse sind die theoretischen Grundlagen von Deep Learning von einem mathematisch rigorosen Standpunkt aus noch nicht hinlänglich verstanden. Eine noch tiefere Einsicht in die zugrunde liegenden Mechanismen könnte ein weiterer Meilenstein zur Akzeptanz Deep-Learning-basierter Algorithmen in Anwendungen mit geringer Fehlertoleranz, wie zum Beispiel in medizinischen Bildgebungsverfahren, sein.

Statistische Lerntheorie untersucht das Lernen einer Funktion von Messungen und beruht darauf, dass der Gesamtfehler in einen Trainings-, Generalisierungs- und Approximationsfehler zerlegt werden kann. Die Analyse der Approximationseigenschaften von neuronalen Feedforward-Netzen ist daher entscheidend für ein tieferes Verständnis Deep-Learning-basierter Algorithmen und Thema der vorliegenden Dissertation. Diese Arbeit untergliedert sich in eine theoretische Analyse in klassischen Funktionenräumen (Teil A) und eine empirische Untersuchung am Beispiel des inversen Problems der Computertomographie (Teil B).

Funktionenräume stellen ein wichtiges Werkzeug in zahlreichen mathematischen Forschungsgebieten, wie etwa der theoretischen Untersuchung inverser Probleme und PDGen, dar. Bestimmte Sobolevräume reellwertiger Ordnung bieten sich beispielsweise für die mathematische Analyse der Computertomographie an. In Teil A leiten wir die optimale Komplexität neuronaler Netze (gemessen in der Anzahl der Gewichte und Schichten) für Approximationen in klassischen Funktionenräumen her. Für hinreichend glatte Aktivierungsfunktionen geben wir untere Komplexitätsschranken für Approximationen mit neuronalen Netzen unter anderem in Besov- und Sobolevräumen an. Unsere Beweise beruhen auf dem Konzept metrischer Entropie und der Vapnik–Chervonenkis-Dimension. Des Weiteren entwickeln wir ein vereinheitlichendes Beweiskonstrukt für eine approximative Zerlegung der Eins mit neuronalen Netzen. Dieses erlaubt es nahezu optimale obere Schranken für Approximation in Sobolevnormen verschiedenen Grades zu bestimmen.

Teil B widmet sich einer empirischen Fallstudie über die Approximationseigenschaften neuronaler Netze. Wir untersuchen, ob Deep-Learning-basierte Methoden rauschfreie inverse Probleme mit fast exakter Genauigkeit lösen können. Dazu betrachten wir ein prototypisches Rekonstruktionsproblem der Computertomographie, welche unter anderem die nicht-invasive Untersuchung des menschlichen Körpers ermöglicht und eine der meistgenutzten medizinischen Bildgebungstechnologien für die Diagnostik ist. Eine Möglichkeit zur Reduzierung der Strahlung ist es, nur eine begrenzte Anzahl an Messungen

durchzuführen, welches die Rekonstruktionsaufgabe zu einem schlecht gestellten inversen Problem macht. Wir zeigen, dass ein iteratives End-to-End Verfahren, basierend auf neuronalen Netzen, Rekonstruktionen ermöglicht, deren numerische Genauigkeit nahe an der Maschienengenauigkeit ist und daher vergleichbar mit Lösungen von Compressed Sensing Methoden ist. Neben einer systematischen Analyse unserer Methodik legen wir unsere konzeptionellen Einsichten dar. Unsere Ergebnisse bestätigen die Zuverlässigkeit von Deep-Learning-basierten Methoden für die Computertomographie und allgemeiner für das Lösen von inversen Problemen.

Acknowledgments

This thesis and, more importantly, the beginning of my journey as a researcher up to this milestone would not be the same without a number of people. Trying to recollect all the acts of kindness, counseling, support, and friendship that I was allowed to experience during the past four years fills my heart with thankfulness. How incredibly lucky was I!

I owe deep gratitude to Maximilian März, who shaped my scientific thinking with his critical mind, filled the role of a mentor when I started this adventure, and helped me grow from a mentee into a collaborator. As a friend, you never hesitated to provide counsel with your clarity when I could not see clearly anymore.

In 2021, I had the pleasure to be welcomed by Klaus-Robert Müller into his group. I would like to thank you dearly for providing help in a time of need and for agreeing to become my "doctor father". I always enjoyed our open conversations and hope that there are many more to come.

Moreover, I am very grateful to Helmut Bölcskei and Philipp Grohs for kindly agreeing to review this thesis. I feel very honored to have you as members of my doctoral committee. Furthermore, I would like to thank Benjamin Blankertz for chairing the doctoral committee.

Collaborating with Mones Raslan was always a pleasure. Our different styles of working complemented each other perfectly and resulted in a smoothness and efficiency that was simply fun. Working together with such a friend brightens even the most intense workdays. I thank you for that. I am also grateful to Martin Genzel, Jan Macdonald, and Maximilian März for bringing me on board of the "near-exact recovery" project. Again, it was a pleasure working together with friends. Kind regards go to Philipp Petersen, a co-author of my very first paper, for fruitful discussions and suggestions. I also wish to express my gratitude to Martin Eigel and Cosmas Heiß for the pleasant collaboration. Furthermore, I would like to thank all members of the (former) Applied Functional Analysis Group at TU Berlin, we were a great team.

During six instructive months at AWS, I was greeted with utmost hospitality and helpfulness by my former team. Here, I would like to thank my colleagues Oliver Borchert, Jan Gasthaus, Tim Januschowski, Shubham Kapoor, Richard Kurle, and Huibin Shen for their support. I am especially indebted to Jan Gasthaus, who gave me a chance in industry, when most of my publications were more of a theoretical nature.

Looking back, I realize that there were many great people who guided me during my studies and inspired me with their love for science. I would like to call out a few of them by name: My former mathematics teacher, Dr. Clemens Janz, encouraged me to study mathematics; Friedrich Philipp brought me closer to research with a demanding but manageable Bachelor thesis project; Benjamin Blankertz motivated me with his enthusiasm and patience in the BCI project; Klaus-Robert Müller taught me the foundations of machine learning.

Johannes Hugger (a.k.a., Hugo), thank you for initiating our great friendship by choosing me as your Analysis 2 homework partner. Knowing that you were also working late, made nightly CoMa programming sessions bearable. I was always inspired by our

comradely and never seriously meant competition.

Finally, I am grateful to Martin Genzel, Mones Raslan, and, in particular, Maximilian März for proofreading this thesis.

Furthermore, I acknowledge support from the Research Training Group "Differential Equation- and Data-driven Models in Life Sciences and Fluid Dynamics: An Interdisciplinary Research Training Group (DAEDALUS)" (GRK 2433) funded by the German Research Foundation (DFG).

I am also thankful to a number of people who are not directly tied to my research but whose contribution is surely not less important: Ronald Kakolewicz taught me how to swim properly and still turns a blind eye on me coming too late to practice regularly. Thank you for continually getting the best out of me in training. I would also like to thank Jian Chen for being a great cycling buddy and friend. I am looking forward to our next tour. A big shout-out goes to my dear friends Alvaro Elze, Raphael Muñoz, and Philipp Thoma! I am happy to have you in my life. Additionally, I would like to thank all friends of mine who were not yet specifically mentioned by name for being a part of my life.

I am profoundly grateful to my family for their unconditional love. My grandparents' dedication and patience gave me the tools to pursue my goals. And with my father's continuous support and trust, I could make them real. I am thankful to my mother for always believing in me. Moreover, I am grateful to have such a great brother at my side. Most of all, I owe my dearest gratitude to Ngoc Dung: For your patience, your advice, and your love. Everyday, I am deeply thankful for your shining presence in my life.

Contents

List of Tables

List of Figures

List of Abbreviations

CT	computed tomography
ELU	exponential linear unit
FBP	filtered backprojection
LoDoPaB	low-dose parallel beam
LPD	learned primal-dual
PDE	partial differential equation
PU	partition of unity
ReLU	rectified linear unit
RePU	rectified power unit
RMSE	root-mean-square-error
TV	total variation
WCRMSE	worst-case RMSE

1

Introduction

In recent years, deep learning methods have been successfully applied to many problems of the natural sciences [Bal18; SB18; NTMC20; Kei+21; Unk+21a]. Prominent examples of such *scientific machine learning* are the development of efficient solution strategies for *inverse problems* [AMÖS19; Ong+20] and *partial differential equations (PDEs)* [BHJK20]. But despite unprecedented empirical performance in numerous practical scenarios, reservations remain about the reliability of these methods in safety-critical applications [ARPAH20; Kno+20; Muc+21; SLBP21].

The study of the approximation power of deep neural networks in various function spaces, typically coined *expressivity*, is one actively researched attempt in the mathematical community to derive an improved understanding of the outstanding effectiveness of deep neural networks. Function spaces characterized by different smoothness properties are ubiquitous in modern mathematics and physics. Sobolev spaces for example, defined by the existence of weak derivatives, build the foundation for the theory of PDEs [Ada75]. Naturally, they also play a major role for the mathematical analysis of inverse problems that are often formulated via partial derivatives [Bel12] or integral equations [Asa11].

In this thesis, we aim at solidifying the understanding of the expressivity of neural networks from a theoretical and empirical perspective. The contribution of this thesis is two-fold:

A Derivation of complexity bounds for neural networks to approximate functions from different smoothness spaces.

B Providing empirical evidence that deep-learning-based methods solve noise-free computed tomography (CT) inverse problems to near-perfect accuracy.

Part A: Approximation Theory for Deep Neural Networks

In this part we relate the complexity of neural networks (measured in the number of non-zero weights) to their approximation power in diverse classical functions spaces and with respect to different distance measures. More formally, for a function space $\mathcal{C}$ and denoting the number of nonzero weights of a neural network Φ by $M(\Phi)$, we tackle the following task:

> *Let $\varepsilon > 0$ and $f \in \mathcal{C}$. We denote by $\Phi_{\varepsilon,f}$ a neural network with the minimal number of weights such that $\|f - \Phi_{\varepsilon,f}\| \leq \varepsilon$. Find the worst case complexity*
>
> $$WC(\varepsilon) := \sup_{f \in \mathcal{C}} M(\Phi_{\varepsilon,f}).$$

We note that $WC(\varepsilon)$ is the worst-case complexity in the sense that it characterizes the necessary and sufficient complexity uniformly over $\mathcal{C}$. In all considered settings $WC(\varepsilon) \to \infty$ as $\varepsilon \to 0$ and the asymptotic speed typically depends on properties of the activation function and the weights. For the weights we consider two scenarios: Firstly, weights representable by a bit-string of moderate length – so called *encodable weights* [PV18; BGKP19; GPEB19] and, secondly, the case of *unconstrained weights*. We approach this problem by providing upper and lower bounds for $WC(\varepsilon)$. Upper bounds are derived via explicit network constructions (for each ε and f) and complemented by lower bounds based on information theoretic concepts.

We make the following contributions:

(i) We transfer lower bounds for the ε-entropy $H_\varepsilon(\mathcal{C}, \mathcal{D})$ from classical functional analysis results to *lower bounds on the number of encodable weights*. These bounds hold with minimal requirements on the activation function and cover for example (fractional) Sobolev, Besov, Hölder, Triebel-Lizorkin, or Zygmund spaces [Tri78; ET96].

(ii) For the case of *unconstrained weights*, we provide lower bounds for approximations in Sobolev spaces based on a VC-dimension argument for piecewise-linear activation functions.

(iii) We derive *almost optimal upper bounds in Sobolev spaces* for neural networks with a wide class of activation functions and encodable weights. For this, we build an abstract, unifying framework which allows to approximate localized Taylor polynomials by neural networks.

(iv) We observe in both, lower and upper bounds, a *trade-off* between the complexity of the approximating neural networks and the order of the approximation norm: Approximations in stronger norms require more weights.

However, the practical relevance of these results needs to be critically reflected. As a motivation for the second part of this thesis, we mention two aspects that prohibit a direct transfer of the above results to applications in the following.

In many real-world scenarios the exact mathematical description of the relevant signal class is highly problem-dependent and notoriously hard to accomplish. Function spaces provide a convenient mathematical construct, that allows a precise analysis, but often only capture overly broad characteristics of the data distribution. Therefore, approximation bounds are generally too pessimistic since the specific structure of the data is not exploited in the analysis. Furthermore, generalization and optimization related components of the function approximation problem are ignored in classical approximation analysis. Recent results, however, point to strong ties between network size (complexity) and optimization together with generalization [BMR21]. To address these issues we conduct an empirical simulation based study for a specific inverse problem in Part B.

Parallel to our theoretical setup, there are many applications where neural networks are used to approximate (discretized) functions from function spaces [EY18; SS18; UVL18]. In contrast, we focus in Part B on learning the solution operator of an inverse problem that in the continuous setting maps *one function space to another*. Approximation results for neural networks in the *operator regime* are an interesting and developing field of research [CC95; Kov+21; LMK22].

Part B: Near-Exact Recovery for Tomographic Inverse Problems via Deep Learning

CT, which allows the non-invasive study of internal structures of objects, is one of the most frequently used medical imaging methods for diagnostics. While the mathematical foundation has already been laid by Johann Radon in 1917 [Rad17], CT is a continuously developing field of research at the intersection of mathematics, engineering and computer science [Nat01; Buz11]. Spectacular successes in computer vision have lately moved deep learning into the center of attention [Lit+17].

The continuous version of the *tomographic measurement operator* F is based on computing line integrals:

$$Fx_0(s,\varphi) = \int_{L(s,\varphi)} x_0(x,y)\,\mathrm{d}(x,y),$$

where x_0 is the unknown image and $L(s,\varphi)$ denotes a line, i.e., φ is the *rotation angle* and s encodes the *sensor position* [Fes17]. The task is the *inverse problem* of reconstructing the image x_0 from the measurements Fx_0. Natterer [Nat80; Nat01] argues that certain fractional Sobolev functions (H^α with α close to $1/2$), which allow (smoothed for $\alpha >= 1/2$) jumps along smooth curves, are a suitable model for the solution space of images and shows that the operator $F : H^\alpha \to H^{\alpha+1/2}$ has a bounded inverse.

Sparse-view CT aims at reducing radiation exposure for patients by sampling only a limited number of measurements. This leads to a severely ill-posed inverse problem causing conventional reconstruction methods like the widely used filtered backprojection (FBP) to introduce serious artifacts. In contrast, sparse-regularization–based algorithms like total variation (TV) minimization, where the solution is given by

$$\arg\min_x \|\nabla x\|_1 \quad \text{such that} \quad Fx = Fx_0,$$

provide perfect recovery from incomplete, noiseless measurements.

The study of this desirable property was popularized by the field of *compressed sensing* [CRT06; Don06; FR13]. Indeed, high precision in the noiseless, undersampled regime can be used to benchmark reconstruction methods and is a driving factor for their acceptance in practice. Therefore, an important open research question for the applicability of deep learning, that we address in this part, is the following:

> *Can deep-learning-based schemes achieve such (near-)perfect solutions of noise-free inverse problems, comparable to model-based algorithms like TV minimization?*

In this regard, Sidky et al. [SLBP21] have recently demonstrated that *post-processing* of FBP images with the prominent UNet-architecture may not yield satisfactory recovery precision in sparse-view CT. This observation gave rise to the recent AAPM Grand Challenge "Deep Learning for Inverse Problems: Sparse-View Computed Tomography Image Reconstruction", with the goal *"to identify the state-of-the-art in solving the CT inverse problem with data-driven techniques"* [Sid+21].

This thesis makes first progress in this direction, building on the winning submission to the AAPM challenge. Our main contributions are as follows:

(i) We show that end-to-end neural networks can achieve *near-perfect accuracy* on the prescribed CT reconstruction task. This underscores the reliability of deep-learning-based solvers for inverse problems, in the sense that they can match the precision of a widely-accepted benchmark (TV minimization) in the noiseless limit.

(ii) A distinctive feature of our approach is that only *very few (five) forward operator evaluations* need to be incorporated to achieve near-perfect recovery. This stands in stark contrast to model-based counterparts, which typically require hundreds or thousands of iterations to converge (resulting in significantly increased computation times).

(iii) We give a *detailed analysis* of the solution strategy, which has significantly outperformed the runner-up teams. Although the challenge amounts to a comparison with 24 competing methods, we also explicitly demonstrate the superiority over several popular baselines in this work. In addition, we show the effectiveness of our learning pipeline beyond synthetically generated image data: The proposed neural network scheme produces state-of-the-art results on the LoDoPaB CT dataset [Leu+21], currently ranked first in the public leaderboard.

(iv) We distill several insights of broader interest and conceptual value. Most notably, we found that *simple building blocks* (e.g., end-to-end training, alternation between learned and model-based components, etc.) and a careful *pre-training strategy* already allow for remarkable performance gains.

List of Publications

The results of this dissertation have been previously published by the author and his collaborators. The author would like to thank all co-authors of the works included in this thesis for agreeing to borrowing ideas, figures, and results. This dissertation is based on the following **four publications**[1]:

[GKP20] I. Gühring, G. Kutyniok, and P. Petersen. **Error bounds for approximations with deep ReLU neural networks in $W^{s,p}$ norms.** *Analysis and Applications*, 18.05 (2020), 803–859.

[GR21] I. Gühring and M. Raslan. **Approximation rates for neural networks with encodable weights in smoothness spaces.** *Neural Networks*, 134 (2021), 107–130.

[GGMM22] M. Genzel, I. Gühring, J. Macdonald, and M. März. **Near-Exact Recovery for Tomographic Inverse Problems via Deep Learning.** *Proceedings of the 39th International Conference on Machine Learning (ICML)*, Vol. 162, 2022, 7368–7381.

[GRK22] I. Gühring, M. Raslan, and G. Kutyniok. **Expressivity of Deep Neural Networks.** *Mathematical Aspects of Deep Learning.* Ed. by P. Grohs and G. Kutyniok. Cambridge: Cambridge University Press, 2022, 149-199.

Further publications by the author that are **not directly included** in this thesis are:

[HGE21] C. Heiß, I. Gühring, and M. Eigel. **A neural multilevel method for high-dimensional parametric PDEs.** *NeurIPS 2021 workshop on The Symbiosis of Deep Learning and Differential Equations*, 2021.

[1] [GGMM22]: A version of this article has also been published at the *NeurIPS 2021 workshop on Deep Learning and Inverse Problems*. See [GGMM21].

Organization of this Thesis

Chapter 2 is based on [GKP20; GR21; GRK22] and presents and discusses the results of Part A. We start by introducing the necessary terminology for neural networks in Section 2.1 and continue by proving lower complexity bounds in Section 2.2. Lower bounds for neural networks with encodable weights for very general function spaces and norms can be found in Section 2.2.1. Our results for networks with arbitrary complex weights are presented in Section 2.2.2. To complement the lower bounds, we derive almost optimal upper approximation bounds for neural networks with fairly general activation functions in Section 2.3. In more detail, we describe the necessary ingredients for these results in Section 2.3.1 and 2.3.2 before outlining the main results as well as the underlying proof strategy in Section 2.3.3. We discuss our results in Section 2.4 and point out some limitations and future work in Section 2.5.

To not interrupt the flow of reading, the proofs of the two main results, Proposition 2.21 and Theorem 2.22, can be found in Appendix A.5 and Appendix A.6, respectively. Basic facts about Sobolev spaces and basic operations one can perform with neural networks have been deferred to Appendices A.1-A.4, respectively. Since our upper bounds depend on abstract properties of the activation function, we provide an analysis of these properties for many practically used activation functions in Appendix A.7.

Chapter 3 is based on [GGMM21] and devoted to Part B. Section 3.1 gives an overview of the AAPM challenge setup and the training/test-data. Section 3.2 provides a conceptual description of our learning pipeline, while more details on the implementation can be found in Appendix B. Our results and several accompanying experiments are reported in Section 3.3. A discussion of our findings and their limitations is provided in Section 3.4.

Chapter 4 provides a meta-level discussion focusing on connections between Part A and Part B, limitations and future directions of research.

2

Part A: Approximation Theory for Deep Neural Networks

This chapter is based on the three works [GKP20; GR21; GRK22] on approximation theory. We start with embedding approximation theory in the greater context of statistical learning theory and proceed by laying out the rich history of this active field of research. The main results for lower and upper complexity bounds are presented in the associated sections.

While for many aspects of the success of deep learning a mathematically fully rigorous theory is not yet available [BMR21; BGKP21; Nak21], approximation properties[1] of neural networks have been studied since around 1960 and are relatively well understood. *Statistical learning theory* formalizes the problem of approximating - in this context also called *learning* - a function from a finite set of samples. Next to statistical and algorithmic considerations, approximation theory plays a major role for the analysis of statistical learning problems. We will clarify this in the following by introducing some fundamental notions.[2]

Assume that $\mathcal{X}$ is an *input space* and $\mathcal{Y}$ a *target space*, $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to [0, \infty]$ is a *loss function* and $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$ a (usually unknown) *probability distribution* on some σ-algebra of $\mathcal{X} \times \mathcal{Y}$. We then aim at finding a minimizer of the risk functional[3]

$$\mathcal{R} : \mathcal{Y}^{\mathcal{X}} \to [0, \infty], \quad f \mapsto \int_{\mathcal{X}\times\mathcal{Y}} \mathcal{L}\left(f(x), y\right) \mathrm{d}\mathbb{P}_{(\mathcal{X},\mathcal{Y})}(x, y),$$

induced by $\mathcal{L}$ and $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$ (where $\mathcal{Y}^{\mathcal{X}}$ denotes the set of all measurable functions from $\mathcal{X}$ to $\mathcal{Y}$). That means we are looking for a function $\hat{f}$ with

$$\hat{f} \in \operatorname{argmin}\left\{\mathcal{R}(f) : \ f \in \mathcal{Y}^{\mathcal{X}}\right\}.$$

In the overwhelming majority of practical applications, however, this optimization problem turns out to be infeasible due to three reasons:

(i) The set $\mathcal{Y}^{\mathcal{X}}$ is simply too large, such that one usually fixes a priori some *hypothesis class* $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ and instead searches for

$$\hat{f}_{\mathcal{H}} \in \operatorname{argmin}\left\{\mathcal{R}(f) : \ f \in \mathcal{H}\right\}.$$

In the context of deep learning, the set $\mathcal{H}$ consists of *deep neural networks*, which we

[1] Throughout the thesis, we will interchangeably use the term *approximation* theory and *expressivity* theory.
[2] [CZ07] provides a concise introduction to statistical learning theory from the point of view of approximation theory.
[3] With the convention that $\mathcal{R}(f) = \infty$ if the integral is not well-defined.

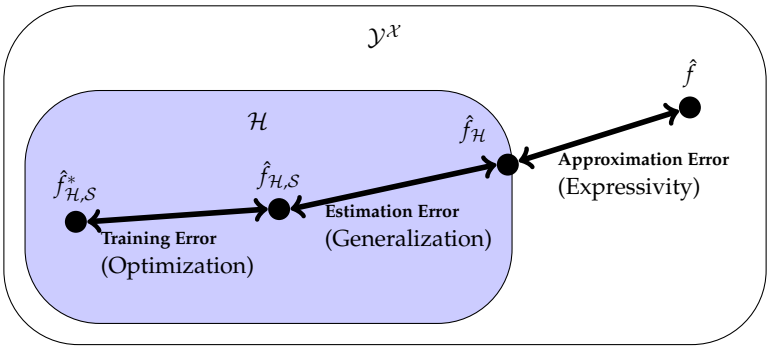


Figure 2.1: **Overall error decomposition.** Decomposition of the overall error into training error, estimation error and approximation error

will introduce in Section 2.1.

(ii) Since $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$ is unknown, one cannot compute the risk of a given function f. Instead, we are given a *training set* $\mathcal{S} = ((x_i, y_i))_{i=1}^m$, which consists of $m \in \mathbb{N}$ i.i.d. samples drawn from $\mathcal{X} \times \mathcal{Y}$ with respect to $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$. Thus, we can only hope to find the minimizer of the *empirical risk* $\mathcal{R}_{\mathcal{S}}(f) = \frac{1}{m}\sum_{i=1}^m \mathcal{L}(f(x_i), y_i))$ given by

$$\hat{f}_{\mathcal{H},\mathcal{S}} \in \operatorname{argmin}\{\mathcal{R}_{\mathcal{S}}(f) : \ f \in \mathcal{H}\}.$$

(iii) In the case of deep learning one needs to solve a complicated non-convex optimization problem to find $\hat{f}_{\mathcal{H},\mathcal{S}}$, which is called *training* and can only be done approximately.

Denoting by $\hat{f}^*_{\mathcal{H},\mathcal{S}} \in \mathcal{H}$ the approximative solution, the overall error can be decomposed as follows (see Figure 2.1 for a visualization)

$$\begin{aligned}\left|\mathcal{R}(\hat{f}) - \mathcal{R}(\hat{f}^*_{\mathcal{H},\mathcal{S}})\right| \leq & \underbrace{\left|\mathcal{R}(\hat{f}^*_{\mathcal{H},\mathcal{S}}) - \mathcal{R}(\hat{f}_{\mathcal{H},\mathcal{S}})\right|}_{\textit{training error}} \\ & + \underbrace{\left|\mathcal{R}(\hat{f}_{\mathcal{H},\mathcal{S}}) - \mathcal{R}(\hat{f}_{\mathcal{H}})\right|}_{\textit{estimation error}} \\ & + \underbrace{\left|\mathcal{R}(\hat{f}_{\mathcal{H}}) - \mathcal{R}(\hat{f})\right|}_{\textit{approximation error}}.\end{aligned}$$

The results discussed in this thesis deal with estimating the approximation error if the set $\mathcal{H}$ consists of deep neural networks. However, practically all of the results presented below ignore the dependence on the unknown probability distribution $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$. This can be justified by different strategies (see also [CZ07]) from which we will depict one here. Under suitable conditions it is possible to bound the approximation error by

$$\left|\mathcal{R}(\hat{f}_{\mathcal{H}}) - \mathcal{R}(\hat{f})\right| \leq \operatorname{error}(\hat{f}_{\mathcal{H}} - \hat{f}),$$

where $\operatorname{error}(\cdot)$ is an expression (e.g. the $\|\cdot\|_\infty$ norm) that is independent of $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$. As an example, assume that $\mathcal{Y} \subset \mathbb{R}$, and the loss function $\mathcal{L}(\cdot, y)$ is Lipschitz continuous for all

$y \in \mathcal{Y}$ with uniform Lipschitz constant $\mathrm{Lip}(\mathcal{L})$. We then get

$$\left|\mathcal{R}(\hat{f}_{\mathcal{H}}) - \mathcal{R}(\hat{f})\right| \leq \mathrm{Lip}(\mathcal{L}) \cdot \left\|\hat{f}_{\mathcal{H}} - \hat{f}\right\|_{\infty}, \tag{2.1}$$

and hence an upper bound of $\|\hat{f}_{\mathcal{H}} - \hat{f}\|_{\infty}$ can be used to upper bound the approximation error.

Expressivity of Neural Networks in Classical Norms

Many attempts at unraveling the extreme efficiency of deep neural networks have been made in the context of approximation theory. The *universal approximation theorem* (see [Cyb89; Fun89; HSW89; Hor91; LLPS93]), which is the starting point of approximation theory of neural networks, states:

> *For every $\hat{f} \in C(K)$ with $K \subset \mathbb{R}^d$ compact and every $\varepsilon > 0$ there exists (under the assumption that the activation function is continuous and not a polynomial) a neural network $\hat{f}_{\mathcal{H},\varepsilon}$ such that $\|\hat{f}_{\mathcal{H},\varepsilon} - \hat{f}\|_{\infty} \leq \varepsilon$.*

Utilizing that neural networks are universal approximators, we can now see from Equation (2.1) that for $\mathcal{H} = C(K)$ the approximation error can be made arbitrarily small. In practice, we are faced with a finite memory and computation budget, which shows the importance of results similar to the theorem above that additionally quantify the complexity of $\hat{f}_{\mathcal{H}}$. The existence of an activation function such that restricted width and depth networks are universal is shown in [MP99] and an explicit activation function based on the countability of the rational numbers with that property is constructed in [GI18]. For ReLU (rectified linear unit) networks with restricted width and unbounded depth universality is established in [KL20].

The necessary and sufficient complexity of (higher-order) sigmoidal neural network approximations for (piecewise) smooth functions has been studied in [Bar94; Mha96; BGKP19][4]. The results in [Mha96] for function approximation in L^p are derived by approximating global (not localized) polynomials with degree increasing concurrently with the approximation accuracy. Our results include these approximation rates as a special case based on an alternative proof strategy. The ansatz in [Mha96] can be used for C^{∞} activation functions with non-vanishing derivatives at some point to obtain network approximations with constant depth and increasing width. Vanishing derivatives of the activation function need to be compensated by increasing depth in order to construct polynomials of increasing degree. This approach is utilized in [TLY19; LTY20], where approximations of weighted L^2-spaces by neural networks with the rectified power unit (RePU) activation function are derived[5]. The function spaces considered therein can be efficiently described by non-localized (Jacobi or Chebychev) polynomials. Complexity bounds for ReLU neural networks based on *localized* polynomial approximation can be found in [Yar17; PV18; OK19; Suz19; Sch20]. The upper bounds in [Yar17, Thm. 1] are covered by our framework as a special case. In [PV18], localization is achieved by approximating characteristic functions. Parts of our proof framework are general enough

[4] In [OK19] rates for locally quadratic activation functions are formulated. However, in the crucial [OK19, Lemma A.3.(d)], there is, in its present form, a gap in the author's reasoning.

[5] which are able to represent polynomials with zero error

to include this approach but we focus on different function classes. Localization by means of wavelet approximations on manifolds is utilized in [SCC18] and by means of general affine systems in [BGKP19; GPEB19]. The approximation error in all of these papers is measured with respect to L^p-norms. Only the papers [PV18; BGKP19; GPEB19] consider the restriction of encodable[6] weights.

Neural Networks and Sobolev Spaces

Expressivity results have so far mainly focused on L^p-type function classes, for $p \in [1, \infty]$. However, recent advances in applications of deep neural networks for inverse problems and PDEs have shown the pressing need to derive a comprehensive understanding of the expressive power of deep neural networks with respect to Sobolev-regular functions. As a motivation for our results, we described the connection of Sobolev spaces to the theory of inverse problems, in particular to the CT inverse problem, in Chapter 1. Here, we will put more emphasis on the connection to elliptic PDEs.

Spaces of functions that admit generalized derivatives fulfilling suitable integrability properties, so-called *Sobolev spaces*, are a crucial concept in modern theory of PDEs (see e.g. [Ada75; Eva99; Rou13]). Given some domain $\Omega \subset \mathbb{R}^d$, integrability order $1 \leq p < \infty$, and regularity $n \in \mathbb{N}$, the Sobolev space $W^{n,p}(\Omega)$ is defined as

$$W^{n,p}(\Omega) := \left\{ f : \Omega \to \mathbb{R} : \int_\Omega |D^\alpha f|^p dx < \infty \text{ for all } \alpha \in \mathbb{N}_0^d \text{ with } |\alpha| \leq n \right\},$$

and equipped with the norm

$$\|f\|_{W^{n,p}(\Omega)} := \left(\sum_{0 \leq |\alpha| \leq n} \int_\Omega |D^\alpha f|^p dx \right)^{1/p}.$$

To study properties of PDEs using functional analytic tools, the *variational formulation* of a PDE is derived. For this, a differential equation is reformulated via a differential operator mapping one function space to another. For a wide range of elliptic PDEs the appropriate spaces in this formulation are Sobolev spaces.

Motivated by the performance of deep learning-based solutions in classical machine learning tasks, neural networks are now also applied for the approximative solution of PDEs. We refer to [BHJK20] for an introduction to the field of deep learning-based methods for PDEs and mention [RTML12; Unk+21b; Sau+22; WMS22] as an incomplete list of concrete applications. In the following, we describe one of these methods in greater detail.

In [EY18] the authors present their meshfree *Deep Ritz method* for approximating solutions of potentially high-dimensional PDEs where the solution u can be expressed as the minimum $u = \arg\min_{v \in V} J(v)$ of a functional $J : V \to \mathbb{R}$ encoding the differential operator and external forces in the PDE. In this setting, V is typically a Sobolev space with regularity $n = 1$. Classical approaches compute an approximation u_h to the solution u by solving the minimization problem over a finite dimensional hypothesis space V_h (instead of V), where V_h is for example a finite element space with underlying mesh of fineness $h > 0$. Standard results in the literature relate the approximation error $\|u_h - u\|_V$ to the

[6] i.e., representable by a bit-string of moderate length. See Section 2.1 for a formal definition.

fineness h of the mesh. In case of the Deep Ritz method, the hypothesis space is a set of functions parameterized by the weights w of a neural network $\mathcal{N}_{\mathrm{PDE}}$. This leads to the following optimization problem

$$w^* = \arg\min_w J(\mathcal{N}_{\mathrm{PDE}}(w)),$$

where $\mathcal{N}_{\mathrm{PDE}}(w^*)$ yields an approximation to the solution u. To derive error bounds relating the network complexity to the approximation error $\|\mathcal{N}_{\mathrm{PDE}}(w^*) - u\|_V$, similar as in the classical setting, one needs to study the expressivity of neural networks with respect to Sobolev-regular norms.

The theoretical foundation for approximating a function with a neural network in Sobolev type norms was already given in a less known version of the universal approximation theorem by Hornik in [Hor91, Thm. 3]. In particular, it was shown that if the activation function ϱ is k-times continuously differentiable, non-constant, and bounded, then any k-times continuously differentiable function f and its derivatives up to order k can be uniformly approximated by a shallow neural network on compact sets. In [COJSP17], it was shown that the theorem also holds for shallow ReLU networks if $k = 1$. Theorem 3 in [Hor91] was also used in [SS18] to show the existence of a shallow network approximating solutions of the PDEs considered in that paper. Note though that in the above results only shallow networks are considered and it is not clear how the number of weights and neurons relates to the approximation error. In [OPS20] approximation rates were derived by re-approximating finite elements. None of these papers examine neural networks with encodable[7] weights.

2.1 Neural Networks: Terminology

We start by formally introducing neural networks closely sticking to the notions introduced in [PV18]. In the following, we will distinguish between a *neural network* as a structured set of weights and the associated function implemented by the network, called its *realization*. Towards this goal, let us fix numbers $L, d = N_0, N_1, \dots, N_L \in \mathbb{N}$.

- A family $\Phi = \big((A_\ell, b_\ell)\big)_{\ell=1}^L$ of matrix-vector tuples of the form $A_\ell \in \mathbb{R}^{N_\ell, N_{\ell-1}}$ and $b_\ell \in \mathbb{R}^{N_\ell}$ is called *neural network*.
- We refer to the entries of A_ℓ, b_ℓ as the weights of Φ and call

  $$M(\Phi) := \sum_{\ell=1}^{L} \left(\|A_\ell\|_0 + \|b_\ell\|_0\right)$$

  its *number of nonzero weights*, $L = L(\Phi)$ its *number of layers* and we call N_ℓ the *number of neurons in layer* ℓ.
- We denote by $d := N_0$ the *input dimension* of Φ and by N_L the *output dimension*.
- Moreover, we set

  $$\|\Phi\|_{\max} := \max_{\ell=1,\dots,L} \max_{\substack{i=1,\dots,N_\ell \\ j=1,\dots,N_{\ell-1}}} \max\{|(A_\ell)_{i,j}|, |(b_\ell)_i|\},$$

[7] See Footnote 6 and Section 2.1 for a formal definition.

which is the *maximum absolute value of all weights*.

- For defining the realization of a network $\Phi = \big((A_\ell, b_\ell)\big)_{\ell=1}^{L}$, we additionally fix an *activation function* $\varrho : \mathbb{R} \to \mathbb{R}$ and a set $\Omega \subset \mathbb{R}^d$. The *realization of the network* $\Phi = \big((A_\ell, b_\ell)\big)_{\ell=1}^{L}$ is the function

$$R_\varrho(\Phi) : \Omega \to \mathbb{R}^{N_L}, \quad x \mapsto x_L\,,$$

where x_L results from the following scheme:

$$\begin{aligned} x_0 &:= x, \\ x_\ell &:= \varrho(A_\ell\, x_{\ell-1} + b_\ell), \quad \text{for } \ell = 1, \dots, L-1, \\ x_L &:= A_L\, x_{L-1} + b_L, \end{aligned}$$

and where ϱ acts componentwise.

- We denote by $\mathcal{NN}_\varrho^d$ the *set of all ϱ-realizations of neural networks with input dimension d and output dimension* 1.[8]

- A *neural network architecture*[9] $\mathcal{A}$ prescribes the number of layers, neurons per layer and which weights in each layer may be nonzero. For a specific choice of weights $w \in \mathbb{R}^{M(\mathcal{A})}$, we denote by $\mathcal{A}(w)$ the neural network with architecture $\mathcal{A}$ and weights w.

Encodability

This information-theoretic viewpoint has already been examined in [PV18; BGKP19; GPEB19] and is motivated by the observation that on a computer only weights of limited complexity (w.r.t. their bit-length) can be stored [10]. We call weights that can be encoded by bit-strings with length logarithmically growing in $1/\varepsilon$, where ε is the approximation accuracy, *encodable*.

To make the notion of encodability more precise, we first introduce coding schemes (see [PV18]): A *coding scheme (for real numbers)* is a sequence $\mathcal{B} = (B_\ell)_{\ell \in \mathbb{N}}$ of maps $B_\ell : \{0,1\}^\ell \to \mathbb{R}$. Now we define sets of neural networks with weights encodable by a coding scheme. Given an arbitrary coding scheme $\mathcal{B} = (B_\ell)_{\ell \in \mathbb{N}}$, and $d \in \mathbb{N}, C_0, \varepsilon, M > 0$, we denote the set of all neural networks Φ with d-dimensional input, one-dimensional output and at most M nonzero weights such that *each nonzero weight of Φ is contained in* $\mathrm{Range}(B_{\lceil C_0 \log_2(1/\varepsilon)\rceil})$ by

$$\mathcal{NN}_{M, \lceil C_0 \log_2(1/\varepsilon)\rceil, d}^{\mathcal{B}}. \tag{2.2}$$

Our Settings

Neural network approximation rates for a function space $\mathcal{C}$ can be studied in different scenarios that generally require different proof strategies. In the following, we present

[8] In the following we will denote by *(ϱ-)neural networks* both neural networks and their corresponding realizations as long it is clear from the context what is meant.

[9] A mathematically more precise definition is given in [GKP20, Definition 2.3].

[10] Encodable weights can still not be stored on a computer with a fixed finite bit-length (e.g. 64 bits), but for each $\varepsilon > 0$ there exists a finite bit-length such that all weights can be stored.

some distinctions that are relevant for our study:

- *Fixed vs. f-adaptive network architecture*: For $\varepsilon > 0$, we call the network architecture *fixed*, if the same network architecture is used for all approximating networks $\{\Phi_{\varepsilon,f} : f \in \mathcal{C}\}$ and f-*adaptive* if the architecture is allowed to depend on the approximated function f. Requiring a fixed architecture is more restrictive but closer to how neural networks are used in practice since the architecture is generally chosen before the training.
- *Encodable vs. unconstrained weights*: We study neural networks with *encodable* weights and with weights of unlimited memory requirements (*unconstrained weights*). Encodable weights are more realistic since in practice neural networks weights need to be stored on a computer with finite memory.

Combining the above categorizations results in four distinct scenarios which are investigated to different extents and for different classes of activation functions in this thesis. However, in the next remark, we observe a (trivial) connection between the above settings which allows to transfer bounds from one setting to the other.

Remark 2.1 Lower bounds for less restrictive assumptions (e.g. f-adaptive architectures with unconstrained weights) directly yield (potentially suboptimal) lower bounds for more restrictive assumptions (e.g. fixed architectures with unconstrained or encodable weights). Conversely, upper bounds for more restrictive assumptions yield (potentially suboptimal) upper bounds for less restrictive assumptions.

2.2 Lower Bounds

In this section, we investigate the necessary complexity of neural network approximations. More mathematically, for a function space $\mathcal{C}$ and activation function $\varrho : \mathbb{R} \to \mathbb{R}$, we consider the following problem:

> *Let $\varepsilon > 0$ and $f \in \mathcal{C}$. We denote by $\Phi_{\varepsilon,f}$ a neural network with the minimal number of weights such that $\|f - R_\varrho(\Phi_{\varepsilon,f})\| \leq \varepsilon$. Find $M_\varepsilon \in \mathbb{N}$ (as large as possible) such that $\sup_{f\in\mathcal{C}} M(\Phi_{\varepsilon,f}) \geq M_\varepsilon$.*

In Section 2.2.1, we derive results for neural networks with encodable weights, very general activation functions, and fixed as well as f-adaptive architectures. In Section 2.2.2, we study the case with unconstrained weights, the ReLU activation function and fixed architecture.

2.2.1 Encodable Weights and General Activation Functions

In this section, we derive lower bounds on the necessary number of nonzero, encodable weights of neural network with fixed and f-adaptive architectures. The considered function spaces include a wide variety of classical smoothness spaces and the accuracy is measured in different smoothness norms. Our result applies to every activation function that is sufficiently smooth to be considered in these norms. We note that the proof of our result is essentially an abstract version of the proof of [PV18, Theorem 4.2]. After encouragement of one of the authors of [PV18] and after studying the paper more closely,

we noticed that it is possible to consider the proof strategy of [PV18, Theorem 4.2] in a more abstract setting which we will outline below. Throughout this section (unless stated otherwise) we fix some $d \in \mathbb{N}$, some domain $\Omega \subset \mathbb{R}^d$ and two normed spaces $\mathcal{C}, \mathcal{D}$ of (equivalence classes of) functions defined on Ω with values in $\mathbb{R}$. Additionally, we assume that $\mathcal{C} \subset \mathcal{D}$.

First of all, we need the notion of the *minimax code length* $L_\varepsilon(\mathcal{C}, \mathcal{D})$ *of* $\mathcal{C}$ *with respect to* $\mathcal{D}$. The minimax code length describes the uniform description complexity of the set $\{f \in \mathcal{C} : \|f\|_{\mathcal{C}} \leq 1\}$ in terms of the number of nonzero bits necessary to encode every f with distortion at most ε in $\mathcal{D}$. It can be directly related to approximation capabilities of arbitrary computing schemes and is defined as follows (see also [PV18, Definition B.2]):

Definition 2.2 (Minimax Code Length) Let $\ell \in \mathbb{N}$. By $\mathfrak{E}^\ell := \left\{E : \mathcal{C} \to \{0,1\}^\ell\right\}$ we denote the set of binary encoders mapping elements of $\mathcal{C}$ to bit strings of length ℓ, and by $\mathfrak{D}^\ell := \{D : \{0,1\}^\ell \to \mathcal{D}\}$ the set of binary decoders mapping bit-strings of length ℓ into $\mathcal{D}$. For $\varepsilon > 0$, we define the *minimax code length* by

$$L_\varepsilon(\mathcal{C}, \mathcal{D}) := \min\left\{\ell \in \mathbb{N} : \exists (E^\ell, D^\ell) \in \mathfrak{E}^\ell \times \mathfrak{D}^\ell : \sup_{f \in \mathcal{C} : \|f\|_{\mathcal{C}} \leq 1} \|D^\ell(E^\ell(f)) - f\|_{\mathcal{D}} \leq \varepsilon\right\}.$$

The next observation demonstrates in the context of neural networks how the minimax code length can be employed to derive lower bounds for approximations with fixed architectures.

Observation 2.3. *Let $\varepsilon > 0$ and $\varrho : \mathbb{R} \to \mathbb{R}$ such that $\mathcal{NN}_\varrho^d \subset \mathcal{D}$. If $\mathcal{A}$ is a neural network architecture with M unspecified nonzero weights[11] (but fixed number of layers, neurons and position of nonzero weights) such that for each $f \in \mathcal{C}$ with $\|f\|_{\mathcal{C}} \leq 1$ there is a set of weights $w_1, \ldots, w_M$, where each weight can be encoded by at most $b \in \mathbb{N}$ bits and $\|R_\varrho(\mathcal{A}(w_1, \ldots, w_M)) - f\|_{\mathcal{D}} \leq \varepsilon$, then*

$$M \geq L_\varepsilon(\mathcal{C}, \mathcal{D})/b. \tag{2.3}$$

Mapping $f \in \mathcal{C}$ to the bit representation of the M weights can be viewed as an encoder, and mapping the encoded weights to $R_\varrho(\mathcal{A}(w_1, \ldots, w_M))$ acts as a decoder with bit length $\ell = Mb$, which shows the claim. This in particular holds true, if $b \leq C \log_2(1/\varepsilon)$ which is the focus of this thesis.

In the following, we exploit this strategy to show that the same bound actually holds true, if we allow for the architecture to depend on the function to be approximated. That means, for each $f \in \mathcal{C}$ the number of layers, neurons and position of M nonzero encodable weights (and the weights themselves) may change but need to be encoded. The next lemma (shown in [PV18, Lemma B.4] under the additional restriction[12] that $\varrho(0) = 0$) shows the number of bits needed to encode this information.

[11] Or any computation scheme that takes as input M parameters.

[12] The lemma is proven by first noting that a network with arbitrary number of neurons and layers, but M non-zero weights, can be replaced by a network with the same number of non-zero weights, but number of neurons and layers bounded by $M + 1$. This can be done by removing neurons that do not contribute to the next layer. This strategy (see also [BGKP19, Proposition 3.6]) allows us to drop the assumption that $\varrho(0) = 0$ from [PV18, Lemma B.4].

Lemma 2.4 *Let $M, K \in \mathbb{N}$, and let $\mathcal{B}$ be an encoding scheme for real numbers and $\varrho : \mathbb{R} \to \mathbb{R}$ an activation function. There is a constant $C = C(d)$, such that there is an injective map $\Gamma : \{R_\varrho(\Phi) : \Phi \in \mathcal{NN}^{\mathcal{B}}_{M,K,d}\} \to \{0,1\}^{CM(K+\lceil \log_2 M \rceil)}$.*

To make the main statement of this section mathematically more precise, we introduce some further notation.

Definition 2.5 Let $C_0 > 0$ be fixed. Additionally, let $f \in \mathcal{C}$, and for some function $\varrho : \mathbb{R} \to \mathbb{R}$ assume that $\mathcal{NN}^d_\varrho \subset \mathcal{D}$. Finally, let $\varepsilon > 0$ and fix some coding scheme $\mathcal{B}$. Then, for $C_0 > 0$, we define the quantities[13]

$$\begin{aligned} M^{\mathcal{B}}_\varepsilon(f) &:= M^{\mathcal{B},\varrho,C_0,\mathcal{C},\mathcal{D}}_\varepsilon(f) \\ &:= \min\left\{ M \in \mathbb{N} : \exists \Phi \in \mathcal{NN}^{\mathcal{B}}_{M,\lceil C_0 \cdot \log_2 \frac{1}{\varepsilon} \rceil, d} : \|f - R_\varrho(\Phi)\|_{\mathcal{D}} \leq \varepsilon \right\}, \end{aligned}$$

and

$$M^{\mathcal{B}}_\varepsilon(\mathcal{C},\mathcal{D}) := M^{\mathcal{B},\varrho,C_0}_\varepsilon(\mathcal{C},\mathcal{D}) := \sup_{f \in \mathcal{C},\ \|f\|_{\mathcal{C}} \leq 1} M^{\mathcal{B},\varrho,C_0,\mathcal{C},\mathcal{D}}_\varepsilon(f).$$

In other words, the quantity $M^{\mathcal{B}}_\varepsilon(f)$ denotes the required number of nonzero weights of a neural network Φ to ε-approximate f with weights that can be encoded with $\lceil C_0 \log_2(1/\varepsilon) \rceil$ bits using the coding scheme $\mathcal{B}$. $M^{\mathcal{B}}_\varepsilon(\mathcal{C},\mathcal{D})$ gives a uniform bound of this quantity over the unit ball in $\mathcal{C}$.

Theorem 2.6 now states that if we can lower bound the minimax code length, then we are also able to lower bound $M^{\mathcal{B}}_\varepsilon(\mathcal{C},\mathcal{D})$. Lower bounds on the minimax code length (and hence for the quantity $M^{\mathcal{B}}_\varepsilon(\mathcal{C},\mathcal{D})$) for specific, frequently used function spaces fulfilling the assumptions of the theorem will be given in Corollary 2.9.

Theorem 2.6 *Let $\varrho : \mathbb{R} \to \mathbb{R}$ such that $\mathcal{NN}^d_\varrho \subset \mathcal{D}$. Additionally, assume that $L_\varepsilon(\mathcal{C},\mathcal{D}) \geq C_1 \varepsilon^{-\gamma}$ for some $\gamma = \gamma(\mathcal{C},\mathcal{D}), C_1 = C_1(\mathcal{C},\mathcal{D}) > 0$ and all $\varepsilon > 0$. Then, for each $C_0 > 0$ there exist constants $C = C(\gamma,\mathcal{C},\mathcal{D},C_0) > 0$ and $\varepsilon_0 = \varepsilon_0(\gamma,\mathcal{C},\mathcal{D},C_0) > 0$, such that for each coding scheme of real numbers $\mathcal{B}$, and for all $\varepsilon \in (0,\varepsilon_0)$ we have*

$$M^{\mathcal{B},\varrho,C_0}_\varepsilon(\mathcal{C},\mathcal{D}) \geq C \cdot \varepsilon^{-\gamma} \Big/ \log_2\left(\frac{1}{\varepsilon}\right).$$

The idea for the proof of this theorem is the same as for Observation 2.3. Here, the encoder is $E : \mathcal{C} \to \{0,1\}^\ell, f \mapsto \Gamma(R_\varrho(\Phi_{\varepsilon,f}))$, where $\Phi_{\varepsilon,f}$ is the neural network ε-approximating f, Γ is the network encoder from Lemma 2.4 and $\ell = CM(\log_2(1/\varepsilon) + \log_2(M))$. The decoder is given by $D : \{0,1\}^\ell \to \mathcal{C}, b \mapsto \Gamma^{-1}(b)$. The bound now follows from $CM(\log_2(1/\varepsilon) + \log_2(M)) \geq C_1 \varepsilon^{-\gamma}$.

Remark 2.7 (Activation Functions) We only require sufficient smoothness of the activation function for the spaces under consideration. Hence, we are in a position to conclude suitable lower bounds for *all* practically used activation functions.

[13] we use the convention that $\min \varnothing = \infty$.

Remark 2.8 (Bounds With Non-Encodable Weights) If one drops the restriction of encodable weights and considers the more general setting of arbitrary weights, a lesser number of weights is required in general. We mention one example here and cover this setting in more depth in the next section.

In [GI16] it is shown that there exists an activation function such that a neural network with three parameters is able to uniformly approximate each function in $\mathcal{C} = C([0,1])$ arbitrary well. Observation 2.3 now shows that there is no finite encoding bit length for the weights necessary to approximate all functions in the unit ball of $C([0,1])$, since in this case $L_\varepsilon(\mathcal{C},\mathcal{C}) = \infty$ for $0 < \varepsilon < 1$.[14]

We proceed by listing a variety of lower bounds for a selection of specific examples for frequently used function spaces. Here, we make use of the fact that, by [Gro15, Remark 5.10], the ε-entropy $H_\varepsilon(\mathcal{C},\mathcal{D})$ is bounded by the minimax code length, i.e., $L_\varepsilon(\mathcal{C},\mathcal{D}) \geq H_\varepsilon(\mathcal{C},\mathcal{D})$. One can deduce similar lower bounds for other choices of $\mathcal{C},\mathcal{D}$. Notable examples that are not covered below include Hölder spaces, Triebel-Lizorkin, or Zygmund spaces (see for instance [Tri78; ET96] and the references therein for further examples).

Corollary 2.9 *Assume that Ω fulfills some regularity conditions.[15] Let $\varrho : \mathbb{R} \to \mathbb{R}$ be chosen such that $\mathcal{NN}_\varrho^d \subset \mathcal{D}$ (where $\mathcal{D}$ is a function space on Ω specified below). Moreover, let $\mathcal{B}$ be an arbitrary coding scheme. Then, the following statements hold:*

(i) ***Besov spaces:*** *Let $s,t \in \mathbb{R}$ with $s < t$ as well as $p_1,p_2,q_1,q_2 \in (0,\infty]$ such that*

$$t - s - d\max\left\{\left(\frac{1}{p_1} - \frac{1}{p_2}\right), 0\right\} > 0.$$

Moreover, let $\mathcal{C} = B^t_{p_1,q_1}(\Omega)$, and $\mathcal{D} = B^s_{p_2,q_2}(\Omega)$. Then, for some $C,\varepsilon_0 > 0$, we have

$$M_\varepsilon^{\mathcal{B}}(\mathcal{C},\mathcal{D}) \geq C\varepsilon^{-\frac{d}{t-s}} \Big/ \log_2\left(\frac{1}{\varepsilon}\right), \qquad \text{for all } \varepsilon \in (0,\varepsilon_0).$$

(ii) ***Sobolev Spaces:*** *Let $s,t \in \mathbb{N}$ with $t > s$ and let $p \in (0,\infty]$. Then, for $\mathcal{C} = W^{t,p}(\Omega)$ and for $\mathcal{D} = W^{s,p}(\Omega)$ there exist some $C,\varepsilon_0 > 0$ with*

$$M_\varepsilon^{\mathcal{B}}(\mathcal{C},\mathcal{D}) \geq C\varepsilon^{-\frac{d}{t-s}} \Big/ \log_2\left(\frac{1}{\varepsilon}\right), \qquad \text{for all } \varepsilon \in (0,\varepsilon_0).$$

***Proof*.** (i) follows immediately from Theorem 2.6 in combination with Theorem [ET96, Section 3.5].

[14] $L_\varepsilon(\mathcal{C},\mathcal{C}) = \infty$ for $0 < \varepsilon < 1$ follows from the fact that the unit ball in $\mathcal{C} = C([0,1])$ is not compact. The same argument can also be used to directly deduce from the construction of the weights in [GI16] that their encoding bit length is not finite.

[15] Many results estimating the ε-entropy are only formulated and proven for C^∞-domains for simplicity of exposition. However, as has been described in [Tri78, Section 4.10.3] and [ET96, Section 3.5], these results remain valid for function spaces on more general domains including cubes.

(ii) follows from Theorem 2.6 together with [EE04, Section 1.3], where we use the estimate on the approximation number (denoted by $a_k(id)$ in [EE04, page 9]) combined with the relation of $a_k(id)$ and the entropy. ∎

2.2.2 Unconstrained Weights, Fixed Architecture, and ReLU Activation Function

This section is devoted to the study of neural network approximations with fixed architecture and unconstrained weights. From now on, we consider functions from the unit ball of the Sobolev space $W^{n,p}$ and introduce for this the notation

$$\mathcal{F}_{n,d,p} := \left\{ f \in W^{n,p}((0,1)^d) : \|f\|_{W^{n,p}((0,1)^d)} \leq 1 \right\}.$$

In Theorem 4 a) in [Yar17] Yarotsky proves that for piecewise linear activation functions a network architecture capable of approximating any function $f \in \mathcal{F}_{n,d,\infty}$ up to a L^∞-error ε has at least $c \cdot \varepsilon^{-d/(2n)}$ weights. We prove an extension of this result to approximations in $W^{1,\infty}$ norm. This stronger norm requires a continuous piecewise linear activation function (with finitely many break points). To simplify the exposition, we only formulate the theorem for the ReLU activation function which is the most prominent representative of this class.

For the sake of readability, we combine the result from [Yar17] with our result in the following theorem.

Theorem 2.10 *Let $\varrho : \mathbb{R} \to \mathbb{R}$, $x \mapsto \max(0, x)$ be the ReLU activation function. Furthermore, let $d \in \mathbb{N}$, $n \in \mathbb{N}_{\geq 2}$ and $k \in \{0, 1\}$. Then, there is a constant $c = c(d, n, k) > 0$ with the following property:*

If $\varepsilon \in (0, 1/2)$ and $\mathcal{A}_\varepsilon = \mathcal{A}_\varepsilon(d, n, k, \varepsilon)$ is an architecture such that for any $f \in \mathcal{F}_{n,d,\infty}$ there is a neural network Φ_ε^f that has architecture $\mathcal{A}_\varepsilon$ and

$$\left\| R_\varrho(\Phi_\varepsilon^f) - f \right\|_{W^{k,\infty}((0,1)^d)} \leq \varepsilon,$$

then $\mathcal{A}_\varepsilon$ has at least $M(\mathcal{A}_\varepsilon) \geq c\varepsilon^{-d/2(n-k)}$ weights.

In the rest of this section, we outline the strategy of the proof of Theorem 2.10. As in [Yar17], we make use of a combinatorial quantity measuring the expressiveness of a set of binary valued functions H defined on some set X, called *VC-dimension* (see e.g. [AB09, Chap. 3.3]). We define

$$\mathrm{VCdim}(H) := \sup \left\{ m \in \mathbb{N} : \begin{array}{l} \text{there exist } x_1, \dots, x_m \in X \text{ such that} \\ \text{for every } y \in \{0,1\}^m \text{ there is a function} \\ h \in H \text{ with } h(x_i) = y_i \text{ for } i = 1, \dots, m \end{array} \right\}.$$

An upper bound of the VC-dimension of a set of functions that can be computed by relatively simple operations is given by [AB09, Theorem 8.4]:

Theorem 2.11 *Suppose h is a function from $\mathbb{R}^M \times \mathbb{R}^d$ to $\{0,1\}$ and let*

$$H = \{x \mapsto h(a,x) : a \in \mathbb{R}^M\}$$

be the class determined by h. Suppose that h can be computed by an algorithm that takes as input the pair $(a,x) \in \mathbb{R}^M \times \mathbb{R}^d$ and returns $h(a,x)$ after no more than t operations of the following types:

- *the arithmetic operations $+, -, \times,$ and $/$ on real numbers and*
- *jumps conditioned on $>, \geq, <, \leq, =,$ and $\neq$ comparisons of real numbers.*

Then $\mathrm{VCdim}(H) \leq 4M(t+2)$.

For the proof of the main result of this section, we make use of the following consequence of the above theorem:

Remark 2.12 If the number of operations t can be bounded by the dimension M of the parametrizing vector a, then $\mathrm{VCdim}(H) \lesssim M^2$.

The overall proof idea of Theorem 2.10 is to construct H_ε based on neural networks with architecture $\mathcal{A}_\varepsilon$ and derive an estimate of the form

$$\alpha(\varepsilon) \leq \mathrm{VCdim}(H_\varepsilon) \leq M(\mathcal{A}_\varepsilon)^2, \tag{2.4}$$

which leads to $\alpha(\varepsilon)^{1/2} \leq M(\mathcal{A}_\varepsilon)$, where $\alpha : \mathbb{R} \to \mathbb{R}$ is a function to be determined during the proof. For activation functions that can be computed by a finite number of operations[16], the number of operations t of a neural network with architecture $\mathcal{A}_\varepsilon$ is linear in the number of weights $M(\mathcal{A}_\varepsilon)$. If the computation of the activation function additionally satisfies the constraints of Theorem 2.10, we get the *upper bound* in Equation (2.4) from Theorem 2.11 together with Remark 2.12. For the *lower bound*, we find a set of $m := \alpha(\varepsilon)$ points $x_1, \ldots, x_m$ such that H_ε can compute all possible dichotomies $y \in \{0,1\}^m$. For this, we construct a smoothed version $f_y : \mathbb{R} \to \mathbb{R}$ of the function $x_i \mapsto y_i$ in $\mathcal{F}_{n,d,\infty}$ (see Figure 2.2), approximate it by a neural network with architecture $\mathcal{A}_\varepsilon$, and threshold it to get the desired binary output.

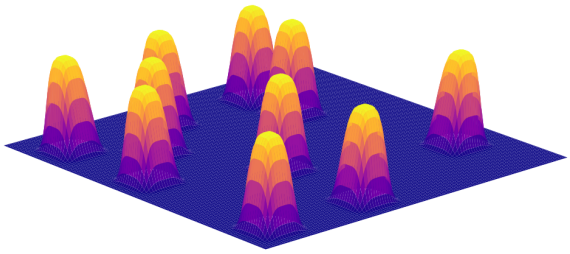


Figure 2.2: **Smooth dichotomy.** The function f_y in $d = 2$ dimensions.

[16] This is e.g. the case for the ReLU or any piecewise polynomial activation function. It holds not true for activation functions that involve the exponential function.

The strategy for the last step differs for approximations in $W^{k,\infty}((0,1)^d)$ with $k = 0$ (shown in [Yar17]) and $k = 1$ (our result). For $k = 0$, suitably thresholding the neural network approximation of f_y directly leads to a correct binary classification of x_i. Precisely, we set $\alpha(\varepsilon) \approx \varepsilon^{-d/n}$ and

$$H_\varepsilon := \left\{ \mathbf{1}_{(-\infty,a]} \circ R_\varrho(\mathcal{A}_\varepsilon(w)) : w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)} \right\},$$

for some (carefully chosen) threshold $a \in \mathbb{R}$, and all together derive the chain of inequalities

$$\varepsilon^{-d/n} \lesssim \mathrm{VCdim}(H_\varepsilon) \lesssim M(\mathcal{A}_\varepsilon)^2.$$

For $k = 1$, we aim at a larger number $m := \alpha(\varepsilon) \approx \varepsilon^{-d/(n-1)}$ of points that can successfully be separated (resulting in a larger lower bound for the number of weights in Theorem 2.10 for $k = 1$). We use that for a $W^{1,\infty}((0,1)^d)$ approximation, the derivatives of f_y and an approximating neural network are close to each other and that the derivatives of f_y are large in the surrounding of a bump and small in the absence of a bump. This more efficient bump detection algorithm allows us to replace n by $n-1$. Choosing H_ε as the thresholded derivatives of $R_\varrho(\mathcal{A}_\varepsilon(w))$ would yield the desired classification. However, the operation of differentiation does not obey the restrictions of Theorem 2.10. Instead, we use that ReLU neural networks are piecewise affine-linear functions and compute the derivative with finite differences. We start by defining a function $g : \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1} \times [0,1]^d \to \mathbb{R}$ by

$$g((w,\delta),x) := \begin{cases} \frac{1}{\delta} \cdot \Big(R_\varrho(\mathcal{A}_\varepsilon(w))(\tilde{x} - \delta e_1) - R_\varrho(\mathcal{A}_\varepsilon(w))(\tilde{x}) \Big), & \text{if } \delta \neq 0 \\ 0, & \text{if } \delta = 0 \end{cases}$$

for $w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)}$, $\delta \in \mathbb{R}$ and $x \in [0,1]^d$, where $\tilde{x}$ are appropriate translations of x. Finally, setting

$$H_\varepsilon := \left\{ \mathbf{1}_{(-\infty,a]} \circ g((w,\delta),\cdot) : (w,\delta) \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1} \right\},$$

yields the inequality

$$\varepsilon^{-d/(n-1)} \lesssim \mathrm{VCdim}(H_\varepsilon) \lesssim M(\mathcal{A}_\varepsilon)^2.$$

2.3 Upper Bounds for General Activation Functions in Sobolev Spaces

Complementary to the lower bounds from the previous section, we provide upper bounds for approximations of Sobolev functions by neural networks with fixed architecture and encodable weights in this section. Since this is the most restricted case, we can transfer the derived bounds to all other scenarios (see also Remark 2.1 and Section 2.4).

In more detail, we show that for an arbitrary accuracy $\varepsilon > 0$, every function from the unit ball of the Sobolev space $W^{n,p}$, denoted by $\mathcal{F}_{n,d,p}$, can be ε-approximated in weaker $W^{k,p}$-Sobolev norms (with $n > k$) by neural networks with fairly general activation function, encodable weights, and fixed architecture. For this, we explicitly construct approximating neural networks with constant depth (i.e., independent of ε) and give upper bounds for the number of nonzero, encodable weights (depending on ε), which, in light of the results of Section 2.2.1, are almost optimal. The main idea is based on the common strategy (see e.g. [Yar17; Sch20]) of approximating f by localized polynomials which in turn are

approximated by neural networks. Our work differs from these other works in three major aspects:

(1) Depending on the smoothness j of the activation function, our approximations include $W^{k,p}$ for $k \leq j$ (instead of maximally $W^{1,p}$).

(2) Constructing a partition of unity (PU) by neural networks with general activation function is tricky (contrary to ReLU networks) and can, in general, only be done approximately (see Section 2.3.1 and Figure 2.3).

(3) Our polynomial approximations and approximate PUs have depth independent of ε, which results in constant-depth approximations of f.

We construct localizing bump functions that form an *(approximate) partition of unity* in Section 2.3.1 and efficiently *approximate polynomials* by neural networks in Section 2.3.2. Afterwards, the statements of the main results as well as a detailed overview of their overall proof strategies are given in Section 2.3.3.

2.3.1 Ingredient I: (Approximate) Partition of Unity

In [Yar17] the ReLU activation function is used to construct continuous, piecewise linear bump functions with compact support that form a PU. However, this approach heavily relies on properties of the ReLU and is only suitable for approximations in Sobolev norms up to order one. For general activation functions, there is, to the best of our knowledge, no canonical way to build a PU by neural networks. As a remedy we introduce approximate partitions of unity which are compatible with all practically used activation functions. In detail, for a gridsize $1/N$ (with $N \in \mathbb{N}$), we divide the domain $(0,1)^d$ into $(N+1)^d$ equally large patches and construct, for each patch Ω_m where $m \in \{0,\dots,N\}^d$, a bump function $\phi_m \in W^{j,\infty}$. Deviating from usually used bump functions, ϕ_m is in general not compactly supported on the corresponding patch and their sum only approximates $\mathbb{1}_{(0,1)^d}$, i.e., $\sum_m \phi_m \approx \mathbb{1}_{(0,1)^d}$. Additionally, we introduce a scaling factor $s \geq 1$, which regulates the closeness of the approximate PU to an exact PU. For $s \to \infty$, we have that $\|\phi_m^s\|_{\Omega_m^c} \to 0$ and $\sum_m \phi_m^s \to \mathbb{1}_{(0,1)^d}$. The overall approximation rates in our main result now also depend on properties of the approximate PU. It will later turn out that the speed of the convergence of the approximate PU is the decisive factor in showing efficient approximation rates. We distinguish between exponential and polynomial speed. Besides the smoothness j and the convergence speed there is another defining quantity τ which we call the *order of the PU*. The order τ specifies at which derivative the scaling factor starts to show. In other words, all derivatives up to order τ absorb the effect of the scaling. In Definition 2.13 we formally introduce the notion of an approximate PU. Additionally to approximate PUs with exponential and polynomial convergence properties we also include exact PUs in this definition since these include (leaky) ReLUs and powers thereof.

Definition 2.13 Let $d \in \mathbb{N}, j, \tau \in \mathbb{N}_0$. We say that the collection of families of functions $(\Psi^{(j,\tau,N,s)})_{N\in\mathbb{N},s\in\mathbb{R}_{\geq 1}}$, where each $\Psi^{(j,\tau,N,s)} := \{\phi_m^s : m \in \{0,\dots,N\}^d\}$ consists of $(N+1)^d$ functions $\phi_m^s : \mathbb{R}^d \to \mathbb{R}$, is an *exponential* (respectively *polynomial*, *exact*) *partition of unity of order* τ *and smoothness* j, or short *exponential* (*polynomial*, *exact*)

(j,τ)-*PU*, if the following conditions are met:

There exists some $D > 0$, $C = C(k,d) > 0$ and $S > 0$ such that for all $N \in \mathbb{N}, s \geq S, k \in \{0,\dots,j\}$ the following properties hold:

(i) $\|\phi_m^s\|_{W^{k,\infty}(\mathbb{R}^d)} \leq CN^k \cdot s^{\max\{0,k-\tau\}}$ for every $\phi_m^s \in \Psi^{(j,\tau,N,s)}$;

(ii) for $\Omega_m^c = \left\{x \in \mathbb{R}^d : \|x - \frac{m}{N}\|_\infty \geq \frac{1}{N}\right\}$, we have

$$\|\phi_m^s\|_{W^{k,\infty}(\Omega_m^c)} \leq \begin{cases} CN^k s^{\max\{0,k-\tau\}} e^{-Ds}, & \text{if } \textit{exponential PU}, \\ CN^k s^{\max\{0,k-\tau\}} s^{-D}, & \text{if } \textit{polynomial PU}, \\ 0, & \text{if } \textit{exact PU}, \end{cases}$$

for every $\phi_m^s \in \Psi^{(j,\tau,N,s)}$.

(iii) We have

$$\left\| \mathbb{1}_{(0,1)^d} - \sum_{m \in \{0,\dots,N\}^d} \phi_m^s \right\|_{W^{k,\infty}((0,1)^d)} \leq \begin{cases} CN^k s^{\max\{0,k-\tau\}} e^{-Ds}, & \text{if } \textit{exponential PU}, \\ CN^k s^{\max\{0,k-\tau\}} s^{-D}, & \text{if } \textit{polynomial PU}, \\ 0, & \text{if } \textit{exact PU}. \end{cases}$$

(iv) There exists a function $\varrho : \mathbb{R} \to \mathbb{R}$ such that for each $\phi_m^s \in \Psi$ there is a neural network Φ_m^s with d-dimensional input and d-dimensional output, with two layers and C nonzero weights, that satisfies

$$\prod_{l=1}^{d} [R_\varrho(\Phi_m^s)]_l = \phi_m^s,$$

and $\|R_\varrho(\Phi_m^s)\|_{W^{k,\infty}((0,1)^d)} \leq CN^k \cdot s^{\max\{0,k-\tau\}}$. Furthermore, for the weights of Φ_m^s it holds that $\|\Phi_m^s\|_{\max} \leq CsN$.

In the next definition, we state sufficient conditions for an activation function ϱ to admit (in the sense of Definition 2.13 (iv)) an exponential (polynomial, exact) PU of order τ with smoothness j for $\tau \in \{0,1\}$ and afterwards we explicitly construct the corresponding PUs. For $\tau = 0$ the activation functions are approximately piecewise constant outside of a neighborhood of zero (e.g., sigmoidal) and for $\tau = 1$ approximately piecewise affine-linear outside of a neighborhood of zero (e.g., exponential linear unit (ELU)). The speed they approach their asymptotes with (see (d) in the next definition) defines the convergence speed of the resulting PU. Furthermore, we require ϱ to be j-smooth.

Definition 2.14 Let $j \in \mathbb{N}_0, \tau \in \{0,1\}$. We say that a function $\varrho : \mathbb{R} \to \mathbb{R}$ is *exponential (polynomial, exact)* (j,τ)*-PU-admissible*, if

(a) ϱ is $\begin{cases} \text{bounded}, & \text{if } \tau = 0, \\ \text{Lipschitz continuous}, & \text{if } \tau = 1; \end{cases}$

(b) There exists $R > 0$ such that $\varrho \in C^j(\mathbb{R} \setminus [-R,R])$;

(c) $\varrho' \in W^{j-1,\infty}(\mathbb{R})$, if $j \geq 1$;

(d) There exist $A = A(\varrho), B = B(\varrho) \in \mathbb{R}$ with $A < B$, some $C = C(\varrho, j) > 0$ and some $D = D(\varrho, j) > 0$ such that

(d.1) $\left|B - \varrho^{(\tau)}(x)\right| \leq Ce^{-Dx}$ (Cx^{-D} if *polynomial*, 0 if *exact*) for all $x > R$;

(d.2) $\left|A - \varrho^{(\tau)}(x)\right| \leq Ce^{Dx}$ ($C'|x|^{-D}$ if *polynomial*, 0 if *exact*) for all $x < -R$;

(d.3) $\left|\varrho^{(k)}(x)\right| \leq Ce^{-D|x|}$ ($C|x|^{-D}$ if *polynomial*, 0 if *exact*) for all $x \in \mathbb{R} \setminus [-R, R]$ and all $k = \tau + 1, \ldots, j$.

Remark 2.15 To give the reader a better intuition for the above definition we mention the similarity to τ-*degree sigmoidal functions* (see [MM92]) defined as $\varrho : \mathbb{R} \to \mathbb{R}$ with

$$\lim_{x \to -\infty} \frac{\varrho(x)}{x^\tau} = 0, \quad \lim_{x \to \infty} \frac{\varrho(x)}{x^\tau} = 1.$$

Roughly speaking, we require the same asymptotic behavior (with the exception that the asymptotes do not need to be $0, 1$) and additionally that the asymptotes are approached with a certain speed.

In Table 2.1 (see Section 2.3.3), we list a large variety of commonly used activation functions and their corresponding PU properties. The proofs of these properties can be found in Appendix A.7.

In the next definition, we give (depending on τ) a recipe for the construction of a one-dimensional approximate bump from which multi-dimensional bumps are derived via a tensor approach. To give the reader a better impression of the definition below and the role of the scaling factor, we present exponential, polynomial and exact bumps and resulting PUs for different activation functions and scaling s in Figure 2.3.

Definition 2.16 Let $j \in \mathbb{N}_0, \tau \in \{0, 1\}$. Assume that $\varrho : \mathbb{R} \to \mathbb{R}$ is exponential, polynomial or exact (j, τ)-PU-admissible. We define, for a scaling factor $s \geq 1$, the one-dimensional bump functions

$$\psi^s : \mathbb{R} \to \mathbb{R},$$

$$\psi^s(x) := \begin{cases} \frac{1}{B-A} \left(\varrho(s(x+3/2)) - \varrho(s(x-3/2))\right), & \text{if } \tau = 0, \\ \frac{1}{s(B-A)} \left(\varrho(s(x+2)) - \varrho(s(x+1)) - \varrho(s(x-1)) + \varrho(s(x-2))\right), & \text{if } \tau = 1. \end{cases}$$

For $N \in \mathbb{N}$, $d \in \mathbb{N}$ and $m \in \{0, \ldots, N\}^d$, we define multi-dimensional bumps $\phi_m^s : \mathbb{R}^d \to \mathbb{R}$ as a tensor product of scaled and shifted versions of ψ^s. Concretely, we set

$$\phi_m^s(x) := \prod_{l=1}^{d} \psi^s \left(3N \left(x_l - \frac{m_l}{N}\right)\right).$$

Finally, for $N \in \mathbb{N}, s \geq 1$, the collection of bump functions is denoted by

$$\Psi^{(j,\tau,N,s)}(\varrho) := \{\phi_m^s : m \in \{0, \ldots, N\}^d\}.$$

In the next lemma we show that the conditions from Definition 2.14 together with the construction in Definition 2.16 are indeed sufficient to generate an (approximate) PU.

Lemma 2.17 *Let $j \in \mathbb{N}_0, \tau \in \{0,1\}$ and a function $\varrho : \mathbb{R} \to \mathbb{R}$ be* exponential (polynomial, exact) (j,τ)-PU-admissible. *Then, the collection of families of functions $(\Psi^{(j,\tau,N,s)}(\varrho))_{N\in\mathbb{N},s\in\mathbb{R}_{\geq 1}}$ defined in Definition 2.16 is an exponential (polynomial, exact) PU of order τ and smoothness j.*

***Proof*.** The proof of this statement is the subject of Appendix A.5.1. We only give the proof for exponential PUs. The statement for the other two cases follows analogously. ∎

We demonstrate in Appendix A.7 the admissibility for many practically-used activation functions. In Table 2.1 below we have included the types of PUs these activation functions induce.

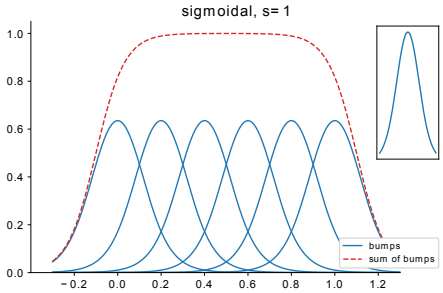
Remark 2.18 Definition 2.14 can be generalized to higher $\tau \geq 2$, resulting in an increasing amount of terms in the definition of a bump. Since most activation functions used in practice are of order $\tau \in \{0,1\}$, we did not introduce this concept for simplicity of exposition. An example of $(\tau \geq 2)$-functions are *τ-order RePUs* (see, e.g., [LTY20]), given by ReLU^τ. Due to its obvious connections to *B-splines* of order $\tau+1$ (see for instance [De 01, Chapter IX]), and their ability to form an exact PU ([De 01, p. 96]) as well as their smoothness properties, it is clear that the resulting system $\left(\Psi^{(\tau,\tau,N,s)}(\mathrm{ReLU}^\tau)\right)_{N\in\mathbb{N},s\geq 1}$ forms an exact (τ,τ)-PU.

We conclude this section by giving an overview of further works that introduce different types of PUs. An approach which is similar to ours for functions of sigmoidal type has been used in [CS13a; CS13b; CSG19]. There, approximate bumps are constructed from differences of scaled and shifted sigmoidals. The key difference is that for a fixed patch the contributions of the neighboring approximate bump functions do not decrease with the number of patches N going to infinity which is an important factor in our construction. In [Lin19], characteristic functions χ_p for each patch are L^∞-approximated in order to achieve localization. However, in this work, the Heaviside function is used as an activation function in the first layer (followed by a different activation function in the next layer), which is not transferable to our work, since it prevents higher-order Sobolev approximations.

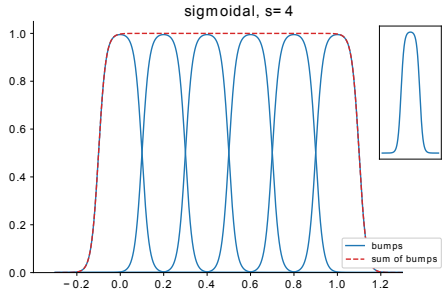
2.3.2 Ingredient II: Approximation of Polynomials

Later on, we approximate our target function by localized polynomials $\sum \phi_m \cdot \mathrm{poly}_m$, where the ϕ_m are the localizing functions from Section 2.3.1[17]. Afterwards, we emulate these localized polynomials by neural networks[18]. For this, we need to approximate polynomials in an efficient way. We start with approximating monomials $x \mapsto x^r$ on $\mathbb{R}$ by two-layered neural networks with activation functions that have a non-vanishing Taylor coefficient of order $r \in \mathbb{N}$. The construction is mainly based on a generalization of a standard approach for approximating the function $x \mapsto x^2$ by using finite differences. This

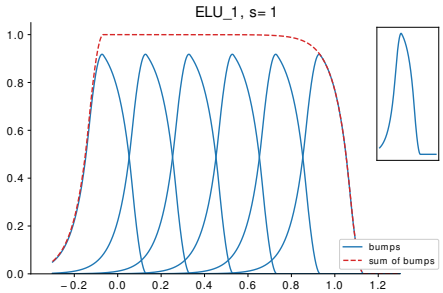
[17] see Appendix A.5.2 for the precise statement and its proof
[18] see Lemma A.26 in Appendix A.5.3 for the final statement and its proof



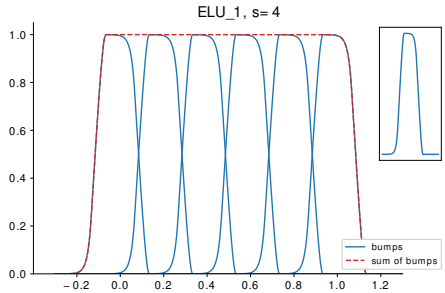
(a) Exponential PU implemented by sigmoid-neural networks with scaling $s = 1$.



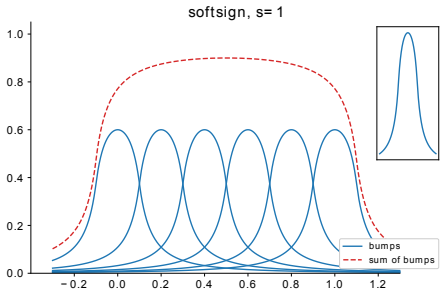
(b) Exponential PU implemented by sigmoid-neural networks with scaling $s = 4$.



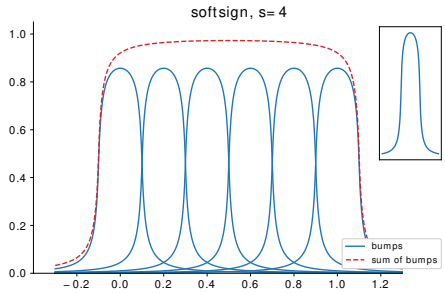
(c) Exponential PU implemented by ELU-neural networks with scaling $s = 1$.



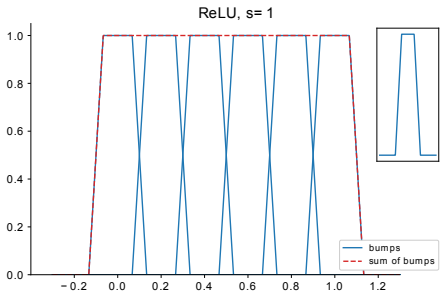
(d) Exponential PU implemented by ELU-neural networks with scaling $s = 4$.



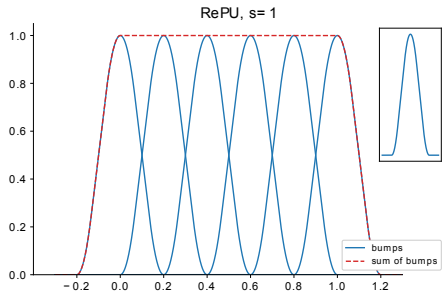
(e) Polynomial PU implemented by softsign-neural networks with scaling $s = 1$.



(f) Polynomial PU implemented by softsign-neural networks with scaling $s = 4$.



(g) Exact PU implemented by ReLU-neural networks used in [Yar17] and Remark 2.23.



(h) Exact PU implemented by quadratic RePU-neural networks ($\tau = 2$).

Figure 2.3: (Previous page.) **Partitions of unity.** All displayed partitions of unity have 6 bumps ($N = 5$). The red curve shows the sum of the bump functions. A single bump function can be seen in the small window in the upper right part of each plot. The first two rows depict an exponential PU for $\tau = 0$ (first row) and $\tau = 1$ (second row). A polynomial PU of order $\tau = 0$ can be seen in the third row. The impact of increasing the scaling factor s can be seen in the second column. In the last row two exact PUs are shown. Here, the sum is constant 1 on $(0, 1)$ and scaling has no impact.

has been studied in [RT18] and variations thereof have been considered, e.g., in [OK19; SZ19].

Proposition 2.19 *Let $\varrho : \mathbb{R} \to \mathbb{R}$ be a function. Assume, that for some $n \in \mathbb{N}$ there exists $x_0 \in \mathbb{R}$ such that ϱ is $n+1$ times continuously differentiable in some open neighborhood U around x_0 and $\varrho^{(r)}(x_0) \neq 0$ for some $r \in \{1, \dots, n\}$. Then, for every $\varepsilon \in (0, 1)$, and every $B > 0$ there exists a constant $C = C(B, \varrho, r, n) > 0$ as well as a neural network Φ_ε^r with $R_\varrho(\Phi_\varepsilon^r)|_{[-B,B]} \in C^{n+1}([-B, B])$ and the following properties:*

(i) $\left\|R_\varrho(\Phi_\varepsilon^r) - x^r\right\|_{C^k([-B,B])} \leq \varepsilon$ *for all* $k = 0, \dots, n$;

(ii) $|R_\varrho(\Phi_\varepsilon^r)|_{W^{k,\infty}([-B,B])} \leq C\frac{r!}{(r-k)!}B^{r-k}$ *for* $k = 0, \dots, r$ *and* $|R_\varrho(\Phi_\varepsilon^r)|_{W^{k,\infty}([-B,B])} \leq \varepsilon$ *for* $k = r+1, \dots, n$;

(iii) $L(\Phi_\varepsilon^r) = 2$, *as well as* $M(\Phi_\varepsilon^r) \leq 3(r+1)$;

(iv) $\|\Phi_\varepsilon^r\|_{\max} \leq C\varepsilon^{-r}$.

Proof. The proof of this result can be found in Appendix A.4.2. ∎

Proposition 2.19 comes handy for two other usages besides monomial approximation:

- We construct neural networks which implement an *approximate multiplication* (see Corollary A.18) via the polarization identity

$$xy = \frac{1}{4}\left((x+y)^2 - (x-y)^2\right) \quad \text{for } x, y \in \mathbb{R}.$$

  This can by now be considered a standard approach in neural network approximation theory (originally used in [Yar17]). For this, the assumptions from Proposition 2.19 need to be fulfilled for $n = 2$ and $r = 2$, which holds true for all activation functions listed in Table 2.1 *except for the (leaky) ReLU*[19]. We use the approximate multiplication to obtain approximations of the multi-dimensional bumps ϕ_m from one-dimensional bumps which are in turn by construction neural networks. Furthermore, we can now deal with the multiplication of and ϕ_m with poly_m (see Corollary A.18 in Appendix A.4.2 and Lemma A.25 in Appendix A.5.3).

- It is often useful to pass output from a layer to a non neighboring layer deeper in the network. Previous works have solved this issue for the ReLU activation function by constructing an identity network (e.g., [PV18]). For general activation functions this is not possible. With help of Proposition 2.19 (for $n = 1$ and $r = 1$) an *approximate*

[19] For these activation functions see Remark 2.23.

identity neural network can be constructed (see Proposition A.19). It is clear that all activation functions listed in Table 2.1 fulfill the requirements.

2.3.3 Main Results Based on Ingredients I & II

The proof of the main statement of this section can be roughly divided into two steps: In Proposition 2.21, the approximating neural networks are constructed with weights whose absolute values are bounded polynomially in ε^{-1}. In Theorem 2.22, the encodability of the weights is enforced. Before we state the actual results we give an overview of the proof of Proposition 2.21, in which we explain the different approximation rates that can be obtained from different PUs. We hope that this excursus will make it easier for the reader to keep track of the different approximation rates presented in the results of this section.

Overview of Our Proof Strategy. *Let $\varepsilon > 0$. The proof of Proposition 2.21 is based on approximating a sum of $N^d = N(\varepsilon)^d$ localized Taylor polynomials (which are close to f) by a neural network $\Phi_{P,\varepsilon}$, such that we get*

$$\begin{aligned}&\left\| f - R_\varrho(\Phi_{P,\varepsilon}) \right\|_{W^{k,\infty}((0,1)^d)} \\ &\quad \leq \underbrace{\left\| f - \sum_m \phi_m^s \mathrm{poly}_m \right\|_{W^{k,\infty}((0,1)^d)}}_{\text{Step 1}} + \underbrace{\left\| \sum_m \phi_m^s \mathrm{poly}_m - R_\varrho(\Phi_{P,\varepsilon}) \right\|_{W^{k,\infty}((0,1)^d)}}_{\text{Step 2}}.\end{aligned}$$

***Step 1**: We start by depicting how our PUs are used together with localized Taylor polynomials. In the process the interplay between the convergence speed of the PUs and the approximation rates that can be obtained becomes clear. When approximating a function f by localized Taylor polynomials poly_m, where the localization is realized by a PU from Section 2.3.1, we estimate the error on a fixed patch $\Omega_{\tilde{m}}$ by*

$$\left\| f - \sum_m \phi_m^s \mathrm{poly}_m \right\|_{W^{k,\infty}(\Omega_{\tilde{m}})} \leq \left\| \left(\mathbb{1}_{(0,1)^d} - \sum_m \phi_m^s \right) f \right\|_{W^{k,\infty}(\Omega_{\tilde{m}})} + \left\| \sum_m \phi_m^s (f - \mathrm{poly}_m) \right\|_{W^{k,\infty}(\Omega_{\tilde{m}})}.$$

The first term can be handled by Definition 2.13 (iii) of the PU. Here, we only focus in detail on the second term. We have

$$\left\| \sum_m \phi_m^s (f - \mathrm{poly}_m) \right\|_{W^{k,\infty}(\Omega_{\tilde{m}})} \leq C \underbrace{\sum_{\|m-\tilde{m}\|_\infty > 1} \|\phi_m^s\|_{W^{k,\infty}(\Omega_{\tilde{m}})}}_{\text{(a)}} + \underbrace{\sum_{\|m-\tilde{m}\|_\infty \leq 1} \|\phi_m^s (f - \mathrm{poly}_{\tilde{m}})\|_{W^{k,\infty}(\Omega_{\tilde{m}})}}_{\text{(b)}}.$$

In the cases of exponential/polynomial PUs, we will make use of the decay property of Definition 2.13 (ii). In general we get

$$\sum_{\|m-\tilde{m}\|_\infty > 1} \|\phi_m^s\|_{W^{k,\infty}(\Omega_{\tilde{m}})} \lesssim N^d \cdot \begin{cases} CN^k s^{\max\{0,k-\tau\}} e^{-Ds}, & \text{if exponential PU,} \\ CN^k s^{\max\{0,k-\tau\}} s^{-D}, & \text{if polynomial PU,} \\ 0, & \text{if exact PU.} \end{cases}$$

The closeness of the approximate bump to an exact bump is determined by the scaling factor s which

we now couple with N.

- *For the* exponential case *we set $s := N^\mu$ for arbitrarily small $\mu > 0$ and can now use that the exponential term decays faster than any polynomial in N grows. In particular, we have*

$$N^d N^k s^{\max\{0,k-\tau\}} e^{-Ds} = N^d N^k N^{\mu(k=2)} e^{-DN^\mu} \leq N^{-(n-k)}$$

  for N large enough.

- *In the* polynomial case *an arbitrarily small exponent is not sufficient to get rid of N^d, instead we must set $s := N^{\frac{d+k+(n-k)}{D}}$ and get*

$$N^d N^k s^{\max\{0,k-\tau\}} s^{-D} = N^d N^k N^{-d-k-(n-k)} = N^{-(n-k)} \quad \text{for} \quad k \leq \tau.$$

  Here, we can only compensate the term N^d for $k \leq \tau$, since only the derivatives up to order τ absorb the effect of the scaling.

- *Finally, in case of an exact PU, term (a) is zero.*

For term (b) we only consider $m = \widetilde{m}$. For $k \geq \tau + 1$, we now pay the price for the scaling in the exponential case, since there is no exponential decay for the derivative of $\phi^s_{\widetilde{m}}$ on the patch $\Omega_{\widetilde{m}}$. From Definition 2.13 (i) together with the Bramble-Hilbert Corollary A.11 we get the estimate

$$\|\phi^s_m (f - \mathrm{poly}_{\widetilde{m}})\|_{W^{k,\infty}(\Omega_{\widetilde{m}})} \lesssim \begin{cases} N^{-(n-k-\mu_{(k=2)})}, & \text{if exponential PU,} \\ N^{-(n-k)}, & \text{for } k \leq \tau, \text{ if polynomial PU,} \\ N^{-(n-k)}, & \text{if exact PU.} \end{cases}$$

Combining the computations for (a) and (b) we get the total estimate in Step 1

$$\left\| \sum_m f - \phi^s_m \mathrm{poly}_m \right\|_{W^{k,\infty}(\Omega_{\widetilde{m}})} \lesssim \begin{cases} N^{-(n-k-\mu_{(k=2)})}, & \text{if exponential PU,} \\ N^{-(n-k)}, & \text{for } k \leq \tau, \text{ if polynomial PU,} \\ N^{-(n-k)}, & \text{if exact PU.} \end{cases}$$

By choosing $N := \lceil \varepsilon^{-1/(n-k-\mu_{(k=2)})} \rceil$ in the exponential case and $N := \lceil \varepsilon^{-1/(n-k)} \rceil$ in the other two cases, we get that the term from Step 1 can be bounded by ε.
Step 2: *To construct the neural network we use the results from Section 2.3.2 to*

(i) approximate Taylor polynomials by neural networks;

(ii) approximate the multi-dimensional PU. Since only the d factors of its tensor structure can be exactly represented by a neural network (see Definition 2.13 (iv)), their multiplication must be approximated;

(iii) approximate the multiplication of (i) with (ii) by neural networks $\Phi_{m,\widetilde{\varepsilon}}$ with accuracy $\widetilde{\varepsilon}$ (chosen below);

(iv) build the sum of all approximations of localized Taylor polynomials by neural networks.

The network $\Phi_{P,\varepsilon}$ thus consists of the subnetworks from step (iii). We get the estimate

$$\left\| \sum_m \phi^s_m \mathrm{poly}_m - R_\varrho(\Phi_{P,\varepsilon}) \right\|_{W^{k,\infty}((0,1)^d)} \leq \sum_m \|\phi^s_m \mathrm{poly}_m - \Phi_{m,\widetilde{\varepsilon}}\|_{W^{k,\infty}((0,1)^d)} \lesssim N^d \widetilde{\varepsilon}.$$

Consequently, we need to chose $\widetilde{\varepsilon} := \varepsilon N^{-d} \approx \varepsilon^{-d/(n-k-\mu_{(k=2)})+1}$ *(some terms are suppressed here for simplicity of exposition). We can only do this, since neither the number of weights of* $\Phi_{m,\widetilde{\varepsilon}}$ *nor its number of layers depends on* $\widetilde{\varepsilon}$ *(only the* values *of the weights do). In other words, each* $\Phi_{m,\widetilde{\varepsilon}}$ *has a constant number of weights and layers. Combining* $\sim N^d$ *of such networks to get* $\Phi_{P,\varepsilon}$ *yields a network with about* $N^d = \varepsilon^{-d/(n-k-\mu_{(k=2)})}$ *weights and constant number of layers for the exponential case (with obvious adaptations for the other two cases).*

Conclusion: *For activation functions* ϱ *with an exponential PU, we obtain optimal rates for Sobolev norms* $k \leq \tau$ *and almost optimal rates for* $k \geq \tau + 1$*; in the polynomial case, we get optimal approximation rates only in* $W^{k,p}$*-norms if* $k \leq \tau$*; in the case of an exact PU, we get optimal approximation rates for Sobolev norms up to order* j *(smoothness of* ϱ*).*

Remark 2.20 (Fixed architecture) Note that the network $\Phi_{P,\varepsilon}$ only depends on a specific f via the weights in the last layer, which are the coefficients of the monomials that together form the Taylor polynomials poly_m. Since the approximate PU and the monomials are independent on f, it is easy to see that there exists a neural network architecture $\mathcal{A}_\varepsilon$ such that $\Phi_{P,\varepsilon}$ has architecture $\mathcal{A}_\varepsilon$ for every of choice of f.

We now give the statement of Proposition 2.21, which can be proven by using the ideas and concepts presented so far in this section. The detailed proofs are executed in Appendices A.5.1-A.5.4, mostly for the case of *exponential* (j,τ)-PUs. The statements for the other two cases can be proven in an analogous way.

Proposition 2.21 *We make the following assumptions:*

- *Let* $d \in \mathbb{N}$, $j, \tau \in \mathbb{N}_0, k \in \{0,\dots,j\}, n \in \mathbb{N}_{\geq k+1}$, $1 \leq p \leq \infty$, *and* $\mu > 0$*;*
- *let* $\varrho : \mathbb{R} \to \mathbb{R}$ *such that* $\left(\Psi^{(j,\tau,N,s)}(\varrho)\right)_{N\in\mathbb{N},s\geq 1}$ *is an exponential (polynomial, exact)* (j,τ)*-PU;*
- *there exists* $x_0 \in \mathbb{R}$ *such that* ϱ *is three times continuously differentiable in a neighborhood of* x_0 *and* $\varrho''(x_0) \neq 0$*.*

Then, there exist constants $L, C, \theta, \widetilde{\varepsilon}$ *depending on* d, n, p, k, μ *with the following properties:*
For every $\varepsilon \in (0, \widetilde{\varepsilon})$ *and every* $f \in \mathcal{F}_{n,d,p}$*, there is a neural network* $\Phi_{\varepsilon,f}$ *with* d*-dimensional input and one-dimensional output, at most* L *layers and at most*

$$\begin{cases} C\varepsilon^{-d/(n-k-\mu_{(k=2)})}, & \text{if exponential PU ,} \\ C\varepsilon^{-d/(n-k)}, & \text{for } k \leq \tau, \text{ if polynomial PU ,} \\ C\varepsilon^{-d/(n-k)}, & \text{if exact PU ,} \end{cases}$$

nonzero weights bounded in absolute value by $C\varepsilon^{-\theta}$ *such that*

$$\|R_\varrho(\Phi_{\varepsilon,f}) - f\|_{W^{k,p}((0,1)^d)} \leq \varepsilon.$$

The main theorem now states that Proposition 2.21 also holds with encodable weights, i.e. for each $\varepsilon > 0$, every element of the set of weights $W_\varepsilon = \bigcup_f W_{\varepsilon,f}$ (where $W_{\varepsilon,f}$ denotes the weights of $\Phi_{\varepsilon,f}$) can be uniquely encoded by $\lceil C \log_2(1/\varepsilon)\rceil$ bits. To state this in a formal way, we use the notation introduced in Equation (2.2).

Theorem 2.22 *We make the following assumptions:*

- *Let $d \in \mathbb{N}$, $j, \tau \in \mathbb{N}_0, k \in \{0, \dots, j\}, n \in \mathbb{N}_{\geq k+1}$, $1 \leq p \leq \infty$, and $\mu > 0$;*
- *let $\varrho : \mathbb{R} \to \mathbb{R}$ such that $\left(\Psi^{(j,\tau,N,s)}(\varrho)\right)_{N \in \mathbb{N}, s \geq 1}$ is an exponential (polynomial, exact) (j,τ)-PU;*
- *there exists $x_0 \in \mathbb{R}$ such that ϱ is three times continuously differentiable in a neighborhood of x_0 and $\varrho''(x_0) \neq 0$.*

Then, there exist constants L, C and $\widetilde{\varepsilon}$, and a coding scheme $\mathcal{B} = (B_\ell)_{\ell \in \mathbb{N}}$ depending on d, n, p, k, μ with the following properties:
For every $\varepsilon \in (0, \widetilde{\varepsilon})$ and every $f \in \mathcal{F}_{n,d,p}$, there is a neural network

$$\Phi_{\varepsilon,f} \in \mathcal{NN}^{\mathcal{B}}_{M_\varepsilon, \lceil C \log_2(1/\varepsilon) \rceil, d}$$

with d-dimensional input, one-dimensional output, at most L layers and at most

$$M_\varepsilon = \begin{cases} C\varepsilon^{-d/(n-k-\mu_{(k=2)})}, & \text{if exponential PU,} \\ C\varepsilon^{-d/(n-k)}, & \text{for } k \leq \tau, \text{ if polynomial PU,} \\ C\varepsilon^{-d/(n-k)}, & \text{if exact PU,} \end{cases}$$

nonzero weights, such that

$$\|R_\varrho(\Phi_{\varepsilon,f}) - f\|_{W^{k,p}((0,1)^d)} \leq \varepsilon.$$

Furthermore, there exists a neural network architecture $\mathcal{A}_\varepsilon$ (independent of f) such that $\Phi_{\varepsilon,f}$ has architecture $\mathcal{A}_\varepsilon$ for each $f \in \mathcal{F}_{n,d,p}$.

Proof. We give a short outline of the proof here, the details can be found in Appendix A.6. Let $\Phi_{\varepsilon,f} = ((A_1, b_1), \dots, (A_{L-1}, b_{L-1}), (A_L, b_L))$ be the network from Proposition 2.21 (where the main work has already been done). From the proof of the proposition (see Equation (A.43)) it follows that that $A_L = A_f \widetilde{A}_L$ and $b_L = A_f \widetilde{b}_L$ where the entries of the block diagonal matrix A_f depend on f and the entries of $A_1, b_1, \dots, A_{L-1}, b_{L-1}, \widetilde{A}_L, \widetilde{b}_L$ are independent from f (i.e., they only depend on $\varepsilon, n, d, p, k, \mu$). We denote the collection of nonzero entries of $A_1, b_1, \dots, A_{L-1}, b_{L-1}, \widetilde{A}_L, \widetilde{b}_L$ by W_ε.

- The number of independent weights $|W_\varepsilon|$ is bounded by $C \cdot \varepsilon^{-d/(n-k-\mu_{(k=2)})}$ since the total number of nonzero weights is bounded by this quantity.
- We round the entries of A_f, b_f with a suitable precision ν to the mesh $[-\varepsilon^{-\theta}, \varepsilon^{-\theta}] \cap \varepsilon^\nu \mathbb{Z}$, where we also use the fact that the weights of $\Phi_{\varepsilon,f}$ are bounded in absolute value by $C\varepsilon^{-\theta}$.
- The nonzero entries of A_L in the last layer of $\Phi_{\varepsilon,f}$ are in the set $G_{\text{mult}} := \{x_1 x_2 : x_1 \in W_\varepsilon, x_2 \in [-\varepsilon^{-\theta}, \varepsilon^{-\theta}] \cap \varepsilon^\nu \mathbb{Z}\}$ with cardinality bounded by $\varepsilon^{-\tilde{s}}$ (similar for b_L).

Hence, the weights of the approximating neural networks can be chosen from a set $\widetilde{W}_\varepsilon$ with less than ε^{-s} real numbers, where $s > 0$ only depends on d, n, p, k, μ and not on f.

Consequently, there exists a surjective mapping $B_\varepsilon : \{0,1\}^{\lceil s \log_2(1/\varepsilon)\rceil} \to W_\varepsilon$. The collection of these maps constitutes the coding scheme.

From Remark 2.20, it follows that the approximations can be done with fixed architecture. ■

Name	**Given by**	**Smoothness Boundedness**	**PU-Decay (j,τ)**	**Approximation Rates** $(k \le j)$
(leaky) ReLU, $a \in [0,1)$	$\max\{ax, x\}$	$C(\mathbb{R}) \cap W^{1,\infty}_{\text{loc}}(\mathbb{R})$ Unbounded	exact $(1,1)$	$\varepsilon^{-d/(n-k)} \log^2(1/\varepsilon)$
exponential linear unit (ELU_a), $a > 0$, $a \neq 1$	$\begin{cases} x, & x \ge 0 \\ a(e^x - 1), & x < 0 \end{cases}$	$C(\mathbb{R}) \cap W^{1,\infty}_{\text{loc}}(\mathbb{R})$ Unbounded	exponential $(1,1)$	$\varepsilon^{-d/(n-k)}$
exponential linear unit (ELU_1)	$\begin{cases} x, & x \ge 0 \\ e^x - 1, & x < 0 \end{cases}$	$C^1(\mathbb{R}) \cap W^{2,\infty}_{\text{loc}}(\mathbb{R})$ Unbounded	exponential (2,1)	$\varepsilon^{-d/(n-k)}$ for $k \le 1$, $\varepsilon^{-d/(n-2-\mu)}$ for $k = 2$
softsign	$\dfrac{x}{1+\lvert x\rvert}$	$C^1(\mathbb{R}) \cap W^{2,\infty}(\mathbb{R})$ Bounded	polynomial $(2,0)$	$\varepsilon^{-d/(n-k)}$ for $k = 0$
inverse square root linear unit, $a > 0$	$\begin{cases} x, & x \ge 0 \\ \frac{x}{\sqrt{1+ax^2}}, & x < 0 \end{cases}$	$C^2(\mathbb{R}) \cap W^{3,\infty}_{\text{loc}}(\mathbb{R})$ Unbounded	polynomial $(3,1)$	$\varepsilon^{-d/(n-k)}$ for $k \le 1$
inverse square root unit, $a > 0$	$\dfrac{x}{\sqrt{1+ax^2}}$	Analytic Bounded	polynomial $(j,0)$ $\forall j \in \mathbb{N}_0$	$\varepsilon^{-d/(n-k)}$ for $k = 0$
sigmoid / logistic	$\dfrac{1}{1+e^{-x}}$	Analytic Bounded	exponential $(j,0)$ $\forall j \in \mathbb{N}_0$	$\varepsilon^{-d/n}$ for $k = 0$, $\varepsilon^{-d/(n-k-\mu)}$ for $k \ge 1$
tanh	$\dfrac{e^x - e^{-x}}{e^x + e^{-x}}$	Analytic Bounded	exponential $(j,0)$ $\forall j \in \mathbb{N}_0$	$\varepsilon^{-d/n}$ for $k = 0$, $\varepsilon^{-d/(n-k-\mu)}$ for $k \ge 1$
arctan	$\arctan(x)$	Analytic Bounded	polynomial $(j,0)$ $\forall j \in \mathbb{N}_0$	$\varepsilon^{-d/(n-k)}$ for $k = 0$
softplus	$\ln(1+e^x)$	Analytic Unbounded	exponential $(j,1)$ $\forall j \in \mathbb{N}_0$	$\varepsilon^{-d/n}$ for $k \le 1$, $\varepsilon^{-d/(n-k-\mu)}$ for $k \ge 2$
swish	$\dfrac{x}{1+e^{-x}}$	Analytic Unbounded	exponential $(j,1)$ $\forall j \in \mathbb{N}_0$	$\varepsilon^{-d/n}$ for $k \le 1$, $\varepsilon^{-d/(n-k-\mu)}$ for $k \ge 2$
rectified power unit (RePU), $a \in \mathbb{N}_{\ge 2}$	$\max\{0,x\}^a$	$C^{a-1}(\mathbb{R}) \cap W^{a,\infty}_{\text{loc}}(\mathbb{R})$ Unbounded	exact (a,a)	$\varepsilon^{-d/(n-k)}$

Table 2.1: **Overview of upper bounds.** Commonly-used activation functions, the type of PU they admit and the approximation rates in $W^{k,p}$ in terms of the number of nonzero weights. The rates are provided by Theorem 2.22 and, for the (leaky) ReLU case, in combination with Remark 2.23. The results for the (leaky) ReLU are consistent with those rates derived in [Yar17] for $k = 0$. $\mu > 0$ is arbitrary and, unless specified otherwise, $k \in \{0, \dots, j\}$ and $n \ge k+1$. The depth of the networks is independent of ε except for the (leaky) ReLU with number of layers in $\mathcal{O}(\log(1/\varepsilon))$.

Remark 2.23 (Plug & Play) Some well-known activation functions, e.g., the (leaky) ReLU, do not fulfill all assumptions stated in Proposition 2.21 and Theorem 2.22 (ϱ should be

Table 2.2: **Overview of proofs.** For all four settings, we list whether we specifically prove a result for this setting (col. "Proof") or if a (possibly suboptimal) bound follows from another setting (col. "Transferable"). Transferring results is possible from more restricted settings to less restricted settings for upper bounds and from less restricted settings to more restricted settings for lower bounds. Furthermore, we indicate if the resulting bounds are tight (col. "Tight"). If the tightness depends on the activation function, different options are possible. For unconstrained weights with f-adaptive architectures we neither prove results nor can they be transferred from another setting. For all other settings lower and upper bounds are either proven or can be transferred.

Setting		Upper bounds		Lower bounds		
Weights	Architecture	Proof	Transferable	Proof	Transferable	Tight
Encodable	Fixed	Thm. 2.22		Obs. 2.3		Yes / Almost
	f-adaptive		Thm. 2.22	Thm. 2.6		Yes / Almost
Unconstrained	Fixed		Thm. 2.22	Thm. 2.10		No
	f-adaptive		Thm. 2.22			

three times continuously differentiable in a neighborhood of some $x_0 \in \mathbb{R}$ with $\varrho''(x_0) \neq 0$). However, we note that our proof strategy only requires the approximation of the square function and an identity function. In case of the (leaky) ReLU this can be done with $\mathcal{O}(\log^2(1/\varepsilon))$ weights and $\mathcal{O}(\log(1/\varepsilon))$ layers (see [Yar17, Proposition 2 and 3] for L^∞ norm and Proposition A.20 for $W^{1,\infty}$ norm). Generally speaking: As long as an activation allows for

- the construction of an (approximate) PU along the lines of Definition 2.13,
- an efficient approximation of the square function and the identity function,

our proof strategy can be employed to yield efficient convergence rates. As such, our framework is very general and unifies the previous approach in [Yar17] as well as extends the previously known rates to a wide class of activation functions and rather general smoothness norms. We list approximation rates that can be deduced from our proof framework for some commonly-used activation in Table 2.1.

2.4 Discussion

This section is dedicated to a summary of our findings and sheds some light onto scenarios that were not covered in the previous sections. Furthermore, we discuss the tightness of the derived approximation bounds. Throughout this section, Table 2.2 serves as a reference that connects each proof to the setting it has implications for and Table 2.3 summarizes which function classes, approximation norms, and activation functions are covered by each setting.

Encodable Weights

Encodable weights integrate the realistic assumption of restricted storage capacities for neural network weights on a computer into approximation results. For two normed function spaces $\mathcal{C} \subset \mathcal{D}$, the minimax code length framework allows to directly conclude lower bounds in Observation 2.3 for approximations of functions from the unit ball of $\mathcal{C}$

Table 2.3: **Overview of results.** For all four settings, we list for upper and lower bounds the function classes, approximation norms, and activation functions we derived results for. "b.f.b." is short for "bounded from below".

Setting		Upper bounds		Lower bounds	
Weights	Architecture	Space / Norm	Act. Funct.	Space / Norm	Act. Funct.
Encodable	Fixed	$W^{n,p}/W^{k,p}$	Table 2.1	$\mathcal{C}/\mathcal{D}$ s.th. $L_\varepsilon(\mathcal{C},\mathcal{D})$ b.f.b.	ϱ s.th. $\mathcal{NN}_\varrho^d \subset \mathcal{D}$
	f-adaptive	$W^{n,p}/W^{k,p}$	Table 2.1	$\mathcal{C}/\mathcal{D}$ s.th. $L_\varepsilon(\mathcal{C},\mathcal{D})$ b.f.b.	ϱ s.th. $\mathcal{NN}_\varrho^d \subset \mathcal{D}$
Unconstrained	Fixed	$W^{n,p}/W^{k,p}$	Table 2.1	$W^{n,\infty}/W^{1,\infty}$	ReLU
	f-adaptive	$W^{n,p}/W^{k,p}$	Table 2.1		

in norm $\mathcal{D}$ by neural networks with *fixed* architecture. Based on the efficient encoding of adaptive architectures from Lemma 2.4, we are able to derive the same bounds in Theorem 2.6 for the case of f*-adaptive* architectures. These bounds hold for all activation functions sufficiently smooth such that $\mathcal{NN}_\varrho^d \subset \mathcal{D}$. A concrete consequence of this result for $\mathcal{C}$ and $\mathcal{D}$ being Sobolev spaces is formulated in Corollary 2.9 (ii).

Complementary to the above lower bounds, we derive upper bounds for fixed architectures (for Sobolev spaces) in Theorem 2.22. The case of f-adaptive architectures is not specifically treated but can be inferred from the case of fixed architectures (see Table 2.2).

For both architecture choices, our bounds are tight up to a log factor for $k \leq \tau$, where k is the smoothness of the approximation norm and τ the order of the PU. For $k \geq \tau + 1$, they are (up to a log factor) tight in the case of exact PUs and we get arbitrarily close to the optimal bound (again up to a log factor) in case of exponential PUs. Consequently, there is no gain in allowing the architecture to depend on f in the case of encodable weights. We conclude that our work holistically investigates approximations with encodable weights: We cover different architecture types, general function spaces and norms, and all practically used activation functions.

Unconstrained Weights

We investigate the case of unconstrained weights in our results only to a limited extent but include pointers on how to infer bounds from the case of encodable weights and references to the relevant literature in our discussion.

Unconstrained weights in combination with certain activation functions allow to drastically reduce the complexity in comparison to encodable weights. As an example, we mention again the result in [GI16], where the existence of an activation function is shown, such that a neural network architecture with three parameters only is able to uniformly approximate each function in $\mathcal{C} = C([0,1])$ arbitrary well.

In Section 2.2.2, we focus on unconstrained weights together with the much more restrictive ReLU activation function and fixed architectures. We show in Theorem 2.10 that for approximating Sobolev functions from $W^{n,\infty}$ in $W^{1,\infty}$ norm at least $\varepsilon^{-d/(2(n-1))}$ weights are necessary. As one might expect, these are fewer weights than compared to the more restricted case of encodable weights (here we need at least $\varepsilon^{-d/(n-1)}$ weights).

No specific upper bounds for this case were shown but we remark that the upper bounds from Theorem 2.22 together with Remark 2.23 yield the (potentially suboptimal) upper bound of $\varepsilon^{-d/(n-1)}$ weights (see also Table 2.2) which is not tight. For the case of approximations in the L^∞ norm, Yarotsky [Yar18] and Yarotsky and Zhevnerchuk [YZ20]

find that the gap between the lower bound $\varepsilon^{-d/(2n)}$ and the upper bound of $\varepsilon^{-d/n}$ can be closed by construction approximating neural networks of greater depth. To check whether the same explanation also holds in the case of $W^{1,\infty}$ approximations is an interesting avenue of further research.

The case of unconstrained weights and f-adaptive architectures is not covered in this thesis. We refer to [Yar17] for results in L^∞ norm.

2.5 Limitations and Future Work

We highlight limitations and elaborate on possible extensions of our work in this section. For a more general discussion of shortcomings that are intrinsic to current approximation theory for neural networks, we refer to Chapter 4.

Filling the Gaps

From the above discussion it becomes clear that not all possible cases of the considered setup are covered equally by our results. To fill in the gaps, we believe it particularly interesting to focus on proof frameworks that cover families of activation functions, function spaces and norms (see e.g. our lower and upper bounds for encodable weights) instead of isolated cases. In some instances, it might be only a matter of exploring the limits of the presented proof frameworks even further: For example, can the proof of our upper bounds for encodable weights be generalized to other function spaces and approximation norms? Can the proof of the lower bounds for unconstrained weights be extended to more general activation functions (currently only ReLU) or more general norms (currently only $W^{k,\infty}$ for $k = 0, 1$)?

Curse of Dimension

One of the main reasons for the current interest in deep neural networks is their outstanding performance in solving high-dimensional problems.

Our asymptotic lower bounds for the number of weights depend exponentially on the dimension d of the input space showing that one cannot expect to circumvent the curse of dimension in the setting considered in this thesis. Moreover, the constants in the upper bounds for the number of layers and weights also increase exponentially with increasing d.

Several ways have been proposed so far to tackle this common problem. One idea is to think of the data as residing on or near a manifold $\mathcal{M}$ embedded in $\mathbb{R}^d$ with dimension $d_\mathcal{M} \ll d$ (see [BCV13]). Considering, for example, image classification problems this idea seems to be rather intuitive since most of the elements in the high-dimensional image space (e.g. $\mathbb{R}^{240\times240\times3}$) are not perceived as images. A similar approach is to narrow down the approximated function space by incorporating invariances (see [Mal12]). If, for example, the approximated function f maps images to labels, it may make sense to assume that f is translation and rotation invariant. The additional structure of the function space can then be exploited in the approximation (see [PV18, Section 5]). Assuming more structure on the function spaces could be an interesting extension of our results, that might mitigate the curse of dimension.

The Role of Depth

For approximations of the unit ball in $W^{n,\infty}((0,1)^d)$ (with $n > 2$) in the L^∞ norm by ReLU neural networks, Yarotsky [Yar17] showed that unbounded depth approximations are asymptotically more efficient in the number of weights than approximations with fixed depth L if

$$\frac{d}{n} < \frac{1}{2(L-1)}.$$

As a consequence, to be more efficient than a shallow network, i.e. a network with depth $L = 2$, one needs $n > 2d$ regularity. Even if this result does not completely explain the success of deep networks over shallow ones, since d is typically very large, it would be interesting to obtain similar results for higher-order Sobolev norms.

We note that for activation functions that adhere to the assumptions of Theorem 2.22, no such effect can be shown. For these activation functions the provided bounds are optimal (at least for encodable weights), and the depth of the constructed networks is independent of the approximation accuracy ε.

Balancing Bits and Weights

From Equation (2.3) in Observation 2.3, we can deduce a trade-off in the number of weights M and the memory budget (in bits per weight) b, quantified by $M \cdot b \geq L_\varepsilon(\mathcal{C}, \mathcal{D})$, for fixed architectures. For the same level of approximation power, reducing the number of weights needs to be compensated by an increase in storage capacity and the other way round. The extreme cases are given by (a) a fixed number of bits, resulting in $M \gtrsim L_\varepsilon(\mathcal{C}, \mathcal{D})$ and (b) a fixed number of weights resulting in bit-length $b \gtrsim L_\varepsilon(\mathcal{C}, \mathcal{D})$. In practice, reducing the memory and computational cost is often a necessary requirement for the deployment of deep learning solutions on low-power or low-memory devices [TL19]. Designing pruning (see e.g. [Yeo+21]) and compression strategies is an active field of research [CWZZ18] that could potentially benefit from a deeper mathematical analysis in the above framework. From a mathematical perspective, it would be interesting to investigate a unified way to prove upper bounds that cover the complete spectrum of bit-length and number-of-weights configurations.

Practical Relevance

The derived results are of purely theoretical nature and it is unclear if the predicted approximation rates can be observed in practice. Furthermore, our results provide no information about how to find approximating neural networks (optimization), nor about the sample complexity (generalization). To our knowledge there is only a small number of works that empirically investigate approximation rates (see e.g. [AD21; GPRSK21]). We believe that extensive empirical studies that compare theoretical findings with practically observable outcomes are an important direction of future research and we discuss this aspect in greater depth in Chapter 4.

3

Part B: Near-Exact Recovery for Tomographic Inverse Problems via Deep Learning

In the previous chapter, we analyzed the expressivity of neural networks in classical function spaces from a purely theoretical point of view. This chapter is based on the work [GGMM22] and deals with a fundamental question in scientific machine learning:

> *Can deep-learning-based methods solve noise-free inverse problems to near-perfect accuracy?*

To provide a broader context, it is worth considering an inverse problem in its prototypical, finite-dimensional form:

$$y = Fx + e, \tag{3.1}$$

where $x \in \mathbb{R}^N$ denotes the unknown (image) signal, $F \in \mathbb{R}^{m\times N}$ the forward operator, and $e \in \mathbb{R}^m$ models the noise. The goal is to reconstruct x from the noisy measurements y. In medical applications, for example, magnetic resonance imaging (MRI) can be modeled by choosing F as the Fourier transform, whereas in CT, F can be described by the Radon transform [Eps07]. A typical solution approach to Equation (3.1) is to encode prior knowledge on the signal class in a (convex) penalty functional $\Psi : \mathbb{R}^N \to \mathbb{R}_{\geq 0}$, and then search for a solution that balances a data fidelity and prior term, i.e., solving the minimization problem

$$\arg\min_x \frac{1}{2}\|Fx - y\|_2 + \lambda\Psi(x).$$

Here, $\lambda > 0$ is a weight parameter, that is adapted to the noise level, degree of ill-posedness and trust in the prior. Popular choices for Ψ are Tikhonov regularization with $\Psi = \|\cdot\|_2$ or sparse-regularization based priors, such as TV-minimization $\Psi = \|\nabla\cdot\|_1$. Numerically, the above convex optimization problem can, for example, be solved by proximal splitting methods [CP11], such as the alternating direction method of multipliers (ADMM) [GM75; GM76]. Contrary to hand-crafted priors, data-driven approaches implicitly learn a prior from a training data set $\{(y_i, x_i)_i\}$ [AMÖS19].

The error of any given reconstruction map $\mathcal{R} : \mathbb{R}^m \to \mathbb{R}^N$ can be decomposed as

$$\|x - \mathcal{R}(y)\|_2 \leq \underbrace{\|x - \mathcal{R}(Fx)\|_2}_{\text{(a)}} + \underbrace{\|\mathcal{R}(Fx) - \mathcal{R}(y)\|_2}_{\text{(b)}}. \tag{3.2}$$

The first term (a) is associated with the *accuracy* (or *precision*) of $\mathcal{R}$ and measures how well x can be estimated in the idealistic situation of noiseless measurements. The second term (b) captures the *robustness* of $\mathcal{R}$ against perturbations e of the measurements. Adequate control over both expressions forms the backbone of inverse problem theory and scientific computing in general.

This thesis is primarily concerned with the accuracy term (a), more specifically, the root-mean-square-error (RMSE). We demonstrate that under suitable conditions it can become sufficiently close to zero (on the image data distribution) when $\mathcal{R}$ corresponds to a fully data-driven solver, trained on a dataset of *noise-free* measurements and signals $\{(y_i, x_i)_i\}$. The competitive setting of the AAPM Grand Challenge "Deep Learning for Inverse Problems: Sparse-View Computed Tomography Image Reconstruction", with the goal *"to identify the state-of-the-art in solving the CT inverse problem with data-driven techniques"* [Sid+21], has provided us with the right benchmark to conduct such a case study.

An Approximation Theoretic Point of View

While our inquiry goes far beyond purely approximation theoretical aspects, sufficient expressive power is a necessary ingredient for a positive answer of the above question. In fact, we argue that particularly the noise-free setting ($e = 0$ in Equation (3.1)) provides a suitable test ground for an empirical study of neural network expressivity. In the considered setup, the interaction of the "data manifold", forward operator, and zero noise, allows regularization-based approaches, such as TV minimization, to perfectly recover signals x from incomplete measurements y. In other words, the inverse of the forward operator F on the "data manifold" exists, can be explicitly computed, and provides an ideal target function to approximate by neural networks. Since overfitting turned out to be a minor concern in our experiments[1], we could exclusively focus on constructing an architecture that is *well trainable* and *expressive enough* to compute the reconstruction map.

A Note on Robustness Regarding the robustness term (b) of Equation (3.2), we refer to the recent work by Genzel, Macdonald, and März [GMM22] for an in-depth case study. It was shown that even standard end-to-end networks can be surprisingly robust against adversarial perturbations (i.e., worst-case noise), comparable to a provably stable benchmark methods. Together with Genzel, Macdonald, and März [GMM22], the present work provides further evidence for the reliability of deep-learning-based solutions to inverse problems. Apart from the numerical perspective, robustness of deep learning approaches is an important and actively researched topic. For example, so-called hallucinations (see e.g. [Mar+19]) could lead to misleading diagnostics in the context of medical applications [Muc+21].

Our Conceptual Approach: The Role of the Forward Operator

Conceptually, our approach to solving Equation (3.1) stems from the following (debatable) observation:

> *High reconstruction accuracy is only possible if the forward model is explicitly incorporated into the solution map, e.g., by an iterative promotion of data-consistency.*

The vital role of the forward operator in data-driven solutions to inverse problems is by no means a new insight. It is well in line with a central pillar of scientific machine learning, namely that neural networks can be often enriched (or constrained) by physical

[1] For all tested architectures, a smaller training error always resulted in a smaller test error.

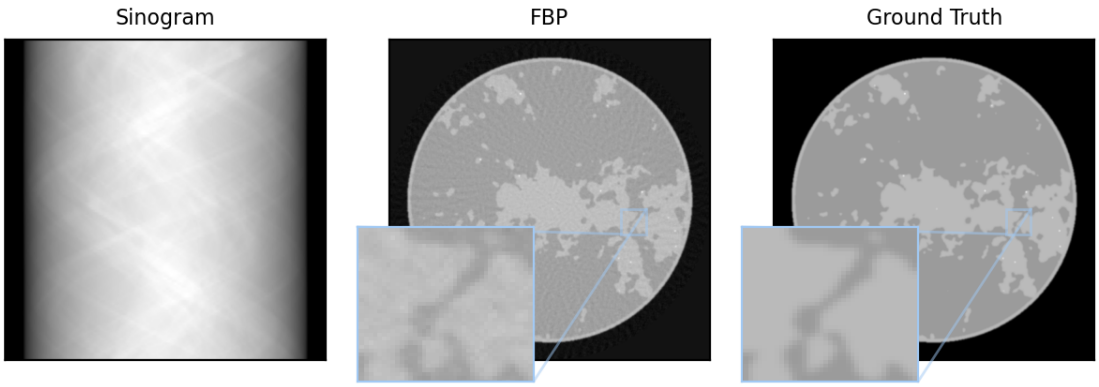


Figure 3.1: **AAPM challenge data.** Example of a 128-view sinogram, FBP reconstruction, and ground-truth phantom image taken from the AAPM challenge training dataset.

modeling [Chm+17; CSMT18; Kei+21; Unk+21a]. Indeed, the seminal works on deep learning techniques for inverse problems are inspired by *unrolling* classical algorithms, e.g., see Gregor and LeCun [GL10], Yang et al. [YSLX16], Adler and Öktem [AÖ18], Aggarwal, Mani, and Jacob [AMJ18], Chen et al. [Che+18], Hammernik et al. [Ham+18], Schlemper et al. [Sch+19], Chun et al. [CHLF20], Gilton, Ongie, and Willett [GOW21a], Hammernik et al. [Ham+21], and Heaton et al. [HWGY21]. At the present time, most state-of-the-art methods rely on *iterative* end-to-end networks and related schemes, e.g., see Knoll et al. [Kno+20], Leuschner et al. [Leu+21], and Muckley et al. [Muc+21] for other recent competition benchmarks.

The winning contribution[2] to the AAPM challenge, is no exception in that respect. We propose a conceptually simple, yet powerful deep learning pipeline, which turns a post-processing UNet [RFB15] into an iterative reconstruction scheme. While many of its individual components have been previously reported in the literature, the overall strategy is novel. Our design differs from more common unrolled networks in several aspects, most notably the following two: (a) we make use of a *pre-trained* UNet as the computational backbone, and (b) data-consistency is inspired by an ℓ^2-gradient step, but employs the *filtered* backprojection (FBP) instead of the regular adjoint. In line with most previous works, our unrolled network only involves very few (five) iteration steps[3]. However, we are the first to show that this is sufficient to match the precision of model-based solvers, which typically need hundreds or thousands of iterations before convergence (and therefore require significantly more computation time).

3.1 AAPM Challenge Setup

Similar to the setting of Sidky et al. [SLBP21], the AAPM challenge data is based on synthetic 2D grayscale images of size $N = 512 \times 512$ simulating real-world mid-plane

[2]The method was designed and submitted to the challenge by M. Genzel, J. Macdonald and M. März. The author of this thesis joined in the aftermath of the challenge and conducted a detailed analysis and further investigations of the method.

[3]See Section 3.3 for more details on this aspect.

breast CT device scans. Four different tissues were modeled: adipose, skin, fibroglandular tissue, and microcalcifications. To obtain smooth transitions at tissue boundaries, Gaussian smoothing was applied. A fanbeam geometry with 128 projections over 360 degrees (i.e. $m = 128 \cdot 360$ in Equation (3.1)) was used to create sinograms and FBPs, see Figure 3.1 for an example. Notably, the exact fanbeam geometry was not revealed to the participants. No noise was added to the data, neither to the phantom images nor to the measurements. In more mathematical terms, for the fanbeam forward operator[4] $F \in \mathbb{R}^{m \times N}$ the relationship between an image $x \in \mathbb{R}^N$ and a sinogram $y \in \mathbb{R}^m$ follows from Equation (3.1) by setting the noise term e to zero, i.e.,

$$y = Fx.$$

The provided training set consisted of 4000 tuples of phantom images, their corresponding 128-view sinograms, and FBP reconstructions. A test set of 100 pairs of sinograms and FBPs (without publicly available ground-truth phantoms) was used for the final challenge evaluation.

Initially, about 50 international teams have participated, out of which 25 have submitted their method to the final evaluation. More details about the challenge setup and results can be found in the official challenge report [SP21].

3.2 Methodology

This section gives an overview of our (three-step) methodology for the AAPM challenge and motivates our design choices.[5]

Step 1: Data-Driven Geometry Identification

In the first step of our reconstruction pipeline, we estimate the unknown forward operator from the provided training data. The continuous version of *tomographic fanbeam measurements* is based on computing line integrals:

$$p(s, \varphi) = \int_{L(s,\varphi)} x_0(x, y)\, \mathrm{d}(x, y),$$

where x_0 is the unknown image and $L(s, \varphi)$ denotes a line in fanbeam coordinates, i.e., φ is the *fan rotation angle* and s encodes the *sensor position*; see Fessler [Fes17] for more details. In an idealized situation, the fanbeam model is specified by the following geometric parameters[6] (see Figure 3.2 for an illustration):

- d_{source} – distance of the X-ray source to the origin,
- d_{detector} – distance of the detector array to the origin,
- n_{detector} – number of detector elements,
- s_{detector} – spacing of detector elements along the array,
- n_{angle} – number of fan rotation angles,

[4] See also Section 3.2 Step 1 for a thorough description of the forward model.

[5] Our code is available at https://github.com/jmaces/aapm-ct-challenge.

[6] We have found that this basic model was enough to accurately describe the AAPM challenge setup. If needed, it would be possible to account for other factors such as non-flat detector arrays, offsets of the axis of rotation from the origin, misalignments of the detector array, etc.

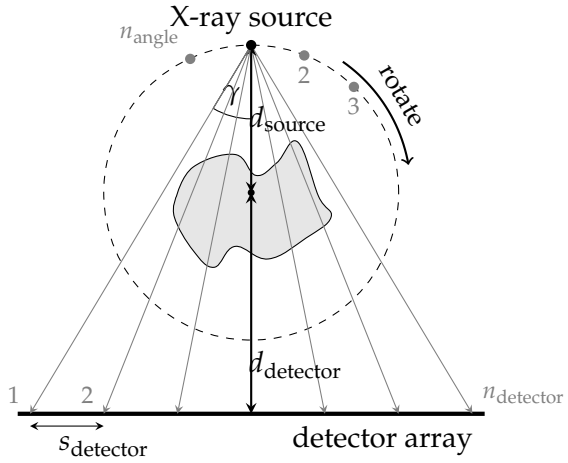


Figure 3.2: **Fanbeam geometry.** Illustration of the parameters determining the geometry of the fanbeam CT model.

- $\varphi \in [0, 2\pi]^{n_{\text{angle}}}$ – discrete list of rotation angles.

Here, it is assumed that integrals are only measured along a finite number of lines, determined by $m := n_{\text{detector}} \cdot n_{\text{angle}}$. In the *sparse-view* challenge setup, the resulting forward operator is *severely ill-posed*, since only the measurements of a few fan rotation angles n_{angle} are acquired. Furthermore, the geometric setup is not disclosed to the challenge participants—it is only known that fanbeam measurements are used.

We have addressed this lack of information by a data-driven estimation strategy that fits the above set of parameters to the given training data. To this end, we first observe that the previous parametrization is redundant, and without of loss of generality, we may assume that $s_{\text{detector}} = 1$ (by rescaling d_{detector} appropriately). Further, if the field-of-view angle γ is known, then the relation

$$d_{\text{detector}} = \frac{n_{\text{detector}} \cdot s_{\text{detector}}}{2 \tan \gamma} - d_{\text{source}} \tag{3.3}$$

can be used to eliminate another parameter. Thus, the fanbeam geometry is effectively determined by the reduced parameter set $(d_{\text{source}}, n_{\text{detector}}, n_{\text{angle}}, \varphi)$. The training data provides pairs of discrete images $x \in \mathbb{R}^{512 \cdot 512 =: N}$ and its simulated fanbeam measurements $y \in \mathbb{R}^{128 \cdot 1024 = m}$, from which the dimensions $n_{\text{angle}} = 128$ and $n_{\text{detector}} = 1024$ can be derived. We determine the field of view as $\gamma = \arcsin(256/d_{\text{source}})$, so that the maximum inscribed circle in the discrete image is exactly contained within each fan of lines, which is a common choice for fanbeam CT. Hence, (3.3) leads to

$$d_{\text{detector}} = 2 \cdot s_{\text{detector}} \cdot \sqrt{d_{\text{source}}^2 - 256^2} - d_{\text{source}} \, .$$

The main difficulty of Step 1 lies in the estimation of the remaining parameters d_{source} and φ). To that end, we have implemented a discrete fanbeam transform from scratch in `PyTorch` (together with its corresponding FBP). A distinctive aspect of our implementation is the use of a vectorized numerical integration that enables the efficient computation of derivatives with respect to the geometric parameters by means of *automatic differentia-*

tion. This feature can be exploited for a data-driven parameter identification, for instance, by a gradient descent. More precisely, we use a ray-driven numerical integration for the forward model and a pixel-driven and sinogram-reweighting-based FBP (with a Hamming filter), see Fessler [Fes17, Sec. 3.9.2]. In addition to the parameters $(d_{\text{source}}, \varphi)$, we also introduce learnable scaling factors s_{fwd} and s_{fbp} for the forward and inverse transform, respectively. They account for ambiguities in choosing the discretization units of distance compared to the actual physical units of distance.

As previously indicated, we estimate the free parameters $\theta_{\texttt{fan}} = (s_{\text{fwd}}, d_{\text{source}}, \varphi) \in \mathbb{R}^{130}$ of the implemented forward operator $F[\theta_{\texttt{fan}}] \in \mathbb{R}^{m \times N}$ in a deep-learning-like fashion: The ability to compute derivatives $\frac{\mathrm{d}F}{\mathrm{d}\theta_{\texttt{fan}}}$ allows us to make use of the $M = 4000$ sinogram-image pairs $\{(y^i, x^i)\}_{i=1}^M$ by solving

$$\min_{\theta_{\texttt{fan}}} \frac{1}{M} \sum_{i=1}^{M} \left\| F[\theta_{\texttt{fan}}](x^i) - y^i \right\|_2^2 \tag{3.4}$$

with a variant of gradient descent (see Remark 3.1 below for details). Finally, we determine s_{fbp} by solving

$$\min_{s_{\text{fbp}}} \frac{1}{M} \sum_{i=1}^{M} \left\| x^i - \texttt{FBP}[\theta_{\texttt{fan}}, s_{\text{fbp}}](y^i) \right\|_2^2 ,$$

while keeping the already identified parameters fixed. We will use the short-hand notation F and $\texttt{FBP}$ for the estimated operators $F[\theta_{\texttt{fan}}]$ and $\texttt{FBP}[\theta_{\texttt{fan}}, s_{\text{fbp}}] \colon \mathbb{R}^m \to \mathbb{R}^N$, respectively.

Remark 3.1 (1) Clearly, the formulation (3.4) is non-convex and therefore it is not clear whether gradient descent enables an accurate estimation of the underlying fanbeam geometry. Indeed, standard gradient descent was found to be very sensitive to the initialization of $\theta_{\texttt{fan}}$ and got stuck in bad local minima. To overcome this issue, we solve (3.4) by a *coordinate descent* instead, which alternatingly optimizes over s_{fwd}, d_{source}, and φ with individual learning rates. This strategy was found to effectively account for large deviations of gradient magnitudes of the different parameters. Indeed, we observed a fast convergence and a reliable identification of $\theta_{\texttt{fan}}$, independently of the initialization.

(2) Subsequent to the estimation of an accurate fanbeam geometry, we still recognized a small systematic error in our forward model. We suspect that it is caused by subtle differences in the numerical integration in comparison to the true forward model of the AAPM challenge. In compensation, we compute the (pixelwise) mean error over the training set, as an additive correction of the model bias.

Step 2: Pre-Training a UNet as Computational Backbone

The centerpiece of our reconstruction scheme is formed by a standard *UNet-architecture* $\text{UNet}[\theta] \colon \mathbb{R}^N \to \mathbb{R}^N$ [RFB15] (see Figure 3.3) which is employed as a residual network to post-process sparse-view FBP images. The learnable parameters θ are trained from the collection of $M = 4000$ sinogram-image pairs $\{(y^i, x^i)\}_{i=1}^M$ provided by the AAPM challenge. This is achieved by standard empirical risk minimization, i.e., by (approximately) solving

$$\min_{\theta} \frac{1}{M} \sum_{i=1}^{M} \left\| x^i - \left[\text{UNet}[\theta] \circ \texttt{FBP}\right](y^i) \right\|_2^2 + \mu \cdot \|\theta\|_2^2 , \tag{3.5}$$

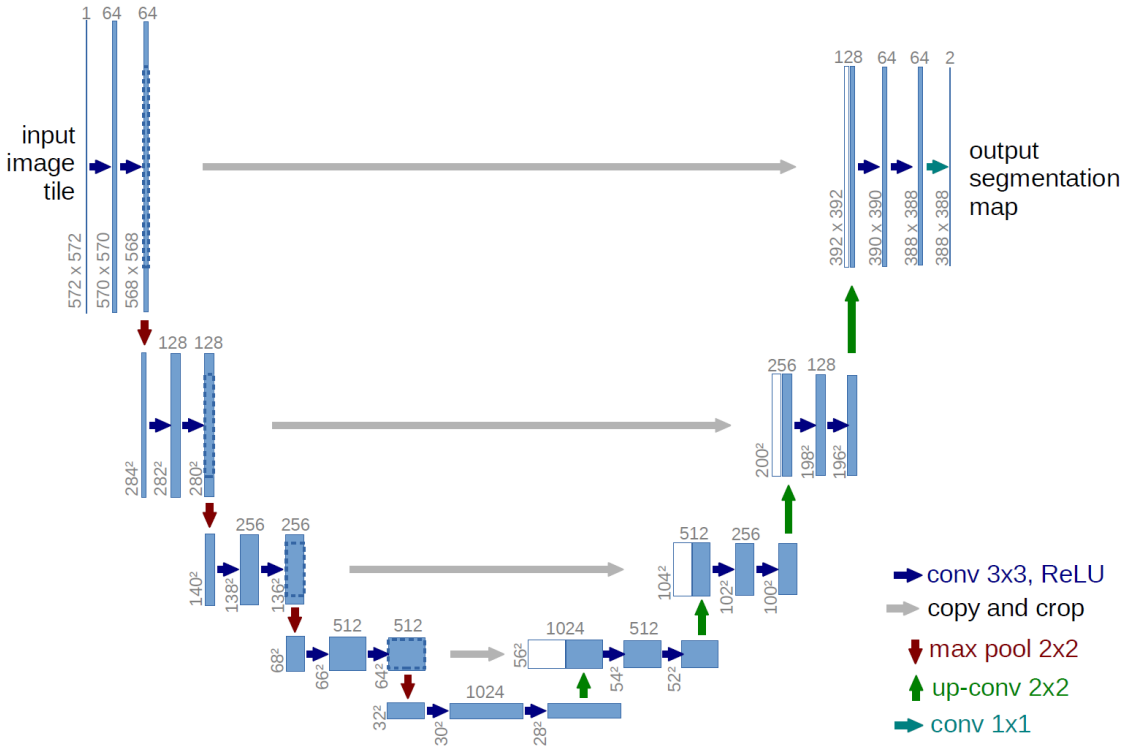


Figure 3.3: **UNet architecture.** The UNet architecture consists of a multi-scale encoder-decoder structure with skip-connections. In the encoder, the input image is successively downsampled and then successively upsampled in the decoder. Levels of matching resolution in the encoder and decoder are connected by skip-connections. Reprinted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature [RFB15], Copyright (2015).

where we choose $\mu = 10^{-3}$, and $\texttt{FBP}\colon \mathbb{R}^m \to \mathbb{R}^N$ is obtained from Step 1. This minimization problem is tackled by 400 epochs of mini-batch stochastic gradient descent and the Adam optimizer [KB14] with initial learning rate 0.0002 and batch size 4.

Remark 3.2 The post-processing strategy of Step 2 was pioneered by Chen et al. [Che+17b] and Kang, Min, and Ye [KMY17] and popularized by Chen et al. [Che+17a] and Jin et al. [JMFU17], among many others. Due to the multi-scale encoder-decoder structure with skip-connections, the UNet-architecture is very efficient in handling image-to-image learning problems. Therefore, solving (3.5) typically works out-of-the-box without requiring sophisticated initialization or optimization strategies (even in seemingly hopeless situations [HA20]). Making use of a more powerful or a more memory-efficient network would be beneficial, e.g., see results for the Tiramisu network in Section 3.3. However, we preferred to keep our workflow as simple as possible and therefore decided to stick to the standard UNet as the main computational building block.

Step 3: Constructing an Iterative Scheme

Our main reconstruction method is called ItNet (short for *iterative network*). It incorporates the estimated forward model F from Step 1 (and the associated inversion $\texttt{FBP}$) via the

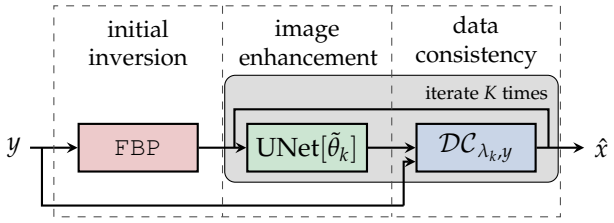


Figure 3.4: **Constructing an iterative scheme.** Schematic reconstruction pipeline of $\mathrm{ItNet}_K[\theta]$ defined in (3.6).

following iterative procedure:

$$\begin{aligned} &\mathrm{ItNet}_K[\theta]\colon \mathbb{R}^m \to \mathbb{R}^N, \\ &y \mapsto \left[\bigcirc_{k=1}^{K} \left(\mathcal{DC}_{\lambda_k,y} \circ \mathrm{UNet}[\tilde{\theta}_k]\right) \circ \mathtt{FBP}\right](y), \end{aligned} \tag{3.6}$$

for the learnable parameters $\theta = \{\tilde{\theta}_k, \lambda_k\}_{k=1}^K$, $K \in \mathbb{N}$ and the k-th *data-consistency* layer

$$\mathcal{DC}_{\lambda_k,y}\colon \mathbb{R}^N \to \mathbb{R}^N,\ x \mapsto x - \lambda_k \cdot \mathtt{FBP}(Fx - y).$$

The ItNet-architecture[7] is illustrated in Figure 3.4. We train it by empirical risk minimization analogously to (3.5) with $\mu = 10^{-4}$. The UNet-parameters $\tilde{\theta}_k$ are initialized by the weights obtained in Step 2. More details on our precise training approach during the challenge submission phase can be found in Appendix B.

We close this section by pointing out several important design choices in our ItNet-architecture:

(i) The centerpiece of ItNet is the *UNet-architecture*. This stands in contrast to earlier generations of unrolled iterative schemes, which rely on basic convolutional blocks instead, e.g., see Yang et al. [YSLX16] and Adler and Öktem [AÖ18]. We have found that it is advantageous to exploit the efficacy of UNet-like image-to-image networks as enhancement blocks. This is in line with recent state-of-the-art models, which also make use of various advanced sub-networks, e.g., see Knoll et al. [Kno+20], Ramzi, Ciuciu, and Starck [RCS20], Sriram et al. [Sri+20], Hammernik et al. [Ham+21], and Muckley et al. [Muc+21].

(ii) The initialization of the UNet-parameters $\tilde{\theta}_k$ with a *pre-trained model* from Step 2 has led to significant performance gains, regarding both training speed and reconstruction accuracy. We refer to Section 3.3 and especially Figure 3.11 for a more details.

(iii) Our *data-consistency* layer is inspired by a gradient step on the loss $x \mapsto \frac{\lambda_k}{2}\|Fx - y\|_2^2$, which would result in the update $x \mapsto x - \lambda_k \cdot F^{\mathbb{T}}(Fx - y)$. We depart from this scheme by replacing the unfiltered backprojection $F^{\mathbb{T}}$ by its filtered counterpart $\mathtt{FBP}$; cf. Ding et al. [Din+20] and Tirer and Giryes [TG21]. This modification leads to significantly improved results (see also [GMM22, Section 4.1]) for two reasons: (a) it counteracts the fact that the unfiltered backprojection is smoothing, and (b) it produces images with pixel values at the right scale. Therefore, we interpret the ItNet as an industry-like iterative CT-algorithm (e.g., see Willemink and Noël [WN19]), rather than a neurally-augmented convex optimization scheme.

[7] We drop the subscript K in ItNet_K whenever it is irrelevant.

Table 3.1: **Average RMSE scores for further evaluation.** "Challenge FBP" corresponds to the FBP reconstructions included in the challenge dataset. The method "UNet ∘ FBP" corresponds to a post-processing UNet as obtained from Step 2 of Section 3.2. For more details on our winning-method "ItNet-post ens." (and its pre-steps "ItNet_4" and "ItNet-post"), see Appendix B.

	Baselines		Our Network Variants				Comparison Networks	
	Challenge FBP	FBP	UNet ∘ FBP	ItNet_4	ItNet-post	ItNet-post ens.	Tiramisu	LPD
RMSE	5.72e-3	3.40e-3	3.50e-4	1.64e-5	1.05e-5	**6.42e-6**	2.24e-4	1.24e-4

3.3 Results and Analysis

This section presents the main findings of our case study. We begin with several challenge-related experiments, followed by a more in-depth analysis of our method.

Winning the AAPM Challenge and Beyond

In terms of quantitative similarity measures, we restrict ourselves to reporting the average RMSE, which was the main evaluation metric for the AAPM challenge [Sid+21; SP21]. With an ensembling of ten ItNet_5 (more precisely a variant thereof referred to as ItNet-post, see Appendix B), we were able to achieve near-exact recovery on the test set, thereby winning the challenge with a margin of about an order of magnitude ahead of the runner-up team. The RMSE scores of all participating teams were spread across more than two orders of magnitude in a range between 6.37e-6 (ours) and 7.90e-4. Remarkably, four out of the five top-performing teams have estimated the forward fanbeam operator and made use of the sinogram data. Two of them computed an approximate TV minimization solution that was further processed by a trained neural network. The resulting solution maps involve much higher computational costs than our ItNet-post, due to a significantly larger number of forward-model evaluations. Note that reaching the first place amounts to a direct comparison with 24 competing methods, see the official AAPM challenge report [SP21] for more details.

Nevertheless, for further analysis, we have benchmarked variants of the ItNet with different in-house baselines and other state-of-the-art methods. More specifically, we consider a post-processing of the FBP by the more powerful *Tiramisu-architecture* [JDVRB17; Bub+19; GMM22] (in comparison to the UNet) as well as the iterative *learned primal-dual* (LPD) algorithm [AÖ18] (modified by replacing the unfiltered backprojection with the FBP). LPD has been recently reported as state-of-the-art in the literature, e.g., see Ramzi, Ciuciu, and Starck [RCS20] and Leuschner et al. [Leu+21]. Table 3.1 shows the average RMSE scores for all methods[8] and Figure 3.5 visualizes reconstructions of an image from the validation set.

After the competition period, the challenge organizer has provided us with 10000 additional test samples to increase the statistical significance of our evaluation. The resulting error distribution is visualized in Figure 3.6. Although there are very few outliers with respect to the performance of our method, even these reconstructions (highlighted in

[8]Note that we report the RMSE on a subset of 125 images from the training set used for validation. Hence, values differ slightly from the actual results on the official test set. In the final challenge evaluation, ItNet-post has achieved an RMSE of 6.37e-6.

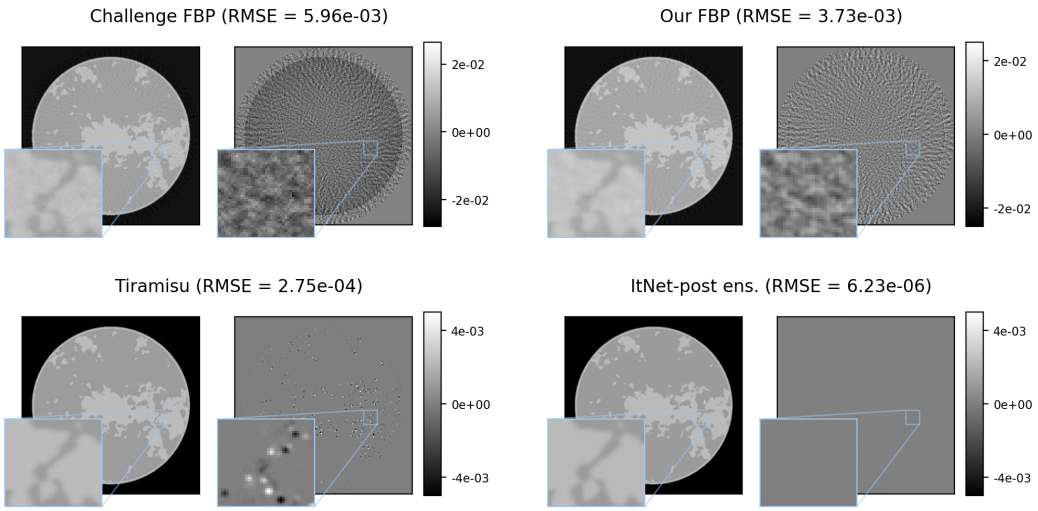


Figure 3.5: **Reconstruction results.** We display individual reconstructions for an image from the validation set. The first row compares the FBP provided by the AAPM challenge with our FBP (= `FBP`, see Step 1 of Section 3.2). The second row compares a post-processing Tiramisu with the (ensemble) ItNet-post. The ground-truth image is omitted because it is visually indistinguishable from the reconstruction of ItNet-post.

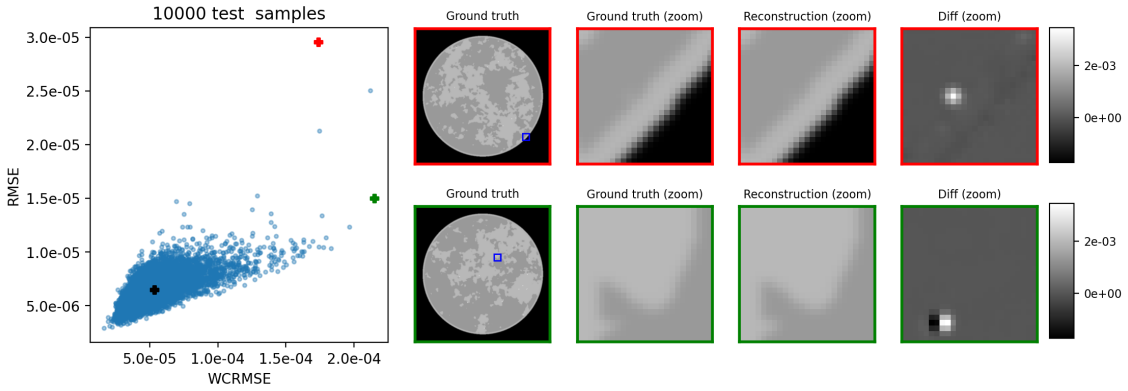


Figure 3.6: **Consistently accurate?** The plot on the left-hand side visualizes the reconstruction errors of the (ensemble) ItNet-post with respect to the RMSE and the WCRMSE (*worst-case* RMSE) over a set of 10000 test images. Note that the WCRMSE was used as secondary challenge metric, computing the highest RMSE value over all 25×25 sub-patches of each image. The error distribution indicates a low variance in the reconstruction performance of our method. On the right-hand side, we show the worst-case 25×25 sub-patches of the images corresponding to the red point (worst RMSE) and green point (worst WCRMSE). The black point represents the average RMSE and WCRMSE.

red and green in Figure 3.6) are visually indistinguishable from the corresponding ground-truth phantoms. This underscores that the ItNet-post solves the CT inverse problem on the given data distribution satisfactorily.

Data-Consistency

A crucial feature of a proper solver for an inverse problem is its consistency with the forward model. In our case, this means that the difference $y - F \cdot \mathrm{ItNet}(y)$ should be as small as possible. We analyze this aspect in Figure 3.7.[9] We observe that the data-consistency error is dominated by the error caused by the estimated forward model F (according to Step 1 of Section 3.2). This indicates that the performance of the ItNet could be further improved if the exact forward operator would be available. To test this hypothesis, we have trained an ItNet on sinogram data that was simulated from the ground-truth phantoms using our own estimated forward operator. As expected, the resulting ItNetSim is more accurate (about factor 2), and according to Figure 3.7 bottom right, implies a much smaller data-consistency error. It is also noteworthy that the loss of data-consistency for the Tiramisu is about a factor 20 larger compared to the ItNet,[10] which highlights a typical downside of simple post-processing approaches.

Forward Operator Needed? ... Yes! But How Often?

The previous considerations have particularly demonstrated that incorporating the forward model is key to highly accurate and data-consistent reconstructions. However, invoking the forward operator often forms the computational bottleneck of a given solution method. It is therefore important to analyze the effective number of forward/adjoint operator calls required for satisfactory precision. We address this by training ItNet_K for different numbers of iterations.[11] In a nutshell, Figure 3.8 confirms that only a few forward operator calls are sufficient for near-exact recovery by the ItNet, which is a notable difference to classical model-based methods like TV-minimization. A closer look reveals that (a) not sharing the UNet-weights consistently outperforms weight sharing[12] by a small margin independent of the number of iterations, and (b) there is a sweetspot at about $K = 5$ after which the performance gain due to increasing K is negligible and only the training time increases.

The Effect of Weight Sharing

General aspects of weight sharing for unrolled algorithms have been extensively discussed in the literature, e.g., see [AMJ18; Ham+21]. Figure 3.9 gives some insights in the context of our specific approach. It clearly indicates that weight sharing also changes the reconstruction dynamic within the neural networks. We observe that (a) earlier iteration steps of the weight-shared ItNets are on average more effective, while the non-weight-shared

[9] Here, we have considered an ensemble of five ItNet_4.

[10] This refers to the ratio $\frac{\mathrm{RMSE}(y, F\cdot\mathrm{Tiramisu}(y)) - \mathrm{RMSE}(y, Fx)}{\mathrm{RMSE}(y, F\cdot\mathrm{ItNet}(y)) - \mathrm{RMSE}(y, Fx)}$.

[11] Due the significant computational effort required to conduct such an experiment, this was done on subsampled 256×256 phantom images and simulated 64-view sinograms.

[12] This means that the UNet-parameters are shared between all iterations, i.e., enforcing $\tilde{\theta}_1 = \cdots = \tilde{\theta}_K$ at the training stage.

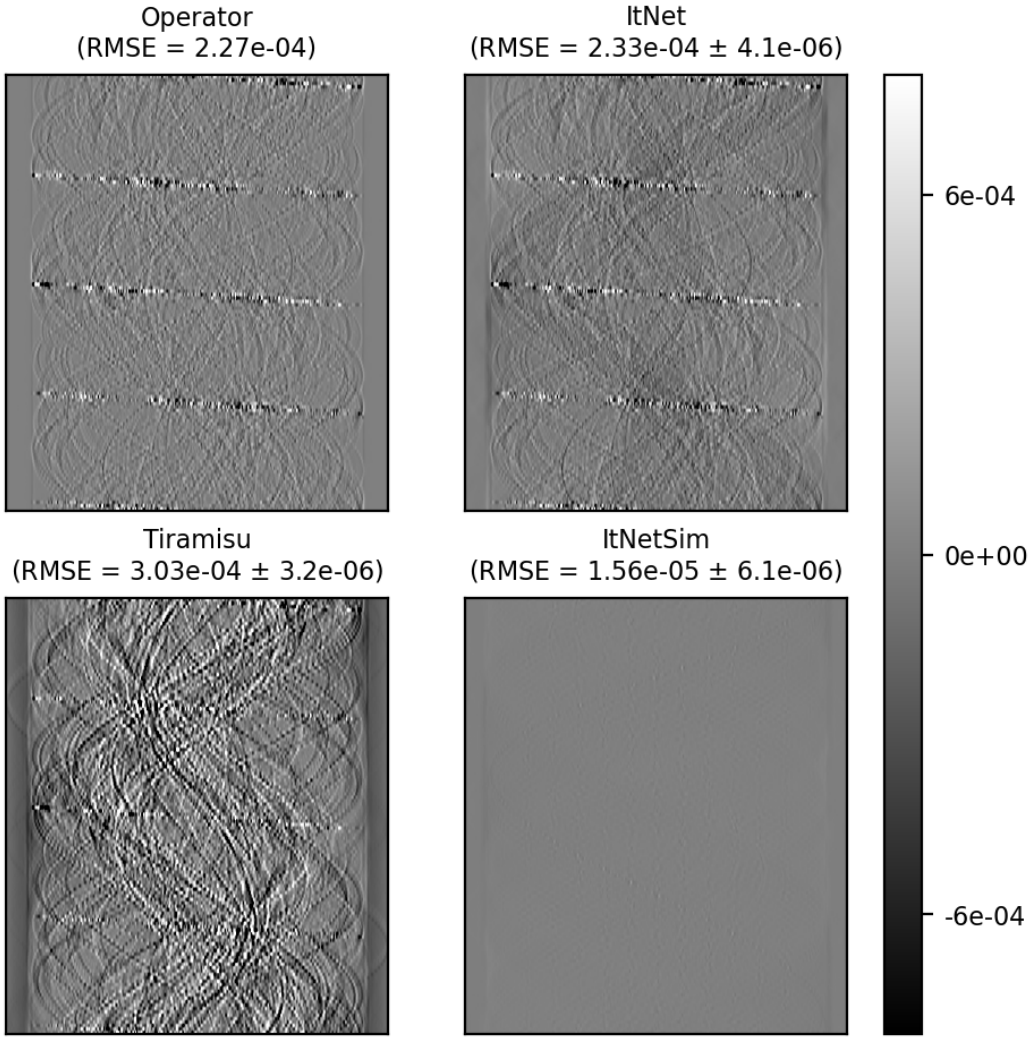


Figure 3.7: **Data consistency.** We analyze the accuracy of our estimated forward operator by displaying the difference $y - Fx$ for a sinogram-image pair (y, x) from the validation set (top left); the corresponding error is the RMSE averaged over all differences. The difference $y - F \cdot \mathrm{ItNet}(y)$ is visually nearly indistinguishable (top right), showing that ItNet inherits the inaccuracies from the forward model. Indeed, ItNetSim exhibits a much smaller data-consistency error due to a perfectly matching forward model (bottom right). In contrast, post-processing via Tiramisu (cf. Table 3.1) reveals a clear lack of data-consistency (bottom left). All images are shown within the same dynamical range.

counterparts draw most of their performance from the later steps, and (b) a larger variance in the early-layer performance without weight sharing hints at a more unstable training. This suggests a trade-off between increasing the model capacity and the difficulty of optimizing the resulting network, while weight sharing forms a simple remedy; cf. Hammernik et al. [Ham+21]. However, we conjecture that an improved training strategy for the non-weight-shared networks might unlock the potential of the early-step UNet-blocks. A canonical possibility would be to include the reconstructions on the intermediate levels

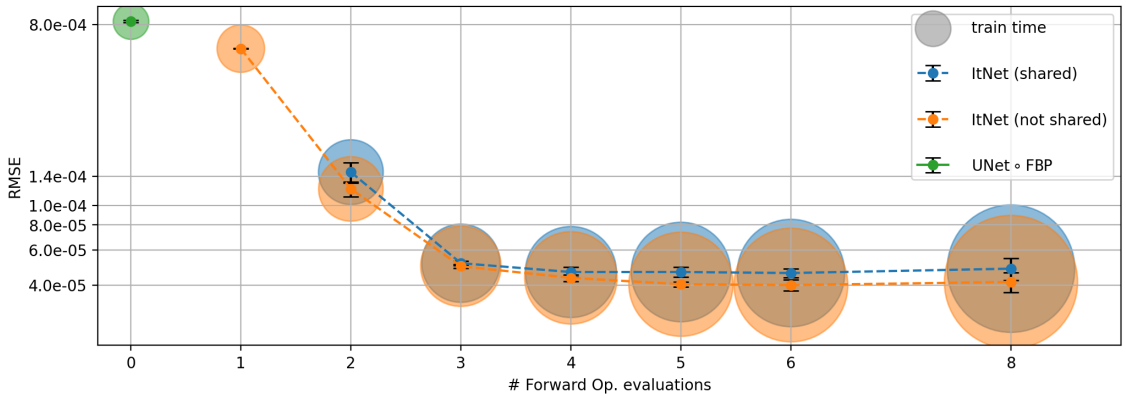


Figure 3.8: **The deeper the better?** Accuracy of ItNet_K for different K with (blue) and without (orange) UNet weight sharing. The radii of the circles are proportional to the training time. The mean RMSE ($\pm$ std. dev.) on a hold-out evaluation data set is reported over 5 different training/validation splits. The original AAPM challenge data was subsampled to the resolution 256×256 for this experiment.

in the loss term. This could lead to an even larger performance gap between the final reconstruction accuracy of ItNets with and without weight sharing.

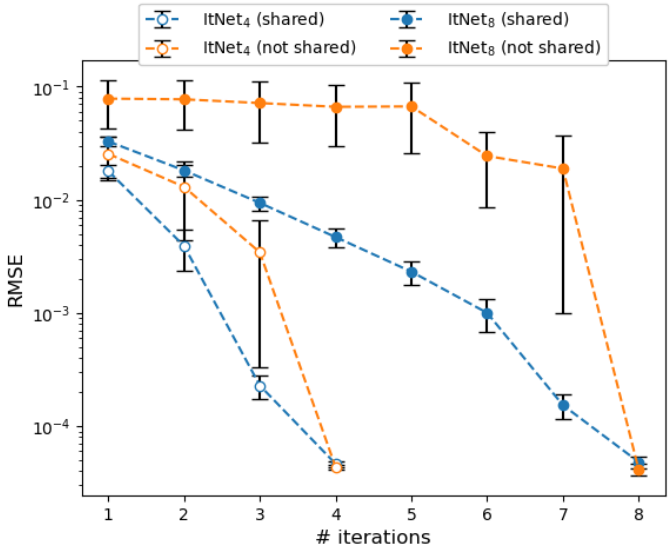


Figure 3.9: **A look inside.** Accuracy of ItNet_4 and ItNet_8 with (blue) and without (orange) UNet weight sharing when using only the first k iteration steps and discarding the rest. The mean RMSE ($\pm$ std. dev.) on a hold-out evaluation data set is reported over 5 different training/validation splits. The original AAPM challenge data was subsampled to the resolution 256×256 for this experiment.

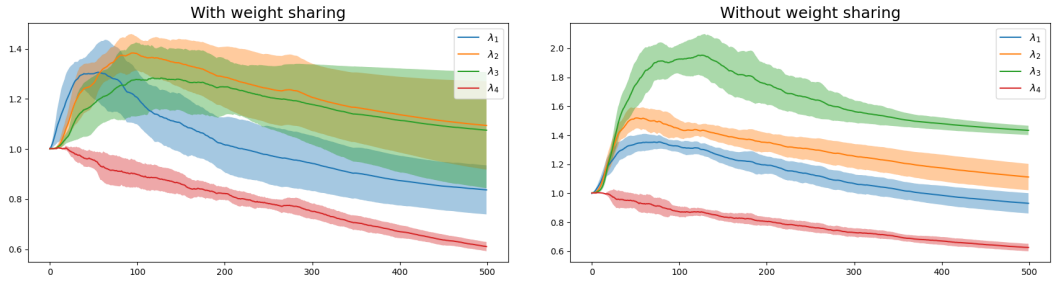


Figure 3.10: **Lambda training trajectory.** Training evolution of λ_k over 500 epochs for ItNet$_4$ with (left) and without (right) UNet weight sharing. All λ_k were initialized by 1. The mean value per epoch ($\pm$ standard deviation) is reported over five different training/validation splits. The original AAPM challenge data was subsampled to the half resolution 256×256 for this experiment.

Consistency Parameter λ_k.

The parameters λ_k control the data consistency update in each iteration. We noticed that in the case of sharing the weights of the UNet-blocks a clear convergence pattern emerges during training which we display in Figure 3.10 for $K = 4$. We have found that $\lambda = [\lambda_1, \lambda_2, \lambda_3, \lambda_4]$ typically converges to values of the form $\{\lambda_1 < \lambda_2 < \lambda_3 \gg \lambda_4\}$ after sufficiently many training epochs of ItNet. Furthermore the final ItNet performance is in our experience stable with respect to different (reasonable) initializations of λ_k.

Pre-Training Matters

When constructing the ItNet according to Step 3 of Section 3.2, we have observed that it is crucial to initialize the UNet-parameters $\tilde{\theta}_k$ by the weights from the post-processing network in Step 2. This does not only increase the speed of convergence of training ItNet, but it also significantly improves the final accuracy, see Figure 3.11 for corresponding loss curves. Thus, our results show that the pre-initialization of the UNet-blocks allows finding better local minima. While the benefits of pre-trained modules are well-known for many standard machine learning tasks, to the best of our knowledge, this has not been reported in the context of inverse problems yet.

Performance on Real-World Image Data

In order to assess the effectiveness of our method on real-world images and noisy (but still simulated) measurements, we have applied it to the *low-dose parallel beam (LoDoPaB)* CT dataset [Leu+21]. This dataset is part of a past challenge and was successfully used to benchmark various deep-learning-based reconstruction schemes. It consists of 42895 two-dimensional human chest CT slices and their low-intensity measurements, see Leuschner et al. [Leu+21] for details on the low-dose setup. For our case study, we have applied Step 2 and Step 3 of the methodology in Section 3.2.[13] The resulting ItNet has reached the

[13] We have trained an ensemble of five ItNet$_3$ with initial learning rate of $8 \cdot 10^{-5}$ and batch size 2. As loss function, a combination of the MSE and SSIM was used.

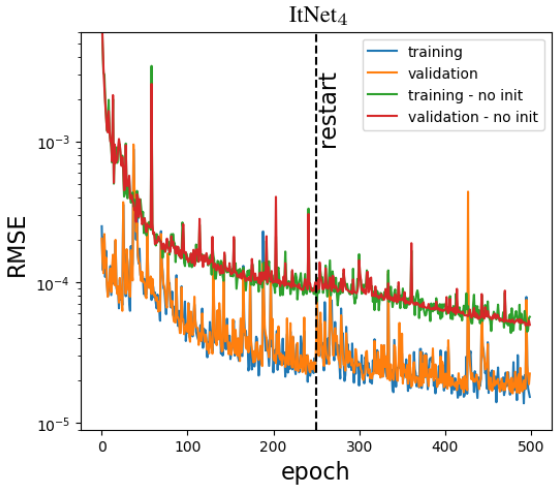


Figure 3.11: **The power of pre-training.** Loss curves when training the ItNet with and without a pre-initialization from Step 2 of Section 3.2. Note that the above loss curve only corresponds to a part of our full training pipeline, see Figure B.1 in Appendix B for the complete picture.

first place in the public leaderboard (still open for submissions),[14] thereby outperforming various other methods, such as the learned primal-dual algorithm [AÖ18] (cf. Table 3.1). A brief analysis and visualization of our reconstructions results can be found in Figure 3.12. Overall, we conclude that our solution strategy can also achieve state-of-the-art performance on natural image data.

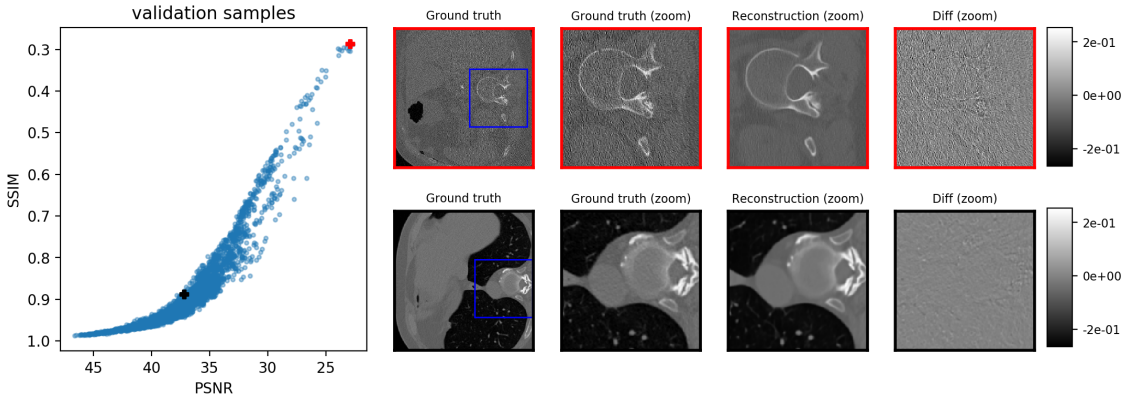


Figure 3.12: **Results for LoDoPaB CT.** The plot on the left-hand side visualizes the reconstruction performance of the ItNet with respect to the challenge metrics (SSIM and PSNR) where each blue point corresponds to one image in the LoDoPaB CT validation set (3522 images). We also show individual reconstructions for the red point (worst SSIM) and black point (closest to average SSIM and PNSR) on the right-hand side. Most notable is that the poor SSIM value of the red point is rather due to a low-quality ground-truth image, than a low-quality reconstruction.

[14] Team-name: `RobustAndStable`; public leaderboard on https://lodopab.grand-challenge.org (accessed on June 7, 2022).

3.4 Discussion, Limitations, and Future Work

We have demonstrated that deep-learning-based solvers can produce near-perfect reconstructions for a noise-free CT inverse problem. While our approach provides evidence of feasibility, several aspects are not studied in this dissertation, some of which are pointed out in the following.

How Accurate Can/Should We Become?

The reconstruction error of ItNet-post reported in Table 3.1 is not exactly zero, yet comparable to the precision of TV minimization, cf. Sidky et al. [SLBP21]. There is no evidence why even more accurate results should not be achievable, for example, by increasing the internal machine precision of `PyTorch` (which is ≈1.19e-7 for float32); but this tweak would certainly also affect model-based algorithms. However, non-perfect recovery is not a severe issue from an applied perspective, since it is typically not required for *practical solutions* to inverse problems. We believe that our submission to the AAPM challenge has obtained satisfactory results in that respect (i.e., reconstructions visually indistinguishable from the ground-truth phantoms; see Figure 3.6). Having said this, the term "near-perfect accuracy" should be used with some care when it comes to real-world scenarios. For instance, realistic CT systems involve analog-to-digital conversion processes and measurement noise, which inevitably leads to reconstruction errors. Therefore, the operating regime of the present thesis is primarily a testing ground for exploring the potential capabilities of learning-based methods.

Fully Data-Driven or Hybrid Method?

Although the ItNet-architecture is inspired by unrolling, it is not clear to us how well its internal mechanisms match those of classical iterative algorithms. Indeed, a distinctive feature of our approach is that only very few (five) iterations can achieve near-perfect recovery. This stands in stark contrast to model-based counterparts, which typically require hundreds or thousands of iterations to converge (resulting in significantly increased computation times). Therefore, we prefer viewing the ItNet as fully data-driven pipeline that is *model-aided* by data-consistency terms, rather than a hybrid method; see also Shlezinger et al. [SWED20]. More generally, we suspect that viewing unrolled networks as neurally-enhanced iterative schemes only partially explains the success of deep learning in inverse problems.

Model Distortions?

Since the purpose of data-driven methods is to adapt to a specific data distribution, the generalization to out-of-distribution features and forward-model distortions cannot be taken for granted [SKM07]. This aspect forms a field of active research, e.g., see Antun et al. [ARPAH20], Darestani, Chaudhari, and Heckel [DCH21], and Gilton, Ongie, and Willett [GOW21b] for initial results, but is not investigated in this thesis.

Generalization to Other (Inverse) Problems?

The scope of our empirical study is limited to the setup of the sparse-view CT inverse problem as prescribed by the AAPM challenge, which has provided an ideal experimental area to test our research hypothesis. Although this has enabled an insightful reliability check, a foundational understanding of learned reconstruction methods is still in its infancy. In particular, it remains speculative to what extent our findings would generalize beyond the sparse-view, mono-energy CT setup. Therefore, similar case studies for different types of inverse problems and real-world data are important steps for future research. Our evaluation on the more realistic LoDoPaB CT dataset can be seen as a first effort in that direction.

4

Conclusion and Outlook

Over the course of this thesis, we have studied approximations by neural networks from a theoretical and an applied perspective. In this last chapter, we summarize our achievements and provide a meta-level discussion. Finally, we outline possible future research directions related to this work.

On Part A In Chapter 2, we theoretically analyzed the complexity of neural networks to approximate functions from a wide variety of classical function spaces under different assumptions on the network architecture and the memory requirement of weights. As a complexity measure, the number of weights and layers was used. Our results allow us to transfer bounds on the well-studied entropy of embeddings directly into lower bounds for neural network approximations with encodable weights and, consequently, cover (fractional) Sobolev spaces, Besov spaces, Hölder spaces, Triebel-Lizorkin, Zygmund spaces and more. An analysis in these function spaces is essential for various mathematical fields such as signal processing, the analysis of PDEs, and inverse problems. For our upper bounds, we focus on Sobolev spaces and presented a framework for the construction of approximate partitions of unity by neural networks with fairly general activation functions. The derived proof framework stands out since it generalizes several previous approaches for more restricted classes of activation functions and we believe that its usage in a Plug & Play fashion (see also Remark 2.23) paves the way for further results in an even broader class of functions spaces. We hope that our findings are general enough to facilitate further insights for various deep learning applications as e.g. in [Pou22], where convergence results of neural networks for the solution of second order elliptic PDEs are derived based on our approximation bounds for Sobolev spaces.

We would like to turn to Allan Pinkus – a pioneer of approximation theory for neural networks – to set the stage for a two-sided interpretation of our theoretical results. In the survey paper [Pin99] in 1999 he gave the following characterization:

> *"Theoretical results ... do not usually have direct applications. In fact they are often far removed from practical considerations. Rather they are meant to tell us what is possible and, sometimes equally importantly, what is not.*
>
> *They are also meant to explain why certain things can or cannot occur, by highlighting their salient characteristics, and this can be very useful."*

In the spirit of the above quote, we summarize two high-level insights from our results in the context of the existing literature:

- Neural networks provide a very flexible framework able to approximate functions from most function spaces of interest where the error can be measured in a wide variety of norms, arbitrarily well.

- Under realistic assumptions (e.g. encodable weights and practically used activation functions) approximation theory with neural networks does not result in approximation theoretic breakthroughs for classical function spaces. The complexity of the approximating neural networks is comparable to classical schemes.

While the first statement already follows from classical universality results and variations thereof [Pin99], the second one, which does not join the usual praise of neural networks, might be less obvious. As evidence in the case of approximations of Sobolev functions, we would like to point to our results from Chapter 2. Here, we show that the optimal complexity of approximating neural networks[1] coincides with the expected power law relation of classical schemes for domain dimension d, function space smoothness n, and smoothness of the approximation norm k given by $\varepsilon^{-d/(n-k)}$. But even works that specifically advocate the efficacy of neural networks for certain tasks are often not able to identify an advantage over established (efficient) algorithms [BGKP19; HGE21; KPRS22]. This is no surprise, given that current proof strategies merely emulate standard results in the language of neural networks (see also next section).

From Part A to Part B As a provocative and debatable conclusion, we would like to state the following hypothesis:

> *Approximation theory for neural networks in classical function spaces has come to a point of diminishing returns. Holistic and / or empirical investigations will lead to further insights.*

Inter alia, we see two possible reasons for this development: *Firstly*, we self-critically observe that the starting point of many works in this field is the question: What results are in reach based on existing mathematical frameworks? This mathematically convenient approach results in a methodology that is detached from the very phenomena it tries to explain [AD21]. Instead, over the course of this thesis, we came to the opinion that theory that is closely tied to experimental observations provides more valuable insights into the inner workings of deep learning. Since problems in deep learning are notoriously hard to model, the price to pay might be less mathematical rigor or empirical studies without theoretical guarantees [GMM22]. *Secondly*, recent works provide evidence that the influence of the network size might be better explained with respect to optimization and generalization properties [BMR21]. This indicates that an isolated study of network size in the context of approximation theory might disregard key factors for the explanation of the success of deep learning.

On a more concrete level, we believe that a systematic empirical investigation of theoretical approximation settings should be the "gold standard" of future research in this area and accompany theoretical studies. This would further the understanding of the practical implications of these results without excluding generalization and optimization effects. In particular, it would be interesting to underpin the following aspects by experiments:

- Theoretical results claim that for each accuracy $\varepsilon > 0$ an approximating neural network Φ_ε can be found (with complexity depending on the specific norm and function space setup). Similar to the driving research question of Part B, we might

[1] The statement holds true for neural networks with encodable weights and depends on properties of the activation function. See Chapter 2 for an in-depth discussion.

ask: *Can neural networks reach the theoretically guaranteed near-perfect accuracy in practical scenarios?* As a starting point for Sobolev spaces, we mention [COJSP17], where a Sobolev loss is used to promote convergence in the appropriate norm.

- In most works on approximation theory for neural networks the determining factors of the approximation rates are identified. These factors depend on the object of study. Approximation rates for parametric PDEs, for example, are determined by the dimension of the solution manifold [KPRS22]. In our results, the driving factors for the network complexity besides the accuracy ε are the domain dimension, function space smoothness, and smoothness of the approximation norm. Generally, increasing the function space smoothness facilitates the approximation task. However, these theoretically predicted dependencies might not always carry over to practice. Indeed, Adcock and Dexter [AD21] find in their experimental study that theoretically better approximation rates cannot always be observed in experiments. Similarly, Grohs and Voigtlaender [GV21] and Berner, Grohs, and Voigtlaender [BGV22] show that approximation rates and sample complexity are not necessarily coupled. This might prevent efficient approximations to be learned from a tractable number of samples. We believe that future research should reveal these gaps, by systematically experimenting with varying factors and quantify the empirically observed impact.

The above discussion also motivates the transition from the theoretical approach in Part A (Chapter 2) to the empirical study of Part B (Chapter 3).

On Part B In Chapter 3, we investigated a deep-learning-based method for the solution of a noise-free prototypical computed tomography setup. We demonstrate that an iterative end-to-end network scheme enables reconstructions close to numerical precision, comparable to compressed sensing strategies. A key feature of our approach is the incorporation of a small number of forward model evaluations in a data consistency step together with a well-trained but comparatively simple architecture. Apart from an in-depth analysis of our methodology, we also demonstrated its state-of-the-art performance on the open-access real-world dataset LoDoPaB CT. In a broader context, we see our work as an incremental but important step towards the bold hypothesis:

> *Deep-learning-based methods can solve noise-free inverse problems (under suitable conditions) to near-perfect accuracy.*

The above statement includes the finding that neural networks are expressive enough to approximate the solution mapping. However, the optimization aspect can not be disregarded for practically meaningful results.

In the following, we mention some possibilities to further substantiate and explore the limits of the above hypothesis:

- A reduction in the number of measurements can save time and costs in e.g. medical applications but at the same time increases the degree of ill-posedness of the reconstruction task. Consequently, it would be of practical importance and scientific interest to explore how much further the number of measurements can be reduced until the performance of neural solvers starts to break down. Particularly interesting would be the comparison to regularization-based approaches, such as TV minimization: *Can deep-learning-based schemes reach near-perfect recovery in a regime where*

regularization-based approaches cannot? There is certainly no general "yes" or "no" answer to this fundamental question. Rather, we expect the answer to depend on the interplay of the "data manifold", the forward operator, and algorithmic choices. We hypothesize that on intrinsically low-dimensional "image manifolds", that can not necessarily be fully described by simple priors, such as gradient sparsity, neural networks might have an advantage due to greater adaptability[2]. Furthermore, we expect that concurrently with increasing the degree of ill-posedness, the probability of artifacts in the form of hallucinations grows. This might explain why hallucinations were found in the fastMRI competition [Muc+21] but not in our study. Future research should focus on extracting practical guidelines from systematic experiments starting on simple and transitioning to more complicated real-world image distributions. This could yield a characterization of the kind of inverse problems where deep learning based methods have an edge on regularization-based approaches.

- The strength of the empirical approach is at the same time its weakness in safety-critical applications. Arguing for the reliability of deep-learning-based approaches by conducting experiments alone, is a Sisyphean task, that needs to be repeated for each change in the setup of interest (e.g. a change in the forward operator, architecture or training methodology). A theoretical framework that derives end-to-end performance guarantees, based on tangible assumptions on the data, would be a milestone for deep learning driven technologies to be accepted in medical applications. Since such a compressed-sensing-style theory seems to be currently out of reach [BMR21; BGKP21; Nak21], we advocate that future research focuses on a more "practical theory" that yields criteria of inverse problems (such as "can be solved by TV minimization") under which certain neural network solvers (such as "iterative with enforced data-consistency") perform the reconstruction task up to a satisfactory accuracy.

- Finally, we outline a way to connect Part B to Part A for the specific case study in this thesis. Indeed, in the introduction, we mentioned that Natterer [Nat80; Nat01] argues that fractional Sobolev spaces provide a suitable framework for the analysis of the CT inverse problem. By lifting our function approximation results to the operator regime, similar to [CC95; Kov+21; LMK22], it might be possible to derive an approximation of the inverse radon transform by neural networks. As a possible starting point, we mention Carroll and Dickinson [CD89], which makes use of the inverse radon transform in their approximation results. Such a result suffers of course from all the above mentioned shortcomings of approximation theoretic result, but combined with empirical evaluations might be a step towards a more comprehensive theory.

We would like to conclude with stating our optimism that future works are going to continue to tackle the gap between theory and practice in deep learning. The only thing left is to express our gratitude to be able to participate in these exciting times of scientific advancements.

[2]We remark that the data from the AAPM challenge was synthetically generated from some low-dimensional parameter space.

A

Proofs for Part A

A.1 Notation and Auxiliary Results

In this subsection, we depict the (mostly standard) notation used throughout this thesis. We set $\mathbb{N} := \{1, 2, \ldots\}$ and $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. For $k \in \mathbb{N}_0$ we define $\mathbb{N}_{\geq k} := \{k, k+1, \ldots\}$. For a set A we denote its cardinality by $|A| \in \mathbb{N} \cup \{\infty\}$ and by $\mathbb{1}_A$ its indicator function of A. If $x \in \mathbb{R}$, then we write $\lceil x \rceil := \min\{k \in \mathbb{Z} : k \geq x\}$ where $\mathbb{Z}$ is the set of integers and $\lfloor x \rfloor := \max\{k \in \mathbb{Z} : k \leq x\}$.

If $d \in \mathbb{N}$ and $\|\cdot\|$ is a norm on $\mathbb{R}^d$, then we denote for $x \in \mathbb{R}^d$ and $r > 0$ by $B_{r,\|\cdot\|}(x)$ the open ball around x in $\mathbb{R}^d$ with radius r, where the distance is measured in $\|\cdot\|$. By $|x|$ we denote the euclidean norm of x and by $\|x\|_\infty$ the maximum norm. We endow $\mathbb{R}^d$ with the standard topology and for $A \subset \mathbb{R}^d$ we denote by $\overline{A}$ the closure of A and by ∂A the boundary of A. For the convex hull of A we write $\operatorname{conv} A$. The diameter of a non-empty set $A \subset \mathbb{R}^d$ is always taken with respect to the euclidean distance, i.e. $\operatorname{diam} A := \operatorname{diam}_{|\cdot|} A := \sup_{x,y \in A} |x - y|$. If $A, B \subset \mathbb{R}^d$, then we write $A \subset\subset B$ if $\overline{A}$ is compact in B.

For $d_1, d_2 \in \mathbb{N}$ and a matrix $A \in \mathbb{R}^{d_1, d_2}$ the number of nonzero entries of A is counted by $\|\cdot\|_0$, i.e.

$$\|A\|_0 := \big|\{(i,j) : A_{i,j} \neq 0\}\big|.$$

If $f : X \to Y$ and $g : Y \to Z$ are two functions, then we write $g \circ f : X \to Z$ for their composition. If additionally $U \subset X$, then $f|_U : U \to Y$ denotes the restriction of f onto U. We use the usual *multiindex* notation, i.e. for $\alpha \in \mathbb{N}_0^d$ we write $|\alpha| := \alpha_1 + \ldots + \alpha_d$ and $\alpha! := \alpha_1! \cdot \ldots \cdot \alpha_d!$. Moreover, if $x \in \mathbb{R}^d$, then we have

$$x^\alpha := \prod_{i=1}^{d} x_i^{\alpha_i}.$$

Let from now on $\Omega \subset \mathbb{R}^d$ be open. For a function $f : \Omega \to \mathbb{R}$, we denote by

$$D^\alpha f := \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_d^{\alpha_d}}.$$

its (weak or classical) derivative of order α. For $n \in \mathbb{N}_0 \cup \{\infty\}$, we denote by $C^n(\Omega)$ the set of n times continuously differentiable functions on Ω. Additionally, if $\overline{\Omega}$ is compact, we set, for $f \in C^n(\Omega)$

$$\|f\|_{C^n(\overline{\Omega})} := \max_{0 \leq |\alpha| \leq n} \sup_{x \in \Omega} |D^\alpha f(x)|.$$

We denote by $L^p(\Omega)$, $1 \leq p \leq \infty$ the standard Lebesgue spaces.

In the following, we will also make use of the following well-known fact stating that the exponential function decays faster than any polynomial.

Proposition A.1 *Let $\alpha, \beta, c, c' > 0$. Then*

$$\lim_{x \to \infty} \frac{c' x^\alpha}{e^{c \cdot x^\beta}} = 0.$$

This implies that for all $\gamma > 0$ there exists some constant $C = C(\alpha, \beta, \gamma) > 0$ such that for all $x > 0$ there holds

$$\frac{c' x^\alpha}{e^{c \cdot x^\beta}} \leq C x^{-\gamma}.$$

A.2 Sobolev Spaces

In this section, we introduce Sobolev spaces (see [Ada75]) which constitute a crucial concept within the theory of PDEs (see e.g. [Eva99; Rou13]).

Definition A.2 Given some domain $\Omega \subset \mathbb{R}^d$, $1 \leq p < \infty$, and $n \in \mathbb{N}$, the Sobolev space $W^{n,p}(\Omega)$ is defined as

$$W^{n,p}(\Omega) := \left\{ f : \Omega \to \mathbb{R} : \|D^\alpha f\|^p_{L^p(\Omega)} < \infty, \text{ for all } \alpha \in \mathbb{N}_0^d \text{ with } |\alpha| \leq n \right\},$$

and is equipped with the norm

$$\|f\|_{W^{n,p}(\Omega)} := \left(\sum_{0 \leq |\alpha| \leq n} \|D^\alpha f\|^p_{L^p(\Omega)} \right)^{1/p}.$$

Additionally, we set

$$W^{n,\infty}(\Omega) := \left\{ f : \Omega \to \mathbb{R} : \|D^\alpha f\|_{L^\infty(\Omega)} < \infty \text{ for all } \alpha \in \mathbb{N}_0^d \text{ with } |\alpha| \leq n \right\},$$

and we equip this space with the norm $\|f\|_{W^{n,\infty}(\Omega)} := \max_{|\alpha| \leq n} \|D^\alpha f\|_{L^\infty(\Omega)}$. Moreover, for $0 \leq k \leq n$, on $W^{n,p}(\Omega)$ we introduce the family of *semi-norms*

$$|f|_{W^{k,p}(\Omega)} := \left(\sum_{|\alpha| = k} \|D^\alpha f\|^p_{L^p(\Omega)} \right)^{1/p}, \qquad |f|_{W^{k,\infty}(\Omega)} := \max_{|\alpha| = k} \|D^\alpha f\|_{L^\infty(\Omega)},$$

respectively. Finally, let

$$W^{n,p}_{\mathrm{loc}}(\Omega) := \{ f : \Omega \to \mathbb{R} : \ f|_{\widetilde{\Omega}} \in W^{n,p}(\widetilde{\Omega}) \ \text{ for all compact } \widetilde{\Omega} \subset \Omega \}.$$

Remark A.3 If Ω is bounded and fulfills a local Lipschitz condition, arguments from [Ada75] show that $W^{2,\infty}(\Omega)$ can be continuously embedded into $C^1(\overline{\Omega})$. This can be seen

as follows: [Ada75, Theorem 4.12] shows that $W^{2,p}(\Omega)$ can be continuously embedded into $C^1(\overline{\Omega})$ for $p > d$. Since also $W^{2,\infty}(\Omega)$ can be continuously embedded into $W^{2,p}(\Omega)$, the claim follows.

Remark A.4 For purely technical reasons we sometimes make use of an extension operator. For this, let $E : W^{n,p}((0,1)^d) \to W^{n,p}(\mathbb{R}^d)$ be the extension operator from [Ste79, Theorem VI.3.1.5] and set $\widetilde{f} := Ef$. Note that for arbitrary $\Omega \subset \mathbb{R}^d$ and $0 \leq k \leq n$ it holds

$$|\widetilde{f}|_{W^{k,p}(\Omega)} \leq \|\widetilde{f}\|_{W^{n,p}(\mathbb{R}^d)} \leq C\|f\|_{W^{n,p}((0,1)^d)}, \tag{A.1}$$

where $C = C(n,p,d)$ is the norm of the extension operator.

A.2.1 Averaged Taylor Polynomial

In this subsection, we develop a polynomial approximation in the spirit of Taylor polynomials but appropriate for Sobolev spaces. A polynomial approximation P_f of a function $f \in \mathcal{F}_{n,d,p}$ is the first step towards an approximation of f realized by a neural network in the proof of Theorem 2.22.

A reference for this entire subsection is [BS08, Chap. 4.1].

Definition A.5 (averaged Taylor polynomial) Let $n \in \mathbb{N}$, $1 \leq p \leq \infty$ and $f \in W^{n-1,p}(\Omega)$, and let $x_0 \in \Omega$, $r > 0$ such that for the ball $B := B_{r,|\cdot|}(x_0)$ it holds that $B \subset\subset \Omega$. The corresponding *Taylor polynomial of order n of f averaged over B* is defined for $x \in \Omega$ as

$$Q^n f(x) := \int_B T_y^n f(x)\phi(y)dy, \tag{A.2}$$

where

$$T_y^n f(x) := \sum_{|\alpha|\leq n-1} \frac{1}{\alpha!} D^\alpha f(y)(x-y)^\alpha \tag{A.3}$$

and ϕ is an arbitrary cut-off function supported in $\overline{B}$, i.e.

$$\phi \in C_c^\infty(\mathbb{R}^d) \quad \text{with} \quad \phi(x) \geq 0 \text{ for all } x \in \mathbb{R}^d, \quad \operatorname{supp}\phi = \overline{B} \quad \text{and} \quad \int_{\mathbb{R}^n} \phi(x)dx = 1.$$

A cut-off function as used in the previous definition always exists. A possible choice is

$$\psi(x) = \begin{cases} e^{-\left(1-(|x-x_0|/r)^2\right)^{-1}}, & \text{if } |x-x_0| < r \\ 0, & \text{else} \end{cases}$$

normalized by $\int_{\mathbb{R}^d} \psi(x)dx$. Next, we derive some properties of the averaged Taylor polynomial.

Remark A.6 From the linearity of the weak derivative we can easily conclude that the averaged Taylor polynomial is linear in f.

Recall that the averaged Taylor polynomial is defined via an integral and some cut-off function (cf. (A.2)) that perform an averaging of a polynomial expression (cf. (A.3)). Additionally, the following lemma shows that the averaged Taylor polynomial of order n is indeed a polynomial of degree less than n in x.

Lemma A.7 *Let $n \in \mathbb{N}$, $1 \leq p \leq \infty$ and $f \in W^{n-1,p}(\Omega)$, and let $x_0 \in \Omega$, $r > 0, R \geq 1$ such that for the ball $B := B_{r,|\cdot|}(x_0)$ it holds that $B \subset\subset \Omega$ and $B \subset B_{R,\|\cdot\|_{\ell^\infty}}(0)$. Then the Taylor polynomial of order n of f averaged over B can be written as*

$$Q^n f(x) = \sum_{|\alpha| \leq n-1} c_\alpha x^\alpha$$

for $x \in \Omega$.

Moreover, there exists a constant $c = c(n,d,R) > 0$ such that the coefficients c_α are bounded with $|c_\alpha| \leq c r^{-d/p} \|f\|_{W^{n-1,p}(\Omega)}$ for all α with $|\alpha| \leq n-1$.

Proof . The first part of the proof of this lemma follows closely the chain of arguments in [BS08, Eq. (4.1.5) - (4.1.8)]. We write for $\alpha \in \mathbb{N}_0^d$

$$(x-y)^\alpha = \prod_{i=1}^d (x_i - y_i)^{\alpha_i} = \sum_{\substack{\gamma,\beta \in \mathbb{N}_0^d, \\ \gamma+\beta=\alpha}} a_{(\gamma,\beta)} x^\gamma y^\beta,$$

where $a_{(\gamma,\beta)} \in \mathbb{R}$ are suitable constants with

$$\left|a_{(\gamma,\beta)}\right| \leq \binom{\gamma+\beta}{\gamma} = \frac{(\gamma+\beta)!}{\gamma!\,\beta!} \tag{A.4}$$

in multi-index notation. Then, combining Equation (A.2) and (A.3) yields

$$\begin{aligned} Q^n f(x) &= \sum_{|\alpha| \leq n-1} \sum_{\gamma+\beta=\alpha} \frac{1}{\alpha!} a_{(\gamma,\beta)} x^\gamma \int_B D^\alpha f(y) y^\beta \phi(y) dy \\ &= \sum_{|\gamma| \leq n-1} x^\gamma \underbrace{\sum_{|\gamma+\beta| \leq n-1} \frac{1}{(\gamma+\beta)!} a_{(\gamma,\beta)} \int_B D^{\gamma+\beta} f(y) y^\beta \phi(y) dy}_{=:c_\gamma}. \end{aligned}$$

For the second part, note that

$$\begin{aligned} \left|\int_B D^{\gamma+\beta} f(y) y^\beta \phi(y) dy\right| &\leq \int_B \left|D^{\gamma+\beta} f(y)\right| \left|y^\beta\right| |\phi(y)| dy \\ &\leq R^{|\beta|} \|f\|_{W^{n-1,p}(B)} \|\phi\|_{L^q(B)}, \end{aligned} \tag{A.6}$$

where we used $B \subset B_{R,\|\cdot\|_{\ell^\infty}}(0)$ and the Hölder's inequality with $1/q = 1 - 1/p$. Next, since $\phi \in L^1(B) \cap L^\infty(B)$ and $\|\phi\|_{L^1} = 1$ we have (see [AJ08, Chap. X.4 Exercise 4])

$$\|\phi\|_{L^q} \leq \|\phi\|_{L^1}^{1/q} \|\phi\|_{L^\infty}^{1-1/q} = \|\phi\|_{L^\infty}^{1/p}.$$

Combining the last estimate with Equation (A.6) yields

$$\begin{aligned} \left|\int_B D^{\gamma+\beta} f(y) y^\beta \phi(y) dy\right| &\leq R^{n-1} \|f\|_{W^{n-1,p}(\Omega)} \|\phi\|_{L^\infty(B)}^{1/p} \\ &\leq c R^{n-1} \|f\|_{W^{n-1,p}(\Omega)} r^{-d/p}, \end{aligned} \tag{A.7}$$

where the second step follows from $\|\phi\|_{L^\infty} \leq cr^{-d}$ for some constant $c = c(d) > 0$ (see [BS08, Section 4.1]). To estimate the absolute value of the coefficients c_γ (defined in Equation A.5), we have

$$\begin{aligned} |c_\gamma| &\leq \sum_{|\gamma+\beta|\leq n-1} \frac{1}{(\gamma+\beta)!} \left|a_{(\gamma,\beta)}\right| \left| \int_B D^{\gamma+\beta} f(y) y^\beta \phi(y) dy \right| \\ &\leq cR^{n-1} \|f\|_{W^{n-1,p}(\Omega)} r^{-d/p} \sum_{|\gamma+\beta|\leq n-1} \frac{1}{\gamma!\beta!} = c' \|f\|_{W^{n-1,p}(\Omega)} r^{-d/p}. \end{aligned}$$

Here, the second step used Equation (A.7) together with Equation (A.4), and $c' = c'(n,d,R) > 0$ is a constant. ∎

The next step is to derive approximation properties of the averaged Taylor polynomial. To this end, recall that for the (standard) Taylor expansion of some function f defined on a domain Ω in x_0 to yield an approximation at some point $x_0 + h$ the whole path $x_0 + th$ for $0 \leq t \leq 1$ has to be contained in Ω (see [Mar74, Thm. 6.8.10]). In case of the averaged Taylor polynomial the expansion point x_0 is replaced by a ball B and we require that the path between each $x_0 \in B$ and each $x \in \Omega$ is contained in Ω. This geometrical condition is made precise in the following definition.

Definition A.8 Let $\Omega, B \subset \mathbb{R}^d$. Then Ω is called *star-shaped with respect to* B if,

$$\overline{\text{conv}}\left(\{x\} \cup B\right) \subset \Omega \quad \text{for all } x \in \Omega.$$

The next definition introduces a geometric notion which becomes important when given a family of subdivisions $\mathcal{T}^h$, $0 < h \leq 1$ of a domain Ω which becomes finer for smaller h. One typically needs to control the nondegeneracy of $(\mathcal{T}^h)_h$ which can be done e.g. with a uniformly bounded chunkiness parameter.

Definition A.9 Let $\Omega \subset \mathbb{R}^d$ be bounded. We define the set

$$\mathcal{R} := \left\{ r > 0 : \begin{array}{l} \text{there exists } x_0 \in \Omega \text{ such that } \Omega \text{ is} \\ \text{star-shaped with respect to } B_{r,|\cdot|}(x_0) \end{array} \right\}.$$

If $\mathcal{R} \neq \varnothing$, then we define

$$r^\star_{\max} := \sup \mathcal{R} \quad \text{and call} \quad \gamma := \frac{\text{diam}(\Omega)}{r^\star_{\max}}$$

the *chunkiness parameter* of Ω.

To emphasize the dependence on the set Ω, we will occasionally write $r^\star_{\max}(\Omega)$ and $\gamma(\Omega)$.

The next lemma shows approximation properties of the averaged Taylor polynomial. A proof can be found in [BS08, Lem. 4.3.8].

Lemma A.10 (Bramble-Hilbert) *Let $\Omega \subset \mathbb{R}^d$ be open and bounded, $x_0 \in \Omega$ and $r > 0$ such that Ω is star-shaped with respect to $B := B_{r,|\cdot|}(x_0)$, and $r > (1/2)r^\star_{max}$. Moreover, let $n \in \mathbb{N}$, $1 \leq p \leq \infty$ and denote by γ the chunkiness parameter of Ω. Then there exists a constant $C = C(n,d,\gamma) > 0$ such that for all $f \in W^{n,p}(\Omega)$*

$$|f - Q^n f|_{W^{k,p}(\Omega)} \leq Ch^{n-k}|f|_{W^{n,p}(\Omega)} \quad \text{for } k = 0,1,\dots,n,$$

where $Q^n f$ denotes the Taylor polynomial of order n of f averaged over B and $h = \operatorname{diam}(\Omega)$.

Finally, the following crucial corollary is a consequence of Lemma A.10 specifically tailored to our needs.

Corollary A.11 (Bramble-Hilbert) *Let $d, n \in \mathbb{N}$ and $1 \leq p \leq \infty$. Furthermore, let $N \in \mathbb{N}$ and set for $m \in \{0,\dots,N\}^d$*

$$\Omega_{m,N} := B_{\frac{1}{N},\|\cdot\|_\infty}\Big(\frac{m}{N}\Big).$$

Then there exists a constant $C = C(n,d) > 0$ such that for all $f \in W^{n,p}(\mathbb{R}^d)$ and $m \in \{0,\dots,N\}^d$ there is a polynomial $p_m(x) = \sum_{|\alpha|\leq n-1} c_\alpha x^\alpha$ such that

$$\big\|f - p_m\big\|_{W^{k,p}(\Omega_{m,N})} \leq C\left(\frac{1}{N}\right)^{n-k} \|f\|_{W^{n,p}(\Omega_{m,N})}, \quad \text{for } k = 0,1,\dots,n$$

and the coefficients c_α are bounded by $|c_\alpha| \leq CN^{d/p}\|f\|_{W^{n,p}(\Omega_{m,N})}$ for all α with $|\alpha| \leq n-1$.

***Proof*.** For each $m \in \{0,\dots,N\}^d$ we set

$$B_{m,N} := B_{\frac{3}{4N},|\cdot|}\Big(\frac{m}{N}\Big),$$

and denote by $p_m = p_{f,m}$ the Taylor polynomial of order n of f averaged over $B_{m,N}$ (cf. Def. A.5). It follows from Proposition A.7 (for $\Omega = \Omega_{m,N}$, $B = B_{m,N}$ and $R = 2$) that we can write $p_m = \sum_{|\alpha|\leq n-1} c_{m,\alpha} x^\alpha$ and that there is a constant $c' = c'(n,d) > 0$ such that

$$|c_{m,\alpha}| \leq c'\big\|f\big\|_{W^{n,p}(\Omega_{m,N})}\left(\frac{3}{4N}\right)^{-d/p} \leq c''\|f\|_{W^{n,p}(\Omega_{m,N})}N^{d/p}$$

for $m \in \{0,\dots,N\}^d$, where $c'' = c''(n,d,p) > 0$ is a suitable constant. To check that the conditions of the Bramble-Hilbert Lemma A.10 are fulfilled, note that $B_{m,N} \subset\subset \Omega_{m,N}$. Furthermore, $B_{m,N}$ is a ball in $\Omega_{m,N}$ such that $\Omega_{m,N}$ is star-shaped with respect to $B_{m,N}$. We have $\operatorname{diam}_{|\cdot|}(\Omega_{m,N}) = (2\sqrt{d})/N$ and $r^\star_{\max}(\Omega_{m,N}) = 1/N$ and, thus,

$$r_{|\cdot|}\left(B_{m,N}\right) = \frac{3}{4N} > \frac{1}{2}\cdot\frac{1}{N} = \frac{1}{2}\cdot r^\star_{\max}(\Omega_{m,N}).$$

Finally, we have for the chunkiness parameter of $\Omega_{m,N}$

$$\gamma(\Omega_{m,N}) = \operatorname{diam}(\Omega_{m,N})\cdot\frac{1}{r^\star_{\max}(\Omega_{m,N})} = \frac{2\sqrt{d}}{N}\cdot N = 2\sqrt{d}. \tag{A.8}$$

Applying the Bramble-Hilbert Lemma A.10 yields for each $m \in \{0,\dots,N\}^d$ the local estimate

$$\|f - p_m\|_{L^p(\Omega_{m,N})} \leq C_1 \left(\frac{2\sqrt{d}}{N}\right)^n |f|_{W^{n,p}(\Omega_{m,N})} \leq C_2 \left(\frac{1}{N}\right)^n \|f\|_{W^{n,p}(\Omega_{m,N})}.$$

Here, $C_1 = C_1(n,d) > 0$ is the constant from Lemma A.10 which only depends on n and d, since the chunkiness parameter of $\Omega_{m,N}$ is a constant depending only on d (see Equation (A.8)) and $C_2 = C_2(n,d) > 0$. In the same way, we get

$$|f - p_m|_{W^{1,p}(\Omega_{m,N})} \leq C_3 \left(\frac{1}{N}\right)^{n-1} \|f\|_{W^{n,p}(\Omega_{m,N})},$$

where $C_3 = C_3(n,d) > 0$ is a suitable constant.

■

A.2.2 Product and Composition Estimates

Now we turn our attention to a version of a product rule tailored to our needs.

Lemma A.12 *Let $k \in \mathbb{N}$, and assume that $f \in W^{k,\infty}(\Omega)$ and $g \in W^{k,p}(\Omega)$ with $1 \leq p \leq \infty$. If $k \geq 3$, additionally assume that $f \in C^k(\Omega)$ or $g \in C^k(\Omega)$. Then $fg \in W^{k,p}(\Omega)$ and there exists a constant $C = C(d,p,k) > 0$ such that*

$$\|fg\|_{W^{k,p}(\Omega)} \leq C \sum_{i=0}^{k} \|f\|_{W^{i,\infty}(\Omega)} \|g\|_{W^{k-i,p}(\Omega)},$$

and, consequently

$$\|fg\|_{W^{k,p}(\Omega)} \leq C \|f\|_{W^{k,\infty}(\Omega)} \|g\|_{W^{k,p}(\Omega)}.$$

***Proof*.** For $k = 0$ the statement is obvious.

The case $k = 1$ is proven similarly to $k = 2$ but easier, so we skip the proof here (see [GKP20, Lemma B.6]).

For $k = 2$ it follows from [GT98, Chap. 7.3] that the usual product rule also holds for the second order derivatives such that we have

$$\begin{aligned}
&|fg|_{W^{2,p}(\Omega)} \\
&\leq C \sum_{i,j=1,\dots,d} \left\| \frac{\partial^2}{\partial x_i \partial x_j} fg \right\|_{L^p(\Omega)} + \left\| \frac{\partial}{\partial x_i} f \frac{\partial}{\partial x_j} g \right\|_{L^p(\Omega)} + \left\| \frac{\partial}{\partial x_j} f \frac{\partial}{\partial x_i} g \right\|_{L^p(\Omega)} + \left\| f \frac{\partial^2}{\partial x_i \partial x_j} g \right\|_{L^p(\Omega)} \\
&\leq C \left(\|f\|_{W^{2,\infty}(\Omega)} \|g\|_{L^p(\Omega)} + \|f\|_{W^{1,\infty}(\Omega)} \|g\|_{W^{1,p}(\Omega)} + \|f\|_{L^\infty(\Omega)} \|g\|_{W^{2,p}(\Omega)} \right).
\end{aligned}$$

Again the overall statement follows easily. The statement for $k \in \mathbb{N}_{\geq 3}$ can directly be concluded from the Leibniz formula (see [Bre12, Lemma 8.18]), which, for a multi-index α

with $|\alpha| \leq k$ yields

$$D^\alpha(fg) = \sum_{|\beta|\leq|\alpha|} \binom{\alpha}{\beta} D^\beta f D^{\alpha-\beta} g.$$

■

The following corollary establishes a chain rule estimate for $W^{k,\infty}$.

Corollary A.13 *Let $d, m \in \mathbb{N}$, $k \in \mathbb{N}_{\geq 2}$ and $\Omega_1 \subset \mathbb{R}^d$, $\Omega_2 \subset \mathbb{R}^m$ both be open, bounded, and convex. Then, there is a constant $C = C(d, m, k) > 0$ with the following properties:*

(i) If $k = 2$ and $f \in W^{2,\infty}(\Omega_1; \mathbb{R}^m) \cap C^1(\Omega_1; \mathbb{R}^m)$ and $g \in W^{2,\infty}(\Omega_2) \cap C^1(\Omega_2)$ such that $\mathrm{Range}(f) \subset \Omega_2$, then for the composition $g \circ f$ it holds that $g \circ f \in W^{2,\infty}(\Omega_1) \cap C^1(\Omega_1)$ and we have

$$|g \circ f|_{W^{1,\infty}(\Omega_1)} \leq C |g|_{W^{1,\infty}(\Omega_2)} |f|_{W^{1,\infty}(\Omega_1;\mathbb{R}^m)},$$

and

$$|g \circ f|_{W^{2,\infty}(\Omega_1)} \leq C \left(|g|_{W^{2,\infty}(\Omega_2)} |f|^2_{W^{1,\infty}(\Omega_1;\mathbb{R}^m)} + |g|_{W^{1,\infty}(\Omega_2)} |f|_{W^{2,\infty}(\Omega_1;\mathbb{R}^m)} \right).$$

(ii) If $k \geq 3$, $f \in C^k(\overline{\Omega}_1; \mathbb{R}^m)$ and $g \in C^k(\overline{\Omega}_2)$ such that $\mathrm{Range}(f) \subset \Omega_2$, then for the composition $g \circ f$ it holds that $g \circ f \in C^k(\Omega_1)$ and

(a) if $|f|_{W^{l,\infty}(\Omega_1;\mathbb{R}^m)} \leq C N^l$ for all $l = 1, \ldots, k$, then

$$|g \circ f|_{W^{k,\infty}(\Omega_1)} \leq C \sum_{l=1}^{k} |g|_{W^{l,\infty}(\Omega_2)} N^k; \tag{A.10}$$

(b) if $\tau \in \mathbb{N}_0$ and $|f|_{W^{l,\infty}(\Omega_1;\mathbb{R}^m)} \leq C N^{l+\mu\max\{0,l-\tau\}}$ for all $l = 1, \ldots, k$, then

$$|g \circ f|_{W^{k,\infty}(\Omega_1)} \leq C \sum_{l=1}^{k} |g|_{W^{l,\infty}(\Omega_2)} N^{k+\mu_{(k=2)}}. \tag{A.11}$$

***Proof*.** (i) can be shown by basic computations using the classical first derivative and [GKP20, Corollary B.5, Lemma B.6]. For (ii), we make use of the multivariate Faa Di Bruno formula (see [CS96, Theorem 2.1]) and get that

$$|g \circ f|_{W^{k,\infty}(\Omega_1)} \leq C \max_{|\nu|=k} \sum_{l=1}^{k} |g|_{W^{l,\infty}(\Omega_2)} \sum_{|\lambda|=l} \sum_{p(\nu,\lambda)} \prod_{j=1}^{k} |f|^{|r_j|}_{W^{|l_j|,\infty}(\Omega_1;\mathbb{R}^m)},$$

where

$$p(\nu,\lambda) := \left\{ \begin{array}{c} (r_1, \ldots, r_k; l_1, \ldots, l_k) : \text{for some } 1 \leq s \leq k, r_i = 0, l_i = 0 \text{ for } 1 \leq i \leq k-s; \\ |r_i| > 0 \text{ for } k-s+1 \leq i \leq k; \text{ and } 0 \leq l_{k-s+1} \leq \ldots \leq l_k \text{ are such that} \\ \textstyle\sum_{i=1}^k r_i = \lambda, \sum_{i=1}^k |r_i| l_i = \nu. \end{array} \right\}.$$

Equation (A.10) now follows from

$$\prod_{j=1}^{k} |f|^{|r_j|}_{W^{|l_j|,\infty}(\Omega_1;\mathbb{R}^m)} \leq C \prod_{j=1}^{k} N^{|l_j||r_j|} = C N^{\sum_{j=1}^{k} |l_j||r_j|} = C N^k.$$

Equation (A.11) for $\tau = 0$ follows from (a) with $N = N^{1+\mu}$. For $\tau \geq 1$, we have

$$\prod_{j=1}^{k} N^{\mu \max\{0,|l_j|-\tau\}|r_j|} = N^{\mu \sum_{j=1}^{k} \max\{0,|l_j|-\tau\}|r_j|}$$

and

$$\sum_{j=1}^{k} \max\{0, |l_j| - \tau\}|r_j| = \sum_{j:|l_j|\geq\tau}^{k} (|l_j| - \tau)|r_j|.$$

If $|l_j| < \tau$ for all $j = 1, \ldots, k$, then $\sum_{j:|l_j|\geq\tau}^{k} (|l_j| - \tau)|r_j| = 0 \leq \mu \max\{0, k - \tau\}$. If there exists some j' with $|l_{j'}| \geq \tau$ and $|r_j| = 0$ for all j with $|l_j| \geq \tau$, then also $\sum_{j:|l_j|\geq\tau}^{k} (|l_j| - \tau)|r_j| = 0 \leq \mu \max\{0, k - \tau\}$. Otherwise, there exists some j' with $|l_{j'}| \geq \tau$ and $|r_{j'}| \geq 1$. We then have

$$\sum_{j:|l_j|\geq\tau}^{k} (|l_j| - \tau)|r_j| \leq \sum_{j:|l_j|\geq 1}^{k} |l_j||r_j| - \tau \sum_{j:|l_j|\geq\tau} |r_j| = k - \tau \sum_{j:|l_j|\geq\tau} |r_j| \leq k - \tau |l_{j'}|\,|r_{j'}| \leq k - \tau$$

from which the statement in combination with (a) follows. ∎

A.3 Proof of Theorem 2.10 (Lower Bounds Based on the VC-Dimension)

For this section, let $\varrho : \mathbb{R} \to \mathbb{R}$, $x \mapsto \max(0, x)$, be the ReLU activation function.

We start by showing an auxiliary result, that is used in the proof of Proposition A.15.

Lemma A.14 *Let $d \in \mathbb{N}$ and Φ be a neural network with d-dimensional input and one-dimensional output. Moreover, let $x \in (0,1)^d$ and $v \in \mathbb{R}^d$. Then, there exists an open set $T = T(x, v) \subset (0,1)^d$ and $\delta = \delta(x, v, T) > 0$ with $x + \lambda\delta v \in \overline{T}$ for $0 \leq \lambda \leq 1$ and $R_\varrho(\Phi)$ is affine-linear on T.*

Proof . We start by defining the set

$$U := \{x \in \mathbb{R}^d : R_\varrho(\Phi) \text{ is affine-linear on a neighborhood of } x\}.$$

Standard results on the number of pieces of ReLU neural networks [MPCB14] yield that U has only finitely many polyhedral, connected components, $(V_i)_{i=1}^k$ for some $k \in \mathbb{N}$, with $U = \bigcup_{i=1}^k V_i$ and $\mathbb{R}^d = \bigcup_{i=1}^k \overline{V_i}$. Note that if follows from the definition of U that V_i is open for $i = 1, \ldots, k$.

Now, set $x_n := x + (1/n)v$. By the pigeonhole principle, there exists $q \in \{1, \ldots, k\}$, such that $\overline{V_q}$ contains infinitely many x_n. It is not hard to see that if a closed polyhedron contains a converging sequence on a line, then it also contains a small section of the line

including the limit point of the sequence. Thus, there exists $\delta > 0$ such that $\{x + \lambda\delta\nu : 0 \leq \lambda \leq 1\} \subset \overline{V_q} \cap (0,1)^d \subset \overline{V_q \cap (0,1)^d}$. Then, setting $T := V_q \cap (0,1)^d$ shows the claim. ∎

The idea of the next proposition is to relate the approximation error ε with the number of weights $M(\mathcal{A}_\varepsilon)$ of an architecture $\mathcal{A}_\varepsilon$ capable of realizing such an approximation. To this end, we construct a set of functions H parameterized by elements of $\mathbb{R}^{M(\mathcal{A}_\varepsilon)+1}$.

If $w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)}$ and $\delta > 0$ is chosen appropriately, then a directional derivative of the function realized by $\mathcal{A}_\varepsilon(w)$ is computed for the evaluation of $h((w,\delta),\cdot) \in H$. By exploiting the approximation capacity of derivatives of functions realized by $\mathcal{A}_\varepsilon(w)$ we can find a lower bound for $\mathrm{VCdim}(H)$ depending on ε (Claim 1).

On the other hand, [AB09, Thm. 8.4] yields an upper bound of the VC-dimension of H in terms of the number of computations and the dimension of the parametrization of H which can be expressed as a function of $M(\mathcal{A}_\varepsilon)$ (Claim 2). Together this gives the desired relation.

Proposition A.15 *Let $d \in \mathbb{N}$ and $n \in \mathbb{N}_{\geq 2}$. Then, there are constants $c = c(n) > 0$ and $C = C(d)$ with the following property:*

Let $N \in \mathbb{N}, 0 < \varepsilon \leq cN^{-(n-1)}$ and $\mathcal{A}_\varepsilon = \mathcal{A}_\varepsilon(d,n,\varepsilon)$ be a neural network architecture with d-dimensional input and one-dimensional output such that for any $f \in \mathcal{F}_{n,d,\infty}$ there is a neural network Φ_ε^f that has architecture $\mathcal{A}_\varepsilon$ and

$$\left\| R_\varrho(\Phi_\varepsilon^f) - f \right\|_{W^{1,\infty}((0,1)^d)} \leq \varepsilon, \tag{A.12}$$

then

$$N^d \leq C \cdot M(\mathcal{A}_\varepsilon)^2.$$

Proof. We prove the proposition by showing that there exists a function $h : \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1} \times [0,1]^d \to \{0,1\}$ with

$$N^d \leq \mathrm{VCdim}\left(\left\{x \mapsto h(w,x) : w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1}\right\}\right) \leq C \cdot M(\mathcal{A}_\varepsilon)^2.$$

To simplify the notation, we set

$$H := \left\{x \mapsto h(w,x) : w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1}\right\}.$$

Step 1 (Construction of h): Let now $0 < \varepsilon < c_1 N^{-(n-1)}/(3\sqrt{d})$ for some constant $c_1 = c_1(n) > 0$ to be chosen later, and $\mathcal{A}_\varepsilon = \mathcal{A}_\varepsilon(d,n,\varepsilon)$ be a neural network architecture as in the claim of the proposition.

For $x \in [0,1]^d$ we define a direction $\nu(x) \in \mathbb{R}^d$ that points from x into $(0,1)^d$ if $x \in [0,1]^d \setminus (0,1)^d$ and equals e_1 if $x \in (0,1)^d$. We set

$$\tilde{\nu}(x) := \begin{cases} e_1, & \text{if } 0 < x_k < 1 \text{ for } k = 1,\dots,d \\ \left[\ \chi_{\{0\}}(x_k) - \chi_{\{1\}}(x_k) : k = 1,\dots,d\ \right], & \text{else,} \end{cases}$$

and define $\nu(x) := \tilde{\nu}(x)/|\tilde{\nu}(x)|$. Moreover, we set $\tilde{x} := x + \nu(x)/(4N)$ for $x \in [0,1]^d$.

To construct h we start be defining a function $g : \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1} \times [0,1]^d \to \mathbb{R}$ and then, to get a binary valued function, define h by thresholding g. In detail, we set

$$g((w,\delta),x) := \begin{cases} \frac{1}{\delta} \cdot \Big(R_\varrho(\mathcal{A}_\varepsilon(w))(\tilde{x} - \delta\nu(x)) - R_\varrho(\mathcal{A}_\varepsilon(w))(\tilde{x}) \Big), & \text{if } \delta \neq 0 \\ 0, & \text{if } \delta = 0 \end{cases}$$

for $w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)}$, $\delta \in \mathbb{R}$ and $x \in [0,1]^d$. Now, we define $h : \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1} \times [0,1]^d \to \{0,1\}$ by

$$h((w,\delta),x) := \begin{cases} 1, & \text{if } g((w,\delta),x) > cN^{-(n-1)}/2 \\ 0, & \text{else} \end{cases}$$

for $w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)}$, $\delta \in \mathbb{R}$ and $x \in [0,1]^d$.

Claim 1 ($N^d \leq \mathbf{VCdim}(H)$): Let $x_1, \dots, x_{N^d} \in [0,1]^d$ such that $|x_m - x_n| \geq 1/N$ for all $m,n = 1,\dots,N^d$ with $m \neq n$ and such that $\tilde{x}_m \in (0,1)^d$ for $m = 1,\dots,N^d$. Moreover, let $y_1,\dots,y_{N^d} \in \{0,1\}$ be arbitrary. We aim to construct $w_y \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)}$ and $\delta_y \in \mathbb{R}$ with

$$h((w_y,\delta_y),x_m) = y_m \quad \text{for} \quad m = 1,\dots,N^d.$$

To this end, we first define a function $f_y \in \mathcal{F}_{n,d,\infty}$ with $f_y(x_m) = y_m \cdot a$ for some constant $a > 0$, and then make use of a neural network Φ^{f_y} that approximates f_y.

Step 2 (Construction of f_y): We start by defining a bump function $\psi \in C^\infty(\mathbb{R}^d)$ by

$$\psi(x) := \begin{cases} e^{-\left(1-4|x|^2\right)^{-1}+1}, & \text{if } |x| < 1/2, \\ 0, & \text{else,} \end{cases}$$

such that $\psi(0) = 1$ and $\operatorname{supp}\psi \subset B_{1/2,|\cdot|}(0)$. For the derivative $D\psi$ of ψ it holds that there exists a function $\phi : (-1/2, 1/2) \to \mathbb{R}_{>0}$ such that[1]

$$(D\psi)(x) = \phi(|x|) \cdot (-x) \quad \text{for all } x \text{ with } |x| < 1/2.$$

Thus, if $x \in \mathbb{R}^d$ with $|x| < 1/2$, then we have for the derivative of ψ in direction $-x/|x|$ at x that $(D_{-x/|x|}\psi)(x) = \phi(|x|)|x| > 0$ only depends on the norm of x.

Next, we define $f_y \in C^\infty(\mathbb{R}^d)$ by

$$f_y(x) := \sum_{m=1}^{N^d} y_m \frac{N^{-n}}{\|\psi\|_{W^{n,\infty}((0,1)^d)}} \psi\left(N(x - x_m)\right)$$

for $x \in \mathbb{R}^d$. We have $|f_y|_{W^{k,\infty}((0,1)^d)} \leq N^{-n}N^k \leq 1$ for $1 \leq k \leq n$ and, consequently, $f_y \in \mathcal{F}_{n,d,\infty}$. Furthermore, for $x \in \mathbb{R}^d$ with $|x| < 1/(2N)$ it holds that

$$(D_{-x/|x|}f_y)(x_m + x) = y_m \frac{N^{-n}}{\|\psi\|_{W^{n,\infty}((0,1)^d)}} \phi(|Nx|)N^2|x|$$

[1] Precisely, $\phi(r) = e^{-\left(1-4r^2\right)^{-1}+1} \cdot \frac{8}{(1-4r^2)^2}$.

and, in particular, if $|x| = 1/(4N)$, then

$$(D_{-x/|x|} f_y)(x_m + x) = y_m \phi(1/4) \cdot \frac{N^{-(n-1)}}{4\|\psi\|_{W^{n,\infty}((0,1)^d)}} = y_m c_1 N^{-(n-1)}. \tag{A.13}$$

Here, we defined the constant $c_1 = c_1(n) > 0$ which was left unspecified in the beginning of the proof by $c_1 := \frac{\phi(1/4)}{4\|\psi\|_{W^{n,\infty}((0,1)^d)}}$.

Step 3 (Existence of w_y and δ_y): We can find a vector $w_y \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)}$ such that for the neural network $\Phi_\varepsilon^{f_y} := \mathcal{A}_\varepsilon(w_y)$ Equation (A.12) holds (with f_y instead of f). In particular, we have

$$|R_\varrho(\Phi_\varepsilon^{f_y}) - f_y|_{W^{1,\infty}((0,1)^d)} \leq \varepsilon. \tag{A.14}$$

Next, for $x \in (0,1)^d$ and $\nu \in \mathbb{R}^d$ we get from Lemma A.14 that there exists an open set $T_{x,\nu} \subset \mathbb{R}^d$ and $\delta = \delta(x, \nu, T_{x,\nu}) > 0$ with $x + \lambda\delta\nu \in \overline{T_{x,\nu}}$ for $0 \leq \lambda \leq 1$ and $R_\varrho(\Phi_\varepsilon^{f_y})$ is affine-linear on $T_{x,\nu}$. We define

$$\delta_y := \min_{m=1,\dots N^d} \delta\left(\tilde{x}_m, -\nu(x_m), T_{\tilde{x}_m, -\nu(x_m)}\right) > 0.$$

Let $m \in \{1, \dots, N^d\}$ and let $F_m : \mathbb{R}^d \to \mathbb{R}$ be an affine-linear function such that $R_\varrho(\Phi_\varepsilon^{f_y})(x) = F_m(x)$ for all $x \in T_{\tilde{x}_m, -\nu(x_m)}$. It then follows from the continuity of $R_\varrho(\Phi_\varepsilon^{f_y})$ that $R_\varrho(\Phi_\varepsilon^{f_y})(x) = F_m(x)$ for all $x \in \overline{T}_{\tilde{x}_m, -\nu(x_m)}$. This, together with the choice of δ_y implies that

$$g((w_y, \delta_y), x_m) = D_{-\nu(x_m)} F_m(\tilde{x}_m) = D_{-\nu(x_m)} F_m.$$

Recall that $T_{\tilde{x}_m, -\nu(x_m)}$ is open and f_y and $R_\varrho(\Phi_\varepsilon^{f_y})$ are continuously differentiable on $T_{\tilde{x}_m, -\nu(x_m)}$. Thus, the weak and strong derivative agree and Equation (A.14) implies

$$|D^i f_y(x) - D^i F_m| = |D^i f_y(x) - D^i R_\varrho(\Phi_\varepsilon^{f_y})(x)| \leq \varepsilon$$

for all $x \in T_{\tilde{x}_m, -\nu(x_m)}$ and $i = 1, \dots, d$. Using the continuity of $D^i f_y$ we get that $|D^i f_y(\tilde{x}_m) - D^i F_m| \leq \varepsilon$ for $i = 1, \dots, d$ and hence

$$|D_\nu f_y(\tilde{x}_m) - D_\nu F_m| \leq \sqrt{d}\varepsilon|\nu| \quad \text{for} \quad \nu \in \mathbb{R}^d. \tag{A.15}$$

An addition of zero yields

$$g((w_y, \delta_y), x_m) = D_{-\nu(x_m)} F_m = D_{-\nu(x_m)} f_y(\tilde{x}_m) + D_{-\nu(x_m)} F_m - D_{-\nu(x_m)} f_y(\tilde{x}_m). \tag{A.16}$$

We get for the case $y_m = 1$ that

$$g((w_y, \delta_y), x_m) \geq D_{-\nu(x_m)} f_y(\tilde{x}_m) - \sqrt{d}\varepsilon \geq c_1 N^{-(n-1)} - c_1 N^{-(n-1)}/3, \tag{A.17}$$

where we used Equation (A.16) together with Equation (A.15) and $|\nu(x_m)| = 1$ for the first step and Equation (A.13) together with the upper bound for ε for the second step.

In a similar way, we get for the case $y_m = 0$ that

$$g((w_y, \delta_y), x_m) \leq \sqrt{d}\varepsilon \leq c_1 N^{-(n-1)}/3, \tag{A.18}$$

where we used that $D_{-\nu(x_m)} f_y(\tilde{x}_m) = 0$.

Finally, combining Equation (A.17) and Equation (A.18) reads as

$$g((w_y, \delta_y), x_m) \begin{cases} > c_1 N^{-(n-1)}/2, & \text{if } y_m = 1, \\ < c_1 N^{-(n-1)}/2, & \text{if } y_m = 0, \end{cases}$$

which proves Claim 1.

Claim 2 ($\mathrm{VCdim}(H) \leq C \cdot M(\mathcal{A}_\varepsilon)^2$): We start by showing that there exists a constant $C' = C'(d)$ such that $h((w, \delta), x)$ can be computed using $C' \cdot M(\mathcal{A}_\varepsilon)$ operations of the following types

- the arithmetic operations $+, -, \times$, and $/$ on real numbers,
- jumps conditioned on $>, \geq, <, \leq, =$, and $\neq$ comparisons of real numbers

for all $w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1}$ and $x \in [0,1]^d$.

There exists an absolute constant $C_1 > 0$ such that at most $C_1 \cdot d$ operations of the specified type are needed to compute $\nu(x)$. Hence, the same holds true for $\tilde{x}$ and $\tilde{x} - \delta\nu(x)$.

Note that the number of neurons that are needed for the computation of $R_\varrho(\mathcal{A}_\varepsilon(w))$ can be bounded by $M(\mathcal{A}_\varepsilon)$. Thus, $R_\varrho(\mathcal{A}_\varepsilon(w))$ can be computed using at most $C_2 \cdot M(\mathcal{A}_\varepsilon)$ operations where $C_2 > 0$ is an absolute constant. Hence, there exists a constant $C_3 = C_3(d)$ such that for the number of operations of the specified type t needed for the computation of $h(x)$ where $x \in [0,1]^d$ it holds that $t \leq C_3 M(\mathcal{A}_\varepsilon)$.

Finally, [AB09, Thm. 8.4] implies

$$\begin{aligned} \mathrm{VCdim}\left(\left\{x \mapsto h(w,x) : w \in \mathbb{R}^{M(\mathcal{A}_\varepsilon)+1}\right\}\right) &\leq 4(M(\mathcal{A}_\varepsilon)+1)(C_3 \cdot M(\mathcal{A}_\varepsilon)+2) \\ &\leq C_4 M(\mathcal{A}_\varepsilon)^2, \end{aligned}$$

where $C_4 = C_4(d) > 0$ is a suitable constant. ∎

The proof of the lower complexity bounds is now a simple consequence of Proposition A.15.

Proof of Theorem 2.10 . The case $k = 0$ corresponds to [Yar17, Thm. 4 a)].

For the case $k = 1$, let $c = c(n, B)$ and $C = C(d)$ be the constants from Proposition A.15 and set

$$N := \left\lfloor \left(\frac{c}{\varepsilon}\right)^{1/(n-1)} \right\rfloor.$$

Then, $N \leq (c/\varepsilon)^{1/(n-1)}$ and, thus, $0 < \varepsilon \leq cN^{-(n-1)}$. Now, Proposition A.15 implies that

$$N^d \leq C M(\mathcal{A}_\varepsilon)^2.$$

We also have $(c/\varepsilon)^{1/(n-1)} \leq 2N$ and hence $c_1 \varepsilon^{-d/(n-1)} \leq N^d$ for a suitable constant

$c_1 = c_1(n) > 0$. Combining this estimate with Equation A.3 yields

$$c_1 \varepsilon^{-d/(n-1)} \leq N^d \leq CM(\mathcal{A}_\varepsilon)^2,$$

which finally results in

$$C' \varepsilon^{-d/2(n-1)} \leq M(\mathcal{A}_\varepsilon)$$

for a constant $C' = C'(d, n) > 0$. ∎

A.4 Neural Network Calculus

In this section, we introduce several operations one can perform with neural networks, namely the *concatenation* and the *parallelization* of neural networks (Section A.4.1). Moreover, Section A.4.2 is devoted to approximations of polynomials. We give the proof of Proposition 2.19 (approximation of monomials by neural networks) and show how to derive approximations of the identity function as well as of approximate multiplications.

A.4.1 Concatenation and Parallelization

We first consider the *concatenation* of two neural networks as given in [PV18].

Definition A.16 Let $\Phi^1 = \left((A_1^1, b_1^1), \ldots, (A_{L_1}^1, b_{L_1}^1)\right)$ and $\Phi^2 = \left((A_1^1, b_1^1), \ldots, (A_{L_1}^1, b_{L_1}^1)\right)$ be two neural networks such that the input dimension of Φ^1 is equal to the output dimension of Φ^2. Then the *concatenation* of Φ^1, Φ^2 is defined as the $L_1 + L_2 - 1$-layer neural network

$$\Phi^1 \bullet \Phi^2 := \left((A_1^2, b_1^2), \ldots, (A_{L_2-1}^2, b_{L_2-1}^2), (A_1^1 A_{L_2}^2, A_1^1 b_{L_2}^2 + b_1^1), (A_2^1, b_2^1), \ldots, (A_{L_1}^1, b_{L_1}^1)\right).$$

It is easy to see that $R_\varrho(\Phi^1 \bullet \Phi^2) = R_\varrho(\Phi^1) \circ R_\varrho(\Phi^2)$.

Now, we introduce the *parallelization* of neural networks with the same number of layers, inspired by the construction in [PV18].

Lemma A.17 *Let $\varrho : \mathbb{R} \to \mathbb{R}$. Additionally, let $\Phi^1, \ldots \Phi^n$ be neural networks with d-dimensional input and $L \in \mathbb{N}$ layers, respectively. Then, there exists a neural network $\mathrm{P}(\Phi^1, \ldots, \Phi^n)$ with d-dimensional input and*

(i) There holds $R_\varrho\left(\mathrm{P}(\Phi^1, \ldots, \Phi^n)\right)(x) = \left(R_\varrho(\Phi^1)(x), \ldots, R_\varrho(\Phi^n)(x)\right)$ for all $x \in \mathbb{R}^d$.;

(ii) L layers;

(iii) $M\left(\mathrm{P}(\Phi^1, \ldots, \Phi^n)\right) = \sum_{i=1}^n M(\Phi^i)$;

(iv) $\left\|\mathrm{P}\left(\Phi^1, \ldots, \Phi^n\right)\right\|_{\max} = \max\left\{\left\|\Phi^1\right\|_{\max}, \ldots, \left\|\Phi^n\right\|_{\max}\right\}$.

Proof. The neural network

$$\mathrm{P}(\Phi^1, \ldots, \Phi^n) := \left((\widetilde{A}_1, \widetilde{b}_1), \ldots, (\widetilde{A}_L, \widetilde{b}_L)\right),$$

with

$$\widetilde{A}_1 := \begin{pmatrix} A_1^1 \\ \vdots \\ A_1^n \end{pmatrix}, \quad \widetilde{b}_1 := \begin{pmatrix} b_1^1 \\ \vdots \\ b_1^n \end{pmatrix} \quad \text{and } \widetilde{A}_\ell := \begin{pmatrix} A_\ell^1 & & & \\ & A_\ell^2 & & \\ & & \ddots & \\ & & & A_L^n \end{pmatrix}, \widetilde{b}_\ell := \begin{pmatrix} b_\ell^1 \\ \vdots \\ b_\ell^n \end{pmatrix},$$

for $1 < \ell \leq L$, fulfills all the desired properties. ∎

A.4.2 Approximate Monomials and Multiplication

We first give the proof of Proposition 2.19:

Proof of Proposition 2.19 . Choose $C_0 > 1$ so that $[x_0 - \frac{nB}{C_0}, x_0 + \frac{nB}{C_0}] \subset U$. Moreover, let $\delta \geq C_0$ be arbitrary. Define the function

$$\varrho_\delta^r : \mathbb{R} \to \mathbb{R}, \quad x \mapsto \frac{\delta^r}{\varrho^{(m)}(x_0)} \sum_{j=0}^{r} (-1)^j \binom{r}{j} \cdot \varrho\left(x_0 - j\frac{x}{\delta}\right).$$

Then $\varrho_\delta^r|_{[-B,B]} \in C^{n+1}([-B,B])$. Using the Taylor expansion and the following identity from [Kat09]

$$\sum_{j=1}^{r} (-1)^j \binom{r}{j} j^k = \begin{cases} 0, & \text{if } 1 \leq k < r, \\ (-1)^r r!, & \text{if } k = r, \end{cases} \tag{A.19}$$

it can easily be shown that $\varrho_\delta^r(x) \approx x^r$ for $\delta > 0$ sufficiently large. In detail, we have by Taylor's Theorem (where ξ_j is between x_0 and $x_0 - j\frac{x}{\delta}$ for $j = 1, \ldots, r$) that

$$\begin{aligned}
&\sum_{j=0}^{r} (-1)^j \binom{r}{j} \cdot \varrho\left(x_0 - j\frac{x}{\delta}\right) \\
&= \varrho(x_0) + \sum_{j=1}^{r} (-1)^j \binom{r}{j} \cdot \left(\sum_{k=0}^{r} \frac{\varrho^{(k)}(x_0)}{k!} \left(\frac{-jx}{\delta}\right)^k + \frac{\varrho^{(r+1)}(\xi_j)}{(r+1)!} \left(\frac{-(r+1)x}{\delta}\right)^{r+1} \right) \\
&= \varrho(x_0) + \sum_{k=0}^{r} \left(\frac{-x}{\delta}\right)^k \frac{\varrho^{(k)}(x_0)}{k!} \sum_{j=1}^{r} (-1)^j \binom{r}{j} j^k + \underbrace{\sum_{j=1}^{r} (-1)^j \binom{r}{j} \frac{\varrho^{(r+1)}(\xi_j)}{(r+1)!} \left(\frac{-(r+1)x}{\delta}\right)^{r+1}}_{=:r_\delta^r(x)} \\
&= \varrho(x_0) \underbrace{\sum_{j=0}^{r} (-1)^j \binom{r}{j}}_{=0} + \sum_{k=1}^{r} \left(\frac{-x}{\delta}\right)^k \frac{\varrho^{(k)}(x_0)}{k!} \underbrace{\sum_{j=1}^{r} (-1)^j \binom{r}{j} j^k}_{\text{use Eq. (A.19)}} + r_\delta^r(x) \\
&= \left(\frac{x}{\delta}\right)^r \varrho^{(r)}(x_0) + r_\delta^r(x).
\end{aligned}$$

Hence, for every $k = 0, \ldots, n$ and every $x \in [-B, B]$, we have

$$\left| (\varrho_\delta^r)^{(k)}(x) - (x^r)^{(k)} \right| = \left| \frac{\delta^r}{\varrho^{(r)}(x_0)} (r_\delta^r)^{(k)}(x) \right|$$

$$\leq \underbrace{\sum_{j=1}^{r} \binom{r}{j} \cdot \left| \frac{\varrho^{(r+1)}(\xi_j)}{(r+1)!} \right|}_{\leq 2^n \|\varrho\|_{C^{n+1}(U)}} \cdot \underbrace{\left| \frac{\delta^r}{\varrho^{(r)}(x_0)} \left(\frac{-(r+1)}{\delta} \right)^{r+1} \right|}_{\leq \frac{(n+1)^{n+1}}{\delta \min_{i=0,\dots,n} |\varrho^{(i)}(x_0)|}} \cdot \underbrace{\left| (x^{r+1})^{(k)} \right|}_{\leq n! \max\{B,1\}^{n+1}}$$

$$\leq 2^n \cdot (n+1)^{n+1} n! \cdot \frac{\|\varrho\|_{C^{n+1}(U)}}{\min_{i=0,\dots,n} |\varrho^{(i)}(x_0)|} \max\{B,1\}^{n+1} \cdot \frac{1}{\delta}$$

$$=: \frac{C'(B,n,\varrho)}{\delta}.$$

This implies, that there exists some $C \geq \max\{C_0, C'(B,n,\varrho)\}$ such that for every $\varepsilon \in (0,1)$ and the neural network $\Phi_\varepsilon^r := ((A_1,b_1),(A_2,b_2))$ with

$$\begin{aligned}
A_1 &:= \left(0, -\frac{\varepsilon}{C}, \dots, -\frac{r\varepsilon}{C}\right)^T \in \mathbb{R}^{r+1,1}, \\
b_1 &:= (x_0, \dots, x_0)^T \in \mathbb{R}^{r+1}, \\
A_2 &:= \frac{C^r}{\varepsilon^r \varrho^{(r)}(x_0)} \left((-1)^0 \binom{r}{0}, (-1)^1 \binom{r}{1}, \dots, (-1)^r \binom{r}{r} \right) \in \mathbb{R}^{1,r+1}, \\
b_2 &:= 0 \in \mathbb{R},
\end{aligned}$$

fulfills

$$\left\| R_\varrho(\Phi_\varepsilon^r) - x^r \right\|_{C^n([-B,B])} \leq \varepsilon.$$

Moreover, $L(\Phi_\varepsilon^r) = 2$ and $M(\Phi_\varepsilon^r) \leq 3(r+1)$.

Additionally, for every $k = 0, \dots, r$ and for every $x \in [-B,B]$ we have

$$\begin{aligned}
\left| \left(R_\varrho(\Phi_\varepsilon^r)\right)^{(k)}(x) \right| &\leq \left\| \left(R_\varrho(\Phi_\varepsilon^r)\right)^{(k)} - (x^r)^{(k)} \right\|_{C^n([-B,B])} + \left| (x^r)^{(k)} \right| \\
&\leq \varepsilon + \frac{n!}{(n-k)!} |\max\{1,B\}|^{r-k}.
\end{aligned}$$

Finally, for all $k = r+1, \dots, n$ we have that

$$\left| \left(R_\varrho(\Phi_\varepsilon^r)\right)^{(k)}(x) \right| \leq \left\| \left(R_\varrho(\Phi_\varepsilon^r)\right)^{(k)} - (x^r)^{(k)} \right\|_{C^n([-B,B])} + \left| (x^r)^{(k)} \right| \leq \varepsilon + 0 = \varepsilon.$$

This completes the proof. ∎

Based on Proposition 2.19, we are now in a position to introduce neural networks that approximate the map which multiplies two real inputs.

Corollary A.18 *Let $\varrho \in W_{\text{loc}}^{j,\infty}(\mathbb{R})$ for some $j \in \mathbb{N}_0$ and $x_0 \in \mathbb{R}$ such that ϱ is three times continuously differentiable in a neighborhood of some $x_0 \in \mathbb{R}$ and $\varrho''(x_0) \neq 0$. Let $B > 0$, then there exists a constant $C = C(B,\varrho) > 0$ such that for every $\varepsilon \in (0, 1/2)$, there is a neural network $\widetilde{\times}$ with two-dimensional input and one-dimensional output that satisfies the following properties:*

(i) $\|R_\varrho(\widetilde{\times})(x,y) - xy\|_{W^{j,\infty}((-B,B)^2;dxdy)} \leq \varepsilon$;
(ii) $\|R_\varrho(\widetilde{\times}_\varepsilon)\|_{W^{j,\infty}((-B,B)^2)} \leq C$;
(iii) $L(\widetilde{\times}) = 2$ *and* $M(\widetilde{\times}) \leq C$;
(iv) $\|\widetilde{\times}\|_{\max} \leq C\varepsilon^{-2}$.

Proof. Let C be the constant from Corollary A.13 and set $\widetilde{\varepsilon} := \varepsilon/2C$. Proposition 2.19 yields that there exists a neural network $\Phi^2_{\widetilde{\varepsilon}}$ with 2 layers and at most 9 nonzero weights such that for all $k \in \{0,\dots,j\}$ we have

$$\left|R_\varrho(\Phi^2_{\widetilde{\varepsilon}}) - x^2\right|_{W^{k,\infty}([-2B,2B];dx)} \leq \widetilde{\varepsilon}.$$

As in [Yar17], we make use of the polarization identity

$$xy = \frac{1}{4}\left((x+y)^2 - (x-y)^2\right) \quad \text{for } x,y \in \mathbb{R}.$$

In detail, we define the neural network

$$\widetilde{\times}_\varepsilon := \left(\left(\frac{1}{4}, \frac{-1}{4}\right), 0\right) \bullet \mathrm{P}\left(\Phi^2_{\widetilde{\varepsilon}}, \Phi^2_{\widetilde{\varepsilon}}\right) \bullet \left(\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, 0\right),$$

which fulfills for all $(x,y) \in \mathbb{R}^2$ that

$$R_\varrho(\widetilde{\times}_\varepsilon)(x,y) = \frac{1}{4}\left(R_\varrho\left(\Phi^2_{\widetilde{\varepsilon}}\right)(x+y) - R_\varrho\left(\Phi^2_{\widetilde{\varepsilon}}\right)(x-y)\right).$$

Now, setting $f : [-2B,2B] \to \mathbb{R}, x \mapsto x^2$ as well as

$$u : [-B,B]^2 \to [-2B,2B],\ (x,y) \mapsto x+y \ \text{ and } \ v : [-B,B]^2 \to [-2B,2B],\ (x,y) \mapsto x-y,$$

we see that for all $(x,y) \in [-B,B]^2$ there holds $xy = 1/4\,(f \circ u(x,y) - f \circ v(x,y))$. We estimate

$$\begin{aligned}
&\left\|R_\varrho(\widetilde{\times}_\varepsilon)(x,y) - xy\right\|_{W^{k,\infty}([-B,B]^2;dxdy)} \\
&= \frac{1}{4}\left\|R_\varrho\left(\Phi^2_{\widetilde{\varepsilon}}\right) \circ u - R_\varrho\left(\Phi^2_{\widetilde{\varepsilon}}\right) \circ v - (f \circ u - f \circ v)\right\|_{W^{k,\infty}([-B,B]^2)} \\
&\leq \frac{1}{4}\left\|R_\varrho\left(\Phi^2_{\widetilde{\varepsilon}}\right) \circ u - f \circ u\right\|_{W^{k,\infty}([-B,B]^2)} + \frac{1}{4}\left\|R_\varrho\left(\Phi^2_{\widetilde{\varepsilon}}\right) \circ v - f \circ v\right\|_{W^{k,\infty}([-B,B]^2)},
\end{aligned}$$

and directly see for $k = 0$ that

$$\left|R_\varrho(\widetilde{\times}_\varepsilon)(x,y) - xy\right|_{W^{0,\infty}([-B,B]^2;dxdy)} \leq \frac{2}{4}\|R_\varrho\left(\Phi^2_{\widetilde{\varepsilon}}\right) - x^2\|_{L^\infty([-B,B]^2;dx)} \leq \frac{1}{2}\widetilde{\varepsilon} \leq \varepsilon.$$

Now, we proceed with the case $k \in \{1,\dots,j\}$. We first note that

$$\begin{aligned}
|u|_{W^{0,\infty}([-B,B]^2)} &= |v|_{W^{0,\infty}([-B,B]^2)} = 2B, \\
|u|_{W^{1,\infty}([-B,B]^2)} &= |v|_{W^{1,\infty}([-B,B]^2)} = 1, \\
|u|_{W^{k,\infty}([-B,B]^2)} &= |v|_{W^{k,\infty}([-B,B]^2)} = 0, \text{ for all } k \geq 2.
\end{aligned}$$

The composition rule from Corollary A.13 then yields that

$$\begin{aligned}\left|R_\varrho(\widetilde{\times}_\varepsilon)(x,y)-xy\right|_{W^{k,\infty}([-B,B]^2;dxdy)} &\leq 2C\sum_{i=1}^{k}\left|R_\varrho\left(\Phi_{\widetilde{\varepsilon}}^2\right)-x^2\right|_{W^{i,\infty}([-2B,2B];dx)}|u|^i_{W^{1,\infty}([-B,B]^2)}\\ &\leq 2C\widetilde{\varepsilon}=\varepsilon.\end{aligned}$$

and, thus, claim (i) is shown. Finally, we have for $k\in\{0,\dots,j\}$

$$\left|R_\varrho(\widetilde{\times}_\varepsilon)\right|_{W^{k,\infty}([-B,B]^2)}\leq\left|R_\varrho(\widetilde{\times}_\varepsilon)-xy\right|_{W^{k,\infty}([-B,B]^2;dxdy)}+|xy|_{W^{k,\infty}([-B,B]^2;dxdy)}\leq C_1,$$

for a constant $C_1=C_1(B)>0$, yielding (ii). Claim (iii),(iv) immediately follow from the construction of $\widetilde{\times}$ in combination with Proposition 2.19 and Lemma A.21.(i). ∎

Another statement that can be deduced from Proposition 2.19 is connected to the construction of neural networks which approximate the identity on $\mathbb{R}^d$.

Corollary A.19 *Let $\varrho:\mathbb{R}\to\mathbb{R}$ be such that ϱ is twice times continuously differentiable in a neighborhood of some $x_0\in\mathbb{R}$ and $\varrho'(x_0)\neq 0$. fulfill the assumptions of Proposition 2.19 for some $n=2$, for $r=1$ and assume that for some $k\leq n$ we have that $\varrho\in W^{k,\infty}_{\mathrm{loc}}(\mathbb{R})$. Then, for every $B>0$, $d\in\mathbb{N}$, for every $L\in\mathbb{N}_{\geq 2}$ and for every $\varepsilon\in(0,1)$ there exists a constant $C=C(B,\varrho)>0$ and a neural network $\Phi_\varepsilon^{L,B,d}$ with d-dimensional input, d-dimensional output and the following properties:*

(i) $\left\|R_\varrho(\Phi_\varepsilon^{L,B,d})-x\right\|_{W^{k,\infty}([-B,B]^d;\mathbb{R}^d)}\leq\varepsilon$;

(ii) $\|R_\varrho(\Phi_\varepsilon^{L,B,d})\|_{W^{k,\infty}([-B,B]^d;\mathbb{R}^d)}\leq C\max\{1,B\}$;

(iii) $L\left(\Phi_\varepsilon^{L,B,d}\right)=L$, *as well as* $M\left(\Phi_\varepsilon^{L,B,d}\right)\leq 4dL-3d$;

(iv) $\left\|\Phi_\varepsilon^{L,B,d}\right\|_{\max}\leq CL\varepsilon^{-1}$.

Proof. W.l.o.g., we assume that $d=1$. The other cases follow from a minor modification of the parallelization of neural networks with the same number of layers. Let $\Phi^1_{\varepsilon/L}$ be the neural network from Proposition 2.19 for $B=B+1$. We define $\Phi_\varepsilon^{L,B,d}:=\Phi^1_{\varepsilon/L}\bullet\ldots\bullet\Phi^1_{\varepsilon/L}$, where we perform $L-2$ concatenations. It is easy to see that $\Phi_\varepsilon^{L,B,d}=((A_1,b_1),(A_2,b_2),\dots,(A_L,b_L))$, where

$$\begin{aligned}A_1&=\left(0,-\frac{\varepsilon}{LC}\right)^T\in\mathbb{R}^{2,1},\\ b_1&=(x_0,x_0)^T\in\mathbb{R}^2,\\ A_\ell&=\begin{pmatrix}0 & 0\\ -\frac{1}{\varrho'(x_0)} & \frac{1}{\varrho'(x_0)}\end{pmatrix}\in\mathbb{R}^{2,2},\quad\text{for }\ell=2,\dots,L-1,\\ b_\ell&=(x_0,x_0)^T\in\mathbb{R}^2,\quad\text{for }\ell=2,\dots,L-1,\\ A_L&=\frac{LC}{\varepsilon\varrho'(x_0)}(1,-1)\in\mathbb{R}^{1,2},\\ b_L&=0\in\mathbb{R},\end{aligned}$$

and where $C > 0$ is a suitable constant provided by Proposition 2.19. By Proposition 2.19 we also have that $R_\varrho(\Phi^1_{\varepsilon/L})(x) \in [-B - \varepsilon/L, B + \varepsilon/L]$ for all $x \in [-B, B]$ as well as

$$\left\| R_\varrho(\Phi^1_{\varepsilon/L}) - x \right\|_{W^{k,\infty}([-B,B])} \leq \frac{\varepsilon}{L}.$$

Iterating this argument shows that $R_\varrho(\Phi^{L,B}_\varepsilon)(x) \in [-B - \varepsilon, B + \varepsilon]$ for all $x \in [-B, B]$ and that

$$\left\| R_\varrho(\Phi^{L,B,d}_\varepsilon) - x \right\|_{W^{k,\infty}([-B,B])} \leq \varepsilon.$$

The other properties follow immediately from (i) in combination with the definition of $\Phi^{L,B,d}_\varepsilon$. ∎

Some well-known activation functions, e.g., the (leaky) ReLU, do not fulfill the assumptions stated in Corollary A.18 (ϱ should be three times continuously differentiable in a neighborhood of some $x_0 \in \mathbb{R}$ with $\varrho''(x_0) \neq 0$) where an approximative multiplication is derived. However, we note that the proof strategy only requires the approximation of the square function. In the following, we show that in case of the (leaky) ReLU this can be done with $\mathcal{O}(\log^2(1/\varepsilon))$ weights and $\mathcal{O}(\log(1/\varepsilon))$ layers. The same arguments as in Corollary A.18, then lead to an approximative multiplication with different complexity bounds. The results presented in the following proposition can be found in a similar way in [SZ19, Prop. 3.1]. However, since our results contain some minor extensions we decided to give the proof here for the sake of completeness.

In [Yar17, Prop. 2], a ReLU neural network is constructed that approximates the square function $x \to x^2$ on the interval $(0, 1)$ in the L^∞ norm. Interestingly, the same construction can be used when measuring the approximation error in the $W^{1,\infty}$ norm. In particular, the depth and the number of weights of the network do not grow asymptotically faster to satisfy the approximation accuracy with respect to this stronger norm.

Proposition A.20 *Let $\varrho : \mathbb{R} \to \mathbb{R}, x \mapsto \max\{0, x\}$ be the ReLU. There exists a constant $C > 0$, such that for all $\varepsilon \in (0, 1/2)$ there is a neural network Φ^{sq}_ε with at most $C \cdot \log_2^2(1/\varepsilon)$ nonzero weights, at most $C \cdot \log_2(1/\varepsilon)$ layers, and with one-dimensional input and output such that*

$$\| R_\varrho(\Phi^{sq}_\varepsilon)(x) - x^2 \|_{W^{1,\infty}((0,1);dx)} \leq \varepsilon \tag{A.20}$$

and $R_\varrho(\Phi^{sq}_\varepsilon)(0) = 0$. Furthermore, it holds that

$$|R_\varrho(\Phi^{sq}_\varepsilon)|_{W^{1,\infty}((0,1))} \leq C. \tag{A.21}$$

***Proof*.** In the proof of [Yar17, Prop. 2] it is shown that there exists a constant $C > 0$, such that for each $m \in \mathbb{N}$ there is a neural network Φ_m with at most $C \cdot m$ nonzero weights and at most $C \cdot m$ layers the realization of which is a piecewise linear interpolation of $x \mapsto x^2$ on $(0, 1)$. In detail, it is shown there, that the network Φ_m satisfies for all $k \in \{0, \dots, 2^m - 1\}$

and all $x \in \left[\frac{k}{2^m}, \frac{k+1}{2^m}\right]$

$$R_\varrho(\Phi_m)(x) = \left(\frac{(k+1)^2}{2^m} - \frac{k^2}{2^m}\right)\left(x - \frac{k}{2^m}\right) + \left(\frac{k}{2^m}\right)^2. \tag{A.22}$$

Thus, $R_\varrho(\Phi_m)$ is a piecewise linear interpolant of f with $2^m + 1$ uniformly distributed breakpoints $\frac{k}{2^m}, k = 0, \dots, 2^m$. In particular, $R_\varrho(\Phi_m)(0) = 0$. Furthermore, it is shown in the proof of [Yar17, Prop. 2] that

$$\|R_\varrho(\Phi_m)(x) - x^2\|_{L^\infty((0,1);dx)} \leq 2^{-2-2m}. \tag{A.23}$$

Since in [Yar17] neural networks with skip connections are considered, we need to adapt the construction to our setting. This can easily be done by replacing skip connections with identity layers. The resulting network has the same depth and the number of weights can be bounded by Cm^2.

We will now show that the approximation error of the derivative can be bounded in a similar way. In particular, we show the estimate

$$|R_\varrho(\Phi_m) - x^2|_{W^{1,\infty}((0,1);dx)} \leq 2^{-m}. \tag{A.24}$$

From Equation (A.22) we get for all $k = 0, \dots, 2^m - 1$

$$\begin{aligned}
|R_\varrho(\Phi_m)(x) - x^2|_{W^{1,\infty}((k/2^m,(k+1)/2^m);dx)} &= \left\| \frac{(k+1)^2}{2^m} - \frac{k^2}{2^m} - 2x \right\|_{L^\infty((k/2^m,(k+1)/2^m);dx)} \\
&= \left\| \frac{2k+1}{2^m} - 2x \right\|_{L^\infty((k/2^m,(k+1)/2^m);dx)} \\
&= \max\left\{ \left| \frac{2k+1}{2^m} - 2\frac{k}{2^m} \right|, \left| \frac{2k+1}{2^m} - 2\frac{k+1}{2^m} \right| \right\} \\
&= 2^{-m}.
\end{aligned}$$

Combining Equation (A.23) and (A.24) yields

$$\|R_\varrho(\Phi_m)(x) - x^2\|_{W^{1,\infty}((0,1);dx)} \leq \max\left\{2^{-2m-2}, 2^{-m}\right\} = 2^{-m}.$$

Clearly, the weak derivative of Φ_m is a piecewise constant function, which assumes its maximum on the last piece. Hence,

$$|R_\varrho(\Phi_m)|_{W^{1,\infty}((0,1))} \leq \frac{(2^m)^2 - (2^m - 1)^2}{2^m} = 2 - \frac{1}{2^m} \leq 2. \tag{A.25}$$

Let now $\varepsilon \in (0, 1/2)$ and choose $m = \lceil \log_2(1/\varepsilon) \rceil$. Now, $\Phi_\varepsilon^{\mathrm{sq}} := \Phi_m$ satisfies the approximation bound in Equation (A.20) and $R_\varrho(\Phi_\varepsilon^{\mathrm{sq}})(0) = 0$. The estimate (A.21) holds because of Equation (A.25). The number of weights can be bounded by

$$M(\Phi_\varepsilon^{\mathrm{sq}}) \leq C \cdot m^2 \leq C \cdot (\log_2(1/\varepsilon) + 1)^2 \leq C' \cdot \log_2(1/\varepsilon),$$

for some suitable constant $C' > 0$. In a similar way, the number layers can be bounded.

This concludes the proof.

■

Before we continue, let us have a closer look at the properties of the concatenation of two neural networks in the following special cases.

Lemma A.21 *Let Φ be a neural network with m-dimensional output.*

(i) If $a \in \mathbb{R}^{1\times m}$, then,

$$M(((a,0))\bullet\Phi) \leq M(\Phi) \quad \text{and} \quad \|((a,0))\bullet\Phi)\|_{\max} \leq m\|\Phi\|_{\max}\max_{i=1,\dots,m} a_i.$$

(ii) Let $\Phi_\varepsilon^{L,B,m}$ be the approximate identity network from Corollary A.19. Then, for some constant $C = C(B,\varrho)$ there holds

$$M(\Phi_\varepsilon^{L,B,m}\bullet\Phi) \leq M(\Phi) + M(\Phi_\varepsilon^{L,B,m}) \text{ and } \|(\Phi_\varepsilon^{L,B,m}\bullet\Phi)\|_{\max} \leq C\max\{\|\Phi\|_{\max}, \varepsilon^{-1}\}.$$

(iii) Let $\widetilde{\times}$ be the approximate multiplication network from Corollary A.18. If $m = 2$, then, for some constant $C = C(B,\varrho)$ there holds

$$M(\widetilde{\times}\bullet\Phi) \leq CM(\Phi) \quad \text{and} \quad \|\widetilde{\times}\bullet\Phi\|_{\max} \leq C\max\{\|\Phi\|_{\max}, \varepsilon^{-2}\}.$$

***Proof*.** For the first part of the proof of (i), see [KPRS22]. The second part is clear.

From now on, let $\Phi = ((A_1,b_1),\dots,(A_{L(\Phi)},b_{L(\Phi)}))$.

For the proof of (ii), let $\Phi_\varepsilon^{L,B,m} = ((A_1^{\mathrm{id}},b_1^{\mathrm{id}}),\dots,(A_L^{\mathrm{id}},b_L^{\mathrm{id}}))$ and recall that

$$\Phi_\varepsilon^{L,B,m}\bullet\Phi = ((A_1,b_1),\dots,(A_{L(\Phi)-1},b_{L(\Phi)-1}),(A_1^{\mathrm{id}}A_{L(\Phi)}, A_1^{\mathrm{id}}b_{L(\Phi)}+b_1^{\mathrm{id}}),(A_2^{\mathrm{id}},b_2^{\mathrm{id}}),\dots,(A_L^{\mathrm{id}},b_L^{\mathrm{id}})).$$

Hence, in order to proof (ii), we only need to examine $(A_1^{\mathrm{id}}A_{L(\Phi)}, A_1^{\mathrm{id}}b_{L(\Phi)}+b_1^{\mathrm{id}})$. From the construction of $\Phi_\varepsilon^{L,B,m}$ we have that $\|A_1^{\mathrm{id}}\|_0 = m$ and that A_1^{id} has block diagonal structure. Additionally, all entries of A_1^{id} are bounded in absolute value by $\frac{\varepsilon}{L\widetilde{C}} \leq 1$ for some $\widetilde{C} \geq 1$. From this, the claim follows.

The proof of (iii) can be done in a similar manner as the proof of (ii). ■

A.5 Proof of Proposition 2.21 (Upper Bounds)

In this section we provide the proofs of those statements of Section 2.3 as well as additional auxiliary statements which together lead to the proof of Proposition 2.21. Appendix A.5.1 is concerned with the proof of Lemma 2.17 which establishes the conditions of the PU. Appendix A.5.2, which contains the proof of Lemma A.22, shows that we are in a position to efficiently approximate $f \in \mathcal{F}_{n,d,p}$ by sums of polynomials multiplied with the functions from the PU. Appendix A.5.3 in turn shows that these sums of localized polynomials can be approximated by neural networks. Appendix A.5.4 concludes the proof of Proposition 2.21.

A.5.1 Approximate Partition of Unity

We start with the proof of Lemma 2.17 which establishes the properties of the exponential (polynomial, exact) (j,τ)-PU.

Proof of Lemma 2.17 . For the proof of the properties ((i)) and ((ii)), we will always assume w.l.o.g. that $m = 0$ unless stated otherwise. Moreover, we only give the proof for the case of an exponential PU. The other cases follow in essentially the same way with some simplifications.

ad ((i)): First of all, assume that $d = 1$. For $\tau = 0$ and $j = 0$ this follows directly from the boundedness of ϱ. For $\tau = 1$ and $j = 0$, we have that ϱ is Lipschitz continuous, and, thus,

$$\begin{aligned}|\phi_0^s(x)| &\leq \frac{1}{s(B-A)}\left(|\varrho(3sNx+2s)-\varrho(3sNx+s)|+|\varrho(3sNx-s)-\varrho(3sNx-2s)|\right)\\ &\leq 2\frac{\mathrm{Lip}(\varrho)\cdot s}{s(B-A)} = 2\frac{\mathrm{Lip}(\varrho)}{(B-A)}.\end{aligned}$$

For $\tau \in \{0,1\}$ and $j \geq 1$ this follows from the case $j = 0$ together with $\varrho' \in W^{j-1,\infty}(\mathbb{R})$ and the chain rule.

Now, let $d \in \mathbb{N}$ be arbitrary. Since we will need it in the proof of ((ii)), we prove the following more general statement (Statement ((i)) follows by considering $I = \{1,\dots,d\}$). Moreover, we will prove this statement only for $k \leq \min\{j,2\}$, since the rest of the proof can be done in exactly the same way by exploiting the tensor structure of ϕ_m^s.

> *Let $I \subset \{1,\dots,d\}$ be arbitrary. Moreover, for $m \in \{0,\dots,N\}^{|I|}$ we define $\phi_{m,I}^s : \mathbb{R}^{|I|} \to \mathbb{R}, x \mapsto \prod_{1\leq l\leq |I|} \psi^s\left(3N\left(x_l - \frac{m_l}{N}\right)\right)$ as well as $\phi_m^s := \phi_{m,I}^s$, if $I = \{1,\dots,d\}$. Then for $k \in \{0,\dots,j\}$ it holds that*
>
> $$\left|\phi_{m,I}^s\right|_{W^{k,\infty}(\mathbb{R}^{|I|})} \leq C^{|I|}\cdot N^k \cdot s^{\max\{0,k-\tau\}}.$$

It is clear that by the definition of $\phi_{m,I}^s$ and what we have shown for $d = 1$ that for $k = 0$ there holds

$$\left|\phi_{m,I}^s\right|_{W^{0,\infty}(\mathbb{R}^{|I|})} \leq C^{|I|}. \tag{A.26}$$

Now, let $i \in I$ be arbitrary. Then, by using the tensor product structure of $\phi_{m,I}^s$ in combination with what we have shown before for $d = 1$, for the case $k = 1$ and (A.26) for $I' := I \setminus \{i\}$ we obtain for a.e. $x \in \mathbb{R}^{|I|}$

$$\begin{aligned}\left|\frac{\partial}{\partial x_i}\phi_{m,I}^s(x)\right| &= \left|\phi_{m,I'}^s(x_1,\dots,x_{i-1},x_{i+1},\dots,x_{|I|})\right|\cdot\left|\left(\psi^s\left(3N\left(\cdot - {}^{m_i}/_N\right)\right)\right)'(x_i)\right|\\ &\leq C^{|I|-1}\cdot CN = C^{|I|}Ns^{\max\{0,k-\tau\}}\end{aligned}$$

which implies that $|\phi_{m,I}^s|_{W^{1,\infty}(\mathbb{R}^{|I|})} \leq C^{|I|}N$.

Finally, let additionally be $r \in I$ be arbitrary. If $i = r$ then we have that (by using (A.26)

in combination with what we have shown for $d = 1$) that

$$\begin{aligned}\left|\frac{\partial^2}{\partial x_i^2}\phi^s_{m,I}(x)\right| &= \left|\phi^s_{m,I'}(x_1,\dots,x_{i-1},x_{i+1},\dots,x_{|I|})\right| \cdot \left|\left(\psi^s\left(3N\left(\cdot - {}^{m_i}/_N\right)\right)\right)''(x_i)\right| \\ &\leq C^{|I|-1}\cdot CN^2 s^{\max\{0,k-\tau\}} = C^{|I|}N^2 s^{\max\{0,k-\tau\}}.\end{aligned}$$

Moreover, if $i \neq r$, then, if we set $I'' := I \setminus \{i,r\}$ we obtain with similar arguments as before that

$$\begin{aligned}&\left|\frac{\partial^2}{\partial x_i \partial x_r}\phi^s_{m,I}(x)\right| \\ &= \left|\phi^s_{m,I''}(x_1,\dots,x_{i-1},x_{i+1},\dots,x_{r-1},x_{r+1},\dots,x_{|I|})\right| \\ &\quad \cdot \left|\left(\psi^s\left(3N\left(\cdot - {}^{m_i}/_N\right)\right)\right)'(x_i)\right| \cdot \left|\left(\psi^s\left(3N\left(\cdot - {}^{m_r}/_N\right)\right)\right)'(x_r)\right| \\ &\leq C^{|I|-2}\cdot CN\cdot CNs^{\max\{0,k-\tau\}} = C^{|I|}N^2 s^{\max\{0,k-\tau\}},\end{aligned}$$

where we assumed w.l.o.g. that $i < r$. This implies $|\phi^s_{m,I}|_{W^{2,\infty}(\mathbb{R}^{|I|})} \leq C^{|I|}N^2 s^{\max\{0,k-\tau\}}$.

ad ((ii)): First of all, assume that $d = 1$. Let $\underline{\tau = 0}$ and let $x \leq -1/N$. Then, since $s > R$, we have that $3Nsx + 3/2s, 3Nsx - 3/2 \leq -R$. We then have by the triangle inequality and the assumption on ϱ that

$$\begin{aligned}|\phi^s_0(x)| &= \left|\frac{\varrho(3Nsx+3/2s) - \varrho(3Nsx-3/2s)}{B-A}\right| \\ &\leq \left|\frac{\varrho(3Nsx+3/2s)-A}{B-A}\right| + \left|\frac{\varrho(3Nsx-3/2s)-A}{B-A}\right| \\ &\leq \frac{C'e^{D(3Nsx+3/2s)} + C'e^{D(3Nsx-3/2s)}}{B-A} \leq \frac{C'e^{D(-3s+3/2s)} + C'e^{D(-3s-3/2s)}}{B-A} \\ &\leq 2C'\frac{e^{-Ds}}{B-A}.\end{aligned}$$

Now, let $k \in \{1,\dots,j\}$. Then, by the assumption on ϱ, we have

$$\begin{aligned}\left|(\phi^s_0)^{(k)}(x)\right| &= (3Ns)^k \left|\frac{\varrho^{(k)}(3Nsx+3/2s) - \varrho^{(k)}(3Nsx-3/2s)}{B-A}\right| \\ &\leq (3Ns)^k \left|\frac{\varrho^{(k)}(3Nsx+3/2s)}{B-A}\right| + \left|\frac{\varrho^{(k)}(3Nsx-3/2s)}{B-A}\right| \\ &\leq \frac{C'(3Ns)^k\left(e^{D(3Nsx+3/2s)} + e^{D(3Nsx-3/2s)}\right)}{B-A} \\ &\leq \frac{C'(3Ns)^k\left(e^{D(-3s+3/2s)} + e^{D(-3s-3/2s)}\right)}{B-A} \\ &\leq \frac{2C'}{B-A}(3Ns)^k e^{-Ds}.\end{aligned}$$

The case $x \geq 1/N$ can be proven in the same way.

Now let $\underline{\tau = 1}$ and let again $x \leq -1/N$. Then $3Nsx + 2s, 3Nsx + s, 3Nsx - 2s, 3Nsx -$

$s \leq -s < -R$. By the mean value theorem there exist $\xi_1 \in (3Nsx + s, 3Nsx + 2s)$ and $\xi_2 \in (3Nsx - 2s, 3Nsx - s)$ such that

$$\phi_0^s(x) = \frac{1}{s(B-A)} \left(\varrho'(\xi_1^x) - \varrho'(\xi_2^x) \right).$$

The remainder of the proof follows in exactly the same way as the proof of the analogous statement for $\tau = 0$. The statement for $x \geq 1/N$ can be done in exactly the same manner. Now, let $d \in \mathbb{N}$ and let $x \in \Omega_m^c$. Then there exists some $l \in \{1, \dots d\}$ with $|x_l - \frac{m_l}{N}| \geq 1/N$. This implies for $I' = \{1, \dots, d\} \setminus \{l\}$ by employing Equation (A.26) that

$$|\phi_m^s(x)| = \left|\phi_{m,I'}^s(x_1, \dots, x_{l-1}, x_{l+1}, \dots, x_d)\right| \cdot \left|\psi^s\left(3N\left(x_l - {}^{m_l}/_N\right)\right)\right| \leq C^{d-1} \cdot Ce^{-Ds}.$$

This shows that $|\phi_m^s|_{W^{0,\infty}(\Omega_m^c)} \leq C^d e^{-Ds}$. By proceeding in a similar manner and with the same techniques as in the proof of ((i)), one can show the remaining Sobolev semi-norm estimates for the higher-order derivatives.

ad ((iii)): First of all, assume that $d = 1$. Let $\underline{\tau = 0}$. It is not hard to see that

$$\sum_{m=0}^{N} \phi_m^s(x) = \frac{1}{B-A} \left(\varrho(3Nsx + 3/2s) - \varrho(3Ns(x-1) - 3/2s) \right).$$

We now have for all $x \in (0,1)$ and using the properties of ϱ that

$$\begin{aligned} \left| 1 - \sum_{m=0}^{N} \phi_m^s(x) \right| &= \left| \frac{B - A - (\varrho(3Nsx + 3/2s) - \varrho(3Ns(x-1) - 3/2s))}{B-A} \right| \\ &\leq \left| \frac{B - \varrho(3Nsx + 3/2s)}{B-A} \right| + \left| \frac{A - \varrho(3Nsx - 3Ns - 3/2s)}{B-A} \right| =: \mathrm{I} + \mathrm{II}. \end{aligned}$$

We continue by estimating I. Since $3Nsx + 3/2s \geq 3/2s > 3/2R$, we obtain that

$$\mathrm{I} \leq \frac{C' e^{-D(3Nsx + 3/2s)}}{B-A} \leq \frac{C' e^{-3/2 \cdot Ds}}{B-A}$$

On the other hand, since $3Nsx - 3Ns - 3/2s \leq -3/2s \leq 0$ we obtain that

$$\mathrm{II} \leq \frac{C' e^{D(3Nsx - 3Ns - 3/2s)}}{B-A} \leq \frac{C' e^{-3/2 \cdot Ds}}{B-A}.$$

For the multidimensional case we have

$$\begin{aligned} \left| 1 - \sum_{m \in \{0,\dots,N\}^d} \phi_m^s(x) \right| &= \left| 1 - \sum_{m \in \{0,\dots,N\}^d} \prod_{l=1}^{d} \psi^s\left(3N\left(x_l - \frac{m_l}{N}\right)\right) \right| \\ &= \left| 1 - \prod_{l=1}^{d} \sum_{m=0}^{N} \psi^s\left(3N\left(x_l - \frac{m}{N}\right)\right) \right| \end{aligned}$$

$$= \left| 1 - \prod_{l=1}^{d} \left(\underbrace{\frac{1}{B-A} \left(\varrho(3Nsx_l + 3/2s) - \varrho(3Ns(x_l - 1) - 3/2s) \right)}_{:=\pi_l, \text{ and } \pi_0 := 1} \right) \right|$$

$$\leq \sum_{l=1}^{d} |\pi_0 \cdot \ldots \cdot \pi_l (1 - \pi_{l+1})| \leq C \cdot e^{-3/2Ds},$$

which follows from the one-dimensional case. Now, let $k \in \{1, \ldots, j\}$ and we consider only the case $d = 1$. The multi-dimensional case follows in exactly the same manner as the analogous considerations in ((i)) and ((ii)). We have that

$$\left| \left(\sum_{m=0}^{N} \phi_m^s \right)^{(k)} (x) \right| \leq \frac{(3Ns)^k}{B-A} \left(\left| \varrho^{(k)}(3Nsx + 3/2s) \right| + \left| \varrho^{(k)}(3Nsx - 3Ns - 3/2s) \right| \right)$$

Since $x > 0$, we have that $3Nsx + 3/2s \geq 3/2s > R$. Since $x < 1$, $3Nsx - 3Ns - 3/2s \leq -3/2R < -R$. Hence, by the assumptions on ϱ we obtain that

$$\left| \left(\sum_{m=0}^{N} \phi_m^s \right)^{(k)} (x) \right| \leq \frac{C'(3Ns)^k}{B-A} \left(e^{-D(3Nsx+3/2s)} + e^{D(3Nsx-3Ns-3/2s)} \right)$$

$$\leq \frac{2C'(3Ns)^k}{B-A} e^{-3/2Ds}.$$

The multidimensional case for $k \in \{0, \ldots, j\}$ follows in a similar manner as above from the tensor structure. Now, let $\underline{\tau = 1}$. It is not hard to see that for all $x \in \mathbb{R}$ there holds

$$\sum_{m=0}^{N} \phi_m^s(x) = \frac{\varrho(3Nsx + 2s) - \varrho(3Nsx + s) - \varrho(3Nsx - 3Ns - s) + \varrho(3Nsx - 3Ns - 2s)}{s(B-A)}.$$

Now, let $x \in (0,1)$. We have that $3Nsx + 2s, 3Nsx + s \geq s > R$ and $3Nsx - 3Ns - s, 3Nsx - 3Ns - 2s \leq -s < -R$. Hence, by the mean value theorem, for every $x \in \mathbb{R}$ there exist $\xi_1 \in (3Nxs + s, 3Nxs + 2s)$ and $\xi_2 \in (3Nxs - 3Ns - 2s, 3Nxs - 3Ns - s)$ such that

$$\sum_{m=0}^{N} \phi_m^s(x) = \frac{1}{B-A} \left(\varrho'(\xi_1) - \varrho'(\xi_2) \right).$$

Now we have that

$$\left| 1 - \sum_{m=0}^{N} \phi_m^s(x) \right| \leq \left| \frac{B - \varrho'(\xi_1)}{B-A} \right| + \left| \frac{A - \varrho'(\xi_2)}{B-A} \right|.$$

The remainder of the statement can be proven in exactly the same way as the analogous statement for $\tau = 0$. **ad ((iv)):** This immediately follows from the definition of the functions ϕ_m^s. ∎

A.5.2 Approximation by Localized Polynomials

In this section, we demonstrate how to approximate a function $f \in \mathcal{F}_{n,d,p}$ by localized polynomials based on the exponential (polynomial, exact) (j,τ)-PU. We only give the proof for the case of an exponential PU. The other cases follow in essentially the same way with some simplifications.

Lemma A.22 *We make the following assumption:*

- *Let $d \in \mathbb{N}$, $j, \tau \in \mathbb{N}_0$, $k \in \{0,\dots,j\}$, $n \in \mathbb{N}_{\geq k+1}$ and $1 \leq p \leq \infty$.*
- *Assume that $(\Psi^{(j,\tau,N,s)})_{N\in\mathbb{N},s\geq 1}$ is an exponential (polynomial, exact) (j,τ)-PU from Definition 2.13. Let $\mu \in (0,1)$. For $N \in \mathbb{N}$, set*

$$s := \begin{cases} N^{\mu}, & \text{if exponential PU,} \\ N^{\frac{2d/p+d+n}{D}}, & \text{if polynomial PU,} \\ 1, & \text{if exact PU,} \end{cases}$$

Then there is a constant $C = C(d,n,p,k) > 0$ and $\widetilde{N} = \widetilde{N}(d,p,\mu,k,\tau) \in \mathbb{N}$ such that for every $f \in W^{n,p}((0,1)^d)$ and every $m \in \{0,\dots,N\}^d$, there exist polynomials $p_{f,m}(x) = \sum_{|\alpha|\leq n-1} c_{f,m,\alpha} x^{\alpha}$ for $m \in \{0,\dots,N\}^d$ with the following properties:

Set $f_N := \sum_{m\in\{0,\dots,N\}^d} \phi_m^s p_{f,m}$. Then, the operator $T_k : W^{n,p}((0,1)^d) \to W^{k,p}((0,1)^d)$ with $T_k f = f - f_N$ is linear and bounded with

$$\|T_k f\|_{W^{k,p}((0,1)^d)} \leq C \|f\|_{W^{n,p}((0,1)^d)} \cdot \begin{cases} \left(\frac{1}{N}\right)^{n-k-\mu_{(k=2)}}, & \text{if exponential PU,} \\ \left(\frac{1}{N}\right)^{n-k}, & \text{for } k \leq \tau, \text{ if polynomial PU,} \\ \left(\frac{1}{N}\right)^{n-k}, & \text{if exact PU,} \end{cases}$$

for all $N \in \mathbb{N}$ with $N \geq \widetilde{N}$.

Before the proof of this statement, we need some preparation. We start with the following observation.

Remark A.23 Since the polynomials utilized in Lemma A.22 are the averaged Taylor polynomials from the Bramble-Hilbert Corollary A.11, we get that there is a constant $C = C(d,n,k) > 0$ such that for any $f \in W^{n,p}((0,1)^d)$ the coefficients of the polynomials $p_{f,m}$ satisfy

$$|c_{f,m,\alpha}| \leq C \|\widetilde{f}\|_{W^{n,p}(\Omega_{m,N})} N^{d/p},$$

for all $\alpha \in \mathbb{N}_0^d$ with $|\alpha| \leq n-1$, and for all $m \in \{0,\dots,N\}^d$, where $\Omega_{m,N} := B_{\frac{1}{N},\|\cdot\|_\infty}\left(\frac{m}{N}\right)$ and $\widetilde{f} \in W^{n,p}(\mathbb{R}^d)$ is an extension of f.

We now state and prove an auxiliary result. The estimation will be very rough and can for sure be improved. This is, however, not necessary for our purpose.

Lemma A.24 *Under the conditions of Lemma A.22 and with the notation from Remark A.23 we have for all $m, \tilde{m} \in \{0, \dots, N\}^d$ the estimate*

$$\|\widetilde{f} - p_{f,m}\|_{W^{k,p}(\Omega_{\tilde{m},N})} \leq CN^{d/p}\|f\|_{W^{n,p}((0,1)^d)},$$

for a constant $C = C(n, d, p, k)$.

Proof . We start with bounding the norm of the polynomial by using the triangle inequality. There holds

$$\|p_{f,m}\|_{W^{k,p}(\Omega_{\tilde{m},N})} = \left\| \sum_{|\alpha|\leq n-1} c_{f,m,\alpha} x^\alpha \right\|_{W^{k,p}(\Omega_{\tilde{m},N};dx)} \leq \sum_{|\alpha|\leq n-1} |c_{f,m,\alpha}| \cdot \|x^\alpha\|_{W^{k,p}(\Omega_{\tilde{m},N};dx)}.$$

Using that $\Omega_{\tilde{m},N} \subset B_{2,\|\cdot\|_\infty}$ we get

$$\|x^\alpha\|_{W^{k,p}(\Omega_{\tilde{m},N};dx)} \leq (n-1)^k 2^{|\alpha|} \leq (n-1)^k 2^{n-1}. \tag{A.27}$$

If we now combine Remark A.23 with Equation (A.27), we get

$$\begin{aligned}\sum_{|\alpha|\leq n-1} |c_{f,m,\alpha}| \|x^\alpha\|_{W^{k,p}(\Omega_{\tilde{m},N};dx)} &\leq C(n-1)^k 2^{n-1} \sum_{|\alpha|\leq n-1} N^{d/p} \|\widetilde{f}\|_{W^{n,p}(\Omega_{m,N})} \\ &\leq CN^{d/p}\|f\|_{W^{n,p}((0,1)^d)},\end{aligned}$$

where we have additionally used Remark A.4 in the last step. Finally, we can estimate, by the triangle inequality

$$\|\widetilde{f} - p_{f,m}\|_{W^{k,p}(\Omega_{\tilde{m},N})} \leq C\|f\|_{W^{k,p}((0,1)^d)} + CN^{d/p}\|f\|_{W^{n,p}((0,1)^d)} \leq CN^{d/p}\|f\|_{W^{n,p}((0,1)^d)},$$

where we again used the extension property from Equation (A.1) for the first step. ∎

Now we are in a position to prove Lemma A.22.

Proof of Lemma A.22 . We use approximation properties of the polynomials from the Bramble-Hilbert Corollary A.11 to derive local estimates and then combine them using an exponential PU to obtain a global estimate. In order to use this strategy also near the boundary, we make use of an extension operator (see Remark A.4).

Step 1 (Local estimates based on Bramble-Hilbert): For each $m \in \{0, \dots, N\}^d$ we set

$$\Omega_{m,N} := B_{\frac{1}{N},\|\cdot\|_\infty}\left(\frac{m}{N}\right)$$

and denote by $p_m = p_{f,m}$ the polynomial from Corollary A.11 so that we can directly state the estimate

$$\|\widetilde{f} - p_m\|_{W^{k,p}(\Omega_{m,N})} \leq C\left(\frac{1}{N}\right)^{n-k} \|\widetilde{f}\|_{W^{n,p}(\Omega_{m,N})}. \tag{A.28}$$

Furthermore, similarly to [GKP20, Lemma C.4], we obtain the estimate

$$\begin{aligned}\left\|\phi_m^s(\widetilde{f}-p_m)\right\|_{W^{k,p}(\Omega_{m,N})} &\leq C\sum_{\kappa=0}^{k}\|\phi_m^s\|_{W^{\kappa,\infty}(\Omega_{m,N})}\|\widetilde{f}-p_m\|_{W^{k-\kappa,p}(\Omega_{m,N})}\\ &\leq C\sum_{\kappa=0}^{k}N^{\kappa+\mu_{(\kappa=2)}}\left(\frac{1}{N}\right)^{n-k+\kappa}\|\widetilde{f}\|_{W^{n,p}(\Omega_{m,N})}\\ &\leq C\left(\frac{1}{N}\right)^{n-k-\mu_{(k=2)}}\|\widetilde{f}\|_{W^{n,p}(\Omega_{m,N})},\end{aligned}$$

where we used the product rule from Lemma A.12 for the first step and the estimate of the derivative of ϕ_m^s from Lemma 2.17 ((i)) together with the Bramble-Hilbert estimate in Equation (A.28) for the second step.

Step 2 (Local estimates based on exponential decay): Since our localizing bump functions ϕ_m^s do not necessarily have compact support on $\Omega_{m,N}$ we also need to bound the influence of $\phi_m^s(\widetilde{f}-p_m)$ on patches $\Omega_{\widetilde{m},N}$ with $\widetilde{m}\neq m$ where we can not use the Bramble-Hilbert lemma. Here, we will make use of the exponential decay of the bump functions ϕ_m^s outside a certain ball centered at m/N (see Lemma 2.17 ((ii))).

This is possible for the case where $\Omega_{\widetilde{m},N}$ is not a neighboring patch of $\Omega_{m,N}$, i.e. $\|\widetilde{m}-m\|_\infty>1$. Then $\Omega_{\widetilde{m},N}\subset\Omega_m^c$ and we have (by using Lemma A.12 in the first step), that

$$\begin{aligned}\left\|\phi_m^s(\widetilde{f}-p_m)\right\|_{W^{k,p}(\Omega_{\widetilde{m},N})} &\leq C\|\phi_m^s\|_{W^{k,\infty}(\Omega_{\widetilde{m},N})}\|\widetilde{f}-p_m\|_{W^{k,p}(\Omega_{\widetilde{m},N})}\\ \text{(Lemma 2.17 ((ii))) with } \Omega_{\widetilde{m},N}\subset\Omega_m^c) &\leq CN^{k+\mu_{(k=2)}}e^{-DN^\mu}\|\widetilde{f}-p_m\|_{W^{k,p}(\Omega_{\widetilde{m},N})}\\ \text{(Lemma A.24)} &\leq C\underbrace{N^{k+\mu_{(k=2)}}N^{d/p}}_{:=\gamma(N)}e^{-DN^\mu}\|f\|_{W^{n,p}((0,1)^d)}.\end{aligned}$$

Then, by Proposition A.1, there exists $N_1=N_1(\mu,d,p)\in\mathbb{N}$ such that $e^{-DN^\mu}\leq C\gamma(N)^{-1}\cdot(N+1)^{-d-d/p}\cdot N^{-(n-k-\mu_{(k=2)})}$ for all $N\geq N_1$. Consequently, we have

$$\left\|\phi_m^s(\widetilde{f}-p_m)\right\|_{W^{k,p}(\Omega_{\widetilde{m},N})}\leq C(N+1)^{-d-d/p}N^{-(n-k-\mu_{(k=2)})}\|f\|_{W^{n,p}((0,1)^d)},$$

for all $N\geq N_1$.

Step 3 (Mixed local estimates): If $\Omega_{\widetilde{m},N}$ is a neighboring patch of $\Omega_{m,N}$, i.e. $\|\widetilde{m}-m\|_\infty=1$, then we have to split the patch in a region $\Omega_{\widetilde{m},N}\cap\Omega_m^c$ where we have exponential decay of the bump function and a region $\Omega_{\widetilde{m},N}\setminus\Omega_m^c\subset\Omega_{m,N}$ where we can make use of the Bramble-Hilbert Lemma. In detail, we have

$$\begin{aligned}\left\|\phi_m^s(\widetilde{f}-p_m)\right\|_{W^{k,p}(\Omega_{\widetilde{m},N})} &\leq \left\|\phi_m^s(\widetilde{f}-p_m)\right\|_{W^{k,p}(\Omega_{\widetilde{m},N}\setminus\Omega_m^c)}+\left\|\phi_m^s(\widetilde{f}-p_m)\right\|_{W^{k,p}(\Omega_{\widetilde{m},N}\cap\Omega_m^c)}\\ &\leq CN^{-(n-k-\mu_{(k=2)})}\left(\|\widetilde{f}\|_{W^{n,p}(\Omega_{m,N})}+(N+1)^{-d-d/p}\|f\|_{W^{n,p}((0,1)^d)}\right),\end{aligned}$$

for all $N\geq N_1$. Here we used Step 1 to bound the first term of the sum and Step 2 for the second.

Step 4 (Global estimate): Using that $\widetilde{f}$ is an extension of f on $(0,1)^d$ we can write

$$\begin{aligned}
&\left\| f - \sum_{m\in\{0,\dots,N\}^d} \phi_m^s p_m \right\|_{W^{k,p}((0,1)^d)} \\
&\leq \left\| \widetilde{f} - \sum_{m\in\{0,\dots,N\}^d} \phi_m^s \widetilde{f} \right\|_{W^{k,p}((0,1)^d)} + \left\| \sum_{m\in\{0,\dots,N\}^d} \phi_m^s (\widetilde{f} - p_m) \right\|_{W^{k,p}((0,1)^d)} \\
&\leq \underbrace{\left\| \widetilde{f}\Big(\mathbb{1}_{(0,1)^d} - \sum_{m\in\{0,\dots,N\}^d} \phi_m^s\Big) \right\|_{W^{k,p}((0,1)^d)}}_{\textbf{Step 4a}} \\
&\quad + \left(\underbrace{\sum_{\widetilde{m}\in\{0,\dots,N\}^d} \left\| \sum_{m\in\{0,\dots,N\}^d} \phi_m^s (\widetilde{f} - p_m) \right\|_{W^{k,p}(\Omega_{\widetilde{m},N})}^p}_{\textbf{Step 4b}} \right)^{1/p},
\end{aligned} \tag{A.29}$$

where the last step follows from $(0,1)^d \subset \bigcup_{\widetilde{m}\in\{0,\dots,N\}^d} \Omega_{\widetilde{m},N}$.

Step 4a (Partition of Unity): For the first term in Equation (A.29), we get by the product rule from Lemma A.12

$$\begin{aligned}
\left\| \widetilde{f}\Big(\mathbb{1}_{(0,1)^d} - \sum_{m\in\{0,\dots,N\}^d} \phi_m^s\Big) \right\|_{W^{k,p}((0,1)^d)} &\leq C \|f\|_{W^{k,p}((0,1)^d)} \left\| \mathbb{1}_{(0,1)^d} - \sum_{m\in\{0,\dots,N\}^d} \phi_m^s \right\|_{W^{k,\infty}((0,1)^d)} \\
&\overset{\text{(Property ((iii)) from Lemma 2.17)}}{\leq} C \|f\|_{W^{k,p}((0,1)^d)} \cdot N^{-(n-k-\mu_{(k=2)})},
\end{aligned} \tag{A.30}$$

for all $N \geq N_2 = N_2(\mu, k, \tau)$. For the second inequality we used the same trick as in Step 2 which is based on Proposition A.1.

Step 4b (Patches): Considering the second term from Equation (A.29), we obtain for each $\widetilde{m} \in \{0,\dots,N\}^d$

$$\begin{aligned}
&\left\| \sum_{m\in\{0,\dots,N\}^d} \phi_m^s (\widetilde{f} - p_m) \right\|_{W^{k,p}(\Omega_{\widetilde{m},N})} \\
&\leq \underbrace{\|\phi_{\widetilde{m}}^s (\widetilde{f} - p_{\widetilde{m}})\|_{W^{k,p}(\Omega_{\widetilde{m},N})}}_{(\star)} + \underbrace{\sum_{\substack{m\in\{0,\dots,N\}^d, \\ \|m-\widetilde{m}\|_\infty = 1}} \|\phi_m^s (\widetilde{f} - p_m)\|_{W^{k,p}(\Omega_{\widetilde{m},N})}}_{(\star\star)} \\
&\quad + \underbrace{\sum_{\substack{m\in\{0,\dots,N\}^d, \\ \|m-\widetilde{m}\|_\infty > 1}} \|\phi_m^s (\widetilde{f} - p_m)\|_{W^{k,p}(\Omega_{\widetilde{m},N})}}_{(\star\star\star)}.
\end{aligned} \tag{A.31}$$

The term $(\star)$ can be handled with Step 1, the term $(\star\star)$ with Step 3 and the third one $(\star\star\star)$ with Step 2. Since $(\star\star)$ and $(\star\star\star)$ require a similar strategy we only demonstrate it for the third term. We get from Step 2

$$\begin{aligned}\sum_{\substack{m\in\{0,\dots,N\}^d,\\ \|m-\widetilde{m}\|_\infty>1}} \|\phi_m^s(\widetilde{f}-p_m)\|_{W^{k,p}(\Omega_{\widetilde{m},N})} &\leq CN^{-(n-k-\mu_{(k=2)})}(N+1)^{-d-d/p}\sum_{\substack{m\in\{0,\dots,N\}^d,\\ \|m-\widetilde{m}\|_\infty>1}} \|f\|_{W^{n,p}((0,1)^d)}\\ &\leq CN^{-(n-k-\mu_{(k=2)})}(N+1)^{-d/p}\|f\|_{W^{n,p}((0,1)^d)}.\end{aligned}$$

We can now bound the sum from Equation (A.31) for each $\widetilde{m}\in\{0,\dots,N\}^d$ by

$$\begin{aligned}&\left\|\sum_{m\in\{0,\dots,N\}^d}\phi_m^s(\widetilde{f}-p_m)\right\|_{W^{k,p}(\Omega_{\widetilde{m},N})}\\ &\leq CN^{-(n-k-\mu_{(k=2)})}\left(2(N+1)^{-d/p}\|f\|_{W^{n,p}((0,1)^d)}+\sum_{\substack{m\in\{0,\dots,N\}^d,\\ \|m-\widetilde{m}\|_\infty\leq 1}}\|\widetilde{f}\|_{W^{n,p}(\Omega_{m,N})}\right).\end{aligned}\tag{A.32}$$

Consequently, we get

$$\begin{aligned}&\sum_{\widetilde{m}\in\{0,\dots,N\}^d}\left\|\sum_{m\in\{0,\dots,N\}^d}\phi_m^s(\widetilde{f}-p_m)\right\|^p_{W^{k,p}(\Omega_{\widetilde{m},N})}\\ &\leq CN^{-(n-k-\mu_{(k=2)})p}\sum_{\widetilde{m}\in\{0,\dots,N\}^d}\left(2(N+1)^{-d/p}\|f\|_{W^{n,p}((0,1)^d)}+\sum_{\substack{m\in\{0,\dots,N\}^d,\\ \|m-\widetilde{m}\|_\infty\leq 1}}\|\widetilde{f}\|_{W^{n,p}(\Omega_{m,N})}\right)^p\\ &\leq CN^{-(n-k-\mu_{(k=2)})p}(3^d+1)^{p/q}\\ &\qquad\cdot\left(\sum_{\widetilde{m}\in\{0,\dots,N\}^d}2^p(N+1)^{-d}\|f\|^p_{W^{n,p}((0,1)^d)}+\sum_{\widetilde{m}\in\{0,\dots,N\}^d}\sum_{\substack{m\in\{0,\dots,N\}^d,\\ \|m-\widetilde{m}\|_\infty\leq 1}}\|\widetilde{f}\|^p_{W^{n,p}(\Omega_{m,N})}\right)\\ &\leq CN^{-(n-k-\mu_{(k=2)})p}\left(\|f\|^p_{W^{n,p}((0,1)^d)}+3^d\sum_{\widetilde{m}\in\{0,\dots,N\}^d}\|\widetilde{f}\|^p_{W^{n,p}(\Omega_{\widetilde{m},N})}\right),\end{aligned}\tag{A.33}$$

where the first step follows from plugging in Equation (A.32), the second step follows from Hölder's inequality (with $q:=1-1/p$) and the last step follows from the definition of $\Omega_{\widetilde{m},N}$. Moreover, we use in the second and the last step the fact that the number of neighbors of a particular patch is bounded by 3^d-1. To conclude Step 4b we note that from the definition of $\Omega_{\widetilde{m},N}$ it follows that there exist 2^d disjoint subsets $\mathcal{M}_i\subset\{0,\dots,N\}^d$ such that $\bigcup_{i=1,\dots,2^d}\mathcal{M}_i=\{0,\dots,N\}^d$ and $\Omega_{m_1,N}\cap\Omega_{m_2,N}=\varnothing$ for all $m_1,m_2\in\mathcal{M}_i$ with

$m_1 \neq m_2$ and all $i = 1, \dots, 2^d$. From this we get

$$\sum_{\tilde{m}\in\{0,\dots,N\}^d} \|\widetilde{f}\|^p_{W^{n,p}(\Omega_{\tilde{m},N})} = \sum_{i=1,\dots,2^d} \sum_{\tilde{m}\in\mathcal{M}_i} \|\widetilde{f}\|^p_{W^{n,p}(\Omega_{\tilde{m},N})} \leq 2^d \|\widetilde{f}\|^p_{W^{n,p}(\bigcup_{\tilde{m}\in\{0,\dots,N\}^d}\Omega_{\tilde{m},N})} \tag{A.34}$$

and, finally, together with Remark A.4

$$\sum_{\tilde{m}\in\{0,\dots,N\}^d} \|\widetilde{f}\|^p_{W^{n,p}(\Omega_{\tilde{m},N})} \leq 2^d \|\widetilde{f}\|^p_{W^{n,p}(\bigcup_{\tilde{m}\in\{0,\dots,N\}^d}\Omega_{\tilde{m},N})} \leq C\|f\|^p_{W^{n,p}((0,1)^d)}. \tag{A.35}$$

Step 4c (Wrap it all up): Combining Equation (A.33) with Equation (A.35) from Step 4b and inserting it into Equation (A.29) together with the estimate in Equation (A.30) from Step 4a finally yields

$$\|f - f_N\|_{W^{k,p}((0,1)^d)} \leq CN^{-(n-k-\mu_{(k=2)})}\|f\|_{W^{n,p}((0,1)^d)},$$

for all $N \geq \widetilde{N} := \max\{N_1, N_2\}$ and a constant $C = C(n,d,p,k) > 0$. The linearity of T_k, $k \in \{0,\dots,j\}$ is a consequence of the linearity of the averaged Taylor polynomial (cf. Remark A.6). ∎

A.5.3 Approximation of Localized Polynomials by Neural Networks

The goal of this subsection is to demonstrate how to approximate sums of localized polynomials $\sum_p \phi_p \mathrm{poly}_p$ by neural networks. Corollary A.18 is the foundation for the following result which implements a neural network that approximates the multiplication of multiple inputs:

Lemma A.25 *Let $d, m, K \in \mathbb{N}, j \in \mathbb{N}_0$ and $N \geq 1$, $\mu \geq 0, c > 0$ be arbitrary, and let $\varrho \in W^{j,\infty}_{\mathrm{loc}}(\mathbb{R})$ fulfill the assumptions of Proposition 2.19 for $n = 3$, $r = 2$. Then there are constants $C(d,m,c,k) > 0$ such that the following holds:*

For any $\varepsilon \in (0, 1/2)$, and any neural network Φ with d-dimensional input and m-dimensional output and with number of layers and nonzero weights all bounded by K, such that

$$\|[R_\varrho(\Phi)]_l\|_{W^{k,\infty}((0,1)^d)} \leq cN^{k+\mu_{(k=2)}},$$

for $k \in \{0,\dots,j\}$, $l = 1,\dots,m$ there exists a neural network $\Psi_{\varepsilon,\Phi}$ with d-dimensional input and one-dimensional output, and with

(i) number of layers and nonzero weights all bounded by CK;

(ii) $\left\|R_\varrho(\Psi_{\varepsilon,\Phi}) - \prod_{l=1}^n [R_\varrho(\Phi)]_l\right\|_{W^{k,\infty}((0,1)^d)} \leq CN^{k+\mu_{(k=2)}}\varepsilon$;

(iii) $|R_\varrho(\Psi_{\varepsilon,\Phi})|_{W^{k,\infty}((0,1)^d)} \leq CN^{k+\mu_{(k=2)}}$;

(iv) $\|\Psi_{\varepsilon,\Phi}\|_{\max} \leq C\max\{\|\Phi\|_{\max}, \varepsilon^{-2}\}$.

Proof. We show by induction over $m \in \mathbb{N}$ that the statement holds. To make the induction argument easier we will additionally show that the network $\Psi_{\varepsilon,\Phi}$ can be chosen such that the first $L(\Phi) - 1$ layers of $\Psi_{\varepsilon,\Phi}$ and Φ coincide.

If $m = 1$, then we can choose $\Psi_{\varepsilon,\Phi} = \Phi$ for any $\varepsilon \in (0, 1/2)$ and the claim holds.

Now, assume that the claim holds for an arbitrary, but fixed $m \in \mathbb{N}$. We show that it also holds for $m+1$. For this, let $\varepsilon \in (0, 1/2)$ and let $\Phi = ((A_1, b_1), (A_2, b_2), \dots, (A_L, b_L))$ be a neural network with d-dimensional input and $(m+1)$-dimensional output and with number of layers, and nonzero weights all bounded by K, where each A_l is an $N_l \times N_{l-1}$ matrix, and $b_l \in \mathbb{R}^{N_l}$ for $l = 1, \dots L$.

Step 1 (Invoking induction hypothesis): We denote by Φ_m the neural network with d-dimensional input and m-dimensional output which results from Φ by removing the last output neuron and corresponding weights. In detail, we write

$$A_L = \begin{bmatrix} A_L^{(1,m)} \\ a_L^{(m+1)} \end{bmatrix} \quad \text{and} \quad b_L = \begin{bmatrix} b_L^{(1,m)} \\ b_L^{(m+1)} \end{bmatrix},$$

where $A_L^{(1,m)}$ is a $m \times N_{L-1}$ matrix and $a_L^{(m+1)}$ is a $1 \times N_{L-1}$ vector, and $b_L^{(1,m)} \in \mathbb{R}^m$ and $b_L^{(m+1)} \in \mathbb{R}^1$. Now we set

$$\Phi_m := \Big((A_1, b_1), (A_2, b_2), \dots, (A_{L-1}, b_{L-1}), \Big(A_L^{(1,m)}, b_L^{(1,m)}\Big)\Big).$$

Using the induction hypothesis we get that there is a neural network

$$\Psi_{\varepsilon,\Phi_m} = ((A_1', b_1'), (A_2', b_2'), \dots, (A_{L'}', b_{L'}'))$$

with d-dimensional input and one-dimensional output, and at most KC layers and nonzero weights such that

$$\left\| R_\varrho(\Psi_{\varepsilon,\Phi_m}) - \prod_{l=1}^{m} [R_\varrho(\Phi_m)]_l \right\|_{W^{k,\infty}((0,1)^d)} \leq C N^{k+\mu_{(k=2)}} \varepsilon,$$

and $|R_\varrho(\Psi_{\varepsilon,\Phi_m})|_{W^{k,\infty}((0,1)^d)} \leq C N^{k+\mu_{(k=2)}}$. Moreover, we have that $\|\Phi_m\|_{\max} \leq \|\Phi\|_{\max}$, so that there we can estimate $\|\Psi_{\varepsilon,\Phi_m}\|_{\max} \leq C \max\{\|\Phi\|_{\max}, \varepsilon^{-2}\}$. Furthermore, we can assume that the first $L-1$ layers of $\Psi_{\varepsilon,\Phi_m}$ and Φ_m coincide and, thus, also the first $L-1$ layers of $\Psi_{\varepsilon,\Phi_m}$ and Φ, i.e. $A_l = A_l'$ for $l = 1, \dots, L-1$.

Step 2 (Combining $\Psi_{\varepsilon,\Phi_m}$ and $[R_\varrho(\Phi)]_{m+1}$): Now, we construct a network $\widetilde{\Psi}_{\varepsilon,\Phi}$ where the first $L-1$ layers of $\widetilde{\Psi}_{\varepsilon,\Phi}$ and $\Psi_{\varepsilon,\Phi_m}$ (and, thus, also of Φ) coincide (by definition), and $\widetilde{\Psi}_{\varepsilon,\Phi}$ has two-dimensional output with $[R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})]_1 = R_\varrho(\Psi_{\varepsilon,\Phi_m})$ and $[R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})]_2 \approx [R_\varrho(\Phi)]_{m+1}$. For this, we add the formerly removed neuron with corresponding weights back to the L-th layer of $\Psi_{\varepsilon,\Phi_m}$ and approximately pass the output through to the last layer. Let $\Phi_\varepsilon^{L'-L+1,c,1} = ((A_1^{\mathrm{id}}, b_1^{\mathrm{id}}), \dots, (A_{L'-L+1}^{\mathrm{id}}, b_{L'-L+1}^{\mathrm{id}}))$ be the network from Corollary A.19. We define

$$\widetilde{\Psi}_{\varepsilon,\Phi} := \left((A_i', b_i')_{i=1}^{L-1}, \left(\begin{bmatrix} A_L' \\ A_1^{\mathrm{id}} a_L^{(m+1)} \end{bmatrix}, \begin{bmatrix} b_L' \\ A_1^{\mathrm{id}} b_L^{(m+1)} + b_1^{(m+1)} \end{bmatrix}\right), \left(\begin{bmatrix} A_{L+1}' \\ A_2^{\mathrm{id}} \end{bmatrix}, \begin{bmatrix} b_{L+1}' \\ b_2^{\mathrm{id}} \end{bmatrix}\right), \dots\right.$$

$$\cdots \left(\begin{bmatrix} A'_{L'} \\ A^{\mathrm{id}}_{L'-L+1} \end{bmatrix}, \begin{bmatrix} b'_{L'} \\ b^{\mathrm{id}}_{L'-L+1} \end{bmatrix} \right) \right).$$

Counting the number of nonzero weights of $\widetilde{\Psi}_{\varepsilon,\Phi}$ we get with Lemma A.21 ((ii)) that

$$\begin{aligned} M(\widetilde{\Psi}_{\varepsilon,\Phi}) &\leq M(\Psi_{\varepsilon,\Phi_m}) + \underbrace{M(\Phi)}_{\text{from } a_L^{(m+1)}, b_L^{(m+1)}} + \underbrace{4(L'-L+1)}_{\text{from approximative identity}} \\ &\leq CK + K + CK \leq CK, \end{aligned} \tag{A.36}$$

where we used in the second step the induction hypothesis twice together with the assumption on Φ. Similarly, we get the statement for $L(\widetilde{\Psi}_{\varepsilon,\Phi})$. Furthermore, $\|\widetilde{\Psi}_{\varepsilon,\Phi}\|_{\max} \leq C \max\{\|\Phi\|_{\max}, \varepsilon^{-2}\}$.

Next, we want to apply the approximate multiplication network from Corollary A.18 to the output of $\widetilde{\Psi}_{\varepsilon,\Phi}$. For this, we need to find a bounding box for the range of $R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})$. We have

$$\|R_\varrho(\Psi_{\varepsilon,\Phi_m})\|_{L^\infty((0,1)^d)} \leq C \quad \text{and} \quad \|[R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})]_2\|_{L^\infty((0,1)^d)} \leq c + \varepsilon \leq c+1,$$

and get for $B := \max\{C, c+1\}$ that Range $R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi}) \subset [-B,B]^2$. Now, we denote by $\widetilde{\times}$ the network from Corollary A.18 with $B = B$ and accuracy ε and define

$$\Psi_{\varepsilon,\Phi} := \widetilde{\times} \bullet \widetilde{\Psi}_{\varepsilon,\Phi}.$$

Step 3 ($\Psi_{\varepsilon,\Phi}$ fulfills induction hypothesis for $m+1$): **ad (i):** Clearly, $\Psi_{\varepsilon,\Phi}$ has d-dimensional input, one-dimensional output and, combining Equation (A.36) with ((iii)) of Corollary A.18 as well as Lemma A.21 (iii), at most CK nonzero weights.

ad (ii): The first $L-1$ layers of $\Psi_{\varepsilon,\Phi}$ and Φ coincide and for the approximation properties it holds that

$$\begin{aligned} &\left\| R_\varrho(\Psi_{\varepsilon,\Phi}) - \prod_{l=1}^{m+1} [R_\varrho(\Phi)]_l \right\|_{W^{k,\infty}((0,1)^d)} \\ &\quad = \left\| R_\varrho(\widetilde{\times}) \circ R_\varrho\big(\widetilde{\Psi}_{\varepsilon,\Phi}\big) - [R_\varrho(\Phi)]_{m+1} \cdot \prod_{l=1}^{m} [R_\varrho(\Phi)]_l \right\|_{W^{k,\infty}((0,1)^d)} \\ &\quad \leq \left\| R_\varrho(\widetilde{\times}) \circ (R_\varrho(\Psi_{\varepsilon,\Phi_m}), [R_\varrho\big(\widetilde{\Psi}_{\varepsilon,\Phi}\big)]_2) - R_\varrho(\Psi_{\varepsilon,\Phi_m}) \cdot [R_\varrho\big(\widetilde{\Psi}_{\varepsilon,\Phi}\big)]_2 \right\|_{W^{k,\infty}((0,1)^d)} \\ &\qquad + \left\| R_\varrho(\Psi_{\varepsilon,\Phi_m}) \left([R_\varrho\big(\widetilde{\Psi}_{\varepsilon,\Phi}\big)]_2 - [R_\varrho(\Phi)]_{m+1} \right) \right\|_{W^{k,\infty}((0,1)^d)} \\ &\qquad\quad + \left\| [R_\varrho(\Phi)]_{m+1} \cdot \big(R_\varrho(\Psi_{\varepsilon,\Phi_m}) - \prod_{l=1}^{m} [R_\varrho(\Phi)]_l \big) \right\|_{W^{k,\infty}((0,1)^d)}. \end{aligned} \tag{A.37}$$

We continue by considering the first term of the Inequality (A.37) and bound the k-seminorm of this term. We apply the chain rule from Corollary A.13 for $g : \mathbb{R}^2 \to \mathbb{R}$ with

$g(x,y) = R_\varrho(\widetilde{\times})(x,y) - x \cdot y$ and $f : \mathbb{R}^d \to \mathbb{R}^2$ with $f = R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})$. We get

$$\begin{aligned}
&\left| R_\varrho(\widetilde{\times}) \circ (R_\varrho(\Psi_{\varepsilon,\Phi_m}), [R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})]_2) - R_\varrho(\Psi_{\varepsilon,\Phi_m}) \cdot [R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})]_2 \right|_{W^{k,\infty}((0,1)^d)} \\
&\quad \leq C \sum_{i=1}^{k} |R_\varrho(\widetilde{\times})(x,y) - x \cdot y|_{W^{i,\infty}((-B,B)^2;dxdy)} N^{k+\mu_{(k=2)}} \\
&\quad \leq Ck \cdot \|R_\varrho(\widetilde{\times})(x,y) - x \cdot y\|_{W^{j,\infty}((-B,B)^2;dxdy)} N^{k+\mu_{(k=2)}} \\
&\quad \leq C\varepsilon N^{k+\mu_{(k=2)}}, \qquad \text{(A.38)}
\end{aligned}$$

where we used the induction hypothesis together with $|[R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})]_2|_{W^{k,\infty}((0,1)^d)} \leq cN^{k+\mu_{(k=2)}}$ (which follows from the properties of the approximate identity network from Corollary A.19 together with the chain rule) in the third step and assumed that $c \leq C$. Combining the statements of the semi-norms then yields the required bound for the norm. For the second term we have by the product rule and the chain rule

$$\begin{aligned}
&\left\| R_\varrho(\Psi_{\varepsilon,\Phi_m}) \left([R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})]_2 - [R_\varrho(\Phi)]_{m+1} \right) \right\|_{W^{k,\infty}((0,1)^d)} \\
&\quad \leq \sum_{i=0}^{k} \|R_\varrho(\Psi_{\varepsilon,\Phi_m})\|_{W^{i,\infty}((0,1)^d)} \cdot \left\| [R_\varrho(\widetilde{\Psi}_{\varepsilon,\Phi})]_2 - [R_\varrho(\Phi)]_{m+1} \right\|_{W^{k-i,\infty}((0,1)^d)} \\
&\quad \leq \sum_{i=0}^{k} cN^{i+\mu_{(i=2)}} \cdot C\varepsilon N^{k-i+\mu_{(k-i=2)}} \leq kcCN^{k+\mu_{(k=2)}}\varepsilon. \qquad \text{(A.39)}
\end{aligned}$$

To estimate the last term of (A.37) we apply the product rule from Lemma A.12 and get

$$\begin{aligned}
&\left\| [R_\varrho(\Phi)]_{m+1} \cdot \left(R_\varrho(\Psi_{\varepsilon,\Phi_m}) - \prod_{l=1}^{m} [R_\varrho(\Phi)]_l \right) \right\|_{W^{k,\infty}((0,1)^d)} \\
&\quad \leq \sum_{i=0}^{k} \|[R_\varrho(\Phi)]_{m+1}\|_{W^{i,\infty}((0,1)^d)} \cdot \left\| R_\varrho(\Psi_{\varepsilon,\Phi_m}) - \prod_{l=1}^{m} [R_\varrho(\Phi)]_l \right\|_{W^{k-i,\infty}((0,1)^d)} \\
&\quad \leq \sum_{i=0}^{k} cN^{i+\mu_{(i=2)}} \cdot CN^{k-i+\mu_{(k-i=2)}}\varepsilon \leq kcCN^{k+\mu_{(k=2)}}\varepsilon. \qquad \text{(A.40)}
\end{aligned}$$

For the second step, we used again the induction hypothesis together with

$$|[R_\varrho(\Phi)]_{m+1}|_{W^{k,\infty}((0,1)^d)} \leq cN^{k+\mu_{(k=2)}}.$$

Combining (A.37) with (A.38), (A.39) and (A.40) yields

$$\left\| R_\varrho(\Psi_{\varepsilon,\Phi}) - \prod_{l=1}^{m+1} [R_\varrho(\Phi)]_l \right\|_{W^{k,\infty}((0,1)^d)} \leq CN^{k+\mu_{(k=2)}}\varepsilon.$$

ad (iii): The estimate

$$|R_\varrho(\Psi_{\varepsilon,\Phi})|_{W^{k,\infty}((0,1)^d)} \leq CN^{k+\mu_{(k=2)}}.$$

can be shown similarly as above.

ad (iv): Finally, we need to derive a bound for the absolute values of the weights. From the definition of $\Psi_{\varepsilon,\Phi}$ together with Lemma A.21 (iii) we get

$$\|\Psi_{\varepsilon,\Phi}\|_{\max} = \|\widetilde{\times} \bullet \widetilde{\Psi}_{\varepsilon,\Phi}\|_{\max} \leq C \cdot \max\{\varepsilon^{-2}, \|\widetilde{\Psi}_{\varepsilon,\Phi}\|_{\max}\}.$$

From $\|\widetilde{\Psi}_{\varepsilon,\Phi}\|_{\max} \leq C \max\{\|\Phi\|_{\max}, \varepsilon^{-2}\}$ (see Step 2) it follows that

$$\|\Psi_{\varepsilon,\Phi}\|_{\max} \leq C \max\{\|\Phi\|_{\max}, \varepsilon^{-2}\}.$$

This concludes the proof. ∎

In the last part of this subsection, we are finally in a position to construct neural networks which approximate sums of localized polynomials.

Lemma A.26 *Let $j, \tau \in \mathbb{N}_0$,, $d, N \in \mathbb{N}$, $k \in \{0, \dots, j\}$, Additionally, let ϱ be such that it fulfills the assumptions of Proposition 2.19 (for $n = 3$, $r = 2$). Let $n \in \mathbb{N}_{\geq k+1}$, $1 \leq p \leq \infty$, and $\mu > 0$. Assume that $\left(\Psi^{(j,\tau,N,s)}\right)_{N\in\mathbb{N},s\geq 1}$ be the exponential (polynomial, exact) (j,τ)-PU from Definition 2.13. For $N \in \mathbb{N}$, set*

$$s := \begin{cases} N^{\mu}, & \text{if exponential PU,} \\ N^{\frac{2d/p+d+n}{D}}, & \text{if polynomial PU,} \\ 1, & \text{if exact PU,} \end{cases}$$

Then, there is a constant $C = C(n,d,p,k) > 0$ with the following properties:

Let $\varepsilon \in (0, 1/2)$, $f \in W^{n,p}((0,1)^d)$ and $p_m(x) := p_{f,m}(x) = \sum_{|\alpha|\leq n-1} c_{f,m,\alpha} x^{\alpha}$ for $m \in \{0, \dots, N\}^d$ be the polynomials from Lemma A.22. Then there is a neural network $\Phi_{P,\varepsilon} = \Phi_{P,\varepsilon}(f,d,n,N,\mu,\varepsilon)$ with d-dimensional input and one-dimensional output, with at most C layers and $C(N+1)^d$ nonzero weights, such that

$$\left\| \sum_{m\in\{0,\dots,N\}^d} \phi_m^s p_m - R_\varrho(\Phi_{P,\varepsilon}) \right\|_{W^{k,p}((0,1)^d)} \leq C\|f\|_{W^{n,p}((0,1)^d)}\varepsilon,$$

and $\|\Phi_{P,\varepsilon}\|_{\max} \leq C\|f\|_{W^{n,p}((0,1)^d)}\varepsilon^{-2}s^2N^{2(d/p+d+k)+d/p+d}$.

***Proof*.** As before, we only provide the proof only for the case of an exponential (j,τ)-PU.

Step 1 (Approximating localized monomials $\phi_m^s(x)x^\alpha$): Let $|\alpha| \leq n-1$ and $m \in \{0, \dots, N\}^d$ and set $\widetilde{\varepsilon} := \varepsilon N^{-(d/p+d+k+\mu_{(k=2)})}$. By Corollary A.19 and inductively using the trick that $|xy - uz| \leq |x(y-z)| + |z(x-u)|$, there is a neural network Φ_α with d-dimensional input and $|\alpha|$-dimensional output, with two layers, at most $4(n-1)$ nonzero weights bounded in absolute value by $C\widetilde{\varepsilon}^{-1}$ such that

$$\left\| x^\alpha - \prod_{l=1}^{|\alpha|} [R_\varrho(\Phi_\alpha)]_l(x) \right\|_{W^{k,\infty}((0,1)^d;dx)} \leq C\widetilde{\varepsilon} \tag{A.41}$$

and

$$\|[R_\varrho(\Phi_\alpha)]_l\|_{W^{k,\infty}((0,1)^d)} \leq \widetilde{\varepsilon} + 1 \leq 2, \quad \text{for all } l = 1, \dots, |\alpha|. \tag{A.42}$$

Let now Φ_m be the neural network from Lemma 2.17 ((iv)) (for $s = N^\mu$) and define the network

$$\Phi_{m,\alpha} := \mathrm{P}(\Phi_m, \Phi_\alpha, \Phi_{n-1-|\alpha|,2}),$$

where the parallelization is provided by Lemma A.17 and

$$\Phi_{n-1-|\alpha|,2} = \left((0_{d,d}, 0_d), (0_{n-1-|\alpha|,d}, 1_{n-1-|\alpha|})\right).$$

Consequently, $\Phi_{m,\alpha}$ has $2 \leq K_0$ layers and $C + 4(n-1) \leq K_0$ nonzero weights for a suitable constant $K_0 = K_0(n,d) \in \mathbb{N}$, $\|\Phi_{m,\alpha}\|_{\max} \leq C \max\{\widetilde{\varepsilon}^{-1}, N^{1+\mu}\}$ and

$$\| \prod_{l=1}^{n-1+d} [R_\varrho(\Phi_{m,\alpha})(x)]_l - \phi_m^s(x) x^\alpha \|_{W^{k,\infty}((0,1)^d);dx} \leq C\widetilde{\varepsilon}.$$

Moreover, as a consequence of Lemma 2.17 ((iv)) together with Equation (A.42) we have

$$\|[R_\varrho(\Phi_{m,\alpha})]_l\|_{W^{k,\infty}((0,1)^d)} \leq C N^{k+\mu_{(k=2)}}, \quad \text{for all } l = 1, \dots, n-1+d.$$

To construct an approximation of the localized monomials $\phi_m^s(x)x^\alpha$, let $\Psi_{\widetilde{\varepsilon},(m,\alpha)}$ be the neural network provided by Lemma A.25 (with $\Phi_{m,\alpha}$ instead of Φ, $m = |\alpha| + d \in \mathbb{N}$, $K = K_0 \in \mathbb{N}$) for $m \in \{0, \dots, N\}^d$ and $\alpha \in \mathbb{N}_0^d, |\alpha| \leq n-1$. Then $\Psi_{\widetilde{\varepsilon},(m,\alpha)}$ has at most C layers (independently of m, α), number of nonzero weights and $\|\Psi_{\widetilde{\varepsilon},(m,\alpha)}\|_{\max} \leq C \max\{N^{1+\mu}, \varepsilon^{-2} N^{2(d/p+d+k+\mu_{(k=2)})}\}$. Moreover,

$$\begin{aligned}
&\left\| \phi_m^s(x)x^\alpha - R_\varrho\left(\Psi_{\widetilde{\varepsilon},(m,\alpha)}\right)(x) \right\|_{W^{k,\infty}((0,1)^d;dx)} \\
&\leq \left\| \phi_m^s(x)x^\alpha - \prod_{l=1}^{n-1+d} [R_\varrho(\Phi_{m,\alpha})(x)]_l \right\|_{W^{k,\infty}((0,1)^d;dx)} \\
&\quad + \| \prod_{l=1}^{n-1+d} [R_\varrho(\Phi_{m,\alpha})]_l - R_\varrho\left(\Psi_{\widetilde{\varepsilon},(m,\alpha)}\right) \|_{W^{k,\infty}((0,1)^d)} \\
&\leq C N^{k+\mu_{(k=2)}} \widetilde{\varepsilon} \leq C\varepsilon N^{-d/p-d},
\end{aligned}$$

where we used Equation (A.41) together with the product rule for the last step.

Step 2 (Constructing $\Phi_{P,\varepsilon}$): We set

$$T := |\{(m,\alpha) : m \in \{0, \dots, N\}^d, \alpha \in \mathbb{N}_0^d, |\alpha| \leq n-1\}|.$$

We note that every network $\Psi_{\widetilde{\varepsilon},(m,\alpha)}$ has the same number of layers. By using Lemma A.17, we parallelize the localized polynomial approximations

$$\mathrm{P}\left(\Psi_{\widetilde{\varepsilon},(m,\alpha)} : m \in \{0, \dots, N\}^d, \alpha \in \mathbb{N}_0^d, \ |\alpha| \leq n-1\right)$$

and note that the resulting network has at most C layers and CT nonzero weights bounded in absolute value by $C \max\{N^{1+\mu}, \varepsilon^{-2} N^{2(d/p+d+k+\mu_{(k=2)})}\} \leq C\varepsilon^{-2} N^{2(d/p+d+k+\mu_{(k=2)})}$. Next, we define the matrix $A_{\mathrm{sum}} \in \mathbb{R}^{1,T}$ by $A_{\mathrm{sum}} := [c_{f,m,\alpha} : m \in \{0, \dots, N\}^d, \alpha \in \mathbb{N}_0^d, |\alpha| \leq$

$n-1]$ and the neural network $\Phi_{\text{sum}} := ((A_{\text{sum}}, 0))$. Finally, we set

$$\Phi_{P,\varepsilon} := \Phi_{\text{sum}} \bullet \mathrm{P}\big(\Psi_{\widetilde{\varepsilon},(m,\alpha)} : m \in \{0,\dots,N\}^d, \alpha \in \mathbb{N}_0^d, |\alpha| \leq n-1\big). \tag{A.43}$$

From Lemma A.21((i)) we get $\Phi_{P,\varepsilon}$ is a neural network with d-dimensional input and one-dimensional output, with at most C layers and, by Lemma A.21, $CT \leq C(N+1)^d$ nonzero weights. For the absolute values of the weights it holds that

$$\begin{aligned}\|\Phi_{P,\varepsilon}\|_{\max} &\leq (N+1)^d C \|f\|_{W^{n,p}((0,1)^d)} N^{d/p} \varepsilon^{-2} N^{2(d/p+d+k+\mu_{(k=2)})} \\ &\leq C\|f\|_{W^{n,p}((0,1)^d)} \varepsilon^{-2} N^{2(d/p+d+k+\mu_{(k=2)})+d/p+d}\end{aligned}$$

where we used the bound for the coefficients $c_{f,m,\alpha}$ from Remark A.23. Moreover, we have

$$R_\varrho(\Phi_{P,\varepsilon}) = \sum_{m\in\{0,\dots,N\}^d} \sum_{|\alpha|\leq n-1} c_{f,m,\alpha} R_\varrho(\Psi_{\widetilde{\varepsilon},(m,\alpha)}).$$

Note that the network $\Phi_{P,\varepsilon}$ only depends on $p_{f,m}$ (and thus on f) via the coefficients $c_{f,m,\alpha}$.

Step 3 (Estimating the approximation error in $\|\cdot\|_{W^{k,p}}$): We get

$$\begin{aligned}&\left\| \sum_{m\in\{0,\dots,N\}^d} \phi_m^s(x) p_m(x) - R_\varrho(\Phi_{P,\varepsilon})(x) \right\|_{W^{k,p}((0,1)^d;dx)} \\ &= \left\| \sum_{m\in\{0,\dots,N\}^d} \sum_{|\alpha|\leq n-1} c_{f,m,\alpha} \Big(\phi_m^s(x) x^\alpha - R_\varrho\big(\Psi_{\widetilde{\varepsilon},(m,\alpha)}\big)(x) \Big) \right\|_{W^{k,p}((0,1)^d;dx)} \\ &\leq \sum_{m\in\{0,\dots,N\}^d} \sum_{|\alpha|\leq n-1} |c_{f,m,\alpha}| \left\| \phi_m^s(x) x^\alpha - R_\varrho\big(\Psi_{\widetilde{\varepsilon},(m,\alpha)}\big)(x) \right\|_{W^{k,p}((0,1)^d;dx)} \\ &\leq \sum_{m\in\{0,\dots,N\}^d} \sum_{|\alpha|\leq n-1} \|\widetilde{f}\|_{W^{n-1,p}(\Omega_{m,N})} N^{d/p} C\varepsilon N^{-d/p-d},\end{aligned}$$

where we used again the bound for the coefficients $c_{f,m,\alpha}$ together with $\|\cdot\|_{W^{k,p}((0,1)^d)} \leq C\|\cdot\|_{W^{k,\infty}((0,1)^d)}$ in the last step. Similar as in Equation (A.34) we finally have

$$\begin{aligned}\left\| \sum_{m\in\{0,\dots,N\}^d} \phi_m^s(x) p_m(x) - R_\varrho(\Phi_{P,\varepsilon})(x) \right\|_{W^{k,p}((0,1)^d;dx)} &\leq C\varepsilon N^{-d} \sum_{m\in\{0,\dots,N\}^d} \|\widetilde{f}\|_{W^{n,p}((0,1)^d)} \\ &\leq C\varepsilon \|f\|_{W^{n,p}((0,1)^d)}.\end{aligned}$$

This concludes the proof. ∎

A.5.4 Putting Everything Together

Now we conclude the proof of Proposition 2.21. Again, we only provide the proof for exponential (j,τ)-PUs. The rest follows in a similar manner by adapting the calculations to come accordingly.

Proof of Proposition 2.21 . We divide the proof into two steps: First, we approximate the function f by a sum of localized polynomials. Afterwards, we proceed by approximating this sum by a neural network.

For the first step, we set

$$N := \left\lceil \left(\frac{\varepsilon}{2\widetilde{C}}\right)^{-1/(n-k-\mu_{(k=2)})} \right\rceil \quad \text{and} \quad s := N^{\mu}, \tag{A.44}$$

where $\widetilde{C} = \widetilde{C}(n,d,p,k) > 0$ is the constant from Lemma A.22. Without loss of generality we may assume that $\widetilde{C} \geq 1$. The same lemma yields that if $\Psi^{(j,\tau)} = \Psi^{(j,\tau)}(d,N,\mu) = \{\phi_m^s : m \in \{0,\dots,N\}^d\}$ is the PU from Lemma 2.17 and $\widetilde{N} = \widetilde{N}(d,p,\mu,k)$ is the constant from Lemma A.22, then there exist polynomials $p_m(x) = \sum_{|\alpha|\leq n-1} c_{f,m,\alpha} x^{\alpha}$ for $m \in \{0,\dots,N\}^d$ such that

$$\left\| f - \sum_{m\in\{0,\dots,N\}^d} \phi_m^s p_m \right\|_{W^{k,p}((0,1)^d)} \leq \widetilde{C}\left(\frac{1}{N}\right)^{n-k-\mu_{(k=2)}} \leq \widetilde{C}\frac{\varepsilon}{2\widetilde{C}} = \frac{\varepsilon}{2}, \tag{A.45}$$

for all $\varepsilon \in (0,\widetilde{\varepsilon})$, where $\widetilde{\varepsilon} = \widetilde{\varepsilon}(d,p,\mu,k) > 0$ is chosen such that $N \geq \widetilde{N}$.

For the second step, let $\widetilde{C}' = \widetilde{C}'(n,d,p,k)$ be the constant from Lemma A.26 and $\Phi_{P,\varepsilon}$ be the neural network provided by Lemma A.26 with $\varepsilon/(2\widetilde{C}')$ instead of ε. Then $\Phi_{P,\varepsilon}$ has at most $\widetilde{C}'$ layers and at most

$$\widetilde{C}'\left(\left(\frac{\varepsilon}{2\widetilde{C}'}\right)^{-1/(n-k-\mu_{(k=2)})} + 2\right)^d \leq \widetilde{C}' 3^d \left(\frac{\varepsilon}{2\widetilde{C}'}\right)^{-d/(n-k-\mu_{(k=2)})} \leq C\varepsilon^{-d/(n-k-\mu_{(k=2)})}$$

nonzero weights. In the first step we have used $(2\widetilde{C}')/\varepsilon \geq 1$. The weights are bounded in absolute value by

$$\begin{aligned}\|\Phi_{P,\varepsilon}\|_{\max} &\leq \widetilde{C}'\varepsilon^{-2} N^{2(d/p+dk+\mu_{(k=2)})+d/p+d} \\ &\leq C\varepsilon^{-2-(2(d/p+d+k+\mu_{(k=2)})+d/p+d)/(n-k-\mu_{(k=2)})} = C\varepsilon^{-\theta},\end{aligned}$$

for a suitable $\theta = \theta(d,p,k,n,\mu) > 0$. Additionally, there holds

$$\left\| \sum_{m\in\{0,\dots,N\}^d} \phi_m^s p_m - R_\varrho(\Phi_{P,\varepsilon}) \right\|_{W^{k,p}((0,1)^d)} \leq \widetilde{C}'\frac{\varepsilon}{2\widetilde{C}'} \leq \frac{\varepsilon}{2}. \tag{A.46}$$

By applying the triangle inequality as well as Equations (A.45) and (A.46) we arrive at

$$\begin{aligned}&\left\| f - R_\varrho(\Phi_{P,\varepsilon}) \right\|_{W^{k,p}((0,1)^d)} \\ &\leq \left\| f - \sum_{m\in\{0,\dots,N\}^d} \phi_m^s p_m \right\|_{W^{k,p}((0,1)^d)} + \left\| \sum_{m\in\{0,\dots,N\}^d} \phi_m^s p_m - R_\varrho(\Phi_{P,\varepsilon}) \right\|_{W^{k,p}((0,1)^d)} \\ &\leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon,\end{aligned}$$

thereby concluding the proof. ∎

A.6 Proof of Theorem 2.22 (Encodability of the Weights)

We now proceed with the proof of Theorem 2.22.

Proof of Theorem 2.22 . Let $C = C(d,n,p,\mu,k) > 0$, $\theta = \theta(d,n,p,k,\mu) > 0$ and $\widetilde{\varepsilon} = \widetilde{\varepsilon}(d,p,\mu,k) > 0$ be the constants from Proposition 2.21 and let $\varepsilon \in (0, \min\{1/3, \widetilde{\varepsilon}\})$. Moreover, for $f \in \mathcal{F}_{n,d,p}$, let

$$\Phi_{\varepsilon,f} := ((A_{\text{sum}}, 0)) \bullet \mathrm{P}\big(\Psi_i : i = 1, \dots, T\big)$$

be the neural network from Proposition 2.21 (defined in Equation (A.43)) with at most L layers and $M(\Phi_{\varepsilon,f}) \leq C \cdot \varepsilon^{-d/(n-k-\mu_{(k=2)})}$ nonzero weights bounded in absolute value by $C\varepsilon^{-\theta}$, such that

$$\|R_\varrho(\Phi_{\varepsilon,f}) - f\|_{W^{k,p}((0,1)^d)} \leq \frac{\varepsilon}{3}.$$

We will make use of the following additional properties of $\Phi_{\varepsilon,f}$:

(i) Only the entries of A_{sum} depend on the function f. In other words, the entries of $\Psi_1, \dots, \Psi_T$ are independent from f. They only depend on $\varepsilon, n, d, p, k, \mu$.

(ii) There exists $s = s(k,n,d,p) > 0$ (we assume w.l.o.g. that the same s can be used) such that

(a) $\big\|R_\varrho(\Psi_i)\big\|_{W^{k,\infty}((0,1)^d)} \leq \varepsilon^{-s}$ for $i = 1, \dots, T$. This follows from Lemma A.25 ((iii)) in combination with Step 1 and 2 of the proof of Lemma A.26 and choice of N in Equation (A.44).

(b) $T \leq \varepsilon^{-s}$. This follows from the definition of T (see Step 2 of the proof of Lemma A.26);

(c) $M(\Phi_{\varepsilon,f}) \leq \varepsilon^{-s}$.

(iii) $A_{\text{sum}} = (a_m)_{m=1}^T \in \mathbb{R}^{1,T}$.

(iv) The last layer $(A_{\text{last}}, b_{\text{last}})$ of $\mathrm{P}\big(\Psi_i : i = 1, \dots, T\big)$ has a block diagonal structure, where each block is a vector (see also Lemma A.17). Thus, in every column of A_{last} there is at most one nonzero entry.

We replace the weights in the last layer of $\Phi_{\varepsilon,f}$ by elements from an appropriate set of weights with cardinality bounded polynomially in ε^{-1} and show that the resulting network is still close enough to f. Afterwards, we construct a coding scheme for the entire set of weights.

Step 1 (Rounding the weights in A_{sum}): We now show that with rounding precision $\nu := 2s + 2$ we have for the neural network

$$\widetilde{\Phi}^{(1)}_{\varepsilon,f} := ((\widetilde{A}_{\text{sum}}, 0)) \bullet \mathrm{P}\big(\Psi_i : i = 1, \dots, T\big)$$

where $\widetilde{A}_{\text{sum}} \in ([-\varepsilon^{-\theta}, \varepsilon^{-\theta}] \cap \varepsilon^\nu \mathbb{Z})^{1,T}$ is the rounded weight matrix $A_{\text{sum}} \in \mathbb{R}^{1,T}$ that

$$\|R_\varrho(\Phi_{\varepsilon,f}) - R_\varrho(\widetilde{\Phi}^{(1)}_{\varepsilon,f})\|_{W^{k,p}((0,1)^d)} \leq \varepsilon/3.$$

Clearly,

$$\begin{aligned}
&\Big\|R_\varrho\big((A_{\mathrm{sum}},0)\bullet \mathrm{P}\big(\Psi_i : i=1,\dots,T\big)\big) - R_\varrho\big((\widetilde{A}_{\mathrm{sum}},0)\bullet \mathrm{P}\big(\Psi_i : i=1,\dots,T\big)\big)\Big\|_{W^{k,p}((0,1)^d)} \\
&\quad\le \left\|\sum_{i=1}^T a_i R_\varrho(\Psi_i) - \sum_{i=1}^T \widetilde{a}_i R_\varrho(\Psi_i)\right\|_{W^{k,\infty}((0,1)^d)} \\
&\quad\le \sum_{i=1}^T |a_i - \widetilde{a}_i| \big\|R_\varrho(\Psi_i)\big\|_{W^{k,\infty}((0,1)^d)} \\
&\quad\le \sum_{i=1}^T \varepsilon^\nu \big\|R_\varrho(\Psi_i)\big\|_{W^{k,\infty}((0,1)^d)} \\
&\quad\le \varepsilon^\nu \varepsilon^{-s}\varepsilon^{-s} \le \varepsilon^2 \le \varepsilon/3,
\end{aligned}$$

where we use in the third step that the rounding precision is ε^ν and in the fourth the properties (ii) and (iii) from above. To get our final network, we replace the bias term $\widetilde{A}_{\mathrm{sum}} b_{\mathrm{last}}$ (which is also bounded in absolute value by $\varepsilon^{-\theta}$) in the last layer of $\widetilde{\Phi}^{(1)}_{\varepsilon,f}$ by the nearest element in $[-\varepsilon^{-\theta},\varepsilon^{-\theta}]\cap\varepsilon^\nu\mathbb{Z}$ and denote the resulting network by $\widetilde{\Phi}_{\varepsilon,f}$. It now easily follows that

$$\|R_\varrho(\widetilde{\Phi}^{(1)}_{\varepsilon,f}) - R_\varrho(\widetilde{\Phi}_{\varepsilon,f})\|_{W^{k,\infty}((0,1)^d)} \le \varepsilon/3$$

which implies by the triangle inequality that

$$\|f - R_\varrho(\widetilde{\Phi}_{\varepsilon,f})\|_{W^{k,p}((0,1)^d)} \le \varepsilon.$$

Step 2 (Construction of coding scheme): We will now show that there is a constant $C_2 = C_2(d,n,p,k,\mu) > 0$ and a coding scheme $\mathcal{B} = (B_\ell)_{\ell\in\mathbb{N}}$ such that for each $\varepsilon > 0$ and each $f \in \mathcal{F}_{n,d,p}$ the nonzero weights of $\widetilde{\Phi}_{\varepsilon,f}$ are in Range $B_{\lceil C_2\log(1/\varepsilon)\rceil}$.

If we denote by W_ε the collection of nonzero weights of $(\Psi_m)_{m=1}^T$ (which are independent of f), then we have $|W_\varepsilon| \le M(\Phi_{\varepsilon,f}) \le \varepsilon^{-s}$. Furthermore, we have $|[-\varepsilon^{-\theta},\varepsilon^{-\theta}]\cap\varepsilon^\nu\mathbb{Z}| = 2\lfloor\varepsilon^{-\theta-\nu}\rfloor + 1 \le \varepsilon^{-s_2}$ with $s_2 := \theta+\nu+2$.

- The matrix weights in the last layer of $\widetilde{\Phi}_{\varepsilon,f}$ are in the set $G_{\mathrm{mult}} := \{x_1x_2 : x_1 \in W_\varepsilon, x_2 \in [-\varepsilon^{-\theta},\varepsilon^{-\theta}]\cap\varepsilon^\nu\mathbb{Z}\}$ with cardinality bounded by $\varepsilon^{-(s+s_2)}$.
- The bias in the last layer is an element of $[-\varepsilon^{-\theta},\varepsilon^{-\theta}]\cap\varepsilon^\nu\mathbb{Z}$.
- The weights of $\widetilde{\Phi}_{\varepsilon,f}$ in the layers $1,\dots,L-1$ are in the set W_ε.

Setting $C_2 := 2(s+s_2)$ it follows that there exists a surjective mapping

$$B_{\lceil C_2\log_2(1/\varepsilon)\rceil} : \{0,1\}^{\lceil C_2\log_2(1/\varepsilon)\rceil} \to G_{\mathrm{mult}} \cup W_\varepsilon \cup \Big([-\varepsilon^{-\theta},\varepsilon^{-\theta}]\cap\varepsilon^\nu\mathbb{Z}\Big),$$

which shows the claim. ∎

A.7 PU-properties of the Activation Functions from Table 2.1

In this section, we examine the PU-properties of the activation functions listed in Table 2.1. The smoothness properties of all functions in Table 2.1 are clear. In particular, all functions

are in $C^\infty(\mathbb{R} \setminus \{0\})$. In order to show that the activation functions to follow allow for exponential (polynomial) PUs, we consider the exponential (polynomial) (j, τ) admissibility conditions of Definition 2.14.

Exact PUs.

(leaky) ReLU and RePUs: These functions admit exact PUs. For the ReLU case, see for instance [Yar17; GKP20]. For RePUs, this follows from the properties of B-splines (see [De 01, Chapter IX]).

Exponential PUs.

ELU$_a$ for $a > 0, a \neq 1$: Here, $j = 1$, $\tau = 1$, $A = 0$ and $B = 1$. Moreover, $R > 0$ can be chosen arbitrarily. Then, for $D = 1$, we have, for all $x > R$, that $|1 - \varrho'(x)| = |1 - 1| = 0$ and, for all $x < -R$ that $|\varrho'(x)| = |ae^x| = ae^{Dx}$.

ELU$_1$: Here, $j = 2$, $\tau = 1$, $A = 0$ and $B = 1$. Moreover, $R > 0$ can be chosen arbitrarily. Then, for $D = 1$, we have, for all $x > R$, that $|1 - \varrho'(x)| = |1 - 1| = 0$ and, for all $x < -R$ that $|\varrho'(x)| = |e^x| = e^{Dx}$. Moreover, we have for all $|x| > R$ that $|\varrho''(x)| \leq e^{-|x|} = e^{-D|x|}$.

sigmoid: Here, $j \in \mathbb{N}_0$ is arbitrary, $\tau = 0$, $A = 0$ and $B = 1$. Moreover, $R > 0$ can be chosen arbitrarily. Then we have, for all $x > R$, that $|1 - \varrho(x)| \leq e^{-x}$ and, for all $x < -R$ that $|\varrho(x)| \leq e^x$. The other statements follow from the fact that, for the sigmoid activation function, the k-th derivative is a finite linear combination of the powers $\varrho, \dots, \varrho^k$ of ϱ (see, e.g., [MW93]). Choosing D suitably then shows the claim.

tanh: Since $\tanh(x) = 2 \cdot \mathrm{sigmoid}(2x) - 1$, the proof of this statement follows from the proof of the statement for the sigmoid activation function for $A = -1$, $B = 1$.

softplus: Here, $j \in \mathbb{N}_0$ is arbitrary, $\tau = 1$, $A = 0$ and $B = 1$. Moreover, $R > 0$ can be chosen arbitrarily. Then, for all $x > R$, there holds $|1 - \varrho'(x)| = |1 - \mathrm{sigmoid}(x)| \leq e^{-x}$ and, for all $x < -R$ that $|\varrho'(x)| = |\mathrm{sigmoid}(x)| \leq e^x$. The proof of (d.3) for the higher-order derivatives follows from the properties of the higher derivatives of the sigmoid function.

swish: Here, $j \in \mathbb{N}_0$ is arbitrary, $\tau = 1$, $A = 0$, and $B = 1$. It is not hard to see that for all $k \in \mathbb{N}$ there holds

$$\mathrm{swish}^{(k)}(x) = x \cdot \mathrm{sigmoid}^{(k)}(x) + k \cdot \mathrm{sigmoid}^{(k-1)}(x).$$

Now, the statement follows from the analogous observations for the sigmoid function combined with the fact that for $r, u > 0$ with $r > u$ there holds

$$\lim_{x \to \infty} \frac{xe^{-rx}}{e^{-ux}} = 0, \qquad \lim_{x \to -\infty} \frac{xe^{rx}}{e^{ux}} = 0.$$

Polynomial PUs.

softsign: Here, $j \in \mathbb{N}_0$ is arbitrary, $\tau = 0$, $A = -1$, $B = 1$. The polynomial convergence properties (d.1)-(d.3) follow immediately from the definition of the softsign function.

inverse square root linear unit: Here, $j = 3$, $\tau = 1$, $A = 0$ and $B = 1$. The polynomial convergence properties (d.1)-(d.3) follow immediately from the definition of the inverse square root linear unit.

inverse square root unit: Here, $j \in \mathbb{N}_0$ is arbitrary, $\tau = 0$, $A = -1$ and $B = 1$. The polynomial convergence properties (d.1)-(d.3) follow immediately from the definition of the inverse square root unit.

arctan: Here, $j \in \mathbb{N}_0$ is arbitrary, $\tau = 0$, $A = -\pi/2$ and $B = \pi/2$. The polynomial convergence properties (d.1)-(d.3) follow immediately from the fact that $\varrho'(x) = 1/(1+x^2)$ which in particular implies polynomial convergence behavior for arctan itself.

B

Exact AAPM Challenge Setup

To ensure reproducability, we give an exact account of how we trained our winning submission to the AAPM challenge (team-name: `robust-and-stable`). Since the systematic investigation of the ItNet-architecture was conducted after the challenge submission phase, it became clear that not all of the substeps outlined below have a notable impact on the performance (see also Section 3.3).

The following details are related to Step 3 of Section 3.2 ("Constructing an Iterative Scheme"). We start by training an ItNet_4 (with weight sharing) for 500 epochs of mini-batch stochastic gradient descent and Adam with an initial learning rate of $8 \cdot 10^{-5}$ and a batch size of 2 (restarting Adam after 250 epochs). Then, we improve the accuracy by the following *post-training* strategy: First, the ItNet_4 is extended by one more iteration:

$$\text{ItNet-post}[\theta] \colon \mathbb{R}^m \to \mathbb{R}^N,$$

$$y \mapsto \left[\bigcirc_{k=1}^{5} \left(\mathcal{DC}_{\lambda_k, y} \circ \text{UNet}[\tilde{\theta}_k]\right) \circ \texttt{FBP}\right](y),$$

where $\tilde{\theta}_k$ is initialized with the optimized weights from ItNet_4 for $k = 1, \dots, 4$, and we set $\tilde{\theta}_5 := \tilde{\theta}_4$. Next, ItNet-post is fine-tuned by keeping the weights $\tilde{\theta}_1 = \tilde{\theta}_2 = \tilde{\theta}_3$ of the first three UNet-blocks fixed and optimizing only over the weights of the last two iterations (without weight sharing).

Aiming at an additional training speed-up, we use the initialization $\lambda = [1.1, 1.3, 1.4, 0.08]$ for the data-consistency parameters of ItNet_4, which was found by pre-training. Similarly, ItNet-post is initialized with the optimized values from ItNet_4 for $k = 1, 2, 3$, together with $\lambda_4 = 1.0$ and $\lambda_5 = 0.1$.

To improve the overall performance of our networks, we have additionally applied the following "tricks" for *fine tuning*, which are ordered by their importance:

(i) Due to statistical fluctuations, the networks typically exhibit slightly different reconstruction errors, despite using the same training pipeline. The final reconstructions are therefore computed by an *ensemble* of ten networks, each trained on a different split of the training set.

(ii) Due to the training with small batch sizes, we replace batch normalization of the UNet-architecture by *group normalization* [WH18].

(iii) We equip the UNet-blocks with a few *memory channels*, i.e., one actually has that $\text{UNet}[\theta] \colon \mathbb{R}^N \times (\mathbb{R}^N)^{c_{\text{mem}}} \to \mathbb{R}^N \times (\mathbb{R}^N)^{c_{\text{mem}}}$; cf. Putzky and Welling [PW17] and Adler and Öktem [AÖ18]. While the original image-enhancement channel is not altered, the output of the additional channels is propagated through the ItNet, playing the role of a hidden state (in the spirit of recurrent neural networks). For our experiments, we use $c_{\text{mem}} = 5$.

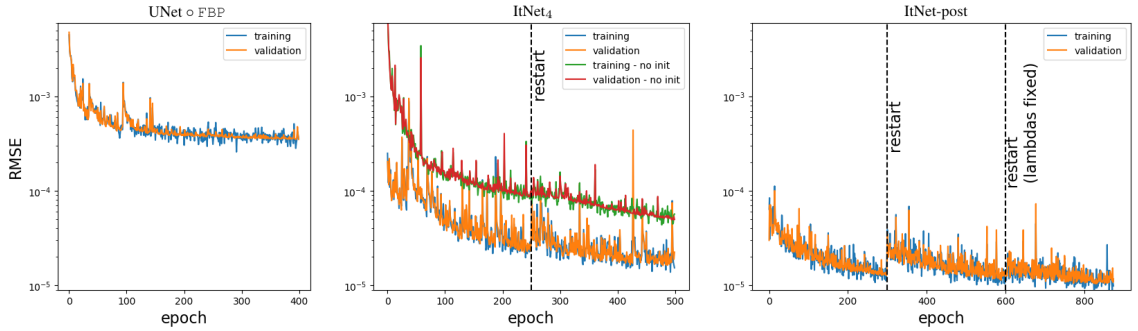


Figure B.1: **Loss curves and network training.** The first two plots demonstrate that ItNet_4 improves the RMSE by approximately an order of magnitude in comparison to a post-processing by UNet. Furthermore, the gain of our UNet-initialization strategy can be seen in the second graph. The last two plots illustrate the advantages of restarting and of the post-training strategy, respectively. Note that we display the RMSE on the training and validation sets instead of the actual ℓ^2-losses, which behave similarly.

(iv) It was beneficial to restart occasionally the training of the networks (see also Fig. B.1).

The following modifications did not lead to a gain in performance and were omitted:

(i) Improving `FBP` in Step 1 of Section 3.2 by making some of it components learnable (e.g., the filter), cf. Würfl et al. [WGCM16]. Although this is advantageous for the reconstruction quality of `FBP` itself, it leads to worse results for the ItNet. This suggests that a combination of model- and data-based methods benefits most from precise and unaltered physical models.

(ii) Adding additional convolutional-blocks in the measurement domain of ItNet.

(iii) Modifying the standard ℓ^2-loss by incorporating the RMSE or the ℓ^1-norm.

(iv) Utilizing different optimizers such as SGD, RAdam [Liu+20], or AdamW [LH19].

In Fig. B.1, we visualize the RMSE loss curves of our training pipeline, i.e.,

$$\text{UNet} \circ \texttt{FBP} \rightarrow \text{ItNet}_4(+\text{restart}) \rightarrow \text{ItNet-post}(+2 \times \text{restart}).$$

Bibliography

[Ada75] R. Adams. *Sobolev Spaces*. New York: Academic Press, 1975 (cit. on pp. 1, 10, 58, 59).

[AD21] B. Adcock and N. Dexter. „The Gap between Theory and Practice in Function Approximation with Deep Neural Networks“. *SIAM J. Math. Data Sci.* 3.2 (2021), 624–655 (cit. on pp. 34, 54, 55).

[AÖ18] J. Adler and O. Öktem. „Learned Primal-Dual Reconstruction“. *IEEE Trans. Med. Imag.* 37.6 (2018), 1322–1332 (cit. on pp. 37, 42, 43, 49, 99).

[AMJ18] H. K. Aggarwal, M. P. Mani, and M. Jacob. „MoDL: Model-Based Deep Learning Architecture for Inverse Problems“. *IEEE Trans. Med. Imag.* 38.2 (2018), 394–405 (cit. on pp. 37, 45).

[AJ08] H. Amann and E. Joachim. *Analysis III*. second. Grundstudium Mathematik. Basel: Birkhäuser, 2008 (cit. on p. 60).

[AB09] M. Anthony and P. Bartlett. *Neural Network Learning: Theoretical Foundations*. first. Cambridge: Cambridge University Press, 2009 (cit. on pp. 17, 66, 69).

[ARPAH20] V. Antun, F. Renna, C. Poon, B. Adcock, and A. C. Hansen. „On instabilities of deep learning in image reconstruction and the potential costs of AI“. *Proc. Natl. Acad. Sci.* 117.48 (2020), 30088–30095 (cit. on pp. 1, 50).

[AMÖS19] S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb. „Solving inverse problems using data-driven models“. *Acta Numer.* 28 (2019), 1–174 (cit. on pp. 1, 35).

[Asa11] A. Asanov. *Regularization, Uniqueness and Existence of Solutions of Volterra Equations of the First Kind*. Berlin: De Gruyter, 2011 (cit. on p. 1).

[Bal18] P. Baldi. „Deep Learning in Biomedical Data Science“. *Annu. Rev. Biomed. Data Sci.* 1.1 (2018), 181–205 (cit. on p. 1).

[Bar94] A. Barron. „Approximation and Estimation Bounds for Artificial Neural Networks“. *Mach. Learn.* 14.1 (1994), 115–133 (cit. on p. 9).

[BMR21] P. L. Bartlett, A. Montanari, and A. Rakhlin. „Deep learning: a statistical viewpoint“. *Acta Numer.* 30 (2021), 87–201 (cit. on pp. 2, 7, 54, 56).

[BHJK20] C. Beck, M. Hutzenthaler, A. Jentzen, and B. Kuckuck. „An overview on deep learning-based approximation methods for partial differential equations“. *arXiv preprint arXiv:2012.12348* (2020) (cit. on pp. 1, 10).

[Bel12] Y. Y. Belov. *Inverse Problems for Partial Differential Equations*. Berlin: De Gruyter, 2012 (cit. on p. 1).

[BCV13] Y. Bengio, A. Courville, and P. Vincent. „Representation Learning: A Review and New Perspectives“. *IEEE Trans. Pattern Anal. Mach. Intell.* 35.8 (2013), 1798–1828 (cit. on p. 33).

[BGKP21] J. Berner, P. Grohs, G. Kutyniok, and P. Petersen. „The modern mathematics of deep learning“. *arXiv preprint arXiv:2105.04026* (2021) (cit. on pp. 7, 56).

[BGV22] J. Berner, P. Grohs, and F. Voigtlaender. „Training ReLU networks to high uniform accuracy is intractable“. *arXiv preprint arXiv:2205.13531* (2022) (cit. on p. 55).

[BGKP19] H. Bölcskei, P. Grohs, G. Kutyniok, and P. Petersen. „Optimal approximation with sparsely connected deep neural networks“. *SIAM J. Math. Data Sci.* 1.1 (2019), 8–45 (cit. on pp. 2, 9, 10, 12, 14, 54).

[BS08] S. Brenner and R. Scott. *The Mathematical Theory of Finite Element Methods*. 3rd. Vol. 15. Texts in Applied Mathematics. New York: Springer Science+Business Media, 2008 (cit. on pp. 59–61).

[Bre12] A. Bressan. *Lecture Notes on Functional Analysis: With Applications to Linear Partial Differential Equations*. Vol. 143. Graduate Studies in Mathematics 143. Providence: American Mathematical Society, 2012 (cit. on p. 63).

[Bub+19] T. A. Bubba, G. Kutyniok, M. Lassas, M. März, W. Samek, S. Siltanen, and V. Srinivasan. „Learning the invisible: A hybrid deep learning-shearlet framework for limited angle computed tomography“. *Inverse Probl.* 35.6 (2019), 064002 (cit. on p. 43).

[Buz11] T. M. Buzug. „Computed Tomography“. *Springer handbook of medical technology*. Berlin: Springer, 2011, 311–342 (cit. on p. 3).

[CRT06] E. J. Candès, J. K. Romberg, and T. Tao. „Robust Uncertainty Principles: Exact Signal Reconstruction From Highly Incomplete Frequency Information“. *IEEE Trans. Inf. Theory* 52.2 (2006), 489–509 (cit. on p. 3).

[CD89] Carroll and Dickinson. „Construction of neural nets using the radon transform“. *International 1989 Joint Conference on Neural Networks*. Vol. 1. 1989, 607–611 (cit. on p. 56).

[Che+18] H. Chen, Y. Zhang, Y. Chen, J. Zhang, W. Zhang, H. Sun, Y. Lv, P. Liao, J. Zhou, and G. Wang. „LEARN: Learned Experts' Assessment-Based Reconstruction Network for Sparse-Data CT“. *IEEE Trans. Med. Imag.* 37.6 (2018), 1333–1347 (cit. on p. 37).

[Che+17a] H. Chen, Y. Zhang, M. K. Kalra, F. Lin, Y. Chen, P. Liao, J. Zhou, and G. Wang. „Low-Dose CT With a Residual Encoder-Decoder Convolutional Neural Network“. *IEEE Trans. Med. Imag.* 36.12 (2017), 2524–2535 (cit. on p. 41).

[Che+17b] H. Chen, Y. Zhang, W. Zhang, P. Liao, K. Li, J. Zhou, and G. Wang. „Low-dose CT via convolutional neural network“. *Biomed. Opt. Express* 8.2 (2017), 679–694 (cit. on p. 41).

[CC95] T. Chen and H. Chen. „Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems“. *IEEE Trans. Neural Netw.* 6.4 (1995), 911–917 (cit. on pp. 2, 56).

[CWZZ18] Y. Cheng, D. Wang, P. Zhou, and T. Zhang. „Model compression and acceleration for deep neural networks: The principles, progress, and challenges“. *IEEE Signal Process. Mag.* 35.1 (2018), 126–136 (cit. on p. 34).

[CSMT18] S. Chmiela, H. E. Sauceda, K.-R. Müller, and A. Tkatchenko. „Towards exact molecular dynamics simulations with machine-learned force fields“. *Nat. Commun.* 9.1 (2018), 1–10 (cit. on p. 37).

[Chm+17] S. Chmiela, A. Tkatchenko, H. E. Sauceda, I. Poltavsky, K. T. Schütt, and K.-R. Müller. „Machine learning of accurate energy-conserving molecular force fields“. *Sci. Adv.* 3.5 (2017), e1603015 (cit. on p. 37).

[CHLF20] I. Y. Chun, Z. Huang, H. Lim, and J. Fessler. „Momentum-Net: Fast and convergent iterative neural network for inverse problems“. *IEEE Trans. Pattern Anal. Mach. Intell.* available online: https://doi.org/10.1109/TPAMI.2020.3012955 (2020) (cit. on p. 37).

[CP11] P. L. Combettes and J.-C. Pesquet. „Proximal splitting methods in signal processing“. *Fixed-point algorithms for inverse problems in science and engineering*. Springer, 2011, 185–212 (cit. on p. 35).

[CS96] G. M. Constantine and T. H. Savits. „A Multivariate Faa Di Bruno Formula With Applications“. *T. Am. Math. Soc.* 348.2 (1996), 503–520 (cit. on p. 64).

[CSG19] D. Costarelli, A. Sambucini, and G.Vinti. „Convergence in Orlicz spaces by means of the multivariate max-product neural network operators of the Kantorovich type and applications“. *Neural Comput. & Applic.* 31 (2019), 5069–5078 (cit. on p. 23).

[CS13a] D. Costarelli and R. Spigler. „Approximation results for neural network operators activated by sigmoidal functions“. *Neural Netw.* 44 (2013), 101–106 (cit. on p. 23).

[CS13b] D. Costarelli and R. Spigler. „Multivariate neural network operators with sigmoidal activation functions“. *Neural Netw.* 48 (2013), 72–77 (cit. on p. 23).

[CZ07] F. Cucker and D. Zhou. *Learning Theory: An Approximation Theory Viewpoint*. Vol. 24. Cambridge Monographs on Applied and Computational Mathematics. Cambridge: Cambridge University Press, 2007 (cit. on pp. 7, 8).

[Cyb89] G. Cybenko. „Approximation by superpositions of a sigmoidal function“. *Math. Control Signals Syst.* 2.4 (1989), 303–314 (cit. on p. 9).

[COJSP17] W. M. Czarnecki, S. Osindero, M. Jaderberg, G. Swirszcz, and R. Pascanu. „Sobolev Training for Neural Networks“. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett. 2017, 4278–4287 (cit. on pp. 11, 55).

[DCH21] M. Z. Darestani, A. S. Chaudhari, and R. Heckel. „Measuring Robustness in Deep Learning Based Compressive Sensing“. *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, 2433–2444 (cit. on p. 50).

[De 01] C. De Boor. *A Practical Guide to Splines*. Applied mathematical sciences. Berlin: Springer, 2001 (cit. on pp. 23, 97).

[Din+20] Q. Ding, G. Chen, X. Zhang, Q. Huang, H. Ji, and H. Gao. „Low-dose CT with deep learning regularization via proximal forward–backward splitting“. *Phys. Med. Biol.* 65.12 (2020), 125009 (cit. on p. 42).

[Don06] D. L. Donoho. „Compressed Sensing“. *IEEE Trans. Inf. Theory* 52.4 (2006), 1289–1306 (cit. on p. 3).

[EY18] W. E and B. Yu. „The Deep Ritz Method: A Deep Learning-Based Numerical Algorithm for Solving Variational Problems“. *Commun. Math. Stat.* 6.1 (2018), 1–12 (cit. on pp. 2, 10).

[EE04] D. E. Edmunds and W. D. Evans. *Hardy Operators, Function Spaces and Embeddings*. Springer Monographs in Mathematics. Berlin: Springer, 2004 (cit. on p. 17).

[ET96] D. E. Edmunds and H. Triebel. *Function Spaces, Entropy Numbers, Differential Operators*. Cambridge Tracts in Mathematics. Cambridge: Cambridge University Press, 1996 (cit. on pp. 2, 16).

[Eps07] C. L. Epstein. *Introduction to the mathematics of medical imaging*. second. Philadelphia: SIAM, 2007 (cit. on p. 35).

[Eva99] L. Evans. *Partial Differential Equations*. Vol. 19. Graduate Studies in Mathematics. Providence: American Mathematical Society, 1999 (cit. on pp. 10, 58).

[Fes17] J. A. Fessler. *Analytical Tomographic Image Reconstruction Methods (Chapter 3 of book draft)*. URL: https://web.eecs.umich.edu/~fessler/book/c-tomo.pdf. 2017 (cit. on pp. 3, 38, 40).

[FR13] S. Foucart and H. Rauhut. *A Mathematical Introduction to Compressive Sensing*. Applied and Numerical Harmonic Analysis. Basel: Birkhäuser, 2013 (cit. on p. 3).

[Fun89] K. Funahashi. „On the approximate realization of continuous mappings by neural networks“. *Neural Netw.* 2.3 (1989), 183–192 (cit. on p. 9).

[GM76] D. Gabay and B. Mercier. „A dual algorithm for the solution of nonlinear variational problems via finite element approximation“. *Comput. Math. Appl.* 2.1 (1976), 17–40 (cit. on p. 35).

[GPRSK21] M. Geist, P. Petersen, M. Raslan, R. Schneider, and G. Kutyniok. „Numerical solution of the parametric diffusion equation by deep neural networks“. *J. Sci. Comput.* 88.1 (2021), 1–37 (cit. on p. 34).

[GGMM21] M. Genzel, I. Gühring, J. Macdonald, and M. März. „Near-Exact Recovery for Sparse-View CT via Data-Driven Methods“. *NeurIPS 2021 workshop on Deep Learning and Inverse Problems*. 2021 (cit. on pp. 4, 5).

[GGMM22] M. Genzel, I. Gühring, J. MacDonald, and M. März. „Near-Exact Recovery for Tomographic Inverse Problems via Deep Learning“. *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Ed. by K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato. Vol. 162. Proceedings of Machine Learning Research. PMLR, 2022, 7368–7381 (cit. on pp. 4, 35).

[GMM22] M. Genzel, J. Macdonald, and M. März. „Solving Inverse Problems With Deep Neural Networks - Robustness Included“. *IEEE Trans. Pattern Anal. Mach. Intell.* (2022). DOI: 10.1109/TPAMI.2022.3148324 (cit. on pp. 36, 42, 43, 54).

[GT98] D. Gilbarg and N. Trudinger. *Elliptic Partial Differential Equations of Second Order*. second. Vol. 224. A Series of Comprehensive Studies in Mathematics. Berlin: Springer, 1998 (cit. on p. 63).

[GOW21a] D. Gilton, G. Ongie, and R. Willett. „Deep Equilibrium Architectures for Inverse Problems in Imaging“. *IEEE Trans. Comput. Imaging* 7 (2021), 1123–1133 (cit. on p. 37).

[GOW21b] D. Gilton, G. Ongie, and R. Willett. „Model Adaptation for Inverse Problems in Imaging“. *IEEE Trans. Comput. Imag.* 7 (2021), 661–674 (cit. on p. 50).

[GM75] R. Glowinski and A. Marroco. „Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires“. *RAIRO Anal. Numer.* 9.R2 (1975), 41–76 (cit. on p. 35).

[GL10] K. Gregor and Y. LeCun. „Learning Fast Approximations of Sparse Coding“. *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*. Ed. by J. Fürnkranz and T. Joachims. Omnipress, 2010, 399–406 (cit. on p. 37).

[Gro15] P. Grohs. *Optimally Sparse Data Representations*. Harmonic and Applied Analysis. Basel: Birkhaeuser, 2015, 199–248 (cit. on p. 16).

[GPEB19] P. Grohs, D. Perekrestenko, D. Elbrächter, and H. Bölcskei. „Deep Neural Network Approximation Theory“. *arXiv preprint arXiv:1901.02220* (2019) (cit. on pp. 2, 10, 12).

[GV21] P. Grohs and F. Voigtlaender. „Proof of the Theory-to-Practice Gap in Deep Learning via Sampling Complexity bounds for Neural Network Approximation Spaces“. *arXiv preprint arXiv:2104.02746* (2021) (cit. on p. 55).

[GR21] I. Gühring and M. Raslan. „Approximation rates for neural networks with encodable weights in smoothness spaces“. *Neural Netw.* 134 (2021), 107–130 (cit. on pp. 4, 5, 7).

[GKP20] I. Gühring, G. Kutyniok, and P. Petersen. „Error bounds for approximations with deep ReLU neural networks in $W^{s,p}$ norms“. *Anal. Appl. (Singap.)* 18.05 (2020), 803–859 (cit. on pp. 4, 5, 7, 12, 63, 64, 84, 97).

[GRK22] I. Gühring, M. Raslan, and G. Kutyniok. „Expressivity of Deep Neural Networks“. *Mathematical Aspects of Deep Learning*. Ed. by P. Grohs and G. Kutyniok. Cambridge: Cambridge University Press, 2022, 149–199 (cit. on pp. 4, 5, 7).

[GI18] N. J. Guliyev and V. E. Ismailov. „Approximation capability of two hidden layer feedforward neural networks with fixed weights“. *Neurocomputing* 316 (2018), 262–269 (cit. on p. 9).

[GI16] N. J. Guliyev and V. E. Ismailov. „A Single Hidden Layer Feedforward Network with Only One Neuron in the Hidden Layer Can Approximate Any Univariate Function“. *Neural Comput.* 28.7 (2016), 1289–1304 (cit. on pp. 16, 32).

[Ham+18] K. Hammernik, T. Klatzer, E. Kobler, M. P. Recht, D. K. Sodickson, T. Pock, and F. Knoll. „Learning a Variational Network for Reconstruction of Accelerated MRI Data“. *Magn. Reson. Med.* 79.6 (2018), 3055–3071 (cit. on p. 37).

[Ham+21] K. Hammernik, J. Schlemper, C. Qin, J. Duan, R. M. Summers, and D. Rueckert. „Systematic evaluation of iterative deep neural networks for fast parallel MRI reconstruction with sensitivity-weighted coil combination“. *Magn. Reson. Med.* 86.4 (2021), 1859–1872 (cit. on pp. 37, 42, 45, 46).

[HA20] A. Hauptmann and J. Adler. „On the unreasonable effectiveness of CNNs“. Preprint arXiv:2007.14745. 2020 (cit. on p. 41).

[HWGY21] H. Heaton, S. Wu Fung, A. Gibali, and W. Yin. „Feasibility-based fixed point networks“. *J. Fixed Point Theory Appl.* 2021.1 (2021), 1–19 (cit. on p. 37).

[HGE21] C. Heiß, I. Gühring, and M. Eigel. „A neural multilevel method for high-dimensional parametric PDEs“. *NeurIPS 2021 workshop on The Symbiosis of Deep Learning and Differential Equations*. 2021 (cit. on pp. 4, 54).

[Hor91] K. Hornik. „Approximation capabilities of multilayer feedforward networks“. *Neural Netw.* 4.2 (1991), 251–257 (cit. on pp. 9, 11).

[HSW89] K. Hornik, M. Stinchcombe, and H. White. „Multilayer feedforward networks are universal approximators“. *Neural Netw.* 2.5 (1989), 359–366 (cit. on p. 9).

[JDVRB17] S. Jégou, M. Drozdzal, D. Vázquez, A. Romero, and Y. Bengio. „The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation“. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, 1175–1183 (cit. on p. 43).

[JMFU17] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser. „Deep Convolutional Neural Network for Inverse Problems in Imaging“. *IEEE Trans. Image Process.* 26.9 (2017), 4509–4522 (cit. on p. 41).

[KMY17] E. Kang, J. Min, and J. C. Ye. „A deep convolutional neural network using directional wavelets for low-dose X-ray CT reconstruction“. *Med. Phys.* 44.10 (2017), e360–e375 (cit. on p. 41).

[Kat09] H. Katsuura. „Summations Involving Binomial Coefficients“. *Coll. Math. J.* 40.4 (2009), 275–278 (cit. on p. 71).

[Kei+21] J. A. Keith, V. Vassilev-Galindo, B. Cheng, S. Chmiela, M. Gastegger, K.-R. Müller, and A. Tkatchenko. „Combining machine learning and computational chemistry for predictive insights into chemical systems“. *Chem. Rev.* 121.16 (2021), 9816–9872 (cit. on pp. 1, 37).

[KL20] P. Kidger and T. J. Lyons. „Universal Approximation with Deep Narrow Networks“. *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*. Ed. by J. D. Abernethy and S. Agarwal. Vol. 125. Proceedings of Machine Learning Research. PMLR, 2020, 2306–2327 (cit. on p. 9).

[KB14] D. P. Kingma and J. Ba. „Adam: A Method for Stochastic Optimization“. Preprint arXiv:1412.6980. 2014 (cit. on p. 41).

[Kno+20] F. Knoll, T. Murrell, A. Sriram, N. Yakubova, J. Zbontar, M. Rabbat, A. Defazio, M. J. Muckley, D. K. Sodickson, C. L. Zitnick, and M. P. Recht. „Advancing machine learning for MR image reconstruction with an open competition: Overview of the 2019 fastMRI challenge“. *Magn. Reson. Med.* 84.6 (2020), 3054–3070 (cit. on pp. 1, 37, 42).

[Kov+21] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar. „Neural Operator: Learning Maps Between Function Spaces“. *arXiv preprint arXiv:2108.08481* (2021) (cit. on pp. 2, 56).

[KPRS22] G. Kutyniok, P. Petersen, M. Raslan, and R. Schneider. „A theoretical analysis of deep neural networks and parametric PDEs“. *Constr. Approx.* 55.1 (2022), 73–125 (cit. on pp. 54, 55, 77).

[LMK22] S. Lanthaler, S. Mishra, and G. E. Karniadakis. „Error estimates for DeepONets: a deep learning framework in infinite dimensions“. *Trans. Math. Appl.* 6.1 (2022) (cit. on pp. 2, 56).

[LLPS93] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken. „Multilayer feedforward networks with a nonpolynomial activation function can approximate any function“. *Neural Netw.* 6.6 (1993), 861–867 (cit. on p. 9).

[Leu+21] J. Leuschner, M. Schmidt, P. S. Ganguly, V. Andriiashen, S. B. Coban, A. Denker, D. Bauer, A. Hadjifaradji, K. J. Batenburg, P. Maass, and M. van Eijnatten. „Quantitative Comparison of Deep Learning-Based Image Reconstruction Methods for Low-Dose and Sparse-Angle CT Applications“. *J. Imaging* 7.3 (2021) (cit. on pp. 4, 37, 43, 48).

[LTY20] B. Li, S. Tang, and H. Yu. „Better Approximations of High Dimensional Smooth Functions by Deep Neural Networks with Rectified Power Units“. *Commun. in Comp. Phys.* 27 (2020), 379–411 (cit. on pp. 9, 23).

[Lin19] S.-B. Lin. „Generalization and Expressivity for Deep Nets “. *IEEE T. Neur. Net. Lear.* 30.5 (2019), 1392–1406 (cit. on p. 23).

[Lit+17] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. Van Der Laak, B. Van Ginneken, and C. I. Sánchez. „A survey on deep learning in medical image analysis“. *Med. Image Anal.* 42 (2017), 60–88 (cit. on p. 3).

[Liu+20] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han. „On the Variance of the Adaptive Learning Rate and Beyond“. *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020 (cit. on p. 100).

[LH19] I. Loshchilov and F. Hutter. „Decoupled Weight Decay Regularization“. *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019 (cit. on p. 100).

[MP99] V. Maiorov and A. Pinkus. „Lower bounds for approximation by MLP neural networks“. *Neurocomputing* 25.1-3 (1999), 81–91 (cit. on p. 9).

[Mal12] S. Mallat. „Group Invariant Scattering“. *Commun. Pure Appl. Math.* 65.10 (2012), 1331–1398 (cit. on p. 33).

[Mar+19] M. Mardani, E. Gong, J. Y. Cheng, S. S. Vasanawala, G. Zaharchuk, L. Xing, and J. M. Pauly. „Deep Generative Adversarial Neural Networks for Compressive Sensing MRI“. *IEEE Transactions on Medical Imaging* 38.1 (2019), 167–179 (cit. on p. 36).

[Mar74] J. Marsden. *Elementary Classical Analysis*. San Francisco: W. H. Freeman and Company, 1974 (cit. on p. 61).

[Mha96] H. Mhaskar. „Neural Networks for Optimal Approximation of Smooth and Analytic Functions“. *Neural Comput.* 8.1 (1996), 164–177 (cit. on p. 9).

[MM92] H. Mhaskar and C. A. Micchelli. „Approximation by Superposition of Sigmoidal and Radial Basis Functions“. *Adv. Appl. Math.* 13.3 (1992), 350–373 (cit. on p. 22).

[MW93] A. A. Minai and R. D. Williams. „On the derivatives of the sigmoid“. *Neural Netw.* 6.6 (1993), 845–853 (cit. on p. 97).

[MPCB14] G. Montúfar, R. Pascanu, K. Cho, and Y. Bengio. „On the Number of Linear Regions of Deep Neural Networks“. *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. 2014, 2924–2932 (cit. on p. 65).

[Muc+21] M. J. Muckley, B. Riemenschneider, A. Radmanesh, S. Kim, G. Jeong, J. Ko, Y. Jun, H. Shin, D. Hwang, M. Mostapha, S. Arberet, D. Nickel, Z. Ramzi, P. Ciuciu, J.-L. Starck, J. Teuwen, D. Karkalousos, C. Zhang, A. Sriram, Z. Huang, N. Yakubova, Y. W. Lui, and F. Knoll. „Results of the 2020 fastMRI Challenge for Machine Learning MR Image Reconstruction“. *IEEE Trans. Med. Imaging* 40.9 (2021), 2306–2317 (cit. on pp. 1, 36, 37, 42, 56).

[Nak21] P. Nakkiran. „Towards an Empirical Theory of Deep Learning“. Doctoral dissertation. Harvard University Graduate School of Arts and Sciences, 2021. URL: https://nrs.harvard.edu/URN-3:HUL.INSTREPOS:37370110 (cit. on pp. 7, 56).

[Nat80] F. Natterer. „A Sobolev space analysis of picture reconstruction“. *SIAM J. Appl. Math.* 39.3 (1980), 402–411 (cit. on pp. 3, 56).

[Nat01] F. Natterer. *The Mathematics of Computerized Tomography*. Classics in Applied Mathematics. Philadelphia: Society for Industrial and Applied Mathematics (SIAM), 2001 (cit. on pp. 3, 56).

[NTMC20] F. Noé, A. Tkatchenko, K.-R. Müller, and C. Clementi. „Machine Learning for Molecular Simulation“. *Annu. Rev. Phys. Chem.* 71.1 (2020), 361–390 (cit. on p. 1).

[OK19] I. Ohn and Y. Kim. „Smooth function approximation by deep neural networks with general activation functions“. *Entropy* 21.7 (2019), 627 (cit. on pp. 9, 25).

[Ong+20] G. Ongie, A. Jalal, R. G. Baraniuk, C. A. Metzler, A. G. Dimakis, and R. Willett. „Deep Learning Techniques for Inverse Problems in Imaging“. *IEEE J. Sel. Areas Inf. Theory* 1.1 (2020), 39–56 (cit. on p. 1).

[OPS20] J. A. Opschoor, P. C. Petersen, and C. Schwab. „Deep ReLU networks and high-order finite element methods“. *Anal. Appl.* 18.05 (2020), 715–770 (cit. on p. 11).

[PV18] P. Petersen and F. Voigtländer. „Optimal approximation of piecewise smooth functions using deep ReLU neural networks“. *Neural Netw.* 108 (2018), 296–330 (cit. on pp. 2, 9–14, 25, 33, 70).

[Pin99] A. Pinkus. „Approximation theory of the MLP model in neural networks“. *Acta Numer.* 8 (1999), 143–195 (cit. on pp. 53, 54).

[Pou22] J. Pousin. „Least squares formulations for some elliptic second order problems, feedforward neural network solutions and convergence results“. *J. Comput. Math. Data Sci.* 2 (2022), 100023 (cit. on p. 53).

[PW17] P. Putzky and M. Welling. „Recurrent Inference Machines for Solving Inverse Problems“. Preprint arXiv:1706.04008. 2017 (cit. on p. 99).

[Rad17] J. Radon. „Über die Bestimmung von Funktionen längs gewisser Mannigfaltigkeiten. Sächsische Gesellschaft der Wissenschaften Math“. *Phys. Klasse, Leipzig* 69 (1917), 262–277 (cit. on p. 3).

[RCS20] Z. Ramzi, P. Ciuciu, and J.-L. Starck. „XPDNet for MRI Reconstruction: an application to the fastMRI 2020 brain challenge“. Preprint arXiv:2010.07290. 2020 (cit. on pp. 42, 43).

[RT18] D. Rolnick and M. Tegmark. „The power of deeper networks for expressing natural functions“. *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018 (cit. on p. 25).

[RFB15] O. Ronneberger, P. Fischer, and T. Brox. „U-Net: Convolutional Networks for Biomedical Image Segmentation“. *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*. Ed. by N. Navab, J. Hornegger, W. M. W. III, and A. F. Frangi. Vol. 9351. Lecture Notes in Computer Science. Springer, 2015, 234–241 (cit. on pp. 37, 40, 41).

[Rou13] T. Roubíček. *Nonlinear Partial Differential Equations with Applications*. second. Vol. 153. International Series of Numerical Mathematics. Basel: Springer Science+Business Media, 2013 (cit. on pp. 10, 58).

[RTML12] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld. „Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning“. *Phys. Rev. Lett.* 108 (5 2012), 058301 (cit. on p. 10).

[SB18] P. Sadowski and P. Baldi. „Deep Learning in the Natural Sciences: Applications to Physics“. *Braverman Readings in Machine Learning. Key Ideas from Inception to Current State: International Conference Commemorating the 40th Anniversary of Emmanuil Braverman's Decease, Boston, MA, USA, April 28-30, 2017, Invited Talks*. Ed. by L. Rozonoer, B. Mirkin, and I. Muchnik. Cham: Springer International Publishing, 2018, 269–297 (cit. on p. 1).

[Sau+22] H. E. Sauceda, L. E. Gálvez-González, S. Chmiela, L. O. Paz-Borbón, K.-R. Müller, and A. Tkatchenko. „BIGDML—Towards accurate quantum machine learning force fields for materials“. *Nat. Commun.* 13.1 (2022), 3733 (cit. on p. 10).

[Sch+19] J. Schlemper, I. Oksuz, J. R. Clough, J. Duan, A. P. King, J. A. Schnabel, J. V. Hajnal, and D. Rueckert. „dAUTOMAP: decomposing AUTOMAP to achieve scalability and enhance performance“. Preprint arXiv:1909.10995. 2019 (cit. on p. 37).

[Sch20] J. Schmidt-Hieber. „Nonparametric regression using deep neural networks with ReLU activation function“. *Ann. Stat.* 48.4 (2020), 1875–1897 (cit. on pp. 9, 19).

[SZ19] C. Schwab and J. Zech. „Deep learning in high dimension: Neural network expression rates for generalized polynomial chaos expansions in UQ“. *Anal. Appl.* 17.1 (2019), 19–55 (cit. on pp. 25, 75).

[SCC18] U. Shaham, A. Cloninger, and R. Coifman. „Provable approximation properties for deep neural networks“. *Appl. Comput. Harmon. Anal.* 44.3 (2018), 537–557 (cit. on p. 10).

[SWED20] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis. „Model-based deep learning“. Preprint arXiv:2012.08405. 2020 (cit. on p. 50).

[SLBP21] E. Sidky, I. Lorente, J. G. Brankov, and X. Pan. „Do CNNs solve the CT inverse problem?“ *IEEE Trans. Biomed. Eng.* 68.6 (2021), 1799–1810 (cit. on pp. 1, 3, 37, 50).

[Sid+21] E. Sidky, X. Pan, J. Brankov, I. Lorente, S. Armato, K. Drukker, L. Hadjiyski, N. Petrick, K. Farahani, R. Munbodh, K. Cha, J. Kalpathy-Cramer, B. Bearce, and AAPM Working Group on Grand challenges. *Deep Learning for Inverse Problems: Sparse-View Computed Tomography Image Reconstruction (DL-sparse-view CT)*. URL: https://www.aapm.org/GrandChallenge/DL-sparse-view-CT/. 2021 (cit. on pp. 3, 36, 43).

[SP21] E. Y. Sidky and X. Pan. „Report on the AAPM deep-learning sparse-view CT (DL-sparse-view CT) Grand Challenge“. *Med. Phys.* accepted, preprint arXiv:2109.09640 (2021) (cit. on pp. 38, 43).

[SS18] J. Sirignano and K. Spiliopoulos. „DGM: A deep learning algorithm for solving partial differential equations“. *J. Comput. Phys.* 375 (2018), 1339–1364 (cit. on pp. 2, 11).

[Sri+20] A. Sriram, J. Zbontar, T. Murrell, A. Defazio, C. L. Zitnick, N. Yakubova, F. Knoll, and P. M. Johnson. „End-to-End Variational Networks for Accelerated MRI Reconstruction“. *Medical Image Computing and Computer Assisted Intervention - MICCAI 2020 - 23rd International Conference, Lima, Peru, October 4-8, 2020, Proceedings, Part II*. Ed. by A. L. Martel, P. Abolmaesumi, D. Stoyanov, D. Mateus, M. A. Zuluaga, S. K. Zhou, D. Racoceanu, and L. Joskowicz. Vol. 12262. Lecture Notes in Computer Science. Springer, 2020, 64–73 (cit. on p. 42).

[Ste79] E. Stein. *Singular Integrals and Differentiability Properties of Functions*. 3rd. Princeton: Princeton University Press, 1979 (cit. on p. 59).

[SKM07] M. Sugiyama, M. Krauledat, and K.-R. Müller. „Covariate Shift Adaptation by Importance Weighted Cross Validation“. *J. Mach. Learn. Res.* 8.35 (2007), 985–1005 (cit. on p. 50).

[Suz19] T. Suzuki. „Adaptivity of deep ReLU network for learning in Besov and mixed smooth Besov spaces: optimal rate and curse of dimensionality“. *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019 (cit. on p. 9).

[TLY19] S. Tang, B. Li, and H. Yu. „ChebNet: Efficient and Stable Constructions of Deep Neural Networks with Rectified Power Units using Chebyshev Approximations“. *arXiv preprint arXiv:1911.05467* (2019) (cit. on p. 9).

[TG21] T. Tirer and R. Giryes. „On the Convergence Rate of Projected Gradient Descent for a Back-Projection Based Objective“. *SIAM J. Imag. Sci.* 14.4 (2021), 1504–1531 (cit. on p. 42).

[Tri78] H. Triebel. *Interpolation Theory, Function Spaces, Differential Operators*. Amsterdam: North-Holland Publishing Company, 1978 (cit. on pp. 2, 16).

[TL19] Y. Tu and Y. Lin. „Deep neural network compression technique towards efficient digital signal modulation recognition in edge device“. *IEEE Access* 7 (2019), 58113–58119 (cit. on p. 34).

[UVL18] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky. „Deep Image Prior“. *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 2018, 9446–9454 (cit. on p. 2).

[Unk+21a] O. T. Unke, S. Chmiela, H. E. Sauceda, M. Gastegger, I. Poltavsky, K. T. Schütt, A. Tkatchenko, and K.-R. Müller. „Machine learning force fields“. *Chem. Rev.* 121.16 (2021), 10142–10186 (cit. on pp. 1, 37).

[Unk+21b] O. T. Unke, M. Bogojeski, M. Gastegger, M. Geiger, T. Smidt, and K.-R. Müller. „SE(3)-equivariant prediction of molecular wavefunctions and electronic densities“. *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan. 2021, 14434–14447 (cit. on p. 10).

[WN19] M. J. Willemink and P. B. Noël. „The evolution of image reconstruction for CT – from filtered back projection to artificial intelligence“. *Eur. Radiol.* 29.5 (2019), 2185–2195 (cit. on p. 42).

[WMS22] L. Winkler, K.-R. Müller, and H. E. Sauceda. „High-fidelity molecular dynamics trajectory reconstruction with bi-directional neural networks“. *Mach. Learn.: Sci. Technol.* (2022) (cit. on p. 10).

[WH18] Y. Wu and K. He. „Group Normalization“. *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIII*. Ed. by V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss. Vol. 11217. Lecture Notes in Computer Science. Springer, 2018, 3–19 (cit. on p. 99).

[WGCM16] T. Würfl, F. C. Ghesu, V. Christlein, and A. K. Maier. „Deep Learning Computed Tomography“. *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2016 - 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part III*. Ed. by S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. B. Ünal, and W. M. W. III. Vol. 9902. Lecture Notes in Computer Science. 2016, 432–440 (cit. on p. 100).

[YSLX16] Y. Yang, J. Sun, H. Li, and Z. Xu. „Deep ADMM-Net for Compressive Sensing MRI“. *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. Ed. by D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett. 2016, 10–18 (cit. on pp. 37, 42).

[Yar17] D. Yarotsky. „Error bounds for approximations with deep ReLU networks“. *Neural Netw.* 94 (2017), 103–114 (cit. on pp. 9, 17, 19, 20, 24, 25, 30, 31, 33, 34, 69, 73, 75, 76, 97).

[Yar18] D. Yarotsky. „Optimal approximation of continuous functions by very deep ReLU networks“. *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*. Ed. by S. Bubeck, V. Perchet, and P. Rigollet. Vol. 75. Proceedings of Machine Learning Research. PMLR, 2018, 639–649 (cit. on p. 32).

[YZ20] D. Yarotsky and A. Zhevnerchuk. „The phase diagram of approximation rates for deep neural networks“. *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. 2020 (cit. on p. 32).

[Yeo+21] S.-K. Yeom, P. Seegerer, S. Lapuschkin, A. Binder, S. Wiedemann, K.-R. Müller, and W. Samek. „Pruning by explaining: A novel criterion for deep neural network pruning“. *Pattern Recognit.* 115 (2021), 107899 (cit. on p. 34).