Universität Stuttgart

mathematik physik fakultät8

Universität Stuttgart

Fakultät 8 - Mathematik und Physik

Approximationsraten und Netzwerkkomplexität für ReLU Netzwerke

Bachelorarbeit

von

Jan Niklas Nertinger

Matrikelnummer:	3552283
Betreuerin:	Prof. Dr. Andrea Barth
Abgabetermin:	27. November 2024

Eigenständigkeitserklärung

Hiermit erkläre ich, Jan Niklas Nertinger, dass ich die vorliegende Bachelorarbeit mit dem Titel *"Approximationsraten und Netzwerkkomplexität für ReLU Netzwerke"* selbstständig und ohne unzulässige Hilfe angefertigt habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel verwendet und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht.

Der Code, der für die Generierung der Daten in dieser Arbeit verwendet wurde, sowie dessen Dokumentation, wurden unter Verwendung von ChatGPT 4.0 (www.chatgpt.com) erstellt.

Die Arbeit wurde weder in gleicher noch in ähnlicher Form einer anderen Prüfungsbehörde vorgelegt, noch wurde sie veröffentlicht.

Stuttgart, den 27. November 2024

Unterschrift

Danksagung

Ich möchte allen, die mich bei der Erstellung dieser Arbeit unterstützt haben, meinen herzlichen Dank aussprechen. Mein besonderer Dank gilt meiner Betreuerin, Frau Prof. Dr. Andrea Barth, für ihre kompetente und engagierte Unterstützung. Ihre fundierten Ratschläge, wertvollen Anregungen und ihre konstruktive Kritik haben entscheidend zum Gelingen dieser Arbeit beigetragen und mich in jeder Phase motiviert.

Ebenso danke ich der Universität Stuttgart herzlich für die Bereitstellung der notwendigen Infrastruktur und Ressourcen, die es ermöglicht haben, die zeit- und rechenintensive Generierung der für diese Arbeit erforderlichen Daten erfolgreich durchzuführen.

Mein Dank gilt auch Lukas Christian Bauer für das Korrekturlesen dieser Arbeit.

Zusammenfassung

Diese Arbeit untersucht die Approximationsraten und die Netzwerkkomplexität neuronaler Feedforward-Netzwerke mit ReLU Aktivierungsfunktion. Im Fokus stehen dabei theoretische Abschätzungen für die benötigte Netzwerkkomplexität — gemessen an den Parametern der Tiefe, Anzahl der Neuronen und Anzahl der Gewichte — um Funktionen bestimmter Klassen mit vorgegebener Genauigkeit zu approximieren. Die Analyse umfasst sowohl Funktionen aus Sobolev-Räumen in Bezug auf die L^∞ Norm als auch glatte Funktionen in allgemeinen L^p Normen. Durch die Herleitung präziser Abhängigkeiten zwischen Approximationsgenauigkeit, Regularitätseigenschaften der Funktionenklassen und Netzwerkkomplexität werden theoretische Obergrenzen formuliert und durch numerische Experimente validiert. Die Ergebnisse liefern wertvolle Einblicke in die Gestaltung effizienter neuronaler Netzwerke und deren Anwendungsmöglichkeiten in datenintensiven Bereichen.

Inhaltsverzeichnis

1 Einleitung

Neuronale Netze haben sich in den letzten Jahren als eine der mächtigsten Methoden zur Approximation komplexer Funktionen etabliert und bilden das Rückgrat zahlreicher Anwendungen in der Datenwissenschaft und künstlichen Intelligenz. Ihre bemerkenswerte Fähigkeit, hochdimensionale und nichtlineare Abbildungen effizient zu modellieren, macht sie zu einem unverzichtbaren Werkzeug in Bereichen wie Bilderkennung, Sprachverarbeitung und Mustererkennung. Die Effektivität eines neuronalen Netzes bei der Approximation einer Funktion f hängt jedoch maßgeblich von seiner Architektur und den verfügbaren Ressourcen ab.

Die **Netzwerkkomplexität** eines neuronalen Netzes quantifiziert die verüfgbaren Ressourcen einer Netzwerkarchitektur. Hierzu zählen Parameter wie die Tiefe des Netzes, die Anzahl der Neuronen und die Gesamtzahl der Gewichte, wobei insbesondere die Anzahl der nichtnull Gewichte von Interesse ist. Diese Kenngrößen beeinflussen direkt den Speicherbedarf und die Rechenkosten und bestimmen somit die praktische Einsetzbarkeit des Netzwerks.

Die **Approximationsraten** beschreiben, wie die Netzwerkkomplexität mit der Approximation einer Funktion mit Genauigkeit $\varepsilon > 0$ zusammenhängt. Ein zentrales Anliegen ist es, obere Schranken für die Netzwerkkomplexität in Abhängigkeit von ε herzuleiten, um zu bestimmen, welche Ressourcen *höchstens* erforderlich sind, um eine vorgegebene Genauigkeit erreichen zu können. Die präzise Charakterisierung dieses Zusammenhangs stellt eine wesentliche Herausforderung in der theoretischen Analyse neuronaler Netze dar und hat weitreichende Implikationen für das Design effizienter Architekturen.

Bestehende Literatur

Die Frage nach den Approximationseigenschaften neuronaler Netzwerke wurde in der Literatur intensiv untersucht. Bereits 1989 zeigte Cybenko in [2], dass flache neuronale Netzwerke universelle Approximatoren sind, sie also jede stetige Funktion auf einem kompakten Definitionsbereich mit beliebiger Genauigkeit approximieren können - vorausgesetzt, es stehen genügend Neuronen zur Verfügung. Spätere Arbeiten, wie [1] von Barron, untersuchten die Approximationsraten für eine Klasse glatter Funktionen in Abhängigkeit von der Netzwerkkomplexität

In jüngerer Zeit legte Yarotsky in [9] den Fokus auf neuronale Netzwerke mit ReLU Aktivierungsfunktionen und zeigte, dass diese in der Lage sind, hochdimensionale Funktionen aus Sobolev-Räumen effizient zu approximieren. Dabei analysierte er insbesondere, wie die Netzwerkkomplexität mit der vorgegebenen Genauigkeit der Approximation skaliert. Ähnliche Untersuchungen führten Petersen und Voigtlaender in [7] für andere

Funktionsklassen durch, wobei sie auch die Rolle der Gewichtsbeschränkung und der Quantisierung der Parameter betrachteten.

Zielsetzung und Ansatz

Das Verständnis der Beziehung zwischen Netzwerkkomplexität und Approximationsgenauigkeit ist von zentraler Bedeutung für die Entwicklung effizienter neuronaler Netze. Die Kernfrage lautet: Welche strukturellen Ressourcen — wie Tiefe, Breite und Anzahl der Gewichte — werden *höchstens* benötigt, um eine gegebene Genauigkeit $\varepsilon > 0$ bei der Approximation bestimmter Funktionen erreichen zu können? Diese Fragestellung ist nicht nur theoretisch interessant, sondern bietet auch praktische Orientierungshilfen für das Design von Netzwerken in realen Anwendungen.

Das Ziel dieser Arbeit ist es, theoretische obere Schranken für die Netzwerkkomplexität herzuleiten, die notwendig sind, um bestimmte Klassen von Funktionen mit einer gewünschten Genauigkeit zu approximieren. Unser Fokus liegt dabei auf zwei spezifischen Kontexten:

- Die Approximation von Funktionen aus Sobolev-Räumen bezüglich der L^∞ Norm.
- Die Approximation von Klassen glatter Funktionen bezüglich allgemeiner L^p Normen.

Dabei orientieren wir uns an zentralen Ergebnissen von Yarotsky in [9] sowie Petersen und Voigtlaender in [7]. Ihre Arbeiten bilden den Ausgangspunkt für unsere Betrachtungen. Wir werden deren Erkenntnisse zu Abhängigkeiten zwischen der Approximationsgenauigkeit ε der zu approximierenden Funktionen und der erforderlichen Netzwerkkomplexität in einer konsistenten und homogenisierten Darstellung zusammenführen und zugänglich machen. Diese theoretischen Ergebnisse werden durch numerische Experimente untermauert, die ihre praktische Relevanz und Anwendbarkeit demonstrieren.

2 Theoretische Grundlagen zu neuronalen Netzen

Dieses Kapitel stützt sich zu großen Teilen auf die Arbeiten von Petersen und Voigtlaender in [7]. Wir werden grundsätzliche Definitionen und Aussagen einführen, die uns dabei helfen werden, die oben angesprochenen Resultate von Yarotsky sowie Petersen und Voigtlaender in kohärenter und strukturierte Form darzusellen.

2.1 Einführung in neuronale Feedforward-Netze

Zunächst erläutern wir die theoretischen Grundlagen zu neuronalen Netzen, wobei wir uns speziell auf "Feedforward"-Netze konzentrieren. Da im weiteren Verlauf dieser Arbeit zahlreiche Aussagen zur Struktur und Größe solcher Netze gemacht werden, sind klare Definitionen von Begriffen wie der Anzahl der Neuronen oder der Anzahl der Gewichte unerlässlich. Dabei interessiert uns in dieser Arbeit grundsätzlich die Anzahl der nichtnull Gewichte, da die Anzahl der Gewichte im Allgemeinen bereits fest durch die Netzwerkarchitektur vorgegeben ist und uns in den folgenden Beweisen daher keinen "Spielraum" lässt.

Eine präzise Definition solcher Begriffe wird in [7] von Petersen und Voigtlaender bereitgestellt und hier für allgemeine Feedforward-Netze erweitert. Für unsere Betrachtungen ist es dabei von entscheidender Bedeutung, strikt zwischen einem neuronalen Netz als strukturierte Menge von Gewichten und der dadurch dargestellten Funktion, der sogenannten *Realisierung* des Netzes, zu unterscheiden.

Definition 2.1. *[Neuronales Feedforward-Netzwerk] Seien $d, m, L \in \mathbb{N}$. Ein* ***neuronales Feedforward-Netzwerk*** *Φ (im weiteren auch einfach als* ***(neuronales) Netz****,* ***Netzwerk*** *oder* ***Feedforward-Netz*** *bezeichnet) mit* ***Eingabedimension*** *d,* ***Ausgabedimension*** *m und* ***Anzahl der Schichten*** *oder* ***Tiefe*** *$L(\Phi) := L$ ist ein Tupel*

$$\Phi = \left(L, \{N_l\}_{l=0}^{L}, \{A_{k,l}\}_{0 \leq k < l \leq L}, \{b_l\}_{l=1}^{L}\right),$$

wobei $N_0 = d$, $N_L = m$, und $N_1, \ldots, N_{L-1} \in \mathbb{N}$ sowie jede Matrix $A_{k,l} \in \mathbb{R}^{N_l \times N_k}$ und jeder Vektor $b_l \in \mathbb{R}^{N_l}$ ist.

Wir definieren die

- ***Anzahl der Neuronen von*** *Φ als $N(\Phi) := \sum_{j=0}^{L} N_j$,*
- ***Anzahl der Gewichte von*** *Φ als $G(\Phi) := \sum_{0 \leq k < l \leq L} (N_k \cdot N_l) + \sum_{j=1}^{L} N_j$,*

- ***Anzahl der nichtnull Gewichte von*** Φ *als die Anzahl der nichtnull Einträge aller* $A_{k,l}$ *und* b_l *gegeben durch* $M(\Phi) := \sum_{0\leq k<l\leq L} \|A_{k,l}\|_{\ell^0} + \sum_{j=1}^{L} \|b_j\|_{\ell^0}$.

Dabei ist $\| \bullet \|_{\ell^0}$ wie üblicherweise ein Maß für die Anzahl der nichtnull Elemente einer Matrix oder eines Vektors.

Definition 2.2. *[Realisierung] Ist* Φ *ein neuronales Feedforward-Netz mit Eingabedimension* d *und Ausgabedimension* m *wie oben und* $\rho : \mathbb{R} \to \mathbb{R}$ *beliebig, dann definieren wir die* ***Realisierung von*** Φ ***mit Aktivierungsfunktion*** ρ *als die Abbildung* $R_\rho(\Phi) : \mathbb{R}^d \to \mathbb{R}^m$ *mit*

$$R_\rho(\Phi)(x) = x_L,$$

wobei x_L *aus dem folgenden Schema resultiert:*

$$\begin{aligned} x_0 &:= x, \\ x_\ell &:= \rho_{N_\ell}\Big(\sum_{k=0}^{\ell-1} A_{k,\ell} x_k + b_\ell\Big), \quad \text{für } \ell = 1, \dots, L-1, \\ x_L &:= \sum_{k=0}^{L-1} A_{k,L} x_k + b_L \end{aligned}$$

Dabei wird ρ *komponentenweise angewandt:* $\rho_m(y) = (\rho(y_1), \dots, \rho(y_m))^T$ *für* $y = (y_1, \dots, y_m) \in \mathbb{R}^m$.

Beispiel 2.3. Für ein neuronales Feedforward-Netz Φ mit $L = 2$ und

$$A_{0,1} := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},\ A_{0,2} := \begin{pmatrix} 1 & 0 \end{pmatrix},\ A_{1,2} := \begin{pmatrix} 0 & 1 \end{pmatrix},\ b_1 := \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ b_2 := \begin{pmatrix} 2 \end{pmatrix},$$

ist für $x_0 := \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ mit ReLU Aktivierungsfunktion $\varrho(x) := \max\{0, x\}$

$$\begin{aligned} x_1 &= \varrho_2(A_{0,1} x_0 + b_1) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \\ x_2 &= A_{0,2} x_0 + A_{1,2} x_1 + b_2 = 1 + 0 + 2 = 3 = R_\varrho(\Phi)(x_0). \end{aligned}$$

Wir werden in dieser Arbeit auch von *Netzwerkarchitekturen* sprechen. Diese sind effektiv neuronale Feedforward-Netzwerke mit unspezifizierten Gewichten.

Definition 2.4. *[Netzwerkarchitektur] Seien* $d, m, L \in \mathbb{N}$. *Eine* ***Netzwerkarchitektur*** $\mathcal{A}$ *mit* ***Eingabedimension*** d, ***Ausgabedimension*** m *und* ***Anzahl der Schichten*** *oder* ***Tiefe*** $L(\Phi) := L$ *ist ein Tupel*

$$\mathcal{A} = \Big(L, \{N_\ell\}_{\ell=0}^{L}, \{\Sigma_{k,l}\}_{0\leq k<l\leq L}, \{\sigma_l\}_{l=1}^{L}\Big),$$

wobei $N_0 = d$, $N_L = m$, *und* $N_1, \dots, N_{L-1} \in \mathbb{N}$ *sowie jede Matrix* $\Sigma_{k,l} \in \{0,1\}^{N_l \times N_k}$ *und jeder Vektor* $\sigma_l \in \{0,1\}^{N_l}$ *ist.*

Ein neuronales Feedforward-Netzwerk $\Phi = \left(L', \{N'_\ell\}_{\ell=0}^{L'}, \{A_{k,\ell}\}_{0\leq k<\ell\leq L'}, \{b_\ell\}_{\ell=1}^{L'}\right)$ ***implementiert*** $\mathcal{A}$*, wenn* $L' = L$ *und* $\{N'_\ell\}_{\ell=0}^{L'} = \{N_\ell\}_{\ell=0}^{L}$ *sowie*

$$(\Sigma_{k,l})_{i,j} = 0 \;\Rightarrow\; (A_{k,l})_{i,j} = 0,$$
$$(\sigma_l)_i = 0 \;\Rightarrow\; (b_l)_i = 0.$$

Die ***Anzahl der Neuronen*** $N(\mathcal{A})$*, die* ***Anzahl der Gewichte*** $G(\mathcal{A})$ *und die* ***Anzahl der nichtnull Gewichte*** $M(\mathcal{A})$ *sind analog zu neuronalen Feedforward-Netzwerken definiert.*

Wir werden unter anderem auch von *Feedforward-Netzwerkarchitekturen* und *MLP-Netzwerkarchitekturen* sprechen. Letzteres bezieht sich auf Netzwerke, die den gleichen Bedingungen wie in der Definition eines Multilayer Perceptrons unterliegen (siehe Definition 2.7), während ersteres eine allgemeine Netzwerkarchitektur meint, bei der typischerweise auch Verbindungen zwischen nicht-benachbarten Schichten bestehen können.

Beispiel 2.5. Das neuronale Feedforward-Netz Φ aus Beispiel 2.3 implementiert die Netzwerkarchitektur $\mathcal{A}$ mit

$$\Sigma_{0,1} := \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix},\ \Sigma_{0,2} := \begin{pmatrix} 1 & 0 \end{pmatrix},\ \Sigma_{1,2} := \begin{pmatrix} 0 & 1 \end{pmatrix},\ \sigma_1 := \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ \sigma_2 := \begin{pmatrix} 1 \end{pmatrix}.$$

Bemerkung 2.6. Im Verlauf dieser Arbeit werden wir häufig davon sprechen, dass sich eine Funktion $f : \mathbb{R}^d \supset D \to W \subset \mathbb{R}^n$ *durch ein neuronales Netz* Φ *darstellen* lässt. Dies bedeutet, dass ein neuronales Netz Φ existiert, dessen Realisierung eingeschränkt auf D gerade die Funktion f ist:

$$R_\rho(\Phi)|_D = f.$$

Sagen wir, dass sich f *durch eine Netzwerkarchitektur* $\mathcal{A}$ *darstellen lässt*, meinen wir, dass ein neuronales Netz Φ existiert, welches $\mathcal{A}$ implementiert und die obige Bedingung erfüllt.

Sagen wir, dass sich eine Funktion f *durch ein neuronales Netz oder eine Netzwerkarchitektur approximieren lässt*, bedeutet dies, dass sich eine Funktion $\tilde{f}$ durch ein Netz oder eine Netzwerkarchitektur darstllen lässt, welche f approximiert.

Die verwendete Aktiverungsfunktion wird in dieser Arbeit ausschließlich die ReLU Aktivierungsfunktion sein (siehe Definition 2.9) und wird daher nicht weiter spezifiziert.

Ein allgemeines Feedforward-Netz erlaubt auch Verbindungen zwischen nicht-benachbarten Schichten. Ein Spezialfall eines solchen Netzes ist das sogenannte Multilayer Perceptron, das nur Verbindungen zwischen benachbarten Schichten zulässt.

Definition 2.7. *[Multilayer Perceptron] Ein* ***Multilayer Perceptron (MLP)*** Φ *ist ein neuronales Feedforward-Netz, für das gilt:*

$$A_{k,\ell} = 0,\ \textit{falls}\ k \neq \ell - 1$$

Für ein Multilayer Perceptron werden die $A_{\ell-1,\ell}$ *auch häufig nur als* A_ℓ *bezeichnet.*

Bemerkung 2.8. Ein Multilayer Perceptron Φ mit L Schichten kann auch einfacher als L-Tupel von Matrix-Vektor-Tupeln dargestellt werden:

$$\Phi = ((A_1, b_1), (A_2, b_2), \dots, (A_L, b_L)) .$$

Dabei ist $A_\ell := A_{\ell-1,\ell}$. Die Berechnung von $R_\rho(\Phi)(x) = x_L$ vereinfacht sich zu

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

und die Anzahl der nichtnull Gewichte zu

$$M(\Phi) := \sum_{j=1}^{L} (\|A_j\|_{\ell^0} + \|b_j\|_{\ell^0}).$$

Zudem definieren wir die **Anzahl der Gewichte eines Multilayer Perceptrons** als

$$G_{\text{MLP}}(\Phi) := \sum_{j=1}^{L} (N_{j-1} \cdot N_j + N_j).$$

Wir bemerken, dass damit für ein Multilayer Perceptron Φ weiterhin $M(\Phi) \leq G_{\text{MLP}}(\Phi)$ gilt.

Im Speziellen werden wir in dieser Arbeit ReLU Netze betrachten, also neuronale Feedforward-Netze mit einer ReLU Aktivierungsfunktion. Diese Funktion, wie von Ding, Qian und Zhou in [3] beschrieben, ist eine der am weitesten verbreiteten Aktivierungsfunktionen im Bereich der neuronalen Netze und bietet dank ihrer Einfachheit und Effizienz erhebliche Vorteile gegenüber anderen Aktivierungsfunktionen.

Definition 2.9. *[ReLU] Die* ***Rectified Linear Unit Aktiverungsfunktion*** *(kurz* ***ReLU Aktivierungsfunktion*** *oder* ***ReLU****)* $\varrho : \mathbb{R} \to \mathbb{R}$ *ist gegeben durch*

$$\varrho(x) := \max\{0, x\}.$$

Im Folgenden sei mit ϱ stets die ReLU Aktivierungsfunktion bezeichnet.

Möchte man ein neuronales Netz in der Praxis auf einem Computer implementieren, stellt man fest, dass die Gewichte nicht frei aus $\mathbb{R}$ gewählt werden können. Stattdessen ist jedem Gewicht eine feste Anzahl an Bits zugeordnet, die festlegt, welche Werte ein Gewicht annehmen kann. Daher werden wir die Gewichte eines Netzes in Teilen dieser Arbeit als *beschränkt und quantisiert* betrachten. Solche Netze können dann mit kontrollierbarem Speicherverbrauch auf einem Computer gespeichert werden. Dabei erlauben wir jedoch keine feste Anzahl an Bits pro Gewicht, sondern lassen diese auf kontrollierte Weise mit der gewünschten Approximationsgenauigkeit wachsen. Diese Idee der quantisierten Gewichte wird in der folgenden Definition formalisiert:

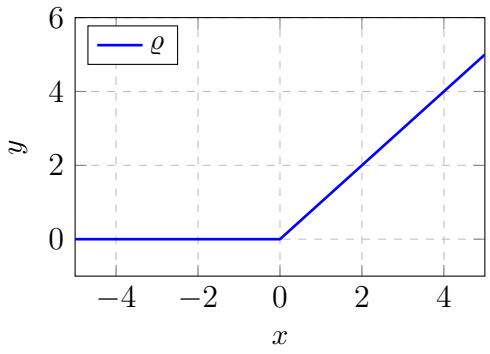

Abbildung 2.1: ReLU Aktivierungsfunktion $\varrho(x) = \max\{0, x\}$.

Definition 2.10. *[quantisierte Gewichte] Seien $\varepsilon \in (0, \infty)$ und $s \in \mathbb{N}$. Ein neuronales Feedforward-Netz $\Phi = (L, \{N_\ell\}_{\ell=0}^{L}, \{A_{k,\ell}\}_{0 \le k < \ell \le L}, \{b_\ell\}_{\ell=1}^{L})$ hat (s, ε)-**quantisierte Gewichte**, wenn alle Gewichte, also die Einträge aller $A_{k,\ell}$ und b_ℓ, aus $[-\varepsilon^{-s}, \varepsilon^{-s}] \cap 2^{-s\lceil \log_2(\frac{1}{\varepsilon})\rceil}\mathbb{Z}$ sind.*

Beispiel 2.11. Sei $\varepsilon = \frac{1}{4}$ und $s = 2$. Dann ist $\varepsilon^{-s} = 16$ und $2^{-s\lceil \log_2(\frac{1}{\varepsilon})\rceil} = \frac{1}{16}$ und damit hat das MLP $\Phi = ((A_1, b_1))$ mit

$$A_1 := \begin{pmatrix} 1 & 0 \\ \frac{3}{4} & \frac{1}{16} \end{pmatrix}, \quad b_1 := \begin{pmatrix} -16 \\ \frac{27}{4} \end{pmatrix},$$

(s, ε)-quantisierte Gewichte

Bemerkung 2.12.

- Insbesondere sind (s, ε)-quantisierte Gewichte für $\varepsilon \in (0, 1]$ und $t, s \in \mathbb{N}$ mit $t \ge s$ immer auch (t, ε)-quantisiert:

$$\varepsilon^{-s} \le \varepsilon^{-t} \text{ und damit } [-\varepsilon^{-s}, \varepsilon^{-s}] \subset [-\varepsilon^{-t}, \varepsilon^{-t}],$$
$$2^{-s\lceil \log_2(\frac{1}{\varepsilon})\rceil} = 2^{(t-s)\lceil \log_2(\frac{1}{\varepsilon})\rceil} \cdot 2^{-t\lceil \log_2(\frac{1}{\varepsilon})\rceil} \text{ und damit } 2^{-s\lceil \log_2(\frac{1}{\varepsilon})\rceil}\mathbb{Z} \subset 2^{-t\lceil \log_2(\frac{1}{\varepsilon})\rceil}\mathbb{Z}.$$

- Des Weiteren sind (s, ε)-quantisierte Gewichte für ein $0 < \delta < \varepsilon$ auch (s, δ)-quantisiert:

$$\varepsilon^{-s} \le \delta^{-s} \text{ und damit } [-\varepsilon^{-s}, \varepsilon^{-s}] \subset [-\delta^{-s}, \delta^{-s}],$$
$$2^{-s\lceil \log_2(\frac{1}{\varepsilon})\rceil} = k \cdot 2^{-t\lceil \log_2(\frac{1}{\delta})\rceil} \text{ für } k \in \mathbb{N} \text{ und damit } 2^{-s\lceil \log_2(\frac{1}{\varepsilon})\rceil}\mathbb{Z} \subset 2^{-s\lceil \log_2(\frac{1}{\delta})\rceil}\mathbb{Z}.$$

  Letzteres gilt, weil $\lceil \log_2(\frac{1}{\varepsilon})\rceil \le \lceil \log_2(\frac{1}{\delta})\rceil$.

- Seien $\varepsilon \in (0, \frac{1}{2})$, $q \in (0, \infty)$, $C \ge 1$ und $s \in \mathbb{N}$. Ist Φ ein Netz mit $(s, \varepsilon^q/C)$-quantisierten Gewichten, so sind die Gewichte, mit $\tilde{s} := \lceil qs + s\log_2(C)\rceil + s$, auch $(\tilde{s}, \varepsilon)$-quantisiert. Das gilt, weil

$$\varepsilon^{-\tilde{s}} \ge \varepsilon^{-qs - s\log_2(C)} = \varepsilon^{-qs} \cdot \left(\frac{1}{\varepsilon}\right)^{s\log_2(C)}$$
$$\overset{\varepsilon < \frac{1}{2}}{\ge} \varepsilon^{-qs} \cdot 2^{s\log_2(C)} = \varepsilon^{-qs} \cdot C^s = \left(\frac{\varepsilon^q}{C}\right)^{-s},$$

und damit $\left[-\left(\frac{\varepsilon^q}{C}\right)^{-s}, \left(\frac{\varepsilon^q}{C}\right)^{-s}\right] \subset [-\varepsilon^{-\tilde{s}}, \varepsilon^{-\tilde{s}}]$, und da

$$\begin{aligned}\frac{s \cdot \lceil \log_2(\frac{1}{\varepsilon^q/C}) \rceil}{\tilde{s} \cdot \lceil \log_2(\frac{1}{\varepsilon}) \rceil} &\leq \frac{s(q \log_2(\frac{1}{\varepsilon}) + \log_2(C) + 1)}{(qs + s\log_2(C) + s)\log_2(\frac{1}{\varepsilon})} \\ &= \frac{sq\log_2(\frac{1}{\varepsilon}) + s\log_2(C) + s}{sq\log_2(\frac{1}{\varepsilon}) + s\log_2(C)\log_2(\frac{1}{\varepsilon}) + s\log_2(\frac{1}{\varepsilon})} \leq 1\end{aligned}$$

und daher $2^{s \cdot \lceil \log_2(\frac{1}{\varepsilon^q/C}) \rceil}\mathbb{Z} \subset 2^{\tilde{s} \cdot \lceil \log_2(\frac{1}{\varepsilon}) \rceil}\mathbb{Z}$. Wir haben dabei im dritten Schritt $\log_2(\frac{1}{\varepsilon}) > 1$ genutzt.

2.2 Konstruktion und Kombination neuronaler Netze

Im Folgenden führen wir einige grundlegende Definitionen und Hilfsaussagen zur Konstruktion und Kombination von neuronalen Feedforward-Netzwerken ein, welche sich im weiteren Verlauf dieser Arbeit als äußerst nützlich erweisen werden. Da sich manche dieser Aussagen für allgemeine Feedforward-Netzwerke als komplex und unübersichtlich herausstellen würden, werden wir uns wo möglich auf Multilayer Perceptrons beschränken.

Zunächst sei angemerkt, dass ein MLP, dessen Realisierung der Identitätsabbildung $\mathrm{Id}_{\mathbb{R}^d}$ entspricht, auf einfache Weise konstruiert werden kann.

Lemma 2.13. *Sei $d \in \mathbb{N}$. Definiere*

$$\Phi_d^{\mathrm{Id}} := ((A_1, b_1), (A_2, b_2))$$

mit

$$A_1 := \begin{pmatrix} \mathrm{Id}_{\mathbb{R}^d} \\ -\mathrm{Id}_{\mathbb{R}^d} \end{pmatrix} \quad b_1 := 0, \quad A_2 := \begin{pmatrix} \mathrm{Id}_{\mathbb{R}^d} & -\mathrm{Id}_{\mathbb{R}^d} \end{pmatrix} \quad b_2 := 0.$$

Dann ist $R_\varrho(\Phi_d^{\mathrm{Id}}) = \mathrm{Id}_{\mathbb{R}^d}$.

Beweis. Sei $\varrho_n : \mathbb{R}^n \to \mathbb{R}^n,\ (y_1, \ldots, y_n) \mapsto (\varrho(y_1), \ldots, \varrho(y_n))$. Dann ist, mit $x_0 := y$,

$$\begin{aligned} x_1 &= \varrho_{2d}(A_1 y + 0) = \begin{pmatrix} \varrho_d(y) \\ \varrho_d(-y) \end{pmatrix} \\ x_2 &= A_2 x_1 + 0 = \varrho_d(y) - \varrho_d(-y) = y = R_\varrho(\Phi_d^{\mathrm{Id}})(y). \end{aligned}$$

□

Bemerkung 2.14. Allgemeiner kann man für alle $d \in \mathbb{N}$ und $L \in \mathbb{N}_{\geq 2}$ ein MLP $\Phi_{d,L}^{\mathrm{Id}}$ mit L Schichten und höchstens $2d \cdot L$ nichtnull Gewichten aus $\{-1, 1\}$ konstruieren, sodass $R_\varrho(\Phi_{d,L}^{\mathrm{Id}}) = \mathrm{Id}_{\mathbb{R}^d}$. Zum Beispiel kann

$$\Phi_{d,L}^{\mathrm{Id}} := \left(\left(\begin{pmatrix} \mathrm{Id}_{\mathbb{R}^d} \\ -\mathrm{Id}_{\mathbb{R}^d} \end{pmatrix}, 0 \right), \underbrace{(\mathrm{Id}_{\mathbb{R}^{2d}}, 0), \ldots, (\mathrm{Id}_{\mathbb{R}^{2d}}, 0)}_{L-2 \text{ Mal}}, \left(\begin{pmatrix} \mathrm{Id}_{\mathbb{R}^d} & -\mathrm{Id}_{\mathbb{R}^d} \end{pmatrix}, 0 \right) \right)$$

verwendet werden.

Um neue MLPs aus bereits existierenden zu konstruieren, werden wir diese häufig verketten. Zu diesem Zweck definieren wir zunächst die *Verkettung* von MLPs.

Definition 2.15. *[Verkettung] Seien $L_1, L_2 \in \mathbb{N}$ und seien $\Phi^1 = ((A_1^1, b_1^1), \dots, (A_{L_1}^1, b_{L_1}^1))$ und $\Phi^2 = ((A_1^2, b_1^2), \dots, (A_{L_2}^2, b_{L_2}^2))$ zwei Multilayer Perceptrons, sodass die Eingabedimension von Φ^1 gleich der Ausgabedimension von Φ^2 ist. Dann bezeichnet $\Phi^1 \bullet \Phi^2$ das folgende Multilayer Perceptron mit $L_1 + L_2 - 1$ Schichten:*

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

*Wir nennen $\Phi^1 \bullet \Phi^2$ die **Verkettung** von Φ^1 und Φ^2.*

Mithilfe von Bemerkung 2.8 lässt sich leicht nachweisen, dass $R_\rho(\Phi^1 \bullet \Phi^2) = R_\rho(\Phi^1) \circ R_\rho(\Phi^2)$ für eine beliebige Aktivierungsfunktion ρ gilt. Sei $x_{L_2}^{(2)} = R_\rho(\Phi^2)(x)$.

$$x_{L_2} = A_1^1 A_{L_2}^2 x_{L_2-1} + A_1^1 b_{L_2}^2 + b_1^1 = A_1^1 (A_{L_2}^2 x_{L_2-1} + b_{L_2}^2) + b_1^1 = A_1^1 x_{L_2}^{(1)} + b_1^1.$$

Damit folgt direkt die obige Aussage. Diese Definition der Verkettung ist also sinnvoll.

Mit der obigen Definition der Verkettung und Lemma 2.13 lässt sich nun eine Art der Verkettung definieren, bei der wir genaue Kontrolle über die Anzahl der Gewichte und die Quantisierung des resultierenden MLPs haben. Es ist hierbei wichtig zu beachten, dass dies nur für die ReLU Aktivierungsfunktion $\rho = \varrho$ gilt.

Definition 2.16. *[spärliche Verkettung] Seien $L_1, L_2 \in \mathbb{N}$ und seien zwei Multilayer Perceptrons $\Phi^1 = ((A_1^1, b_1^1), \dots, (A_{L_1}^1, b_{L_1}^1))$ und $\Phi^2 = ((A_1^2, b_1^2), \dots, (A_{L_2}^2, b_{L_2}^2))$ gegeben, sodass die Eingabedimension von Φ^1 gleich der Ausgabedimension von Φ^2 ist. Sei Φ_d^{Id} wie in Lemma 2.13. Dann ist die **spärliche Verkettung** von Φ^1 und Φ^2 definiert als*

$$\Phi^1 \odot \Phi^2 := \Phi^1 \bullet \Phi_d^{\mathrm{Id}} \bullet \Phi^2.$$

Bemerkung 2.17. Es ist leicht zu sehen, dass

$$\Phi^1 \odot \Phi^2 = \left((A_1^2, b_1^2), \dots, (A_{L_2-1}^2, b_{L_2-1}^2), \left(\begin{pmatrix} A_{L_2}^2 \\ -A_{L_2}^2 \end{pmatrix}, \begin{pmatrix} b_{L_2}^2 \\ -b_{L_2}^2 \end{pmatrix} \right), \left(\left(\begin{pmatrix} A_1^1 & -A_1^1 \end{pmatrix}, b_1^1 \right), (A_2^1, b_2^1), \dots, (A_{L_1}^1, b_{L_1}^1) \right) \right)$$

$L_1 + L_2$ Schichten hat, und, dass $R_\varrho(\Phi^1 \odot \Phi^2) = R_\varrho(\Phi^1) \circ R_\varrho(\Phi^2)$. Zudem können wir mit dieser Darstellung und der Ungleichung $a + b \leq 2 \max\{a.b\}$ ableiten, dass $M(\Phi^1 \odot \Phi^2) \leq 2M(\Phi^1) + 2M(\Phi^2) \leq 4 \cdot \max\{M(\Phi^1), M(\Phi^2)\}$. Induktiv folgt daraus:

$$M(\Phi^1 \odot \cdots \odot \Phi^n) \leq 4^{n-1} \cdot \max\{M(\Phi^1), \dots, M(\Phi^n)\}.$$

Da sich die obige Herleitung der spärlichen Verkettung für allgemeine Feedforward-Netze als relativ unübersichtlich herausstellen würde, erweitern wir die Definition "händisch" auf allgemeine Feedforward-Netze.

Definition 2.18. *[spärliche Verkettung für allgemeine Feedforward-Netze] Sei $d \in \mathbb{N}$ und seien*

$$\Phi^1 = \left(L_1, \{N_\ell^{(1)}\}_{\ell=0}^{L_1}, \{A_{k,\ell}^1\}_{0\le k<\ell\le L_1}, \{b_\ell^1\}_{\ell=1}^{L_1}\right),$$
$$\Phi^2 = \left(L_2, \{N_\ell^{(2)}\}_{\ell=0}^{L_2}, \{A_{k,\ell}^2\}_{0\le k<\ell\le L_2}, \{b_\ell^2\}_{\ell=1}^{L_2}\right),$$

*zwei neuronale Feedforward-Netzwerke, wobei Φ^1 Eingabedimension d und Φ^2 Ausgabedimension d hat. Dann ist die **spärliche Verkettung** von Φ^1 und Φ^2 gegeben durch $\Phi^1 \odot \Phi^2 := \left(L, \{N_\ell\}_{\ell=0}^{L}, \{A_{k,\ell}\}_{0\le k<\ell\le L}, \{b_\ell\}_{\ell=1}^{L}\right)$ mit $L := L_1 + L_2$ und*

$$N_\ell := N_\ell^{(2)} \text{ für } \ell = 0,\dots,L_2-1 \text{ und } N_\ell := N_\ell^{(1)} \text{ für } \ell = L_2+1,\dots,L_2+L_1,$$
$$A_{k,\ell} := A_{k,\ell}^2 \text{ für } 0 \le k < \ell \le L_2 - 1,$$
$$A_{k,\ell} := A_{k,\ell}^1 \text{ für } L_2+1 \le k < \ell \le L_2+L_1,$$
$$b_\ell := b_\ell^2 \text{ für } \ell = 1,\dots,L_2-1 \text{ und } b_\ell := b_\ell^1 \text{ für } \ell = L_2+1,\dots,L_2+L_1,$$

sowie $N_{L_2} := 2d$ und

$$A_{k,L_2} := \begin{pmatrix} A_{k,L_2}^2 \\ -A_{k,L_2}^2 \end{pmatrix} \text{ für } k = 0,\dots,L_2-1,$$
$$A_{L_2,\ell} := \begin{pmatrix} A_{0,\ell-L_2}^1 & -A_{0,\ell-L_2}^1 \end{pmatrix} \text{ für } \ell = L_2+1,\dots,L_2+L_1,$$
$$b_{L_2} := \begin{pmatrix} b_{L_2}^2 \\ -b_{L_2}^2 \end{pmatrix}.$$

Außerdem seien alle nicht genannten $A_{k,\ell} := 0$ in der entsprechenden Dimension.

Bemerkung 2.19. Auch für diese Definition der spärlichen Verkettung gilt $R_\varrho(\Phi^1 \odot \Phi^2) = R_\varrho(\Phi^1) \circ R_\varrho(\Phi^2)$. Seien $x_k^{(2)}$ und $x_k^{(1)}$ jeweils die Zwischenberechnungen für Φ^2 und Φ^1 nach Definition 2.2. Dann ist

$$\begin{aligned} x_{L_2} &= \varrho_{2d}\left(\sum_{k=0}^{L_2-1} A_{k,L_2} x_k + b_{L_2}\right) \\ &= \varrho_{2d}\left(\sum_{k=0}^{L_2-1} \begin{pmatrix} A_{k,L_2}^2 \\ -A_{k,L_2}^2 \end{pmatrix} x_k^{(2)} + \begin{pmatrix} b_{L_2}^2 \\ -b_{L_2}^2 \end{pmatrix}\right) = \begin{pmatrix} \varrho_d(x_{L_2}^{(2)}) \\ \varrho_d(-x_{L_2}^{(2)}) \end{pmatrix}. \end{aligned}$$

Mit $x_0^{(1)} := x_{L_2}^{(2)} = R_\varrho(\Phi^2)(x)$ folgt nun für alle $L_2 + 1 \le \ell \le L_2 + L_1 - 1$:

$$\begin{aligned} x_\ell &= \varrho_{N_\ell}\left(\sum_{k=0}^{\ell-1} A_{k,\ell} x_k + b_\ell\right) = \varrho_{N_\ell}\left(\sum_{k=L_2}^{\ell-1} A_{k,\ell} x_k + b_\ell\right) \\ &= \varrho_{N_\ell}\left(\sum_{k=L_2+1}^{\ell-1} A_{k,\ell} x_k + b_\ell + \begin{pmatrix} A_{0,\ell}^1 & -A_{0,\ell}^1 \end{pmatrix} \begin{pmatrix} \varrho_d(x_{L_2}^{(2)}) \\ \varrho_d(-x_{L_2}^{(2)}) \end{pmatrix}\right) \\ &= \varrho_{N_\ell}\left(\sum_{k=L_2+1}^{\ell-1} A_{k,\ell} x_k + b_\ell + A_{0,\ell}^1 x_{L_2}^{(2)}\right) = \varrho_{N_{\ell'}^{(1)}}\left(\sum_{k=0}^{\ell'-1} A_{k,\ell'}^1 x_k^{(1)} + b_{\ell'}^1\right), \end{aligned}$$

wobei $\ell' := \ell - L_2 \in \{1, \dots, L_1 - 1\}$. Im zweiten Schritt haben wir hierbei verwendet, dass $A_{k,\ell} = 0$ für $k < L_2$. Im letzten Schritt haben wir verwendet, dass die x_k sequenziell berechnet werden, womit die Aussage induktiv folgt.

Eine analoge Rechnung gilt für $x_{L_2+L_1}$ (ohne Anwendung von ϱ_{N_ℓ}) und damit ist $R_\varrho(\Phi^1 \odot \Phi^2)(x) = R_\varrho(\Phi^1)(R_\varrho(\Phi^2)(x))$.

Außerdem erfüllt die spärliche Verkettung

$$\begin{aligned} M(\Phi^1 \odot \Phi^2) &\leq M(\Phi^1) + M(\Phi^2) + \sum_{k=0}^{L_2-1} \|A^2_{k,L_2}\|_{\ell^0} + \sum_{\ell=1}^{L_1} \|A^1_{0,\ell}\|_{\ell^0} + \|b^2_{L_2}\|_{\ell^0} \\ &\leq M(\Phi^1) + M(\Phi^2) + \sum_{k=0}^{L_2-1} \|A^2_{k,L_2}\|_{\ell^0} + \sum_{\ell=1}^{L_1} \|A^1_{0,\ell}\|_{\ell^0} + d, \\ N(\Phi^1 \odot \Phi^2) &\leq N(\Phi^1) + N(\Phi^2) + \|b^2_{L_2}\|_{\ell^0} \leq N(\Phi^1) + N(\Phi^2) + d. \end{aligned}$$

Die Anzahl der zusätzlich eingeführten nichtnull Gewichte hängt also von der Dimension d sowie der Anzahl der Verbindungen in die letzte Schicht von Φ^2 und aus der ersten Schicht von Φ^1 ab. Eine *eingehende Verbindung in Schicht ℓ* meint dabei ein nichtnull Gewicht in einem $A_{k,\ell}$ und eine *ausgehende Verbindung aus Schicht ℓ* ein nichtnull Gewicht in einem $A_{\ell,k}$.

Beispiel 2.20. Seien $\Phi^1 = ((A^1_1, b^1_1), (A^1_2, b^1_2))$ und $\Phi^2 = ((A^2_1, b^2_1), (A^2_2, b^2_2))$ Multilayer Perceptrons:

$$\begin{aligned} \Phi^1 &:= \left(\left(\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right), \left(\begin{bmatrix} 2 & -1 \end{bmatrix}, \begin{bmatrix} 3 \end{bmatrix} \right) \right), \\ \Phi^2 &:= \left(\left(\begin{bmatrix} 2 & 0 \\ 0 & -1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right), \left(\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 2 \\ -1 \end{bmatrix} \right) \right). \end{aligned}$$

Dann sind deren Verkettungen gegeben durch

$$\begin{aligned} \Phi^1 \bullet \Phi^2 &= \left(\left(A^2_1, b^2_1 \right), \left(\begin{bmatrix} 1 & 0 \\ -1 & -1 \end{bmatrix}, \begin{bmatrix} 3 \\ 1 \end{bmatrix} \right), \left(A^1_2, b^1_2 \right) \right), \\ \Phi^1 \odot \Phi^2 &= \left(\left(A^2_1, b^2_1 \right), \left(\begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 2 \\ -1 \\ 2 \\ -1 \end{bmatrix} \right), \left(\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & -1 & 0 & -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right), \left(A^1_2, b^1_2 \right) \right), \end{aligned}$$

Durch Nachrechnen lässt sich bestätigen, dass $R_\varrho(\Phi^1 \bullet \Phi^2) = R_\varrho(\Phi^1 \odot \Phi^2) = R_\varrho(\Phi^1) \circ R_\varrho(\Phi^2)$.

Neben der Verkettung können zwei Netze auch parallel geschaltet werden. Dieses Konzept werden wir ebenfalls für allgemeine Feedforward-Netze benötigen und daher auch für diese definieren.

Definition 2.21. *[Parallelisierung] Sei $L \in \mathbb{N}$ und seien*

$$\Phi^1 = \left(L, \{N_\ell^{(1)}\}_{\ell=0}^L, \{A_{k,\ell}^1\}_{0\leq k<\ell\leq L}, \{b_\ell^1\}_{\ell=1}^L\right),$$
$$\Phi^2 = \left(L, \{N_\ell^{(2)}\}_{\ell=0}^L, \{A_{k,\ell}^2\}_{0\leq k<\ell\leq L}, \{b_\ell^2\}_{\ell=1}^L\right),$$

zwei neuronale Feedforward-Netzwerke mit L Schichten, Eingabedimension d und Ausgabedimensionen m_1 und m_2. Wir definieren

$$P(\Phi^1, \Phi^2) := \left(L, \{N_\ell\}_{\ell=0}^L, \{A_{k,\ell}\}_{0\leq k<\ell\leq L}, \{b_\ell\}_{\ell=1}^L\right),$$

wobei

$$N_\ell := N_\ell^{(1)} + N_\ell^{(2)} \quad \text{für} \quad \ell = 1, \ldots, L \quad \text{und} \quad N_0 := d$$
$$A_{0,\ell} := \begin{pmatrix} A_{0,\ell}^1 \\ A_{0,\ell}^2 \end{pmatrix}, \quad \text{für} \quad 1 \leq \ell \leq L,$$
$$A_{k,\ell} := \begin{pmatrix} A_{k,\ell}^1 & 0 \\ 0 & A_{k,\ell}^2 \end{pmatrix}, \quad \text{für} \quad 1 \leq k < \ell \leq L,$$
$$b_\ell := \begin{pmatrix} b_\ell^1 \\ b_\ell^2 \end{pmatrix} \quad \text{für} \quad 1 \leq \ell \leq L.$$

*$P(\Phi^1, \Phi^2)$ ist ein neuronales Feedforward-Netzwerk mit L Schichten, Eingabedimension d und Ausgabedimension $m_1 + m_2$ und wird als die **Parallelisierung** von Φ^1 und Φ^2 bezeichnet.*

Per Konstruktion gelten $M(P(\Phi^1, \Phi^2)) = M(\Phi^1) + M(\Phi^2)$ und $N(P(\Phi^1, \Phi^2)) \leq N(\Phi^1) + N(\Phi^2)$ sowie

$$R_\rho(P(\Phi^1, \Phi^2))(x) = (R_\rho(\Phi^1)(x), R_\rho(\Phi^2)(x)) \text{ für alle } x \in \mathbb{R}^d.$$

Analog hierzu definieren wir die *disjunkte Parallelisierung*, bei der die parallelisierten Netze im Gegensatz zu oben auch disjunkte Eingaben haben. Diese Konstruktion kann beispielsweise nützlich sein, wenn dieselbe Funktion auf verschiedene Eingaben angewendet werden soll.

Definition 2.22. *[disjunkte Parallelisierung] Sei $L \in \mathbb{N}$ und seien*

$$\Phi^1 = \left(L, \{N_\ell^{(1)}\}_{\ell=0}^L, \{A_{k,\ell}^1\}_{0\leq k<\ell\leq L}, \{b_\ell^1\}_{\ell=1}^L\right),$$
$$\Phi^2 = \left(L, \{N_\ell^{(2)}\}_{\ell=0}^L, \{A_{k,\ell}^2\}_{0\leq k<\ell\leq L}, \{b_\ell^2\}_{\ell=1}^L\right),$$

zwei neuronale Feedforward-Netzwerke mit L Schichten, Eingabedimensionen d_1 und d_2 und Ausgabedimensionen m_1 und m_2. Wir definieren

$$\mathcal{P}(\Phi^1, \Phi^2) := \left(L, \{N_\ell\}_{\ell=0}^L, \{A_{k,\ell}\}_{0\leq k<\ell\leq L}, \{b_\ell\}_{\ell=1}^L\right),$$

wobei

$$N_\ell := N_\ell^{(1)} + N_\ell^{(2)} \quad \textit{für} \ \ell = 0, \dots, L,$$
$$A_{k,\ell} := \begin{pmatrix} A_{k,\ell}^1 & 0 \\ 0 & A_{k,\ell}^2 \end{pmatrix}, \quad \textit{für} \ 0 \leq k < \ell \leq L,$$
$$b_\ell := \begin{pmatrix} b_\ell^1 \\ b_\ell^2 \end{pmatrix} \quad \textit{für} \ 1 \leq \ell \leq L.$$

$\mathcal{P}(\Phi^1, \Phi^2)$ *ist ein neuronales Feedforward-Netzwerk mit* L *Schichten, Eingabedimension* $d_1 + d_2$ *und Ausgabedimension* $m_1 + m_2$ *und wird als die* ***disjunkte Parallelisierung*** *von* Φ^1 *und* Φ^2 *bezeichnet.*

Die disjunkte Parallelisierung erfüllt ebenfalls $M(\mathcal{P}(\Phi^1, \Phi^2)) = M(\Phi^1) + M(\Phi^2)$ und außerdem $N(\mathcal{P}(\Phi^1, \Phi^2)) = N(\Phi^1) + N(\Phi^2)$. Zudem gilt für alle $x \in \mathbb{R}^{d_1}$ und $y \in \mathbb{R}^{d_2}$:

$$R_\rho(\mathcal{P}(\Phi^1, \Phi^2))((x, y)) = (R_\rho(\Phi^1)(x), R_\rho(\Phi^2)(y)).$$

Die (disjunkte) Parallelisierung zweier Multilayer Perceptrons ist ebenfalls ein Multilayer Perceptron.

Beispiel 2.23. Seien $\Phi^1 = ((A_1^1, b_1^1), (A_2^1, b_2^1))$ und $\Phi^2 = ((A_1^2, b_1^2), (A_2^2, b_2^2))$ die beiden MLPs aus Beispiel 2.20:

$$\Phi^1 := \left(\left(\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right), \left(\begin{bmatrix} 2 & -1 \end{bmatrix}, \begin{bmatrix} 3 \end{bmatrix} \right) \right),$$
$$\Phi^2 := \left(\left(\begin{bmatrix} 2 & 0 \\ 0 & -1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right), \left(\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 2 \\ -1 \end{bmatrix} \right) \right).$$

Dann sind

$$P(\Phi^1, \Phi^2) = \left(\left(\begin{bmatrix} 1 & 0 \\ 0 & -1 \\ 2 & 0 \\ 0 & -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right), \left(\begin{bmatrix} 2 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \\ -1 \end{bmatrix} \right) \right),$$
$$\mathcal{P}(\Phi^1, \Phi^2) = \left(\left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right), \left(\begin{bmatrix} 2 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \\ -1 \end{bmatrix} \right) \right).$$

3 Obere Schranken für Approximationsraten mit ReLU Netzen

In diesem Kapitel beschäftigen wir uns mit theoretischen Ergebnissen zu Approximationsraten von neuronalen Feedforward-Netzwerken mit ReLU Aktivierungsfunktion. Genauer formulieren wir zentrale Ergebnisse zu oberen Schranke für die Netzwerkkomplexität, d.h. zur erforderlichen Tiefe sowie zur Anzahl an Gewichten und Neuronen, um eine bestimmte Genauigkeit ε in der Approximation von Funktionen mit einer gewissen Regularität erreichen zu können.

Zuerst betrachten wir, wie Yarotsky in [9], Funktionen aus den Einheitsbällen der Sobolev-Räume $\mathcal{W}^{n,\infty}([0,1]^d)$ und deren Approximation in der L^∞-Norm. Anschließend beschäftigen wir uns, wie Petersen und Voigtlaender in [7], mit den verwandten Räumen $\mathcal{F}_{d,\beta,B}$, die Funktionen einer bestimmten "Glattheit" auf den Intervallen $[-\frac{1}{2},\frac{1}{2}]^d$ umfassen. Hierbei untersuchen wir die Approximation in der L^p-Norm für $p \in (0,\infty)$ und berücksichtigen insbesondere auch die Quantisierung der Gewichte.

Die folgenden Teilkapitel stützen sich maßgeblich auf die oben genannten Arbeiten von Yarotsky [9] sowie Petersen und Voigtlaender [7].

3.1 Approximation in Sobolev-Räumen bezüglich der L^∞ Norm

Zunächst übernehmen wir wesentliche Teile der Arbeit von Yarotsky [9], bereiten zentrale Aspekte seiner Ergebnisse neu auf und präzisieren deren Formulierungen und Beweise. Unser Ziel ist es, eine obere Schranke für die minimal benötigte Tiefe sowie Anzahl an nichtnull Gewichten und Neuronen eines neuronalen Feedforward-Netzwerks für die L^∞-Approximation von Funktionen in den Einheitsbällen der Sobolev-Räume $\mathcal{W}^{n,\infty}([0,1]^d)$ zu formulieren. Diese Sobolev-Räume umfassen $L^\infty([0,1]^d)$-Funktionen, deren schwache Ableitungen bis zum Grad n ebenfalls in diesem L^∞-Raum liegen. Die Norm in diesen Räumen ist gegeben durch

$$\|f\|_{\mathcal{W}^{n,\infty}([0,1]^d)} = \max_{|\alpha|\leq n} \operatorname*{ess\,sup}_{x\in[0,1]^d} |\partial^\alpha f(x)|.$$

Hierbei sind $\alpha \in \mathbb{N}_0^d$ Multiindizes und $\partial^\alpha f$ die schwachen Ableitungen von f. Dieser Raum lässt sich äquivalent auch als der Raum der Funktionen aus $C^{n-1}([0,1]^d)$ beschreiben,

deren schwache Ableitungen bis zum Grad n lipschitzstetig sind. Diese Charakterisierung wird uns im nächsten Unterkapitel die Ähnlichkeit dieser Räume zu den dort betrachteten Funktionenklassen aufzeigen. In dieser Arbeit betrachten wir jedoch nicht die allgemeinen Sobolev-Räume, sondern nur deren Einheitsbälle

$$F_{n,d} := \{f \in \mathcal{W}^{n,\infty}([0,1]^d) : \|f\|_{\mathcal{W}^{n,\infty}([0,1]^d)} \leq 1\}.$$

Wir werden zunächst zeigen, dass sich die Funktion $f(x) = x^2$ effizient durch ReLU Netze approximieren lässt und dieses Resultat anschließend nutzen, um eine Approximation der Multiplikation zu konstruieren. Diese Multiplikation werden wir dann verwenden, um Funktionen aus $F_{n,d}$ mithilfe ihrer lokalen Taylor-Polynome zu approximieren.

Zunächst betrachten wir daher ein Resultat zur Annäherung der Quadratfunktion mittels neuronaler Feedforward-Netzwerke.

Lemma 3.1. *Sei $f : [0,1] \to \mathbb{R}$ gegeben durch $f(x) = x^2$. Es gibt eine universelle Konstante $c > 0$, sodass für jedes $\varepsilon \in (0, \frac{1}{2})$ ein neuronales Feedforward-Netzwerk Φ_ε existiert mit:*

- *Die Anzahl der nichtnull Gewichte und Neuronen von Φ_ε ist höchstens $c \cdot \ln\left(\frac{1}{\varepsilon}\right)$:*

$$N(\Phi_\varepsilon), M(\Phi_\varepsilon) \leq c \cdot \ln\left(\frac{1}{\varepsilon}\right).$$

- *Φ_ε hat höchstens $L \leq c \cdot \ln\left(\frac{1}{\varepsilon}\right)$ Schichten.*
- *Für die Realisierung mit ReLU Aktivierungsfunktion von Φ_ε gilt:*

$$\|R_\varrho(\Phi_\varepsilon) - f\|_{L^\infty([0,1])} \leq \varepsilon.$$

Beweis. Wir Betrachten die "Zahn"-Funktion $g : [0,1] \to [0,1]$,

$$g(x) := \begin{cases} 2x, & x \leq \frac{1}{2}, \\ 2(1-x), & x > \frac{1}{2}, \end{cases}$$

und ihre t-fachen Iterationen g_t,

$$g_t(x) = \underbrace{g \circ \cdots \circ g}_{t \text{ Mal}}(x).$$

Telgarsky zeigte in [8], dass diese g_t "Sägezahn"-Funktionen mit 2^{t-1} gleichmäßig verteilten Spitzen sind:

$$g_t(x) := \begin{cases} 2^t(x - \frac{2k}{2^t}), & x \in [\frac{2k}{2^t}, \frac{2k+1}{2^t}], \quad k = 0, 1, \ldots, 2^{t-1} - 1, \\ 2^t(\frac{2k}{2^t} - x), & x \in [\frac{2k-1}{2^t}, \frac{2k}{2^t}], \quad k = 1, 2, \ldots, 2^{t-1}. \end{cases} \tag{3.1}$$

Siehe hierfür auch Abbildung 3.1 (c). Linearkombinationen dieser Sägezahn-Funktionen wollen wir nun verwenden, um $f(x) = x^2$ zu approximieren. Sei hierfür f_m die stückweise lineare Interpolation von f an 2^m+1 gleichmäßig verteilten Stützstellen $\frac{k}{2^m}$, $k = 0, \dots, 2^m$:

$$f_m\left(\frac{k}{2^m}\right) = \left(\frac{k}{2^m}\right)^2. \tag{3.2}$$

Wir zeigen nun, dass $\varepsilon_m := \|f - f_m\|_{L^\infty} = 2^{-2m-2}$. Hierfür stellen wir zunächst fest, dass dank der Konvexität von f gilt, dass $f \leq f_m$. Mit $h_m := f_m - f$ ist damit insbesondere $\|f - f_m\|_{L^\infty} = \max_{x\in[0,1]}(f_m(x) - f(x)) = \max_{x\in[0,1]} h_m(x)$. Sei nun $x \in [\frac{k}{2^m}, \frac{k+1}{2^m}] =: I_k$, $k = 0, 1, \dots, 2^m - 1$. Per Konstruktion ist $h_m(\frac{k}{2^m}) = h_m(\frac{k+1}{2^m}) = 0$, sodass wir die Randfälle nicht weiter betrachten müssen.

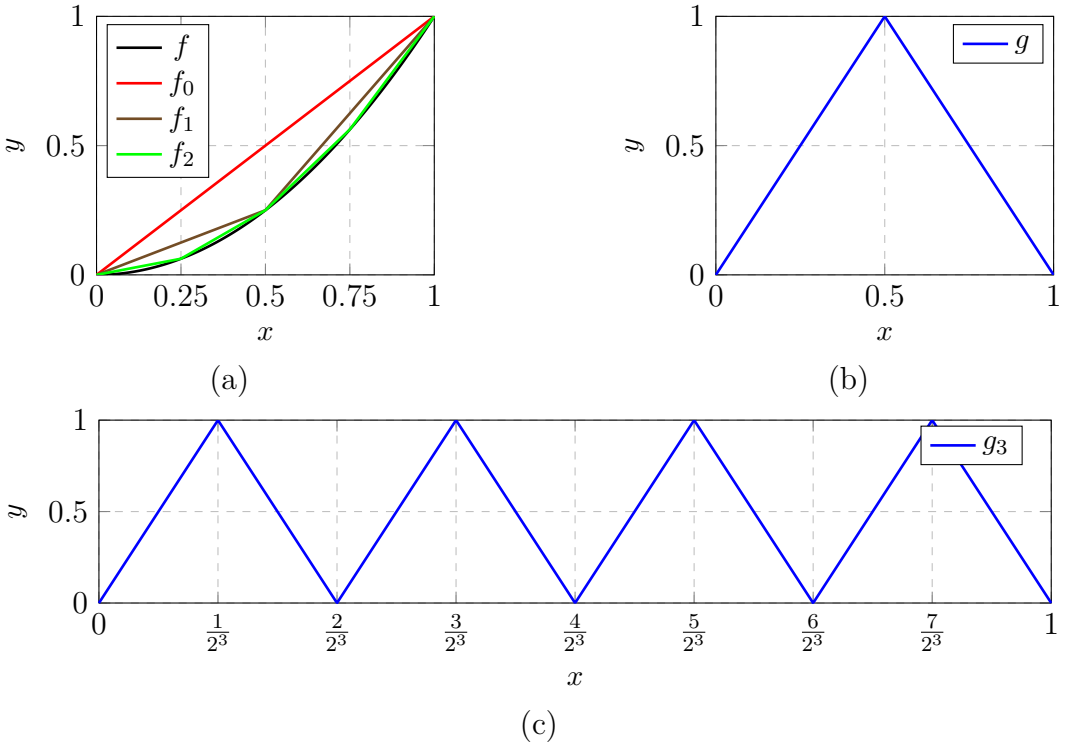

Abbildung 3.1: (a) $f(x) = x^2$ und die stückweise linearen Interpolationen f_m. (b) Zahn-Funktion $g = g_1$. (c) Sägezahn-Funktion g_3.

Beschränken wir uns nun auf ein Intervall I_k, so beschreibt $h_m|_{I_k}$ eine nach unten geöffnete Parabel und wir können das Maximum auf I_k finden, indem wir $h_m|_{I_k}$ in Scheitelform bringen (siehe Abbildung 3.2). Hierbei nutzen wird die bekannte Formel für

eine affine Funktion durch zwei Punkte:

$$\begin{aligned} h_m|_{I_k}(x) = f_m|_{I_k}(x) - f|_{I_k}(x) &= \left(\frac{(\frac{k+1}{2^m})^2 - (\frac{k}{2^m})^2}{\frac{k+1}{2^m} - \frac{k}{2^m}} (x - \frac{k}{2^m}) + (\frac{k}{2^m})^2 \right) - x^2 \\ &= -x^2 + \frac{2k+1}{2^m} x - \frac{k^2+k}{2^{2m}} \\ &= -\left(x - \frac{2k+1}{2^{m+1}} \right)^2 + \frac{1}{2^{2m+2}}. \end{aligned}$$

Damit wird das Maximum $\max_{x \in I_k} h_m(x) = 2^{-2m-2}$ an der Stelle $\frac{2k+1}{2^{m+1}}$, also in der Mitte des Intervalls I_k, angenommen. Da dieses von k unabhängig ist, ist dies auch das globale Maximum $\|f - f_m\|_{L^\infty} = \max_{x \in [0,1]} h_m(x) = 2^{-2m-2}$. Außerdem halten wir fest:

$$h_m \left(\frac{2k+1}{2^{m+1}} \right) = \frac{1}{2^{2m+2}}. \tag{3.3}$$

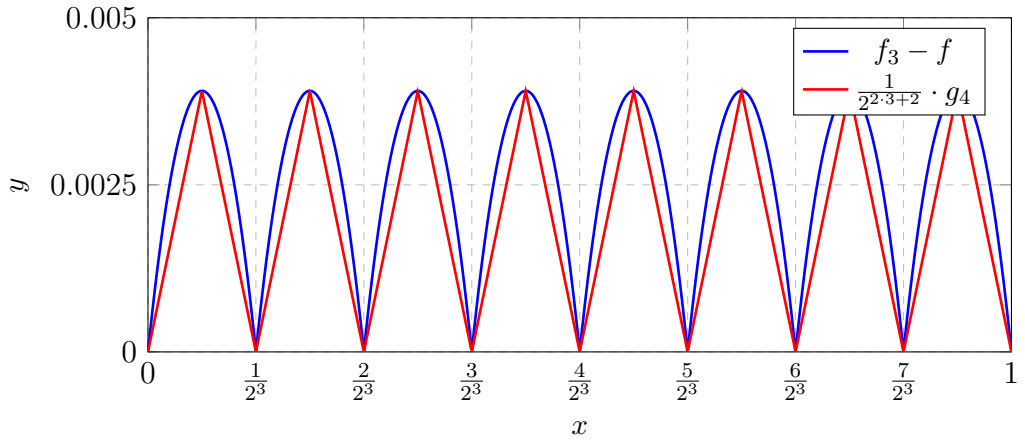

Abbildung 3.2: Differenz $f_m - f$ und Sägezahn-Funktion $\frac{1}{2^{2m+2}} g_{m+1}$ für $m = 3$. Subtraktion von $\frac{1}{2^{2m+2}} g_{m+1}$ scheint der logische nächste Schritt zu sein, um die Approximation zu verbessern.

Als nächstes stellen wir fest, dass $f_m(x) - \frac{g_{m+1}(x)}{2^{2m+2}} =: \tilde{f}_{m+1}(x) = f_{m+1}(x)$, $m \in \mathbb{N}_0$. Dies verifizieren wir: Falls $x = \frac{2k}{2^{m+1}}$ mit $k = 0, 1, \ldots, 2^m$, so gilt (3.2) für $\tilde{f}_{m+1}$, denn

$$\begin{aligned} \tilde{f}_{m+1}(x) &= f_m \left(\frac{2k}{2^{m+1}} \right) - \frac{g_{m+1}\left(\frac{2k}{2^{m+1}}\right)}{2^{2m+2}} \\ &\overset{(3.1)}{=} f_m \left(\frac{k}{2^m} \right) - 0 \overset{(3.2) \text{ für } f_m}{=} f(x). \end{aligned}$$

Falls $x = \frac{2k+1}{2^{m+1}}$ mit $k = 0, 1, \ldots, 2^m - 1$, dann gilt ebenfalls (3.2), denn

$$\begin{aligned}
\tilde{f}_{m+1}(x) &= f_m(x) - \frac{g_{m+1}(x)}{2^{2m+2}} = h_m(x) + f(x) - \frac{g_{m+1}(x)}{2^{2m+2}} \\
&= f\left(\frac{2k+1}{2^{m+1}}\right) + h_m\left(\frac{2k+1}{2^{m+1}}\right) - \frac{g_{m+1}\left(\frac{2k+1}{2^{m+1}}\right)}{2^{2m+2}} \\
&\overset{(3.3),\ (3.1)}{=} f(x) + \frac{1}{2^{2m+2}} - \frac{1}{2^{2m+2}} \\
&= f(x).
\end{aligned}$$

Siehe hierfür auch Abbildung 3.2. Damit wird (3.2) von $\tilde{f}_{m+1}$ erfüllt. Da sowohl g_{m+1} als auch f_m auf den Intervallen $J_\ell := [\frac{\ell}{2^{m+1}}, \frac{\ell+1}{2^{m+1}}]$, $\ell = 0, 1, \ldots, 2^{m+1} - 1$, affin sind, muss damit auch $\tilde{f}_{m+1}$ auf diesen Intervallen affin sein. Weil die lineare Interpolation einer Funktion mit gegebenen Stützstellen eindeutig ist und $\tilde{f}_{m+1}(x) = f(x)$ für alle Stützstellen $x = \frac{k}{2^{m+1}}$, $k = 0, \ldots, 2^{m+1}$, folgt also

$$f_{m+1} = \tilde{f}_{m+1} = f_m - \frac{g_{m+1}}{2^{2m+2}}$$

und damit

$$f_m - f_{m+1} = \frac{g_{m+1}}{2^{2m+2}}.$$

Da $f_0(x) = x$, folgt damit, dass

$$f_m(x) = x - \sum_{t=1}^{m} \frac{g_t(x)}{2^{2t}}. \tag{3.4}$$

Wir wollen nun ein neuronales Feedforward-Netz Ψ_m mit $R_\varrho(\Psi_\varepsilon) = f_m$ konstruieren. Hierfür konstruieren wir zunächst ein Multilayer Perceptron $\Phi^{(t)}$ mit $R_\varrho(\Phi^{(t)}) = g_t$. Sei $\Phi^{(1)} := ((A_1, b_1), (A_2, b_2), (A_3, b_3))$ mit

$$\begin{aligned}
A_1 &:= \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad b_1 := \begin{pmatrix} 0 \\ -\frac{1}{2} \end{pmatrix}, \\
A_2 &:= \begin{pmatrix} 2 & -4 \end{pmatrix}, \quad b_2 := \begin{pmatrix} 0 \end{pmatrix}, \\
A_3 &:= \begin{pmatrix} 1 \end{pmatrix}, \quad b_3 := \begin{pmatrix} 0 \end{pmatrix},
\end{aligned}$$

wobei $L(\Phi^{(1)}) = 3, N(\Phi^{(1)}) = 5$ und $M(\Phi^{(1)}) = 6$. Mit einer einfachen Fallunterscheidung lässt sich verifizieren, dass $R_\varrho(\Phi^{(1)})(x) = \varrho(2\varrho(x) - 4\varrho(x - \frac{1}{2})) = g(x) = g_1(x)$. Wir definieren nun

$$\Phi^{(t)} := \underbrace{\Phi^{(1)} \odot \cdots \odot \Phi^{(1)}}_{t \text{ Mal}}.$$

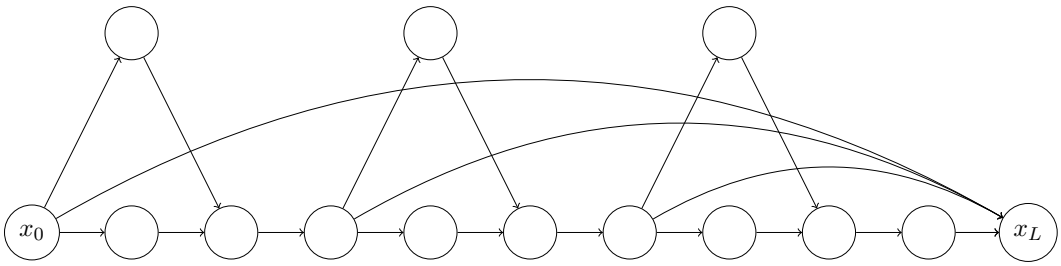

Abbildung 3.3: Schematische Darstellung von Ψ_m für $m = 3$.

Bemerkung 2.17 liefert uns, dass $L(\Phi^{(t)}) = 3t$. Mit Bemerkung 2.19 sowie $\|A_1\|_{\ell^0} = 2$, $\|A_3\|_{\ell^0} = 1$ und $\|b_1\|_{\ell^0} = 1$ folgt $M(\Phi^{(1)} \odot \Phi^{(1)}) \leq M(\Phi^{(1)}) + (\Phi^{(1)}) + 4$ und damit induktiv

$$M(\Phi^{(t)}) \leq t \cdot M(\Phi^{(1)}) + (t-1) \cdot 4 = 6t + 4(t-1) = 10t - 4.$$

Insbesondere ist wegen $N(\Phi^{(1)}) \leq M(\Phi^{(1)})$ mit Bemerkung 2.19 auch $N(\Phi^{(t)}) \leq M(\Phi^{(t)}) \leq 10t - 4$.

Sei nun $\Psi_m = \left(L, \{N_\ell\}_{\ell=0}^{L}, \{A_{k,\ell}\}_{0 \leq k < \ell \leq L}, \{b_\ell\}_{\ell=1}^{L}\right)$ ein neuronales Feedforward-Netz mit $L = 3m+1$, $\{N_\ell\}_{\ell=0}^{L-1}$ und $\{b_\ell\}_{\ell=1}^{L-1}$ wie in $\Phi^{(m)}$ und $A_{k,\ell}$ wie in $\Phi^{(m)}$ für $\ell = k+1$, $k = 0, 1, \ldots, L-2$. Per Konstruktion ist bei der Berechnung der Realisierung, nach Definition 2.2, für $t = 1, \ldots, m$ gerade $x_{3t} = g_t(x)$. Hierbei merken wir an, dass nach unserer Definition eines neuronalen Netzes (in der letzten Schicht wird die Aktivierungsfunktion nicht angewandt) tatsächlich $x_{3t} = \varrho(g_t(x))$, was allerdings kein Problem darstellt, da $\varrho(g_t(x)) = g_t(x)$ wegen $g_t \geq 0$. Zudem ist $x_0 = x$. Damit wählen wir in Hinblick auf 3.4 nun

$$\begin{aligned} A_{0,L} &:= \begin{pmatrix} 1 \end{pmatrix}, \\ A_{3t,L} &:= \begin{pmatrix} -\frac{1}{2^{2t}} \end{pmatrix}, \quad t = 1, \ldots, m. \end{aligned}$$

Wir wählen außerdem $b_L := \begin{pmatrix} 0 \end{pmatrix}$, $N_L = 1$ und alle nicht genannten $A_{k,\ell} := 0$ in der entsprechenden Dimension. Eine schmeatische Darstellung des Netzwerks ist in Abbildung 3.3 gegeben. Mit dieser Konstruktion ist für alle $x \in [0, 1]$

$$\begin{aligned} R_\varrho(\Psi_m)(x) = x_L &= \sum_{k=0}^{L-1} A_{k,L} x_k + b_L = x_0 + \sum_{k=1}^{m} \left(-\frac{1}{2^{2k}}\right) \cdot x_{3k} \\ &= x - \sum_{k=1}^{m} \frac{g_k(x)}{2^{2k}} = f_m(x), \end{aligned}$$

wie gewünscht. Per Konstruktion hat Ψ_m höchstens $M(\Phi^{(m)}) + m + 1 \leq 11m - 3$ nichtnull Gewichte und damit auch Neuronen. Nun müssen wir nur noch m entsprechend eines gegebenen $\varepsilon \in (0, \frac{1}{2})$ wählen und unsere Aussagen zu den Eigenschaften des Netzwerks verifizieren. Wir suchen also m, sodass die Approximationsgenauigkeit $\varepsilon_m = 2^{-2m-2} \leq \varepsilon$ ist. Einfache Umformungen liefern uns $m \geq \ln(\frac{1}{\varepsilon}) \cdot \frac{1}{2\ln(2)} - 1$, also zum Beispiel $m =$

$m(\varepsilon) := \lceil \ln(\frac{1}{\varepsilon}) - 1 \rceil$. Sei nun also $\Phi_\varepsilon := \Psi_{m(\varepsilon)}$. Dann ist

$$\|R_\varrho(\Phi_\varepsilon) - f\|_{L^\infty} = \|f_{m(\varepsilon)} - f\|_{L^\infty} \leq \varepsilon_{m(\varepsilon)} \leq \varepsilon,$$

wie gewünscht. Außerdem ist

$$L(\Phi_\varepsilon) = L(\Psi_{m(\varepsilon)}) = 3 \cdot m(\varepsilon) + 1 \leq 3 \cdot \ln\left(\frac{1}{\varepsilon}\right) + 1 \leq 4 \cdot \ln\left(\frac{1}{\varepsilon}\right)$$

und

$$N(\Phi_\varepsilon) \leq M(\Phi_\varepsilon) = M(\Psi_{m(\varepsilon)}) \leq 11 \cdot m(\varepsilon) - 3 \leq 11 \cdot \ln\left(\frac{1}{\varepsilon}\right).$$

Damit existiert also eine universelle Konstante $c > 0$, sodass die Tiefe sowie die Anzahl der Neuronen und Gewichte von Φ_ε höchstens $c \cdot \ln\left(\frac{1}{\varepsilon}\right)$ ist, zum Beispiel $c = 11$. □

Mithilfe der Expansion

$$xy = \frac{1}{2}((x+y)^2 - x^2 - y^2) \tag{3.5}$$

können wir Lemma 3.1 nun nutzen, um eine allgemeine Multiplikation mithilfe von ReLU Netzwerken umzusetzen.

Lemma 3.2. *Seien $M > 0$ und $\varepsilon \in (0, \frac{1}{2})$. Es existiert ein neuronales Feedforward-Netzwerk $\tilde{\times}$ mit Eingabedimension 2 und Ausgabedimension 1, sodass gilt:*

- *Die Tiefe und die Anzahl der nichtnull Gewichte und Neuronen von $\tilde{\times}$ sind nicht größer als $c_1 \ln(\frac{1}{\varepsilon}) + c_2$ für eine absolute Konstante c_1 und einer Konstante $c_2 = c_2(M)$.*
- *Wenn $x = 0$ oder $y = 0$, dann ist $R_\varrho(\tilde{\times})(x, y) = 0$.*
- *Für alle $x, y \in [-M, M]$ gilt: $|R_\varrho(\tilde{\times})(x, y) - xy| \leq \varepsilon$.*

Beweis. Sei Φ_δ ein Netzwerk nach Lemma 3.1 mit $\delta = \frac{\varepsilon}{6M^2}$. Damit gilt also, mit $\tilde{f}_\delta := R_\varrho(\Phi_\delta)$, dass für alle $x \in [0, 1]$

$$|\tilde{f}_\delta(x) - x^2| < \delta, \quad x \in [0, 1]. \tag{3.6}$$

Da $\tilde{f}_\delta$ nach dem Beweis von Lemma 3.1 von der Form $\tilde{f}_\delta(x) = x - \sum_{s=1}^m \frac{g_s(x)}{2^{2s}}$ ist, gilt außerdem

$$\tilde{f}_\delta(0) = 0. \tag{3.7}$$

Sei nun $f_\varepsilon : [-M, M]^2 \to \mathbb{R}$ gegeben durch

$$f_\varepsilon(x, y) := 2M^2 \left(\tilde{f}_\delta\left(\frac{|x+y|}{2M}\right) - \tilde{f}_\delta\left(\frac{|x|}{2M}\right) - \tilde{f}_\delta\left(\frac{|y|}{2M}\right) \right).$$

Es ist mit der Dreiecksungleichung und Expansion 3.5

$$\begin{aligned}
|f_\varepsilon(x,y) - xy| &= \left|2M^2\left(\tilde{f}_\delta\left(\frac{|x+y|}{2M}\right) - \tilde{f}_\delta\left(\frac{|x|}{2M}\right) - \tilde{f}_\delta\left(\frac{|y|}{2M}\right)\right) - \frac{1}{2}((x+y)^2 - x^2 - y^2)\right| \\
&= 2M^2\left|\tilde{f}_\delta\left(\frac{|x+y|}{2M}\right) - \frac{(x+y)^2}{4M^2} - \tilde{f}_\delta\left(\frac{|x|}{2M}\right) + \frac{x^2}{4M^2} - \tilde{f}_\delta\left(\frac{|y|}{2M}\right) + \frac{y^2}{4M^2}\right| \\
&\leq 2M^2\left(\left|\tilde{f}_\delta\left(\frac{|x+y|}{2M}\right) - \frac{(x+y)^2}{4M^2}\right| + \left|\tilde{f}_\delta\left(\frac{|x|}{2M}\right) - \frac{x^2}{4M^2}\right|\right. \\
&\qquad \left. + \left|\tilde{f}_\delta\left(\frac{|y|}{2M}\right) - \frac{y^2}{4M^2}\right|\right) \\
&\leq 2M^2 \cdot 3 \cdot \frac{\varepsilon}{6M^2} = \varepsilon.
\end{aligned}$$

Dabei benutzen wir im letzten Schritt Eigenschaft 3.6 und, dass $\left(\frac{|z|}{2M}\right)^2 = \frac{z^2}{4M^2}$ und $\frac{|x+y|}{2M}, \frac{|x|}{2M}, \frac{|y|}{2M} \in [0,1]$. Außerdem ist mithilfe von 3.7 und der Definition von f_ε klar, dass $f_\varepsilon(x,y) = 0$, falls $x = 0$ oder $y = 0$.

Wir wollen nun also ein neuronales Feedforward-Netzwerk $\tilde{\times}$ mit $R_\varrho(\tilde{\times})|_{[-M,M]^2} = f_\varepsilon$ konstruieren, welches die Anforderungen an Tiefe und Anzahl an Neuronen und nichtnull Gewichte erfüllt. Sei hierfür zunächst $\Phi_{3,\delta} := \mathcal{P}(\mathcal{P}(\Phi_\delta, \Phi_\delta), \Phi_\delta)$ die disjunkte Parallelisierung. Dann hat $\Phi_{3,\delta}$ höchstens $3 \cdot c \cdot \ln(\frac{1}{\delta}) = 3 \cdot c \cdot \ln(\frac{6M^2}{\varepsilon}) \leq \tilde{c}_1 \cdot \ln(\frac{1}{\varepsilon}) + \tilde{c}_2$ Neuronen, nichtnull Gewichte und Schichten für eine universelle Konstante $\tilde{c}_1$ sowie $\tilde{c}_2 := \tilde{c}_2(M)$, und es ist

$$R_\varrho(\Phi_{3,\delta})(x,y,z) = \left(\tilde{f}_\delta(x), \tilde{f}_\delta(y), \tilde{f}_\delta(z)\right)^T$$

Sei nun außerdem $\Phi_{\text{input}} := ((A_1^{\text{in}}, b_1^{\text{in}}), (A_2^{\text{in}}, b_2^{\text{in}}))$ ein MLP mit

$$A_1^{\text{in}} := \frac{1}{2M} \cdot \begin{pmatrix} 1 & 1 \\ -1 & -1 \\ 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{pmatrix},\ b_1^{\text{in}} := 0,$$

$$A_2^{\text{in}} := \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix},\ b_2^{\text{in}} := 0.$$

Dann ist $R_\varrho(\Phi_{\text{input}})(x,y) = (\frac{|x+y|}{2M}, \frac{|x|}{2M}, \frac{|y|}{2M})^T \in [0,1]^3$ sowie $M(\Phi_{\text{input}}) = 14$ und $N(\Phi_{\text{input}}) = 11$. Sei außerdem $\Phi_{\text{sum}} := ((A_1^{\text{sum}}, b_1^{\text{sum}}))$ mit

$$A_1^{\text{sum}} := \begin{pmatrix} 2M^2 & -2M^2 & -2M^2 \end{pmatrix},\ b_1^{\text{sum}} := 0,$$

wobei $M(\Phi_{\text{sum}}) = 3$ und $N(\Phi_{\text{sum}}) = 4$.

Sei nun $\tilde{\times} := \Phi_{\text{sum}} \odot \Phi_{3,\delta} \odot \Phi_{\text{input}}$ die spärliche Verkettung dieser drei Netze. Für dieses Netz gilt mithilfe von Bemerkung 2.19 mit einer Konstante $c_2 := c_2(M)$ und einer universellen Konstante c_1 für die Anzahl von

- Schichten:

$$\begin{aligned} L(\tilde{\times}) &\leq L(\Phi_{\text{sum}}) + L(\Phi_{3,\delta}) + L(\Phi_{\text{input}}) \\ &\leq \tilde{c}_1 \cdot \ln\left(\frac{1}{\varepsilon}\right) + \tilde{c}_2 + 3 \leq c_1 \cdot \ln\left(\frac{1}{\varepsilon}\right) + c_2; \end{aligned}$$

- nichtnull Gewichten:

$$\begin{aligned} M(\tilde{\times}) &\leq M(\Phi_{\text{sum}}) + M(\Phi_{3,\delta}) + 3 \cdot \left\lceil \frac{L(\Phi_{3,\delta})}{3} + 1 \right\rceil + 3 + 3 + M(\Phi_{\text{input}}) + 6 + 3 + 3 \\ &\leq \tilde{c}_1 \cdot \ln\left(\frac{1}{\varepsilon}\right) + 3 \cdot \left\lceil \frac{L(\Phi_{3,\delta})}{3} + 1 \right\rceil + \tilde{c}_2 + 35 \\ &\leq c_1 \cdot \ln\left(\frac{1}{\varepsilon}\right) + c_2; \end{aligned}$$

  Dabei haben wir die Konstruktion von Φ_δ nach dem Beweis von Lemma 3.1 sowie die Definition von $\Phi_{3,\delta}$ als disjunkte Parallelisierung von drei Netzen Φ_δ verwendet, um nach Bemerkung 2.19 die Summe $\sum_{k=0}^{L(\Phi_{3,\delta})-1} \|A_{k,L(\Phi_{3,\delta})}\|_{\ell^0}$ abzuschätzen. Anschließend haben wir $L(\Phi_{3,\delta}) \leq \tilde{c}_1 \cdot \ln(\frac{1}{\varepsilon}) + \tilde{c}_2$ verwendet.

- Neuronen:

$$\begin{aligned} N(\tilde{\times}) &\leq N(\Phi_{\text{sum}}) + N(\Phi_{3,\delta}) + 3 + N(\Phi_{\text{input}}) + 3 \\ &\leq \tilde{c}_1 \cdot \ln\left(\frac{1}{\varepsilon}\right) + \tilde{c}_2 + 21 \leq c_1 \cdot \ln\left(\frac{1}{\varepsilon}\right) + c_2. \end{aligned}$$

Außerdem ist $R_\varrho(\tilde{\times}) = R_\varrho(\Phi_{\text{sum}}) \circ R_\varrho(\Phi_{3,\delta}) \circ R_\varrho(\Phi_{\text{input}}) = f_\varepsilon$, wie gewünscht.

□

Schließlich präsentieren wir eines der Hauptresultate von Yarotsky in [9] und unser zentrales Ergebnis für die L^∞ Approximation von Funktionen aus Sobolev-Räumen. Besonders bemerkenswert ist, dass die Anzahl der benötigten Neuronen und Gewichte einer Netzwerkarchitektur mit steigender schwacher Differenzierbarkeit der zu approximierenden Funktion abnimmt. Zudem ist hervorzuheben, dass dieses Resultat die Approximation einer gesamten Klasse an Funktionen $F_{n,d}$ mit einer einzigen Netzwerkarchitektur erlaubt.

Satz 3.3. *Für beliebige $d, n \in \mathbb{N}$ und $\varepsilon \in (0, \frac{1}{2})$ existieren eine Netzwerkarchitektur $\mathcal{A}$ und eine Konstante $c := c(d, n)$, für die gilt:*

- *$\mathcal{A}$ hat höchstens $c \cdot \ln(\frac{1}{\varepsilon})$ Schichten.*
- *Die Anzahl der Neuronen und nichtnull Gewichte von $\mathcal{A}$ ist höchstens $c \cdot \varepsilon^{-d/n} \cdot \ln(\frac{1}{\varepsilon})$.*

- *$\mathcal{A}$ kann jede Funktion $f \in F_{n,d}$ mit einem L^∞ Fehler von ε approximieren. In anderen Worten: Für jede Funktion $f \in F_{n,d}$ existiert ein neuronales Feedforward-Netz Φ_f, welches $\mathcal{A}$ implementiert und*

$$\|R_\varrho(\Phi_f) - f\|_{L^\infty([0,1]^d)} \leq \varepsilon.$$

Beweis. Wir werden in diesem Beweis in zwei Schritten vorgehen. Zunächst werden wir f mithilfe einer Zerlegung der Eins und lokalen Taylor-Polynomen auf Gitterpunkten in $[0,1]^d$ approximieren. Anschließend werden wir unsere bisherigen Ergebnisse nutzen, um eine Approximation für die obige Approximation mithilfe eines neuronalen Feedforward-Netzwerks zu konstruieren.

Sei $N \in \mathbb{N}$. Betrachte die Zerlegung der Eins

$$\sum_{\mathbf{m} \in M} \gamma_{\mathbf{m}}(x) \equiv 1, \quad x \in [0,1]^d,$$

mit $\mathbf{m} = (m_1, \ldots, m_d) \in M := \{0, 1, \ldots, N\}^d$, und $\gamma_{\mathbf{m}}$ definiert als

$$\gamma_{\mathbf{m}} := \prod_{k=1}^{d} \psi\left(3N\left(x_k - \frac{m_k}{N}\right)\right),$$

wobei

$$\psi(x) = \begin{cases} 1, & |x| < 1, \\ 2 - |x|, & 1 \leq |x| \leq 2, \\ 0, & |x| > 2. \end{cases}$$

Dabei merken wir an, dass $\psi(x) = \varrho(x+2) - \varrho(x+1) - \varrho(x-1) + \varrho(x-2)$ und damit auch $\psi\left(3N\left(x_k - \frac{m_k}{N}\right)\right)$ durch ein neuronales Netz mit 2 Schichten und beschränkter Anzahl an Neuronen und nichtnull Gewichten exakt dargestellt werden kann.

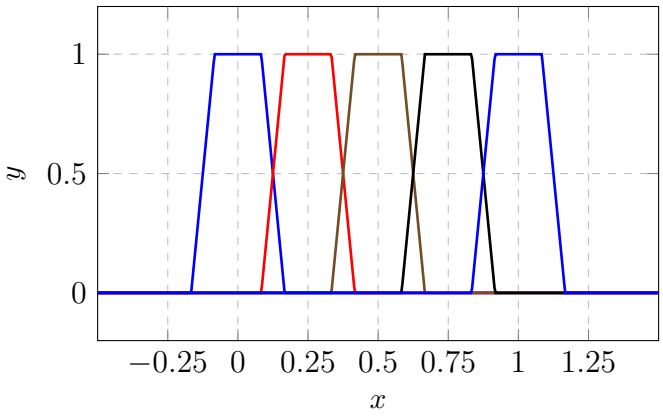

Abbildung 3.4: Zerlegung der Eins mit den Funktionen $\gamma_0, \ldots, \gamma_4$ für $d = 1, N = 4$.

Zunächst verifizieren wir, dass es sich hierbei um eine Zerlegung der Eins handelt. Wir stellen fest, dass $\psi\left(3N\left(x_k - \frac{m_k}{N}\right)\right) = 0$ für $|x_k - \frac{m_k}{N}| \geq \frac{2}{3N}$. Betrachten wir nun zuerst den Fall $d = 1$ (siehe hierfür auch Abbildung 3.4). Dann ist

$$\sum_{\mathbf{m}\in M} \gamma_{\mathbf{m}}(x) = \sum_{m=0}^{N} \psi\left(3N\left(x - \frac{m}{N}\right)\right).$$

Sei nun $x \in [0,1]$ beliebig. Dann ist $x \in [\frac{m^*}{N}, \frac{m^*+1}{N})$ für genau ein m^* oder $x = 1$. Sei zunächst ersteres der Fall. Dann ist $\psi\left(3N\left(x_k - \frac{m}{N}\right)\right) = 0$ für alle $m \notin \{m^*, m^*+1\}$, da dann $|x - \frac{m}{N}| > \frac{2}{3N}$. Es gilt also zu zeigen, dass

$$\begin{aligned} 1 = \sum_{\mathbf{m}\in M} \gamma_{\mathbf{m}}(x) &= \sum_{m=0}^{N} \psi\left(3N\left(x - \frac{m}{N}\right)\right) \\ = \psi\left(3N\left(x - \frac{m^*}{N}\right)\right) &+ \psi\left(3N\left(x - \frac{m^*+1}{N}\right)\right). \end{aligned} \tag{3.8}$$

Wir schreiben nun x eindeutig als $x = \frac{m^*}{N} + y,\ y \in [0, \frac{1}{N})$ und unterscheiden in drei Fälle:

Fall 1: $y \in [0, \frac{1}{3N})$

Dann ist $3N(x - \frac{m^*}{N}) = 3N(\frac{m^*}{N} + y - \frac{m^*}{N}) = 3Ny < 1$ und $3N(x - \frac{m^*+1}{N}) = 3N(\frac{m^*}{N} + y - \frac{m^*+1}{N}) = 3N(\frac{1}{N} - y) > 2$, wobei wir in letzterem Fall nutzen, dass $\frac{1}{N} - y > \frac{2}{3N}$. Daher ist

$$\psi\left(3N\left(x - \frac{m^*}{N}\right)\right) + \psi\left(3N\left(x - \frac{m^*+1}{N}\right)\right) = 1 + 0 = 1,$$

wie gewünscht.

Fall 2: $y \in [\frac{1}{3N}, \frac{2}{3N}]$

Wir gehen ähnlich vor und erhalten $3N(\frac{m^*}{N} + y - \frac{m^*}{N}) = 3Ny \in [1,2]$ und $3N(\frac{m^*}{N} + y - \frac{m^*+1}{N}) = 3N(y - \frac{1}{N}) = 3Ny - 3 \in [-2,-1]$. Damit ist also

$$\begin{aligned} &\psi\left(3N\left(x - \frac{m^*}{N}\right)\right) + \psi\left(3N\left(x - \frac{m^*+1}{N}\right)\right) \\ &= 2 - |3Ny| + 2 - |3Ny - 3| \\ &= 4 - 3(|Ny| + |1 - Ny|) \\ &= 4 - 3(Ny + 1 - Ny) \\ &= 1, \end{aligned}$$

wobei wir genutzt haben, dass $1 > Ny$.

Fall 3: $y \in (\frac{2}{3N}, \frac{1}{3N})$

Dieser Fall verläuft analog zu Fall 1, nur mit vertauschten Rollen.

Als Letztes betrachten wir nun noch $x = 1$. Hier sieht man schnell, dass

$$\sum_{\mathbf{m}\in M} \gamma_{\mathbf{m}}(x) = \psi\left(3N\left(1 - \frac{N}{N}\right)\right) = 1.$$

Nun sei $d \geq 1$ beliebig. Dies können wir auf den eindimensionalen Fall reduzieren. Es ist, mithilfe von 3.8,

$$\begin{aligned}
\sum_{\mathbf{m}\in M}\prod_{k=1}^{d}\psi\left(3N\left(x_k-\frac{m_k}{N}\right)\right) &= \sum_{m_1=0}^{N}\cdots\sum_{m_d=0}^{N}\prod_{k=1}^{d}\psi\left(3N\left(x_k-\frac{m_k}{N}\right)\right)\\
&= \sum_{m_1=0}^{N}\psi\left(3N\left(x_1-\frac{m_1}{N}\right)\right)\sum_{m_2=0}^{N}\cdots\sum_{m_d=0}^{N}\prod_{k=2}^{d}\psi\left(3N\left(x_k-\frac{m_k}{N}\right)\right)\\
&= 1\cdot\sum_{m_2=0}^{N}\cdots\sum_{m_d=0}^{N}\prod_{k=2}^{d}\psi\left(3N\left(x_k-\frac{m_k}{N}\right)\right)\\
&\vdots\\
&= \sum_{m_d=0}^{N}\prod_{k=d}^{d}\psi\left(3N\left(x_k-\frac{m_k}{N}\right)\right) = 1.
\end{aligned}$$

Die Summe ist endlich und somit auch lokal endlich. Da zudem $0 \leq \psi \leq 1$ gilt, folgt, dass auch $0 \leq \gamma \leq 1$ ist. Damit handelt es sich hierbei um eine Zerlegung der Eins.

Wir bemerken, dass

$$\|\psi\|_{L^\infty} = 1 \text{ und } \|\gamma_{\mathbf{m}}\|_{L^\infty} = 1 \quad \forall \mathbf{m} \in M, \tag{3.9}$$

und

$$\operatorname{supp}\gamma_{\mathbf{m}} \subset \left\{x : \left|x_k - \frac{m_k}{N}\right| < \frac{1}{N} \ \ \forall k\right\}. \tag{3.10}$$

Sei nun $f \in \mathcal{F}_{d,n}$.Wir betrachten für $\mathbf{m} \in M$ das Taylorpolynom $(n-1)$-ten Grades von f am Punkt $x = \frac{\mathbf{m}}{N}$:

$$p_{\mathbf{m}}(x) = \sum_{|\alpha|=0}^{n-1}\frac{1}{\alpha!}(\partial^\alpha f)\left(\frac{\mathbf{m}}{N}\right)\cdot\left(x-\frac{\mathbf{m}}{N}\right)^\alpha, \tag{3.11}$$

wobei $\alpha \in \mathbb{N}_0^d$ und üblichen Notationen $\alpha! = \prod_{k=1}^d \alpha_k!$ und $(x-\frac{\mathbf{m}}{N})^\alpha = \prod_{k=1}^d (x_k - \frac{m_k}{N})^{\alpha_k}$ verwendet werden. Wir definieren nun eine Approximation für f als

$$f_1 := \sum_{\mathbf{m}\in M}\gamma_{\mathbf{m}}p_{\mathbf{m}}.$$

Wir beschränken den Approximationsfehler für alle $x \in [0,1]^d$ durch

$$\begin{aligned}
|f(x) - f_1(x)| &= \left|\sum_{\mathbf{m}\in M}\gamma_{\mathbf{m}}(x)(f(x)-p_{\mathbf{m}}(x))\right|\\
&\leq \sum_{\mathbf{m}:|x_k-\frac{m_k}{N}|<\frac{1}{N}\ \forall k}|f(x)-p_{\mathbf{m}}(x)|\\
&\leq 2^d \max_{\mathbf{m}:|x_k-\frac{m_k}{N}|<\frac{1}{N}\ \forall k}|f(x)-p_{\mathbf{m}}(x)|\\
&\leq \frac{2^d d^n}{n!}\left(\frac{1}{N}\right)^n.
\end{aligned} \tag{3.12}$$

Dabei nutzen wir im zweiten Schritt die Eigenschaft 3.10 des Trägers und die Schranke 3.9. Im dritten Schritt verwenden wir, dass $|x_k - \frac{m_k}{N}| < \frac{1}{N}$ für höchstens zwei m_k in jeder Koordinate gilt (siehe 3.8). Im vierten Schritt greifen wir auf eine Abschätzung für das Taylor-Restglied zurück und nutzen, dass $\|f\|_{\mathcal{W}^{n,\infty}([0,1]^d)} \leq 1$. Die Abschätzung im vierten Schritt führen wir wie folgt aus. Wir nutzen die Darstellung des Integralrestglieds. In vielen Versionen des Satzes von Taylor erfordert die Betrachtung des Restglieds vom Grad n die Bedingung $f \in C^n$. Dies ist jedoch nicht zwingend notwendig, wie in [5] von Folland erläutert wird. Sei $x \in [0,1]^d$ und $\mathbf{m}$ so, dass $|x_k - \frac{\mathbf{m}_k}{N}| < \frac{1}{N}$ für alle k gilt. Es ist

$$
\begin{aligned}
|f(x) - p_{\mathbf{m}}(x)| &= |R_n(x)| \\
&= \left| n \cdot \sum_{|\alpha|=n} \frac{1}{\alpha!} \left(x - \frac{\mathbf{m}}{N}\right)^{\alpha} \int_0^1 (1-t)^{n-1} \partial^{\alpha} f \left(\frac{\mathbf{m}}{N} + t \cdot \left(x - \frac{\mathbf{m}}{N}\right)\right) \mathrm{d}t \right| \\
&\leq n \cdot \sum_{|\alpha|=n} \frac{1}{\alpha!} \prod_{k=1}^{d} \left| \left(x_k - \frac{m_k}{N}\right)^{\alpha_k} \right| \cdot \int_0^1 |1-t|^{n-1} \left| \partial^{\alpha} f \left(\frac{\mathbf{m}}{N} + t \cdot \left(x - \frac{\mathbf{m}}{N}\right)\right) \right| \mathrm{d}t \\
&\leq n \cdot \sum_{|\alpha|=n} \frac{1}{\alpha!} \left(\frac{1}{N}\right)^n \int_0^1 |1-t|^{n-1} \max_{\alpha:|\alpha|=n} \operatorname{ess\,sup}_{x \in [0,1]^d} |\partial^{\alpha} f(x)| \, \mathrm{d}t \\
&\leq \left(\frac{1}{N}\right)^n \cdot n \cdot \sum_{|\alpha|=n} \frac{1}{\alpha!} \int_0^1 |1-t|^{n-1} \mathrm{d}t = \left(\frac{1}{N}\right)^n \cdot n \cdot \frac{1}{n!} \cdot \sum_{|\alpha|=n} \frac{n!}{\alpha!} \cdot \frac{1}{n} \\
&= \left(\frac{1}{N}\right)^n \cdot \frac{1}{n!} \cdot \sum_{|\alpha|=n} \binom{n}{\alpha} = \left(\frac{1}{N}\right)^n \cdot \frac{1}{n!} \cdot \underbrace{(1+1+\ldots+1)}_{d \text{ Mal}}{}^n \\
&= \frac{d^n}{n!} \cdot \left(\frac{1}{N}\right)^n,
\end{aligned}
$$

wie gewünscht. Da die obige Schranke von $\mathbf{m}$ unabhängig ist, gilt sie auch für das Maximum aller $\mathbf{m}$, die die obige Bedingung erfüllen. Im fünften Schritt haben wir dabei $\max_{\alpha:|\alpha|=n} \operatorname{ess\,sup}_{x \in [0,1]^d} |\partial^{\alpha} f(x)| \leq \|f\|_{\mathcal{W}^{n,\infty}([0,1]^d)} \leq 1$ und im achten Schritt das Multinomialtheorem verwendet.

Wählen wir nun

$$
N := \left\lceil \left(\frac{n!}{2^d d^n} \cdot \frac{\varepsilon}{2} \right)^{-1/n} \right\rceil,
$$

so folgt mit 3.12 also, dass

$$
\|f - f_1\|_{L^\infty} \leq \frac{\varepsilon}{2}. \tag{3.13}
$$

Außrdem stellen wir fest, dass mit 3.11 und $f \in F_{n,d}$ gilt, dass die Koeffizenten der Polynome $p_{\mathbf{m}}$, die wir von nun an als $a_{\mathbf{m},\alpha}$ bezeichnen, gleichmäßig beschränkt sind:

$$
p_{\mathbf{m}}(x) = \sum_{|\alpha|=0}^{n-1} a_{\mathbf{m},\alpha} \left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}, \quad |a_{\mathbf{m},\alpha}| \leq 1.
$$

Damit reicht es also aus, eine Netzwerkarchitektur zu konstruieren, welche die Funktionen der Form f_1 mit einem Fehler von $\frac{\varepsilon}{2}$ approximieren kann, wobei die $p_{\mathbf{m}}$ und N wie oben gegeben sind. Wir schreiben f_1 aus als

$$f_1(x) = \sum_{\mathbf{m}\in M} \sum_{|\alpha|=0}^{n-1} a_{\mathbf{m},\alpha} \cdot \gamma_{\mathbf{m}}(x) \cdot \left(x - \frac{\mathbf{m}}{N}\right)^{\alpha} \tag{3.14}$$

und erkennen, dass f_1 damit eine Linearkombination aus höchstens $n^d \cdot (N+1)^d$ Termen $\gamma_{\mathbf{m}}(x)\left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}$ ist. Jeder dieser Terme ist ein Produkt aus höchstens $d+n-1$ stückweise linearen Faktoren in einer Variablen: $\gamma_{\mathbf{m}}$ liefert uns d Faktoren $\psi(3Nx_k - 3m_k)$ und $(x - \frac{\mathbf{m}}{N}))^{\alpha}$ liefert uns höchstens $n-1$ Ausdrücke der Form $x_k - \frac{m_k}{N}$. Wir merken an, dass wir alle dieser Faktoren mit einem neuronalen Netz mit jeweils durch c_1 beschränkter Anzahl an Schichten, Neuronen und Tiefe exakt darstellen können.

Wir werden nun Lemma 3.2 verwenden, um eine Approximation eines solchen Produkts $\gamma_{\mathbf{m}}(x)\left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}$ darzustellen. Sei hierfür $\tilde{\times}$ wie in Lemma 3.2, mit $M = d+n$ und $\delta = \frac{\varepsilon}{2^{d+1}n^d(d+n)}$.

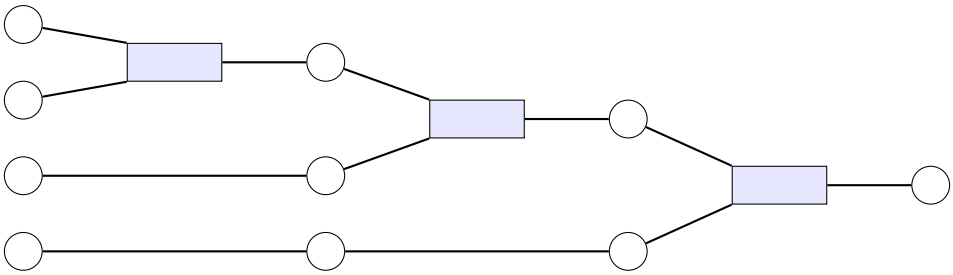

Abbildung 3.5: Schematische Darstellung eines Netzwerk $\Phi^{\alpha}_{\text{prod}}$ mit Eingabedimension 4. Die blauen Kästen stellen die Netze $\tilde{\times}$ dar, während in den anderen Verbindungen die Eingaben mit den Φ_0 unverändert weitergegeben werden. Die Darstellung gibt die Architektur nicht exakt wieder und dient nur dem Verständnis.

Wir werden die konkrete Konstruktion und exakte Betrachtung der Schranken für Gewichte, Schichten und Neuronen eines solchen Netzes kurz erläutern, jedoch nicht komplett ausführen, da dessen vollständige und detaillierte Betrachtung äußerst unübersichtlich werden würde. Hierfür definieren wir ein Netz Φ_0 mit Eingabe- und Ausgabedimension 1 und der gleichen Länge L wie $\tilde{\times}$. In diesem Netz sei nur $A_{0,L} = (1)$ und alle anderen Parameter seien 0. Die Realisierung dieses Netzes ist also die Identitätsabbildung. Anschließend bilden wir die disjunkte Parallelisierung mit i solcher Netze für $i = 1, \ldots, d + |\alpha| - 2$:

$$\tilde{\times}_i := \mathcal{P}(\tilde{\times}, \mathcal{P}(\Phi_0, \mathcal{P}(\Phi_0, \ldots))).$$

Dabei ist die Anzahl der zusätzlich eingeführten nichtnull Gewichte und Neuronen nur von d und n abhängig. Diese Netze haben nun Eingabedimension $i+2$ und Ausgabedimension $i+1$. Wir bilden die spärliche Verkettung

$$\Phi^{\alpha}_{\text{prod}} := \tilde{\times} \odot \tilde{\times}_1 \odot \ldots \odot \tilde{\times}_{d+|\alpha|-2},$$

deren Realisierung nun ein Produkt aus $|\alpha| + d$ Faktoren approximiert. Dabei bemerken wir, dass $\tilde{\times}_i$, abgesehen von den durch die Parallelisierung eingeführten Gewichten, keine Verbindungen von der ersten oder letzten Schicht zu nicht-benachbarten Schichten hat, und die Anzahl der Neuronen in der zweiten und vorletzten Schicht fix ist. Dies kann in der Konstruktion von $\tilde{\times}$ im Beweis von Lemma 3.2 nachgelesen werden. Mit Bemerkung 2.19 hängt damit die Anzahl der durch die spärliche Verkettung eingeführten nichtnull Gewichte, Neuronen und Schichten nur von n und d ab. Eine schematische Darstellung des Netzes $\Phi^{\alpha}_{\text{prod}}$ ist in Abbildung 3.5 zu finden.

Zuletzt definieren wir noch ein Netz $\Phi^{\mathbf{m},\alpha}_{\text{input}}$, dessen Realisierung folgende Funktion ist:

$$g : \mathbb{R}^d \to \mathbb{R}^{d+|\alpha|-1},$$
$$g(x) := \left(\psi(3Nx_1 - 3m_1), \dots, \psi(3Nx_d - 3m_2), x_{j_1} - \frac{m_{j_1}}{N}, \dots, x_{j_{|\alpha|}} - \frac{m_{j_{|\alpha|}}}{N}\right)^T.$$

Dabei sind die Indizes i_k die aus α resultierenden Indizes, sodass alle Faktoren aus $(x - \frac{\mathbf{m}}{N})^{\alpha}$ in ihrer entsprechenden Vielfachheit vorkommen - ist zum Beispiel $\alpha_1 = 2$ und $\alpha_2 = 3$, so ist $j_1 = j_2 = 1$ und $j_3 = j_4 = j_5 = 2$. Da die Funktionen $\psi(\cdots)$ und die $x_k - \frac{m_k}{N}$ jeweils durch beschränkte Netze dargestellt werden können, ist die Tiefe dieses Netzes unabhängig von d und n beschränkt und die Anzahl der Neuronen und nichtnull Gewichte durch eine Konstante $c_2(d, n)$ beschränkt.

Das aus der spärlichen Verkettung resultierende Netzwerk $\Phi_{\mathbf{m},\alpha} := \Phi^{\alpha}_{\text{prod}} \odot \Phi^{\mathbf{m},\alpha}_{\text{input}}$ hat mit Lemma 3.2 also höchstens $c_3 \cdot \ln(\frac{1}{\delta})$ Schichten, nichtnull Gewichte und Neuronen für eine Konstante $c_3 = c_3(d, n)$. Wir nutzen hierbei, dass die Anzahl der eingehenden Verbindungen in die letzte Schicht von $\Phi^{\mathbf{m},\alpha}_{\text{input}}$ und die Anzahl der ausgehenden Verbindungen der ersten Schicht von $\Phi^{\alpha}_{\text{prod}}$ sowie die enstprechende Ein-/Ausgabedimension der Netze nur von n und d abhängen. Wir merken außerdem an, dass $\delta \leq \varepsilon \leq \frac{1}{2}$ und daher $\ln(\frac{1}{\delta}) \geq \frac{1}{2} > 0$, weswegen die Konstante c_2' aus Lemma 3.2 hier fallen gelassen werden kann. Außerdem halten wir fest, dass $\Phi^{\alpha}_{\text{prod}}$ und damit auch $\Phi_{\mathbf{m},\alpha}$ per Konstruktion keine Verbindungen von nicht-benachbarten Schichten in die letzte Schicht hat, und dessen vorletzte Schicht endlich viele Neuronen hat (siehe Diskussion zu $\tilde{\times}_i$ oben). Die Anzahl der Verbindungen in die letzte Schicht ist also universell beschränkt.

Ist nun $h := R_{\varrho}(\tilde{\times})$, dann ist $\tilde{f}_{\mathbf{m},\alpha} := R_{\varrho}(\Phi_{\mathbf{m},\alpha})$ gegeben durch

$$\tilde{f}_{\mathbf{m},\alpha}(x) = h\left(\psi(3Nx_1 - 3m_1), h\left(\psi(3Nx_2 - 3m_2), \dots, h\left(x_{i_1} - \frac{x_{i_1}}{N}, \dots\right)\dots\right)\right). \quad (3.15)$$

Wir möchten nun den Fehler dieser Approximation für $\gamma_{\mathbf{m}}(x)\left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}$ abschätzen. Wir bemerken hierfür zunächst, dass $|\psi(3Nx_k - 3m_k)| \leq 1$ und $|x_k - \frac{m_k}{N}| \leq 1$ für $x \in [0, 1]^d$. Zudem folgt mit der Approximationseigenschaft aus Lemma 3.2, dass für $|a| \leq 1$ und $|b| \leq M$ gilt, dass $|h(a, b)| \leq |b| + \delta$. Zudem ist $\delta < 1$. Wenden wir dies wiederholt auf unsere Approximation der Multiplikation in 3.15 an, so sehen wir, dass die Eingaben der Multiplikationen durch $M = d + n$ beschränkt sind und die Approximationseigenschaft

aus Lemma 3.2 damit für alle h in dieser Verkettung gelt. Es ist dann

$$\begin{aligned}
&\left|\tilde{f}_{\mathbf{m},\alpha}(x) - \gamma_{\mathbf{m}}(x)\left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}\right| \\
&\quad= \Big|h(\psi(3Nx_1 - 3m_1), h(\psi(3Nx_2 - 3m_2), h(\psi(3Nx_3 - 3m_3), \ldots))) \\
&\qquad - \psi(3Nx_1 - 3m_1) \cdot \psi(3Nx_2 - 3m_2) \cdot \psi(3Nx_3 - 3m_3) \cdot \ldots\Big| \\
&\quad\leq \Big|h(\psi(3Nx_1 - 3m_1), h(\psi(3Nx_2 - 3m_2), h(\psi(3Nx_3 - 3m_3), \ldots))) \\
&\qquad - \psi(3Nx_1 - 3m_1) \cdot h(\psi(3Nx_2 - 3m_2), h(\psi(3Nx_3 - 3m_3), \ldots))\Big| \\
&\qquad + \Big|\psi(3Nx_1 - 3m_1)\Big| \cdot \Big|h(\psi(3Nx_2 - 3m_2), h(\psi(3Nx_3 - 3m_3), \ldots)) \\
&\qquad - \psi(3Nx_2 - 3m_2) \cdot h(\psi(3Nx_3 - 3m_3), \ldots)\Big| \\
&\qquad + \ldots \\
&\quad\leq (d+n) \cdot \delta
\end{aligned} \tag{3.16}$$

Außerdem ist $h(a,b) = 0$ für $a = 0$ oder $b = 0$, weswegen für $x \notin \operatorname{supp} \gamma_{\mathbf{m}}$

$$\tilde{f}_{\mathbf{m},\alpha}(x) = \gamma_{\mathbf{m}}(x)\left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}. \tag{3.17}$$

Wir definieren nun unsere vollständige Approximation

$$\tilde{f} := \sum_{\mathbf{m}\in M} \sum_{|\alpha|=0}^{n-1} a_{\mathbf{m},\alpha} \cdot \tilde{f}_{\mathbf{m},\alpha}.$$

Mit dieser Approximation erhalten wir für alle $x \in [0,1]^d$

$$\begin{aligned}
|\tilde{f}(x) - f_1(x)| &= \left|\sum_{\mathbf{m}\in M} \sum_{|\alpha|=0}^{n-1} a_{\mathbf{m},\alpha} \cdot \left(\tilde{f}_{\mathbf{m},\alpha}(x) - \gamma_{\mathbf{m}}(x)\left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}\right)\right| \\
&= \left|\sum_{\mathbf{m}:x\in\operatorname{supp}\gamma_{\mathbf{m}}} \sum_{|\alpha|=0}^{n-1} a_{\mathbf{m},\alpha} \cdot \left(\tilde{f}_{\mathbf{m},\alpha}(x) - \gamma_{\mathbf{m}}(x)\left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}\right)\right| \\
&\leq 2^d \max_{\mathbf{m}:x\in\operatorname{supp}\gamma_{\mathbf{m}}} \sum_{|\alpha|=0}^{n-1} \left|\tilde{f}_{\mathbf{m},\alpha}(x) - \gamma_{\mathbf{m}}(x)\left(x - \frac{\mathbf{m}}{N}\right)^{\alpha}\right| \\
&\leq 2^d n^d (d+n)\delta = \frac{\varepsilon}{2}.
\end{aligned}$$

Dabei haben wir im ersten Schritt 3.14 eingesetzt, im zweiten Schritt 3.17 verwendet, im dritten Schritt die Schranke $|a_{\mathbf{m},\alpha}| \leq 1$ und im vierten Schritt wieder verwendet, dass x nur für höchstens 2^d verschiedene $\mathbf{m}$ im Träger von $\gamma_{\mathbf{m}}$ liegen kann. Im vierten Schritt

haben wir schließlich Abschätzung 3.16 und $|\{\alpha \in \mathbb{N}_0^d : |\alpha| \leq n-1\}| \leq n^d$ verwendet. Zusammen mit 3.13 folgt also

$$\|\tilde{f} - f\|_{L^\infty([0,1]^d)} \leq \|\tilde{f} - f_1\|_{L^\infty([0,1]^d)} + \|f_1 - f\|_{L^\infty([0,1]^d)} \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon.$$

Um nun ein neuronales Netz zu erhalten, welches $\tilde{f}$ darstellt, parallelisieren wir zunächst alle Netze $\Phi_{\mathbf{m},\alpha}$ und erhalten dadurch das Netz $\tilde{\Phi}$. Dabei ist die Anzahl der Verbindungen in die letzte Schicht von $\tilde{\Phi}$ beschränkt durch ein Vielfaches von $n^d \cdot (N+1)^d$. Wir bemerken, dass die Anzahl der Schichten, Neuronen und nichtnull Gewichte dieser Netze $\Phi_{\mathbf{m},\alpha}$ durch $c_3 \cdot \ln(\frac{1}{\delta})$ beschränkt ist und wir nicht mehr als $n^d \cdot (N+1)^d$ dieser Netze verwendet haben. Daher ist die Tiefe von $\tilde{\Phi}$ durch $c_3 \cdot \ln(\frac{1}{\delta})$ und die Anzahl der Neuronen und nichtnull Gewichte durch $n^d \cdot (N+1)^d \cdot c_3 \cdot \ln(\frac{1}{\delta})$ beschränkt. Sei zuletzt $\Phi_{\text{coeff}} := ((A_1, b_1))$ ein Multilayer Perceptron mit $b_1 := 0$ und A_1 einer Matrix mit einer Zeile und den Koeffizienten $a_{\mathbf{m},\alpha}$ als Einträgen. Offensichtlich hat Φ_{coeff} eine Schicht und höchstens $n^d \cdot (N+1)^d$ Neuronen und nichtnull Gewichte. Schließlich sei Φ_f die spärliche Verkettung von Φ_{coeff} und $\tilde{\Phi}$. Wir stellen fest, dass die Anzahl der eingehenden Verbindung in die letzte Schicht von $\tilde{\Phi}$ und die Anzahl der ausgehenden Verbindung der ersten Schicht von Φ_{coeff} sowie die entsprechende Ein-/Ausgabedimension durch ein Vielfaches von $n^d(N+1)^d$ beschränkt sind. Mit Bemerkung 2.19 ist also auch die Anzahl der durch die spärliche Verkettung hinzugefügten nichtnull Gewichte und Neuronen durch ein Vielfaches hiervon beschränkt. Per Konstruktion ist nun $R_\varrho(\Phi_f) = \tilde{f}$ und für Konstanten $c_4 := c_4(d,n), c_5 := c_5(d,n)$ und schließlich eine geeignete Konstante $c := c(d,n)$ gilt

$$\begin{aligned}
L(\Phi_f) &\leq c_4 \cdot \ln\left(\frac{1}{\delta}\right) = c_4 \cdot \ln\left(\frac{2^{d+1} n^d (d+n)}{\varepsilon}\right) \\
&= c_4 \cdot \ln\left(\frac{1}{\varepsilon}\right) + c_4 \cdot \ln\left(2^{d+1} n^d (d+n)\right) \leq c_5 \cdot \ln\left(\frac{1}{\varepsilon}\right) \\
&\leq c \cdot \ln\left(\frac{1}{\varepsilon}\right); \\
N(\Phi_f), M(\Phi_f) &\leq n^d \cdot (N+1)^d \cdot c_4 \cdot \ln\left(\frac{1}{\delta}\right) \\
&\leq n^d \cdot \left(\left(\frac{n! \cdot \varepsilon}{2^{d+1} d^n} + 1\right)^{-1/n} + 1\right)^d \cdot c_5 \cdot \ln\left(\frac{1}{\varepsilon}\right) \\
&\leq c \cdot \varepsilon^{-d/n} \cdot \ln\left(\frac{1}{\varepsilon}\right).
\end{aligned}$$

Dabei haben wir verwendet, dass $\ln\left(\frac{1}{\varepsilon}\right) > \frac{1}{2} > 0$, und, dass $\varepsilon^{-d/n} > \varepsilon^{-k/n}$ für alle $\varepsilon \in (0, \frac{1}{2})$ und $k < d$.

Zuletzt bemerken wir, dass die Struktur des Netzwerks ausschließlich von d, n und ε abhängt – nicht jedoch von der Funktion f. Diese spielte nur bei der Wahl der Koeffizienten $a_{\mathbf{m},\alpha}$ eine Rolle. Wählen wir nun eine Netzwerkarchitektur $\mathcal{A}$ mit der gleichen Struktur wie Φ_f so, dass alle Gewichte, die wir in der Konstruktion von Φ_f als nichtnull Gewichte bezeichnet haben, auf 1 gesetzt werden und alle anderen Gewichte auf 0 gesetzt werden,

haben wir eine Netzwerkarchitektur gefunden, die die Implikationen des Satzes erfüllt. Insbesondere gelten also auch $L(\mathcal{A}) \leq c \cdot \ln(\frac{1}{\varepsilon})$ und $N(\mathcal{A}), M(\mathcal{A}) \leq c \cdot \varepsilon^{-d/n} \cdot \ln(\frac{1}{\varepsilon})$.

□

3.2 Approximation von glatten Funktionen in allgemeinen L^p Normen

Nun beschäftigen wir uns mit den Arbeiten von Petersen und Voigtlaender ([7]), auf die sich das folgende Kapitel stützt.

Im Gegensatz zu unserem vorherigen Ansatz werden wir nun keine allgemeinen Feedforward-Netze, sondern ausschließlich Multilayer Perceptrons betrachten – also Netze, in denen nur Verbindungen zwischen benachbarten Schichten erlaubt sind. Genauer untersuchen wir obere Schranken für die Approximationsraten von Multilayer Perceptrons in der L^p Norm für $p \in (0, \infty)$. Unser Ziel ist es insbesondere, ein Ergebnis zur Approximation einer Klasse *glatter* Funktionen zu formulieren.

Hierfür definieren wir zunächst die Norm $\|\cdot\|_{C^{0,\beta}}$. Sei dazu $\beta \in (0, \infty)$ mit $\beta = n + \sigma$, wobei $n \in \mathbb{N}_0$, $\sigma \in (0, 1]$, und $d \in \mathbb{N}$. Für $f \in C^n\left(\left[-\frac{1}{2}, \frac{1}{2}\right]^d\right)$ setzen wir

$$\|f\|_{C^{0,\beta}} := \max\left\{\max_{|\alpha|\leq n} \|\partial^\alpha f\|_{\sup}, \max_{|\alpha|=n} \mathrm{Lip}_\sigma(\partial^\alpha f)\right\} \in [0, \infty],$$

wobei wir die Notation

$$\mathrm{Lip}_\sigma(g) := \sup_{x,y\in\Omega,\, x\neq y} \frac{|g(x) - g(y)|}{|x - y|^\sigma} \quad \text{für} \quad g : \Omega \subset \mathbb{R}^d \to \mathbb{R}$$

verwenden. Für $B > 0$ definieren wir dann die Klasse **glatter** Funktionen

$$\mathcal{F}_{\beta,d,B} := \left\{f \in C^n\left(\left[-\frac{1}{2}, \frac{1}{2}\right]^d\right) : \|f\|_{C^{0,\beta}} \leq B\right\},$$

wobei β die **Glattheit** der Funktionen bezeichnet. Wir merken an, dass dieser Raum für $\beta \in \mathbb{N}$, d.h. für $\sigma = 1$ und $n = \beta - 1$, gerade dem Raum der Funktionen in $C^{\beta-1}([-\frac{1}{2}, \frac{1}{2}]^d) = C^n([-\frac{1}{2}, \frac{1}{2}]^d)$ entspricht, deren Ableitungen bis zum Grad $\beta - 1 = n$ durch B beschränkt und lipschitzstetig mit einer maximalen Lipschitzkonstante B sind. In diesem Sinne sind die Räume $\mathcal{F}_{\beta,d,B}$ für $\beta \in \mathbb{N}$ und $B = 1$ ähnlich zu den zuvor betrachteten Sobolev-Einheitsbällen $F_{n,d}$, die $(n-1)$-mal stetig differenzierbare Funktionen mit lipschitzstetigen schwachen Ableitungen bis zum Grad n umfassen, deren Lipschitzkonstante durch 1 beschränkt ist. Im Gegensatz zu den $\mathcal{F}_{\beta,d,B}$ wird hier allerdings insbesondere nur $(n-1)$-fache stetige Differenzierbarkeit gefordert. Des Weiteren beschränkt die Norm in den $\mathcal{F}_{\beta,d,B}$ das Supremum anstelle des essenziellen Supremums wie in den Sobolev-Räumen.

In diesem Unterkapitel betrachten wir insbesondere auch quantisierte Gewichte. Dies ist von Interesse, da die Gewichte in realen Implementierungen nur eine endliche Anzahl

an Werten annehmen können. Folglich haben die nachfolgenden Resultate eine höhere praktische Anwendbarkeit als die zuvor formulierten Ergebnisse.

Nachdem wir im Folgenden eine Hilfsaussage zur Beschränkung von Multilayer-Perceptrons auf einen Wertebereich $[-B, B]$ aufstellen, werden wir – analog zum vorherigen Unterkapitel – ein Resultat zur Approximation der Multiplikation beweisen. Mit diesem Ergebnis werden wir schließlich die Approximation von Polynomen betrachten und diese gemeinsam mit sogenannten "Abschneidefunktionen" verwenden, um in unserem finalen Satz glatte Funktionen mithilfe ihrer Taylor-Polynome zu approximieren.

Zunächst formulieren wir ein einfaches Hilfslemma, das uns ermöglicht, Multilayer Perceptrons mit einer beschränkten Realisierung zu erhalten. Dieses Lemma wird erst im späteren Verlauf der Arbeit wieder benötigt.

Lemma 3.4. *Es gibt eine universelle Konstante $c > 0$, sodass folgendes gilt:*

Für beliebige $d, s, m \in \mathbb{N}, B > 0, \varepsilon \in (0, \frac{1}{2})$ und ein beliebiges Multilayer Perceptron Ψ mit Eingabedimension d, Ausgabedimension m und (s, ε)-quantisierten Gewichten existiert ein Multilayer Perceptron Φ mit derselben Ein- und Ausgabedimension, für das gilt:

- $M(\Phi) \leq 2M(\Psi) + c \cdot m$ *und* $L(\Phi) \leq L(\Psi) + 2$.
- *Alle Gewichte von Φ sind (s_0, ε)- quantisiert, wobei* $s_0 := \max\{\lceil \log_2(\lceil B \rceil) \rceil, s\}$.
- $R_\varrho(\Phi) = (\underbrace{\tau_B \times \ldots \times \tau_B}_{m \text{ Mal}}) \circ R_\varrho(\Psi)$, *wobei die Funktion*

$$\tau_B : \mathbb{R} \to [-\lceil B \rceil, \lceil B \rceil], \quad y \mapsto \operatorname{sign}(y) \cdot \min\{|y|, \lceil B \rceil\}$$

1-Lipschitz ist und $\tau_B(y) = y$ für alle $y \in \mathbb{R}$ mit $|y| \leq \lceil B \rceil$.

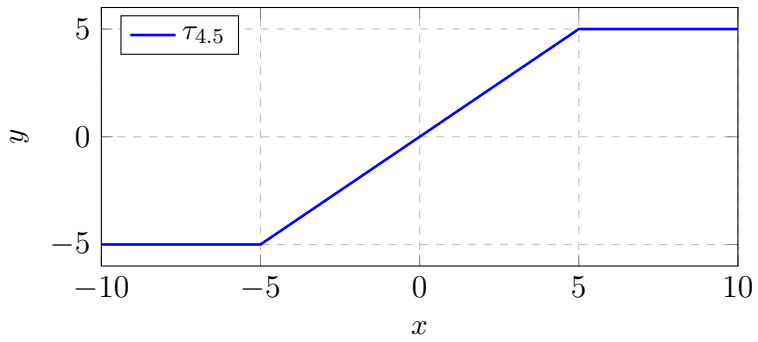

Abbildung 3.6: Die Funktion τ_B für $B = 4.5$ beschränkt alle Eingaben auf $|x| \leq \lceil 4.5 \rceil = 5$

Beweis. Wir konstruieren zunächst ein Multilayer Perceptron $\Phi^B := ((A_1, b_1), (A_2, b_2))$ in folgender Weise:

- $A_1 \in \mathbb{R}^{2m \times m}$ ist die Matrix, die jede Komponente des Eingabevektors dupliziert, also die lineare Abbildung $\mathbb{R}^m \to \mathbb{R}^{2m}$, $(x_1, \dots, x_m) \mapsto (x_1, x_1, x_2, x_2, \dots, x_m, x_m)$ darstellt;
- $A_2 \in \mathbb{R}^{2m \times m}$ ist die Matrix, die die Differenz zweier aufeinanderfolgender Komponenten bildet, also die lineare Abbildung $\mathbb{R}^{2m} \to \mathbb{R}^m$, $(x_1, \dots, x_{2m}) \mapsto (x_1 - x_2, x_3 - x_4, \dots, x_{2m-1} - x_{2m})$ darstellt;
- $b_1 := \lceil B \rceil \cdot (1, -1, 1, -1, \dots, 1, -1)^T \in \mathbb{R}^{2m}$ und $b_2 := -\lceil B \rceil \cdot (1, \dots, 1)^T \in \mathbb{R}^m$.

Es ist leicht zu sehen, dass $\|A_1\|_{\ell^0}, \|A_2\|_{\ell^0} \leq 2m$ sowie $\|b_1\|_{\ell^0} \leq 2m$ und $\|b_2\|_{\ell^0} \leq m$. Damit ist $M(\Phi^B) \leq 7m$ und offensichtlich ist $L(\Phi^B) = 2$. Zudem sind die Gewichte von Φ^B per Konstruktion $(\lceil \log_2(\lceil B \rceil) \rceil, \varepsilon)$-quantisiert für beliebiges $\varepsilon \in (0, \frac{1}{2})$.

Es ist $R_\varrho(\Phi^B) = \tau_B \times \ldots \times \tau_B$, wobei das kartesische Produkt m Faktoren enthält: Für ein $i \in \{1, \dots, m\}$ ist

$$R_\varrho(\Phi^B)(x)_i = \varrho(x_i + \lceil B \rceil) - \varrho(x_i - \lceil B \rceil) - \lceil B \rceil = \tau_B(x_i),$$

wie gewollt (siehe auch Abbildung 3.7). Sei also $\Phi := \Phi^B \odot \Psi$. Mithilfe von Bemerkung 2.17 gelten damit die Behauptungen für die Realisierung, die Tiefe und die Anzahl der nichtnull Gewichte (sei zum Beispiel $c = 14$). Die Behauptung zur Quantisierung folgt mit Bemerkung 2.12. □

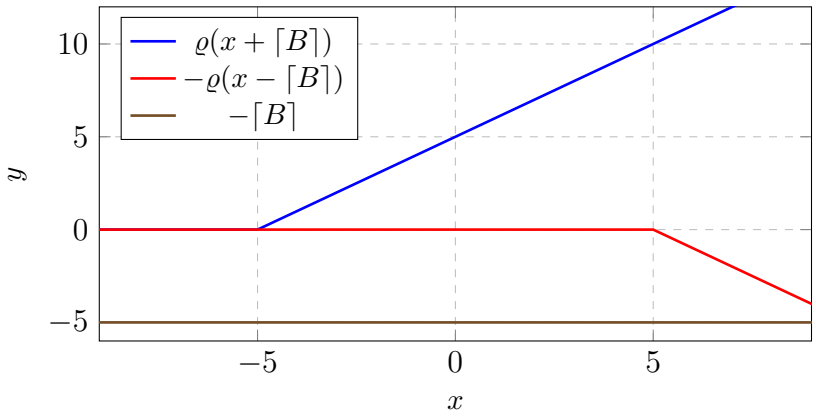

Abbildung 3.7: Die Terme $\varrho(x+\lceil B \rceil)$, $-\varrho(x-\lceil B \rceil)$ und $-\lceil B \rceil$ für $B = 4.5$. Deren Summe ist gerade τ_B.

Im folgenden Lemma formulieren wir, ähnlich wie im vorherigen Kapitel, eine Aussage zur Approximation der Multiplikation mittels eines Multilayer Perceptrons. Im Gegensatz zu unserem vorherigen Resultat konstruieren wir hier jedoch ein Multilayer Perceptron mit einer in gewisser Weise beschränkten Tiefe. Genauer gesagt gibt es in diesem Lemma einen bestimmten "Trade-Off" zwischen der Tiefe und der Anzahl der nichtnull Gewichte.

Der Beweis wird in einigen Aspekten ähnlich aufgebaut sein; aufgrund der Quantisierung und anderer struktureller Unterschiede der Aussagen können wir jedoch nur eingeschränkt Ergebnisse aus dem Beweis von Lemma 3.1 wiederverwenden.

Lemma 3.5. *Sei $\theta > 0$. Für jedes $L \in \mathbb{N}$ mit $L > (2\theta)^{-1}$ und $M \geq 1$ existieren Konstanten $c = c(L, M, \theta) \in \mathbb{N}$ und $s = s(M) \in \mathbb{N}$, sodass für jedes $\varepsilon \in (0, \frac{1}{2})$ ein Multilayer Perceptron $\tilde{\times}$ existiert mit:*

- *$\tilde{\times}$ hat höchstens $c \cdot \varepsilon^{-\theta}$ nichtnull, (s, ε)-quantisierte Gewichte.*
- *$\tilde{\times}$ hat $2L + 7$ Schichten.*
- *Für alle $x, y \in [-M, M]$ mit $x = 0$ oder $y = 0$ gilt $R_\varrho(\tilde{\times})(x, y) = 0$.*
- *Für alle $x, y \in [-M, M]$ gilt $|xy - R_\varrho(\tilde{\times})(x, y)| \leq \varepsilon$.*

Beweis. Zunächst betrachten wir wieder die "Sägezahn"-Funktion $g : [0, 1] \to [0, 1]$ gegeben durch

$$g(x) := \begin{cases} 2x, & x \leq \frac{1}{2}, \\ 2(1 - x), & x > \frac{1}{2}, \end{cases}$$

und ihre t-fachen Iterationen $g_t(x) = \underbrace{g \circ \cdots \circ g}_{t \text{ Mal}}(x)$. Wie bereits erwähnt, gilt:

$$g_t(x) := \begin{cases} 2^t(x - \frac{2i}{2^t}), & x \in [\frac{2i}{2^t}, \frac{2i+1}{2^t}], \ i = 0, 1, \ldots, 2^{t-1} - 1, \\ 2^t(\frac{2i}{2^t} - x), & x \in [\frac{2i-1}{2^t}, \frac{2i}{2^t}], \ i = 1, 2, \ldots, 2^{t-1}. \end{cases} \tag{3.18}$$

Wir behaupten nun, dass

$$g_t(x) = 2^t \left(\varrho(x) + \sum_{k=1}^{2^{t-1}-1} 2 \cdot \varrho\left(x - \frac{2k}{2^t}\right) - \sum_{\ell=1}^{2^{t-1}} 2 \cdot \varrho\left(x - \frac{2\ell - 1}{2^t}\right) \right) \quad \text{für alle } x \in [0, 1].$$

Sei hierfür zunächst $x \in [\frac{2i}{2^t}, \frac{2i+1}{2^t}]$, $i \in \{0, 1, \ldots, 2^{t-1} - 1\}$. Dann ist

$$\begin{aligned} g_t(x) &= 2^t \left(x + \sum_{k=1}^{i} \left(2x - \frac{4k}{2^t}\right) - \sum_{\ell=1}^{i} \left(2x - \frac{4\ell - 2}{2^t}\right) \right) \\ &= 2^t \left(x - \frac{2i}{2^t}\right). \end{aligned}$$

Für $x \in [\frac{2i-1}{2^t}, \frac{2i}{2^t}]$, $i \in \{1, \ldots, 2^{t-1}\}$ erhält man dann mit ähnlicher Betrachtung ein analoges Ergebnis.

Sei dann $\Phi_t := ((A_1, b_1), (A_2, b_2))$ mit

$$\begin{aligned} A_1 &:= (1, \ldots, 1)^T, \\ b_1 &:= \left(0, -\frac{2 \cdot 1}{2^t}, \ldots, -\frac{2 \cdot (2^{t-1} - 1)}{2^t}, -\frac{2 \cdot 1 - 1}{2^t}, \ldots, -\frac{2 \cdot 2^{t-1} - 1}{2^t}\right)^T, \\ A_2 &:= \left(2^t, \underbrace{2^{t+1}, \ldots, 2^{t+1}}_{2^{t-1}-1 \text{ Mal}}, \underbrace{-2^{t+1}, \ldots, -2^{t+1}}_{2^{t-1} \text{ Mal}}\right), \\ b_2 &:= (0), \end{aligned}$$

wobei $A_1, b_1 \in \mathbb{R}^{2^t \times 1}$ und $A_2 \in \mathbb{R}^{1 \times 2^t}$, sodass $g_t = R_\varrho(\Phi_t)|_{[0,1]}$. Damit existiert also für jedes g_t, $t \in \mathbb{N}$, ein MLP Φ_t mit Eingabedimesion und Ausgabedimension 1, 2 Schichten, $1 + 2^t + 1 \leq 4 \cdot 2^t$ Neuronen und höchstens $3 \cdot 2^t \leq 4 \cdot 2^t$ nichtnull Gewichten, welches g_t darstellt. Außerdem können alle Gewichte in $[-2^{t+1}, 2^{t+1}] \cap 2^{-t}\mathbb{Z} \subset [-2^{m+1}, 2^{m+1}] \cap 2^{-m}\mathbb{Z}$ gewählt werden, wenn $1 \leq t \leq m$ für $m \in \mathbb{N}$. Setzen wir außerdem $g_0 := \mathrm{Id}_{[0,1]}$, sehen wir mithilfe von Lemma 2.13, dass auch für $t = 0$ ein MLP mit den oben genannten Eigenschaften existiert.

Nun wählen wir

$$\begin{aligned} s_0 &:= 1 + \lceil \log_2 M \rceil \in \mathbb{N}, \\ M_0 &:= 2^{s_0}, \\ m &:= s_0 + \lceil \log_2(\frac{1}{\varepsilon})/2 \rceil \in \mathbb{N}, \\ N &:= \left\lceil \frac{m}{L} \right\rceil \in \mathbb{N}. \end{aligned}$$

Wir bemerken, dass $M_0 = 2 \cdot 2^{\lceil \log_2 M \rceil} \geq 2 \cdot 2^{\log_2 M} = 2M$. Durch Division mit Rest können wir nun jedes $1 \leq t \leq m$ darstellen als $t = kN + r$ mit $k \in \mathbb{N}_0$ und $r \in \{0, \dots, N-1\}$. Dabei gilt $k = \frac{t-r}{N} \leq \frac{t}{N} \leq \frac{m}{N} \leq L$. Wir können nun g_t als Verkettung $g_t = \underbrace{g \circ \cdots \circ g}_{k \text{ Mal}} \circ g_r$ betrachten und nach diesem Muster $\Phi_0^{(t)} := \underbrace{\Phi_N \odot \cdots \odot \Phi_N}_{k \text{ Mal}} \odot \Phi_r$ konstruieren, sodass $g_t = R_\varrho(\Phi_0^{(t)})|_{[0,1]}$. Mit Bemerkung 2.17 hat $\Phi_0^{(t)}$ die Anzahl von Schichten $2(k+1) \leq 2(L+1) \leq 2L+3$ und die Anzahl der nichtnull Gewichte

$$M(\Phi_0^{(t)}) \leq 4^k \cdot \max\{M(\Phi_N), M(\Phi_r)\} \leq 4^k \cdot 4 \cdot 2^N \leq 4^{L+1} \cdot 2^N.$$

Sei nun $\Phi^{\mathrm{Id}}_{1,2L+3-2(k+1)}$ wie in Bemerkung 2.14. Dann hat $\Phi^{(t)} := \Phi^{\mathrm{Id}}_{1,2L+3-2(k+1)} \odot \Phi_0^{(t)}$ genau $2L+3$ Schichten, erfüllt $R_\varrho(\Phi^{(t)})|_{[0,1]} = g_t$, und hat höchstens

$$\begin{aligned} 2M(\Phi_0^{(t)}) + 2M(\Phi^{\mathrm{Id}}_{1,2L+3-2(k+1)}) &\leq 4^{L+2} \cdot 2^N + 2 \cdot (2L+3) \\ &\leq (2L + 3 + 4^{L+2}) \cdot 2^N =: c_1 \cdot 2^N \end{aligned} \tag{3.19}$$

nichtnull Gewichte. Alle Gewichte sind aus $[-2^{m+1}, 2^{m+1}] \cap 2^{-m}\mathbb{Z}$.

Wie im Beweis von Lemma 3.1 wollen wir diese g_t nun nutzen, um $f : [0,1] \to [0,1]$, $f(x) = x^2$ zu approximieren. Wir haben bereits gezeigt, dass für die Funktionen

$$f_m : [0,1] \to [0,1], \; f(x) = x - \sum_{t=1}^{m} \frac{g_t(x)}{2^{2t}}$$

gilt:

$$\|f - f_m\|_{\sup} = 2^{-2-2m}. \tag{3.20}$$

Seien nun

$$\Psi := P(\Phi^{\mathrm{Id}}_{1,2L+3}, P(\Phi^{(1)}, P(\Phi^{(2)}, \dots, P(\Phi^{(m-1)}, \Phi^{(m)}) \dots)))$$

und $\Phi_{\mathrm{sum}} := ((A_{\mathrm{sum}}, 0))$ mit $A_{\mathrm{sum}} := (1, -2^{-2\cdot 1}, -2^{-2\cdot 2}, \dots, -2^{-2m}) \in \mathbb{R}^{1\times(m+1)}$. Das MLP $\Phi_0 := \Phi_{\mathrm{sum}} \odot \Psi$ erfüllt $R_\varrho(\Phi_0)|_{[0,1]} = f_m$, hat $(2L+3)+1 = 2L+4$ Schichten und hat höchstens $2\cdot(2\cdot(2L+3) + m\cdot c_1 \cdot 2^N) + 2\cdot(m+1) \leq c_2 \cdot m \cdot 2^N$ nichtnull Gewichte. Letztere Schranke kommt dabei durch

$$\begin{aligned} M(\Phi_0) &\leq 2M(\Psi) + 2M(\Phi_{\mathrm{sum}}) \\ &\leq 2\cdot(M(\Phi^{\mathrm{Id}}_{1,2L+3}) + M(\Phi^{(1)}) + \ldots + M(\Phi^{(m)})) + 2M(\Phi_{\mathrm{sum}}) \end{aligned}$$

und Ungleichung 3.19 zustande. Alle Gewichte liegen in $[-2^{m+1}, 2^{m+1}] \cap 2^{-2m}\mathbb{Z}$ und es ist $c_2 = c_2(L) > 0$.

Nun gehen wir ähnlich wie im Beweis von Lemma 3.2 vor und verwenden Expansion 3.5, um die Multiplikation $\tilde{\times}$ zu definieren. Sei hierfür

$$h : \left[-\frac{M_0}{2}, \frac{M_0}{2}\right]^2 \to \mathbb{R}, \;\; h(x,y) = \frac{M_0^2}{2}\left(f_m\left(\frac{|x+y|}{M_0}\right) - f_m\left(\frac{|x|}{M_0}\right) - f_m\left(\frac{|y|}{M_0}\right)\right).$$

Hierfür lässt sich folgendermaßen ein MLP $\tilde{\times}$ mit $h = R_\varrho(\tilde{\times})|_{[-M_0/2, M_0/2]}$ definieren. Seien

$$\begin{aligned} &i : \mathbb{R}^2 \to \mathbb{R}^3, \;\; i(x,y) = \left(\frac{|x+y|}{M_0}, \frac{|x|}{M_0}, \frac{|y|}{M_0}\right), \\ &F_m : \mathbb{R}^3 \to \mathbb{R}^3, \;\; F_m(x,y,z) = (f_m(x), f_m(y), f_m(z)), \\ &j : \mathbb{R}^3 \to \mathbb{R}, \;\; j(x,y,z) = \frac{M_0^2}{2}\cdot(x-y-z), \end{aligned}$$

sodass also $h = j \circ F_m \circ i\,|_{[-M_0/2, M_0/2]}$. Die Funktion i lässt sich als Realisierung eines MLPs mit 2 Schichten darstellen (unter Verwendung der Identität $|x| = \varrho(x) + \varrho(-x)$). Die Funktion F_m lässt sich durch die disjunkte Parallelisierung $\mathcal{P}(\Phi_0, \mathcal{P}(\Phi_0, \Phi_0))$ mit $2L+4$ Schichten darstellen, und die Funktion j lässt sich durch ein MLP mit einer Schicht darstellen. Bilden wir nun die spärliche Verkettung dieser drei Netze, so erhalten wir ein Netz $\tilde{\times}$ mit $2 + (2L+4) + 1 = 2L+7$ Schichten, dessen Realisierung, eingeschränkt auf $\left[-\frac{M_0}{2}, \frac{M_0}{2}\right]$, gerade h ist. Unter Verwendung der vorherigen Schranke für Φ_0 ist die Anzahl der nichtnull Gewichte von $\tilde{\times}$ höchstens $c_3 \cdot m \cdot 2^N$, wobei $c_3 = c_3(L) \in \mathbb{N}$. Zudem sind alle Gewichte in $[-2^{2m+2s_0}, 2^{2m+2s_0}] \cap 2^{-2m-s_0}\mathbb{Z}$, da $\frac{M_0^2}{2} = (2^{s_0})^2 = 2^{2s_0} \leq 2^{2m+2s_0}$ und $\frac{1}{M_0} = 2^{-s_0} = k \cdot 2^{-2m-s_0}$ für ein $k \in \mathbb{Z}$. Weiter gilt für $x, y \in [-M, M] \subset [-M_0/2, M_0/2]$ mit $x \cdot y = 0$, dass $R_\varrho(\tilde{\times})(x,y) = h(x,y) = 0$, da per Konstruktion $f_m(0) = 0$.

Seien schließlich $x, y \in [-M, M] \subset [-M_0/2, M_0/2]$. Dann ist

$$|x+y| \leq |x| + |y| \leq 2M \leq M_0$$

und daher, analog zu unserem Vorgehen im Beweis von Lemma 3.2,

$$\begin{aligned}
|h(x,y)-xy| =& \left|h(x,y) - M_0^2 \cdot \frac{x}{M_0} \cdot \frac{y}{M_0}\right| \\
=& M_0^2 \left| \frac{1}{2}\left(f_m\left(\frac{|x+y|}{M_0}\right) - f_m\left(\frac{|x|}{M_0}\right) - f_m\left(\frac{|y|}{M_0}\right)\right)\right. \\
& \left. - \frac{1}{2}\left(\left(\frac{x}{M_0}+\frac{y}{M_0}\right)^2 - \left(\frac{x}{M_0}\right)^2 - \left(\frac{y}{M_0}\right)^2\right)\right| \\
\leq& \frac{M_0^2}{2}\left(\left|f_m\left(\frac{|x+y|}{M_0}\right) - \left(\frac{|x+y|}{M_0}\right)^2\right| + \left|f_m\left(\frac{|x|}{M_0}\right) - \left(\frac{|x|}{M_0}\right)^2\right| \right. \\
& \left. + \left|f_m\left(\frac{|y|}{M_0}\right) - \left(\frac{|y|}{M_0}\right)^2\right|\right) \\
\leq& \frac{M_0^2}{2} \cdot (2^{-2m-2} + 2^{-2m-2} + 2^{-2m-2}) \leq \left(\frac{M_0}{2^m}\right)^2 \leq \varepsilon
\end{aligned}$$

Dabei haben wir im zweiten Schritt die Expansion 3.5, im dritten Schritt $z^2 = |z|^2$ und im vierten Schritt 3.20 verwendet. Im letzten Schritt nutzen wir, dass

$$2^m = 2^{s_0 + \lceil \log_2(\frac{1}{\varepsilon})/2 \rceil} \geq 2^{s_0} \cdot 2^{\log_2(\frac{1}{\varepsilon})/2} = M_0 \cdot \varepsilon^{-1/2}.$$

Wie wir bereits bemerkt haben, ist die Anzahl der Schichten von $\tilde{\times}$ gerade $2L+7$. Außerdem sind alle Gewichte von $\tilde{\times}$ sind (s,ε)-quantisiert: Wir haben bereits festgestellt, dass alle Gewichte in $[-2^{2m+2s_0}, 2^{2m+2s_0}] \cap 2^{-2m-s_0}\mathbb{Z}$ liegen. Mit $m = s_0 + \lceil \log_2(\frac{1}{\varepsilon}/2) \rceil \leq 1 + s_0 + \frac{1}{2} \cdot \log_2(\frac{1}{\varepsilon})$ und $0 < \varepsilon < \frac{1}{2}$ folgt, dass

$$2^{2m+2s_0} \leq 2^{2+4s_0+\log_2(\frac{1}{\varepsilon})} \leq 2^{2+4s_0} \cdot \varepsilon^{-1} \leq \varepsilon^{-s}$$

für $s := 3 + 4s_0$. Da s_0 nur von M abhängt, ist dann auch $s = s(M)$. Zudem ist mit der Definition von m und, weil $\log_2(\frac{1}{\varepsilon}) \geq 1$

$$2m + s_0 \leq 3s_0 + 2 + \log_2(\frac{1}{\varepsilon}) \leq (4s_0+3) \cdot \log_2(\frac{1}{\varepsilon}) \leq s \cdot \lceil \log_2(\frac{1}{\varepsilon}) \rceil$$

und damit $2^{-2m-s_0}\mathbb{Z} \subset 2^{-s\lceil \log_2(\frac{1}{\varepsilon}) \rceil}\mathbb{Z}$.

Zuletzt gilt für die Anzahl der nichtnull Gewichte von $\tilde{\times}$

$$\begin{aligned}
M(\tilde{\times}) &\leq c_3 \cdot m \cdot 2^N = c_3 \left(s_0 + \left\lceil \frac{\log_2(1/\varepsilon)}{2} \right\rceil\right) \cdot 2^{\lceil m/L \rceil} \\
&\leq 2c_3 \cdot (1+s_0) \cdot \log_2(1/\varepsilon) \cdot 2^{m/L} \\
&= 2c_3 \cdot (1+s_0) \cdot \log_2(1/\varepsilon) \cdot 2^{s_0/L} \cdot 2^{\lceil \log_2(1/\varepsilon)/2 \rceil / L} \\
&\leq 4c_3(1+s_0) \cdot 2^{s_0} \cdot \log_2(1/\varepsilon) \cdot 2^{\log_2(1/\varepsilon) 7(2L)} \\
&= 4c_3 \cdot (1+s_0) \cdot 2^{s_0} \cdot \log_2(1/\varepsilon) \cdot \varepsilon^{-1/(2L)} \leq c(L,M,\theta) \cdot \varepsilon^{-\theta}.
\end{aligned}$$

Wir haben hierbei im dritten Schritt wieder $\log_2(\frac{1}{\varepsilon}) \geq 1$ und im fünften Schritt $2^{\lceil \log_2(1/\varepsilon)/2 \rceil / L} \leq 2^{(\log_2(1/\varepsilon)/2+1)/L} \leq 2^{\log_2(1/\varepsilon)/(2L)} \cdot 2$ verwendet. Im letzten Schritt nutzen wir $s_0 = s_0(M)$ und die Tatsache, dass $\log_2(1/\varepsilon) \cdot \varepsilon^{-1/(2L)} \leq c_4(L,\theta) \cdot \varepsilon^{-\theta}$ für eine geeignete Konstante $c_4(L,\theta) > 0$. Dies zeigen wir: Da $1/(2L) < \theta$, ist die Ungleichung, mit $\gamma := \theta - 1/(2L) > 0$, äquivalent zu $\log_2(1/\varepsilon) \cdot \varepsilon^{\gamma} \leq c_4(L,\theta)$. Es ist also zu zeigen, dass $r(\varepsilon) := \log_2(1/\varepsilon) \cdot \varepsilon^{\gamma} = -\frac{\log_2(\varepsilon)}{\varepsilon^{-\gamma}}$ für $\varepsilon \in (0, \frac{1}{2})$ beschränkt ist. Relevant für die Betrachtung ist dabei nur der linke Rand, da r auf $[\delta, \frac{1}{2}], \delta > 0$ nur ein Produkt zweier stetiger Funktionen auf einem kompakten Intervall ist. Dafür wenden wir den Satz von L'Hospital an und erhalten

$$\lim_{\varepsilon \searrow 0} r(\varepsilon) = \lim_{\varepsilon \searrow 0} -\frac{\varepsilon^{-1}}{\ln(2) \cdot \varepsilon^{-\gamma-1}} = \lim_{\varepsilon \searrow 0} -\frac{\varepsilon^{\gamma}}{\ln(2)} = 0.$$

□

Besonders interessant für uns ist nun das nachfolgende Lemma, welches uns zeigt, wie dank Lemma 3.5 Monome mit Multilayer Perceptrons approximiert werden können.

Lemma 3.6. *Seien $n, d, \ell \in \mathbb{N}$ beliebig. Dann existieren Konstanten $s = s(n) \in \mathbb{N}$, $c = C(d,n,\ell) \in \mathbb{N}$, und $L = L(d,n,\ell) \in \mathbb{N}$, sodass*

$$L \leq (1 + \lceil \log_2 n \rceil) \cdot \left(9 + \tfrac{\ell}{d}\right),$$

und für jedes $\varepsilon \in \left(0, \frac{1}{2}\right)$ und jedes $\alpha \in \mathbb{N}_0^d$ mit $|\alpha| \leq n$ ein Multilayer Perceptron Φ_ε^α mit Eingabedimension d und Ausgabedimension 1 existiert, für das gilt:

- *Φ_ε^α hat höchstens L Schichten.*
- *Die Anzahl der nichtnull Gewichte von Φ_ε^α ist höchstens $c \cdot \varepsilon^{-\frac{d}{\ell}}$.*
- *Alle Gewichte von Φ_ε^α sind (s,ε)-quantisiert.*
- *Für alle $x \in [-\frac{1}{2}, \frac{1}{2}]^d$ gilt:*

$$|R_\varrho(\Phi_\varepsilon^\alpha)(x) - x^\alpha| \leq \varepsilon.$$

Beweis. Sei $d \in \mathbb{N}$ fest und sei $s = s(2) \in \mathbb{N}$ wie in Lemma 3.5 für $M = 2$. Wir führen den Beweis per Induktion über $n \in \mathbb{N}$.

Für $n = 1$ gilt entweder $\alpha = 0$ oder $\alpha = e_i$ für $i \in \{1, \dots, d\}$. Ist $\alpha = 0$, so ist $x^\alpha = 1$. Diese Funktion kann exakt durch ein MLP Φ_ε^α mit einer Schicht und einem (korrekt quantisierten) nichtnull Gewicht dargestellt werden. Ist $\alpha = e_i$, so ist $x^\alpha = x_i$. Auch diese Funktion kann mit einem MLP mit genau einem Layer und einem (korrekt quantisierten) nichtnull Gewicht dargestellt werden, also $R_\varrho(\Phi_\varepsilon^\alpha)(x) = x_i$. Für $n = 1$ gilt die Behauptung also.

Nun sei $k \in \mathbb{N}_{\geq 2}$ und wir nehmen an, dass die Behauptung für alle $1 \leq n < k$ gilt. Wir zeigen, dass sie dann auch für $n = k$ gilt. Ist $|\alpha| < k$, so folgt die Aussage direkt durch

die Induktionshypothese, da der Fall durch $n = k - 1$ abgedeckt ist. Sei im Folgenden also $|\alpha| = n$. Wähle nun $\alpha^{(1)}, \alpha^{(2)} \in \mathbb{N}_0^d$ so, dass $|\alpha^{(2)}| = 2^{\lceil \log_2 k \rceil - 1}$ und $\alpha^{(1)} + \alpha^{(2)} = \alpha$. Zunächst bestätigen wir, dass so eine Wahl von $\alpha^{(1)}$ und $\alpha^{(2)}$ möglich ist: Es ist klar, dass $2^{\lceil \log_2 k \rceil - 1} \in \mathbb{N}$. Außerdem gilt $2^{\lceil \log_2 k \rceil - 1} < k = |\alpha|$, denn definitionsgemäß ist

$$\lceil \log_2 k \rceil - 1 < \log_2 k \leq \lceil \log_2 k \rceil, \tag{3.21}$$

und damit auch

$$2^{\lceil \log_2 k \rceil - 1} < 2^{\log_2 k} = k.$$

Außerdem ist $\log_2 |\alpha^{(2)}| = \lceil \log_2 k \rceil - 1$ und es gilt $|\alpha^{(1)}| \leq |\alpha^{(2)}| < k$: Hierfür bemerken wir zunächst, dass $|\alpha^{(1)}| = k - |\alpha^{(2)}| = k - 2^{\lceil \log_2 k \rceil - 1}$. Mit 3.21 erhalten wir

$$k = 2^{\log_2 k} \leq 2^{\lceil \log_2 k \rceil},$$

oder durch Subtrahieren von $2^{\lceil \log_2 k \rceil - 1}$

$$k - 2^{\lceil \log_2 k \rceil - 1} \leq 2^{\lceil \log_2 k \rceil} - 2^{\lceil \log_2 k \rceil - 1} = (2-1) \cdot 2^{\lceil \log_2 k \rceil - 1} = 2^{\lceil \log_2 k \rceil - 1},$$

also gerade $|\alpha^{(1)}| \leq |\alpha^{(2)}|$.

Durch Anwenden der Induktionshypothese mit $n = |\alpha^{(2)}|$ gibt es also $s_1 = s_1(k) \in \mathbb{N}$, $c_1 = c_1(d, k, \ell) \in \mathbb{N}$ und $L_0 = L_0(d, k, \ell) \in \mathbb{N}$ mit

$$L_0 \leq (1 + \lceil \log_2 k \rceil - 1)(9 + \frac{\ell}{d}), \tag{3.22}$$

sodass für alle $\varepsilon \in (0, \frac{1}{2})$ zwei MLPs $\Phi_\varepsilon^1, \Phi_\varepsilon^2$ existieren mit

$$|R_\varrho(\Phi_\varepsilon^1)(x) - x^{\alpha^{(1)}}| \leq \frac{\varepsilon}{6},$$
$$|R_\varrho(\Phi_\varepsilon^2)(x) - x^{\alpha^{(2)}}| \leq \frac{\varepsilon}{6},$$

für alle $x \in [-\frac{1}{2}, \frac{1}{2}]^d$, wobei beide MLPs höchstens Tiefe L_0 und höchstens $c_1 \varepsilon^{-d/\ell}$ nichtnull, $(s_1, \frac{\varepsilon}{6})$-quantisierte Gewichte haben. Wir stellen fest, dass die Gewichte von Φ_ε^1 und Φ_ε^2 nach Bemerkung 2.12 auch (s_2, ε)-quantisiert sind für ein geeignetes $s_2 = s_2(k) \in \mathbb{N}$. Nun haben Φ_ε^1 und Φ_ε^2 im Allgemeinen nicht Tiefe L_0. Falls nötig, sei für $t \in \{1, 2\}$ dann $\lambda_t = L_0 - L(\Phi_\varepsilon^t)$ und wir ersetzen Φ_ε^t durch $\Phi_{1,\lambda_t}^{\mathrm{Id}} \odot \Phi_\varepsilon^t$ mit $\Phi_{1,\lambda_t}^{\mathrm{Id}}$ wie in Bemerkung 2.14, sodass nun beide MLPs Tiefe L_0 haben. Wir bemerken, dass die Quantisierung der Gewichte hiervon unberührt bleibt, und, dass dank Bemerkung 2.17 und $L_0 = L(d, k, \ell)$ eine Konstante $c_1' = c_1'(d, k, \ell)$ existiert, sodass die Anzahl der nichtnull Gewichte der Φ_ε^t durch $c_1' \cdot \varepsilon^{-d/\ell}$ beschränkt ist.

Sei nun $\tilde{\times}$ das Netzwerk aus Lemma 3.5 mit Genauigkeit $\delta := \frac{\varepsilon}{6}$ sowie $M = 2$ und $\theta = \frac{d}{\ell}$. Da $(2\theta)^{-1} = \frac{\ell}{2d}$, können wir in Lemma 3.5 die Wahl $L = 1 + \lfloor \frac{\ell}{2d} \rfloor > (2\theta)^{-1}$ treffen. Das MLP $\tilde{\times}$ kann also so gewählt werden, dass es höchstens $c_2 \cdot \varepsilon^{-d/\ell}$ nichtnull, (s, δ)-quantisierte Gewichte und höchstens $7 + 2 \cdot (1 + \lfloor \frac{\ell}{2d} \rfloor)$ Schichten hat, wobei s wie im

Start des Beweises und $c_2 = c_2(d, \ell)$ ist. Mithilfe von 2.12 sehen wir, dass die Gewichte von $\tilde{\times}$ mit einer geeigneten Konstante $s_3 = s_3(k) \in \mathbb{N}$ auch (s_3, ε)-quantisiert sind.

Nun definieren wir

$$\Phi_\varepsilon^\alpha := \tilde{\times} \odot P(\Phi_\varepsilon^1, \Phi_\varepsilon^2).$$

Per Konstruktion hat Φ_ε^α höchstens

$$7 + 2 \cdot \left(1 + \left\lfloor \frac{\ell}{2d} \right\rfloor\right) + L_0 \leq 9 + \frac{\ell}{d} + \lceil \log_2 k \rceil \cdot \left(9 + \frac{\ell}{d}\right) = (1 + \lceil \log_2 k \rceil) \cdot \left(9 + \frac{\ell}{d}\right)$$

Schichten, und wir können wie gewünscht ein $L = L(d, k, \ell)$ wählen. Dabei haben wir im ersten Schritt $\lfloor \frac{\ell}{2d} \rfloor \leq \frac{\ell}{2d}$ und 3.22 verwendet. Wir bemerken, dass für $x \in [-\frac{1}{2}, \frac{1}{2}]$

$$\left|R_\varrho(\Phi_\varepsilon^t)(x)\right| \leq \left|x^{\alpha^{(t)}}\right| + \frac{\varepsilon}{6} < 2, \tag{3.23}$$

und sehen, dass für alle $x \in [-\frac{1}{2}, \frac{1}{2}]$ gilt:

$$\begin{aligned}
&|R_\varrho(\Phi_\varepsilon^\alpha)(x) - x^\alpha\| \\
&\quad = \left|R_\varrho(\tilde{\times})\left(R_\varrho(\Phi_\varepsilon^1)(x), R_\varrho(\Phi_\varepsilon^2)(x)\right) - x^\alpha\right| \\
&\quad \leq \left|R_\varrho(\tilde{\times})\left(R_\varrho(\Phi_\varepsilon^1)(x), R_\varrho(\Phi_\varepsilon^2)(x)\right) - R_\varrho(\Phi_\varepsilon^1)(x) \cdot R_\varrho(\Phi_\varepsilon^2)(x)\right| \\
&\qquad + \left|R_\varrho(\Phi_\varepsilon^1)(x) \cdot R_\varrho(\Phi_\varepsilon^2)(x) - x^\alpha\right| \\
&\quad \leq \frac{\varepsilon}{6} + \left|R_\varrho(\Phi_\varepsilon^1)(x) \cdot R_\varrho(\Phi_\varepsilon^2)(x) - R_\varrho(\Phi_\varepsilon^1)(x) \cdot x^{\alpha^{(2)}}\right| + \left|R_\varrho(\Phi_\varepsilon^1)(x) \cdot x^{\alpha^{(2)}} - x^\alpha\right| \\
&\quad \leq \frac{\varepsilon}{6} + \left|R_\varrho(\Phi_\varepsilon^1)(x)\right| \cdot \frac{\varepsilon}{6} + \left|x^{\alpha^{(2)}}\right| \cdot \frac{\varepsilon}{6} \leq \varepsilon.
\end{aligned}$$

Dabei haben wir im zweiten Schritt die Dreiecksungleichung, im dritten Schritt die Approximationseigenschaft von $\tilde{\times}$ und 3.23 (es ist $M = 2$) auf den ersten Term und die Dreiecksungleichung auf den zweiten Term, im vierten Schritt die Approximationseigenschaften von Φ_ε^2 und Φ_ε^1 sowie $x^\alpha = x^{\alpha^{(1)}} \cdot x^{\alpha^{(2)}}$ und im fünften Schritt nochmals 3.23 angewandt. Da alle Gewichte von Φ_ε^α entweder (s_2, ε)- oder (s_3, ε)-quantisiert sind, gibt es eine Konstante $s = s(k)$ (zum Beispiel $s = \max\{s_2, s_3\}$), sodass diese auch (s, ε)-quantisiert sind. Zuletzt stellen wir mithilfe von Bemerkung 2.17 fest, dass eine Konstante $c = c(d, k, \ell) > 0$ existiert, sodass Φ_ε^α nicht mehr als $c \cdot \varepsilon^{-d/\ell}$ nichtnull Gewichte besitzt. □

Mit diesem Ergebnis über Monome können wir nun auch MLPs mit einer kontrollierbaren Anzahl an Schichten konstruieren, die Polynome approximieren. Außerdem erzielen wir ein interessantes Resultat zur Approximation mehrerer Polynome mit einem einzigen MLP. Sei W die Anzahl der Gewichte, die benötigt wird, um ein einzelnes Polynom darzustellen; dann werden zur Darstellung von m Polynomen nicht, wie man vielleicht erwarten könnte, $m \cdot W$ Gewichte, sondern lediglich $O(m + W)$ Gewichte benötigt.

Lemma 3.7. *Seien $d, m \in \mathbb{N}$, und $B, \beta > 0$. Sei*

$$\left\{c_{\ell,\alpha} : \ell \in \{1,\dots,m\}, \alpha \in \mathbb{N}_0^d, |\alpha| < \beta\right\} \subset [-B, B]$$

eine Menge von Koeffizienten, und $\{x_\ell\}_{\ell=1}^m \subset [-\frac{1}{2}, \frac{1}{2}]^d$ eine Menge von Stützpunkten.

Dann existieren Konstanten $c = c(d, \beta, B) > 0$, $s = s(d, \beta, B) \in \mathbb{N}$, $L = L(d, \beta) \in \mathbb{N}$, wobei

$$L \leq 1 + (1 + \lceil \log_2 \beta \rceil)\left(10 + \tfrac{\beta}{d}\right),$$

sodass für alle $\varepsilon \in (0, \frac{1}{2})$ ein Multilayer Perceptron $\Phi_\varepsilon^{\mathrm{p}}$ mit Ausgabedimension m existiert, welches die folgenden Eigenschaften erfüllt:

- *$\Phi_\varepsilon^{\mathrm{p}}$ hat höchstens L Schichten.*
- *Die Anzahl der nichtnull Gewichte von $\Phi_\varepsilon^{\mathrm{p}}$ ist höchstens $c \cdot \left(\varepsilon^{-d/\beta} + m\right)$.*
- *Alle Gewichte von $\Phi_\varepsilon^{\mathrm{p}}$ sind (s, ε)-quantisiert.*
- *Für alle $\ell = 1, \dots, m$ und alle $x \in [-\frac{1}{2}, \frac{1}{2}]^d$ gilt:*

$$\left| [R_\rho(\Phi_\varepsilon^{\mathrm{p}})]_\ell(x) - \sum_{|\alpha|<\beta} c_{\ell,\alpha} \left(x - x_\ell\right)^\alpha \right| < \varepsilon.$$

Beweis. Wir schreiben $\beta = n + \sigma$, mit $n \in \mathbb{N}_0$ und $\sigma \in (0, 1]$. Seien $\varepsilon \in (0, \frac{1}{2})$ und $\{c_{\ell,\alpha} : \ell \in \{1,\dots,m\}, \alpha \in \mathbb{N}_0^d, |\alpha| < \beta\}$ sowie $\{x_\ell\}_{\ell=1}^m$ wie im Lemma gegeben. Mit dem Binomischen Lehrsatz für Multiindizes erhalten wir

$$(x - x_\ell)^\alpha = \sum_{\gamma \leq \alpha} \binom{\alpha}{\gamma} (-x_\ell)^{\alpha-\gamma} x^\gamma \quad \text{für alle } x \in \mathbb{R}^d \text{ und } \alpha \in \mathbb{N}_0^d.$$

Da für $\alpha \in \mathbb{N}_0^d$ die Aussagen $|\alpha| < \beta$ und $|\alpha| \leq n$ äquivalent sind, ist für all $x \in \mathbb{R}^d$ und $\ell \in \{1, \dots, m\}$

$$\begin{aligned}
\sum_{|\alpha|<\beta} c_{\ell,\alpha}(x - x_\ell)^\alpha &= \sum_{|\alpha|\leq n} \left(c_{\ell,\alpha} \sum_{\gamma \leq \alpha} \binom{\alpha}{\gamma} (-x_\ell)^{\alpha-\gamma} x^\gamma \right) \\
&= \sum_{|\alpha|\leq n} \sum_{\gamma \leq \alpha} c_{\ell,\alpha} \binom{\alpha}{\gamma} (-x_\ell)^{\alpha-\gamma} x^\gamma = \sum_{|\gamma|\leq n} \sum_{\substack{|\alpha|\leq n \\ \alpha \geq \gamma}} c_{\ell,\alpha} \binom{\alpha}{\gamma} (-x_\ell)^{\alpha-\gamma} x^\gamma \\
&= \sum_{|\gamma|\leq n} \Bigg(x^\gamma \underbrace{\sum_{\substack{|\alpha|\leq n \\ \alpha \geq \gamma}} c_{\ell,\alpha} \binom{\alpha}{\gamma} (-x_\ell)^{\alpha-\gamma}}_{=:\tilde{c}_{\ell,\gamma}} \Bigg).
\end{aligned}$$

Da die x_ℓ beschränkt sind, können die Terme $\tilde{c}_{\ell,\gamma}$ für alle $\ell \in \{1, \dots, m\}$ und $\gamma \in \mathbb{N}_0^d$ mit $|\gamma| \leq n$ gleichmäßig durch eine Konstante $C = C(d, \beta, B) \geq 1$ beschränkt werden. Dabei

beschränkt B die Koeffizienten $c_{\ell,\alpha}$, und durch β und d werden sowohl die Anzahl der Summanden als auch die Terme $\binom{\alpha}{\gamma}(-x_\ell)^{\alpha-\gamma}$ beschränkt. Mit mit den oben definierten Koeffizienten gilt

$$\sum_{|\alpha|<\beta} c_{\ell,\alpha}(x-x_\ell)^\alpha = \sum_{|\gamma|\leq n} \tilde{c}_{\ell,\gamma} x^\gamma \quad \text{für alle } x \in \mathbb{R}^d. \tag{3.24}$$

Als Nächstes zeigen wir, dass wir für alle $\gamma \in \mathbb{N}_0^d$ mit $|\gamma| \leq n$ "quantisierte Approximationen" $\tilde{\tilde{c}}_{\ell,\gamma,\varepsilon}$ der $\tilde{c}_{\ell,\gamma}$ finden können, sodass

$$\Big| \sum_{|\gamma|\leq n} \tilde{c}_{\ell,\gamma} x^\gamma - \sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} x^\gamma \Big| < \frac{\varepsilon}{2} \tag{3.25}$$

für alle $x \in [-\frac{1}{2}, \frac{1}{2}]$, wobei $\tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} \in [-\varepsilon^{-s_1}, \varepsilon^{s_1}] \cap 2^{-s_1\lceil \log_2(1/\varepsilon)\rceil}\mathbb{Z}$ und $s_1 = s_1(d,\beta,B) \in \mathbb{N}$. Zunächst bemerken wir, dass wegen $\varepsilon \in (0, \frac{1}{2})$ gilt, dass $\varepsilon^{-s} > 2^s$ für $s \in \mathbb{N}$. Daher können wir $s_1 = s_1(d,\beta,B)$, unabhängig von ε, so wählen, dass $\varepsilon^{-s_1} > C(d,\beta,B) \geq \tilde{c}_{\ell,\gamma}$ für alle γ und ℓ. Mit $\vartheta := 2^{-s_1\lceil \log_2(1/\varepsilon)\rceil} \leq 2^{-s_1 \log_2(1/\varepsilon)} = \varepsilon^{s_1}$ können wir die $\tilde{\tilde{c}}_{\ell,\gamma,\varepsilon}$ also wählen, sodass

$$|\tilde{c}_{\ell,\gamma} - \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon}| \leq \vartheta \leq \varepsilon^{s_1}$$

für alle γ und ℓ. Wir betrachten nun

$$\begin{aligned}
\Big| \sum_{|\gamma|\leq n} \tilde{c}_{\ell,\gamma} x^\gamma - \sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} x^\gamma \Big| &= \Big| \sum_{|\gamma|\leq n} (\tilde{c}_{\ell,\gamma} - \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon}) x^\gamma \Big| \\
&\leq \sum_{|\gamma|\leq n} |\tilde{c}_{\ell,\gamma} - \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon}| \cdot |x^\gamma| \\
&\leq \sum_{|\gamma|\leq n} |\tilde{c}_{\ell,\gamma} - \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon}| \leq N \cdot \vartheta \leq N \cdot \varepsilon^{s_1}
\end{aligned}$$

mit der Anzahl der Summanden $N = N(d,\beta)$. Um 3.25 zu zeigen, können wir also auch verifizieren, dass wir s_1 von ε unabhängig wählen können, sodass $N \cdot \varepsilon^{s_1} \leq \frac{\varepsilon}{2}$, oder äquivalent (sei $s_1 \geq 2$)

$$N \cdot \varepsilon^{s_1 - 1} \leq \frac{1}{2}.$$

Wir wählen hierfür $s_1 \geq 1 + \log_2(2N) = 1 + \log_2(2N(d,\beta))$. Dann ist, da $\varepsilon \in (0, \frac{1}{2})$

$$\begin{aligned}
N \cdot \varepsilon^{s_1 - 1} &\leq N \cdot \varepsilon^{\log_2(2N)} \\
&\leq N \cdot \left(\frac{1}{2}\right)^{\log_2(2N)} = \frac{N}{2N} = \frac{1}{2},
\end{aligned}$$

wie gewünscht. Damit können also ein passendes $s_1 = s_1(d,\beta,B)$ und passende $\tilde{\tilde{c}}_{\ell,\gamma,\varepsilon}$ gewählt werden, sodass 3.25 erfüllt ist. Wir bemerken, dass dann insbesondere auch

$|\tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} - \tilde{c}_{\ell,\gamma}| \leq 1$ für alle γ. Nun schreiben wir $\{\gamma \in \mathbb{N}_0^d : |\gamma| \leq n\} = \{\gamma_1, \dots, \gamma_N\}$ mit $\gamma_i \neq \gamma_j$ für $i \neq j$. Hiermit definieren wir das MLP

$$\begin{aligned}\Phi^{\ell,\varepsilon} &:= ((A^{\ell,\varepsilon}, b^\ell)), \quad \text{wobei}\\ A^{\ell,\varepsilon} &:= (\tilde{\tilde{c}}_{\ell,\gamma_1,\varepsilon}, \dots, \tilde{\tilde{c}}_{\ell,\gamma_N,\varepsilon}) \in \mathbb{R}^{1\times N}, \quad b^\ell := 0 \in \mathbb{R}^1.\end{aligned}$$

Nun wenden wir Lemma 3.6 mit $d' = d$ und $n' = \ell' = n+1 = \lceil \beta \rceil$ an. Es gibt also für beliebige $\delta \in (0, \frac{1}{2})$ und $\gamma \in \mathbb{N}_0^d$ mit $|\gamma| \leq n+1$ ein MLP Φ_δ^γ mit Eingabedimension d, Ausgabedimension 1, höchstens $c_1 \cdot \delta^{-d/(n+1)}$ nichtnull, (s_2, δ)-quantisierten Gewichten und höchstens L_1 Schichten, sodass für alle $x \in [-\frac{1}{2}, \frac{1}{2}]^d$

$$|R_\varrho(\Phi_\delta^\gamma)(x) - x^\gamma| \leq \delta. \tag{3.26}$$

Dabei sind $c_1 = c_1(d', n', \ell') = c_1(d, \beta) > 0$, $s_2 = s_2(n') = s_2(\beta) \in \mathbb{N}$, und $L_1 = L_1(d', n', \ell') = L_1(d, \beta) \in \mathbb{N}$ Konstanten und

$$L_1 \leq (1 + \lceil \log_2 \beta \rceil) \cdot \left(9 + \frac{n+1}{d}\right) \leq (1 + \lceil \log_2 \beta \rceil) \cdot \left(10 + \frac{\beta}{d}\right).$$

Hierbei haben wir im ersten Schritt die Aussage des Lemmas und $\lceil \log_2 \beta \rceil \geq \lceil \log_2(\lceil \beta \rceil) \rceil = \lceil \log_2(n+1) \rceil$ verwendet. Letzteres gilt, weil $2^{\lceil \log_2 \beta \rceil} \geq \beta$, weswegen $2^{\lceil \log_2 \beta \rceil} \geq \lceil \beta \rceil$ und damit $\lceil \log_2 \beta \rceil \geq \log_2(\lceil \beta \rceil)$, woraus die Aussage folgt.

Wie im Beweis des vorherigen Lemmas ersetzen wir, falls nötig, das MLP Φ_δ^γ durch das MLP $\Phi_{1,\lambda_\gamma}^{\mathrm{Id}} \odot \Phi_\delta^\gamma$ mit $\Phi_{1,\lambda_\gamma}^{\mathrm{Id}}$ wie in Bemerkung 2.14 und $\lambda_\gamma = L_1 - L(\Phi_\delta^\gamma)$, sodass alle MLPs Tiefe L_1 haben. Dadurch muss die Konstante c_1 eventuell angepasst werden; diese Anpassung ist allerdings in Abhängigkeit von d und L_1 und da $L_1 = L_1(d, \beta)$ ist weiterhin $c_1 = c_1(d, \beta)$. Andere relevante Eigenschaften der MLPs bleiben unverändert.

Nun wählen wir $\delta := \frac{\varepsilon}{4CN}$ und definieren

$$\begin{aligned}\Phi_\varepsilon^a &:= P(\Phi^{1,\varepsilon}, P(\Phi^{2,\varepsilon}, \dots, P(\Phi^{m-1,\varepsilon}, \Phi^{m,\varepsilon}) \dots)),\\ \Phi_\varepsilon^b &:= P(\Phi_\delta^{\gamma_1}, P(\Phi_\delta^{\gamma_2}, \dots, P(\Phi_\delta^{\gamma_{N-1}}, \Phi_\delta^{\gamma_N}) \dots)).\end{aligned}$$

Schließlich definieren wir $\Phi_\varepsilon^{\mathrm{p}} := \Phi_\varepsilon^a \odot \Phi_\varepsilon^b$. Es ist $[R_\varrho(\Phi_\varepsilon^{\mathrm{p}})]_l(x) = R_\varrho(\Phi^{\ell,\varepsilon})(R_\varrho(\Phi_\varepsilon^b)(x))$ und daher mit der Definition von $\Phi^{\ell,\varepsilon}$

$$\begin{aligned}&\left|[R_\rho(\Phi_\varepsilon^{\mathrm{p}})]_\ell(x) - \sum_{|\alpha|<\beta} c_{\ell,\alpha}\,(x - x_\ell)^\alpha\right|\\ &\quad= \left|\sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} R_\varrho(\Phi_\delta^\gamma)(x) - \sum_{|\alpha|<\beta} c_{\ell,\alpha}\,(x - x_\ell)^\alpha\right| \overset{3.24}{=} \left|\sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} R_\varrho(\Phi_\delta^\gamma)(x) - \sum_{|\gamma|\leq n} \tilde{c}_{\ell,\gamma} x^\gamma\right|\\ &\quad\leq \left|\sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} R_\varrho(\Phi_\delta^\gamma)(x) - \sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} x^\gamma\right| + \left|\sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} x^\gamma - \sum_{|\gamma|\leq n} \tilde{c}_{\ell,\gamma} x^\gamma\right|.\end{aligned}$$

Der rechte Term ist mit 3.25 und $|x^\gamma| \leq 1$ beschränkt durch

$$\left|\sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} x^\gamma - \sum_{|\gamma|\leq n} \tilde{c}_{\ell,\gamma} x^\gamma\right| \leq \sum_{|\gamma|\leq n} \left|\tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} - \tilde{c}_{\ell,\gamma}\right| < \frac{\varepsilon}{2}.$$

Der linke Term ist mithilfe von 3.26 und $|\tilde{\tilde{c}}_{\ell,\gamma,\varepsilon}| \leq 1 + |\tilde{c}_{\ell,\gamma}| \leq 1 + C \leq 2C$ beschränkt durch

$$\left| \sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} R_\varrho(\Phi_\delta^\gamma)(x) - \sum_{|\gamma|\leq n} \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} x^\gamma \right| \leq \sum_{|\gamma|\leq n} \left| \tilde{\tilde{c}}_{\ell,\gamma,\varepsilon} \right| \cdot |R_\varrho(\Phi_\delta^\gamma)(x) - x^\gamma| \leq 2NC\delta = \frac{\varepsilon}{2},$$

und damit ist für alle $x \in [-\frac{1}{2}, \frac{1}{2}]$

$$\left| [R_\rho(\Phi_\varepsilon^{\mathrm{p}})]_\ell(x) - \sum_{|\alpha|<\beta} c_{\ell,\alpha} \, (x - x_\ell)^\alpha \right| < \varepsilon.$$

Wähle nun $s(d, \beta, B) \geq \max\{s_1(d, \beta, B), s_2(\beta)\}$. Dank Bemerkung 2.12 sind die Gewichte der $\Phi^{\ell,\varepsilon}$ und damit von Φ_ε^a auch (s, ε)-quantisiert. Aus dem gleichen Grund sind die Gewichte der Φ_δ^γ und damit von Φ_ε^b ebenfalls (s, δ)-quantisiert, und daher mit Bemerkung 2.12 und, weil $\delta = \frac{\varepsilon}{C_2}$ für ein $C_2 = C_2(d, \beta, B) > 0$, auch (s, ε)-qunatisiert. Die Gewichte von $\Phi_\varepsilon^{\mathrm{p}}$ sind also (s, ε)-quantisiert für $s = s(d, \beta, B)$.

Für die Anzahl der nichtnull Gewichte von Φ_ε^a gilt $M(\Phi_\varepsilon^a) \leq m \cdot N$, wobei $N = N(d, \beta)$. Die Anzahl der nichtnull Gewichte der Φ_ε^γ ist mit Lemma 3.6, mit geeigneter Wahl von $C_3(d, \beta, B)$, durch

$$\begin{aligned} c'(d', n', \ell') \cdot \delta^{-\frac{d'}{\ell'}} &= c'(d, \beta) \cdot \left(\frac{\varepsilon}{4 \cdot C(d, \beta, B) \cdot N(d, \beta)} \right)^{-\frac{d}{n+1}} \\ &\leq C_3(d, \beta, B) \cdot \varepsilon^{-\frac{d}{n+1}} \leq C_3(d, \beta, B) \cdot \varepsilon^{-\frac{d}{\beta}}, \end{aligned}$$

beschränkt und die Anzahl der nichtnull Gewichte von Φ_ε^b damit durch $N(d, \beta) \cdot C_3(d, \beta, B) \cdot \varepsilon^{-\frac{d}{\beta}}$. Zusammenfassend ist die Anzahl der nichtnull Gewichte von $\Phi_\varepsilon^{\mathrm{p}}$ mit Bemerkung 2.17 also, mit entsprechender Wahl von $c := c(d, \beta, B)$, wie gewünscht beschränkt:

$$M(\Phi_\varepsilon^{\mathrm{p}}) \leq 2 \cdot N(d, \beta) \cdot m + 2 \cdot C_3(d, \beta, B) \cdot \varepsilon^{-\frac{d}{\beta}} \leq c \cdot \left(\varepsilon^{-\frac{d}{\beta}} + m \right).$$

Zuletzt gilt, weil Φ_ε^a nur eine Schicht hat, und, weil $L_1 \leq (1 + \lceil \log_2 \beta \rceil) \cdot (10 + \frac{\beta}{d})$, dass $\Phi_\varepsilon^{\mathrm{p}}$ höchsten $1 + (1 + \lceil \log_2 \beta \rceil) \cdot (10 + \frac{\beta}{d})$ Schichten hat.

□

Nun wollen wir zeigen, dass MLPs eine "Abschneidung", also eine Multiplikation mit einer Indikatorfunktion $\chi_{[a_1,b_1]\times\cdots\times[a_d,b_d]}$, approximieren können, sofern der Fehler in der L^p-Norm mit $p < \infty$ gemessen wird. Zunächst führen wir zwei Abschätzungen auf, die wir im Folgenden häufig verwenden werden. Zuerst bemerken wir hierfür, dass die Menge $[-\frac{1}{2}, \frac{1}{2}]^d$ zusammen mit dem Lebesgue-Maß ein Wahrscheinlichkeitsraum ist, wir also die Jensen'sche Ungleichung (siehe [4, Satz 1.3]) anwenden können. Damit ist, für $0 < p \leq q < \infty$ und $f : [-\frac{1}{2}, \frac{1}{2}]^d \to \mathbb{R}$ integrierbar bezüglich L^p und L^q

$$\left(\int_{[-\frac{1}{2},\frac{1}{2}]^d} |f|^p \mathrm{d}\lambda \right)^{\frac{q}{p}} \leq \int_{[-\frac{1}{2},\frac{1}{2}]^d} (|f|^p)^{\frac{q}{p}} \, \mathrm{d}\lambda,$$

da $\varphi(x) = x^{\frac{q}{p}}$ konvex ist. Dies ist dann äquivalent zu $\|f\|^q_{L^p([-\frac{1}{2},\frac{1}{2}]^d)} \leq \|f\|^q_{L^q([-\frac{1}{2},\frac{1}{2}]^d)}$ und damit folgt

$$\|f\|_{L^p([-\frac{1}{2},\frac{1}{2}]^d)} \leq \|f\|_{L^q([-\frac{1}{2},\frac{1}{2}]^d)}. \tag{3.27}$$

Des Weiteren wollen wir zeigen, dass für L^p-integrierbare Funktionen $f_i : [-\frac{1}{2},\frac{1}{2}]^d \to \mathbb{R}$, $i \in \{1,\dots,N\}$ gilt:

$$\left\|\sum_{i=1}^N f_i\right\|_{L^p} \leq N^{\max\{1,p^{-1}\}} \cdot \max\{\|f_i\|_{L^p} : i = 1,\dots,N\}. \tag{3.28}$$

Zuerst betrachten wir hierfür den Fall $p \in (0,1)$. Hier erfüllt $\| \bullet \|_{L^p}$ nicht die Dreiecksungleichung. Wir zeigen allerdings, dass

$$\|f+g\|^p_{L^p} \leq \|f\|^p_{L^p} + \|g\|^p_{L^p}$$

gilt. Dafür nutzen wir, dass die konkave Funktion $\phi : [0,\infty) \to [0,\infty)$, $\phi(x) = x^p$ subadditiv ist, also $\phi(x+y) \leq \phi(x)+\phi(y)$. Dies gilt aus folgendem Grund: Weil $\phi(0) = 0$ und weil ϕ konkav ist, ist für $t \in [0,1]$

$$t\phi(x) = t\phi(x) + (1-t)\phi(0) \leq \phi(tx + (1-t)\cdot 0) = \phi(tx).$$

Der Fall $x = y = 0$ ist trivial, sei also $x \neq 0$ oder $y \neq 0$ und $t = \frac{x}{x+y} \in [0,1]$. Dann gilt hiermit

$$\begin{aligned}\phi(x) &= \phi(t(x+y)) \geq t\phi(x+y),\\ \phi(y) &= \phi((1-t)(x+y)) \geq (1-t)\phi(x+y),\end{aligned}$$

und damit $\phi(x+y) \leq \phi(x)+\phi(y)$. Mit der Subadditivität ist also $|g(x)+f(x)|^p \leq |g(x)|^p + |f(x)|^p$ und daher

$$\int_{[-\frac{1}{2},\frac{1}{2}]^d} |f+g|^p \mathrm{d}\lambda \leq \int_{[-\frac{1}{2},\frac{1}{2}]^d} |f|^p \mathrm{d}\lambda + \int_{[-\frac{1}{2},\frac{1}{2}]^d} |g|^p \mathrm{d}\lambda,$$

was genau der obigen Aussage entspricht. Damit folgt dann die Abschätzung

$$\left\|\sum_{i=1}^N f_i\right\|^p_{L^p} \leq \sum_{i=1}^N \|f_i\|^p_{L^p} \leq N \cdot \max_{1\leq i\leq N} \|f_i\|^p_{L^p},$$

woraus folgt, dass

$$\left\|\sum_{i=1}^N f_i\right\|_{L^p} \leq N^{1/p} \cdot \max_{1\leq i\leq N} \|f_i\|_{L^p}.$$

Im Fall $p \geq 1$ gilt die Dreiecksungleichung für $\| \bullet \|_{L^p}$ (siehe Minkowski'sche Ungleichung [4, Satz 7.17]) und damit

$$\left\|\sum_{i=1}^N\right\|_{L^p} \leq N \cdot \max_{1\leq i\leq N} \|f_i\|_{L^p}.$$

Da $1/p > 1$ genau dann wenn $p \in (0,1)$, können wir beide Fälle zusammenfassen und erhalten Abschätzung 3.28. Nun beweisen wir das oben angesprochene Resultat zur "Abschneidung" von Funktionen.

Lemma 3.8. *Seien $d \in \mathbb{N}$, $p \in (0, \infty)$, $B \geq 1$, und sei $\varepsilon \in (0, \frac{1}{2})$ beliebig. Weiterhin seien für jedes $i = 1, \dots, d$ Intervalle $[a_i, b_i] \subseteq [-\frac{1}{2}, \frac{1}{2}]$ gegeben mit $a_i \leq b_i$.*

Dann existieren Konstanten $c = c(d) \in \mathbb{N}$, $s = s(d, B, p) \in \mathbb{N}$, und ein Multilayer Perceptron Λ_ε mit Eingabedimension $d + 1$ mit folgenden Eigenschaften:

- *Λ_ε hat höchstens vier Schichten.*
- *Die Anzahl der nichtnull Gewichte von Λ_ε ist höchstens c.*
- *Alle Gewichte von Λ_ε sind (s, ε)-quantisiert.*
- *Für jedes Multilayer Perceptron Φ mit Eingabedimension d, Ausgabedimension 1 und $\|R_\varrho(\Phi)\|_{L^\infty([-\frac{1}{2},\frac{1}{2}]^d)} \leq B$ gilt:*

$$\left\| R_\varrho(\Lambda_\varepsilon)\left(\bullet, R_\varrho(\Phi)(\bullet)\right) - \chi_{\prod_{i=1}^d [a_i, b_i]} \cdot R_\varrho(\Phi) \right\|_{L^p([-\frac{1}{2},\frac{1}{2}]^d)} \leq \varepsilon.$$

Beweis. Um ein MLP mit entsprechend quantisierten Gewichten zu konstruieren, benötigen wir zunächst modifizierte Intervallgrenzen $\tilde{a}_i, \tilde{b}_i$. Hierfür seien $p_0 := \lceil p \rceil \in \mathbb{N}$, $s_1 = s_1(d, N, p) := 4d + p_0(1 + 8\lceil B \rceil) \in \mathbb{N}$ und $\tilde{\varepsilon} := 2^{-s_1 \lceil \log_2(1/\varepsilon) \rceil}$. Damit ist dann $\tilde{\varepsilon}^{-1} = 2^{s_1 \lceil \log_2(1/\varepsilon) \rceil} \leq 2^{s_1(1 + \log_2(1/\varepsilon))} \leq 2^{2 s_1 \log_2(1/\varepsilon)} = \varepsilon^{-2s_1}$. Außerdem gilt mithilfe von $\lceil \log_2(\frac{1}{\varepsilon}) \rceil \geq 1$ (zweiter und dritter Schritt), $2^{-\lceil \log_2(1/\varepsilon) \rceil} \leq \varepsilon$ (dritter Schritt), $2^{2x} = 4^x \geq e^x \geq 1 + x \geq x$ und damit $2^{-2x} \leq \frac{1}{x}$ für $x > 0$ (dritter und vierter Schritt) und $\lceil B \rceil \geq B$ (vierter Schritt), dass

$$\begin{aligned} \tilde{\varepsilon} &= 2^{-(4d + p_0(1+8\lceil B \rceil))\lceil \log_2(1/\varepsilon) \rceil} = 2^{-2 \cdot 2d} \cdot \left(2^{-2 \cdot 4 \lceil B \rceil \lceil \log_2(1/\varepsilon) \rceil} \cdot 2^{-\lceil \log_2(1/\varepsilon) \rceil}\right)^{p_0} \\ &\leq \frac{1}{2d} \cdot \left(2^{-2 \cdot 4 \lceil B \rceil} \varepsilon\right)^{p_0} \leq \frac{(\frac{\varepsilon}{4B})^{p_0}}{2d} \leq 1. \end{aligned} \tag{3.29}$$

Im fünften Schritt verwenden wir außerdem, dass $\varepsilon > 1$ und $B, d \geq 1$. Schließlich können wir nach der Definition von $\tilde{\varepsilon}$ für jedes $i \in \{1, \dots, d\}$ Intervallgernzen $\tilde{a}_i, \tilde{b}_i \in [-\frac{1}{2}, \frac{1}{2}] \cap 2^{-s_1 \lceil \log_2(1/\varepsilon) \rceil} \mathbb{Z} = [-\frac{1}{2}, \frac{1}{2}] \cap \tilde{\varepsilon}\, \mathbb{Z}$ wählen, sodass $|a_i - \tilde{a}_i| \leq \tilde{\varepsilon}$ und $|b_i - \tilde{b}_i| \leq \tilde{\varepsilon}$.

Nun bemerken wir, dass $\tilde{\varepsilon}^{-1} \in \mathbb{N}$ und $\tilde{\varepsilon}^{-1} \leq \varepsilon^{-2s_1}$ (siehe oben). Zudem folgt mit $\tilde{a}_i = k \cdot \tilde{\varepsilon}$, wobei $k \in \mathbb{Z}$, dass $\tilde{\varepsilon}^{-1} \cdot \tilde{a}_i = \tilde{\varepsilon}^{-1} \cdot k \cdot \tilde{\varepsilon} = k \in \mathbb{Z}$, und

$$k \leq \frac{1}{2} / 2^{-s_1 \lceil \log_2(1/\varepsilon) \rceil} = 2^{s_1 \lceil \log_2(1/\varepsilon) \rceil - 1} \leq 2^{s_1 \log_2(1/\varepsilon)} = \varepsilon^{-s_1}.$$

Mit analoger Rechnung folgt $\tilde{\varepsilon}^{-1} \cdot \tilde{b}_i = \ell$, wobei $\ell \in \mathbb{Z}$, und $\ell \leq \varepsilon^{-s_1}$. Damit sind also $\tilde{\varepsilon}^{-1}, \tilde{\varepsilon}^{-1}\tilde{a}_i$ und $\tilde{\varepsilon}^{-1}\tilde{b}_i$ Elemente von $\mathbb{Z} \cap [-\varepsilon^{-2s_1}, \varepsilon^{-2s_1}] \subset \mathbb{Z} \cap [-\varepsilon^{-3s_1}, \varepsilon^{-3s_1}]$. Weiter sind $1 + \tilde{\varepsilon}^{-1}\tilde{a}_i$ und $1 + \tilde{\varepsilon}^{-1}\tilde{b}_i$ Elemente von $\mathbb{Z} \cap [-\varepsilon^{-(1+s_1)}, \varepsilon^{-(1+s_1)}] \subset \mathbb{Z} \cap [-\varepsilon^{-3s_1}, \varepsilon^{-3s_1}]$, denn

$$1 + k \leq 1 + \varepsilon^{-s_1} \leq 2\varepsilon^{-s_1} \leq \varepsilon^{-1}\varepsilon^{-s_1} = \varepsilon^{-(1+s_1)},$$

und analog für $\tilde{b}_i$. Deshalb ist die Funktion

$$t_i : \left[-\frac{1}{2}, \frac{1}{2}\right] \to \mathbb{R},\ x \mapsto \varrho\left(\frac{x - \tilde{a}_i}{\tilde{\varepsilon}}\right) - \varrho\left(\frac{x - \tilde{a}_i - \tilde{\varepsilon}}{\tilde{\varepsilon}}\right) - \varrho\left(\frac{x - \tilde{b}_i + \tilde{\varepsilon}}{\tilde{\varepsilon}}\right) + \varrho\left(\frac{x - \tilde{b}_i}{\tilde{\varepsilon}}\right)$$

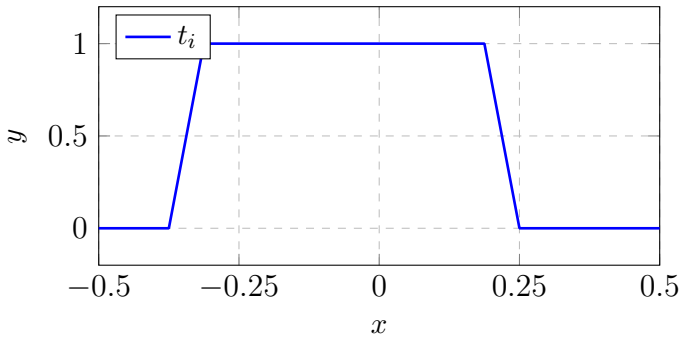

Abbildung 3.8: Darstellung der Funktion t_i mit $\tilde{a}_i = -\frac{3}{8}$, $\tilde{b}_i = \frac{1}{4}$ und $\tilde{\varepsilon} = \frac{1}{16}$.

die Realisierung eines MLPs mit 2 Schichten und höchstens 12 nichtnull, $(3s_1, \varepsilon)$-quantisierten Gewichten.

Nehmen wir nun an, dass $\tilde{b}_i - \tilde{a}_i > 2\varepsilon$, dann ist, falls

$$\begin{aligned}
&x \in [-\tfrac{1}{2}, \tilde{a}_i) : && t_i(x) = 0;\\
&x \in [\tilde{a}_i, \tilde{a}_i + \tilde{\varepsilon}) : && t_i(x) = \tfrac{x-\tilde{a}_i}{\tilde{\varepsilon}};\\
&x \in [\tilde{a}_i + \tilde{\varepsilon}, \tilde{b}_i - \tilde{\varepsilon}] : && t_i(x) = \tfrac{x-\tilde{a}_i}{\tilde{\varepsilon}} - \tfrac{x-\tilde{a}_i-\tilde{\varepsilon}}{\tilde{\varepsilon}} = 1;\\
&x \in (\tilde{b}_i - \tilde{\varepsilon}, \tilde{b}_i] : && t_i(x) = 1 - \tfrac{x-(\tilde{b}_i-\tilde{\varepsilon})}{\tilde{\varepsilon}};\\
&x \in (\tilde{b}_i, \tfrac{1}{2}] : && t_i(x) = 1 - \tfrac{x-(\tilde{b}_i-\tilde{\varepsilon})}{\tilde{\varepsilon}} + \tfrac{x-\tilde{b}_i}{\tilde{\varepsilon}} = 0.
\end{aligned}$$

Als nächstes definieren wir die Funktion $n_\varepsilon : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$, welche die Realisierung von Λ_ε sein wird. Hierfür setzen wir $B_0 := 2^{\lceil \log_2 B \rceil}$. Falls $\tilde{b}_i - \tilde{a}_i > 2\varepsilon$ für alle $i \in \{1, \dots, d\}$, definieren wir

$$n_\varepsilon(x, y) := B_0 \cdot \varrho\left(\sum_{i=1}^{d} t_i(x_i) + \varrho\left(\frac{y}{B_0}\right) - d\right) - B_0 \cdot \varrho\left(\sum_{i=1}^{d} t_i(x_i) + \varrho\left(-\frac{y}{B_0}\right) - d\right).$$

Gibt es ein $i \in \{1, \dots, d\}$, sodass $\tilde{b}_i - \tilde{a}_i < 2\tilde{\varepsilon}$, so wählen wir $n_\varepsilon \equiv 0$. In beiden Fällen lässt sich diese Funktion durch ein MLP Λ_ε mit höchstens vier Schichten darstellen. Letzterer Fall ist trivial. Für den ersten Fall bemerken wir, dass wir neben den t_i auch die Funktionen $\psi_1 : \mathbb{R} \to \mathbb{R},\ x \mapsto \varrho(\frac{x}{B_0})$ und $\psi_2 : \mathbb{R} \to \mathbb{R},\ x \mapsto \varrho(-\frac{x}{B_0})$ durch ein MLP mit zwei Schichten darstellen können. Seien τ_i, $i \in \{1, \dots, d\}$ und Ψ_1, Ψ_2 solche MLPs. Nun betrachten wir deren (teilweise) disjunkte Parallelisierung $\Gamma = \mathcal{P}(\tau_1, \mathcal{P}(\tau_2, \dots, \mathcal{P}(\tau_d, , P(\Psi_1, \Psi_2)) \dots))$. Hierbei ist zu beachten, dass nur zwischen Ψ_1 und Ψ_2 die reguläre Parallelisierung verwendet wurde. Dieses MLP hat ebenfalls zwei Schichten und hat die Realisierung

$$\gamma := R_\varrho(\Gamma) : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d \times \mathbb{R}^2, \quad (x, y) \mapsto \left(t_1(x_1), \dots, t_d(x_d), \varrho(\frac{y}{B_0}), \varrho(-\frac{y}{B_0})\right).$$

Als nächstes stellen wir fest, dass wir auch die Funktion

$$\omega : \mathbb{R}^d \times \mathbb{R}^2 \to \mathbb{R} \quad (x, y) \mapsto B_0 \cdot \varrho\left(\sum_{i=1}^{d} x_i + y_1 - d\right) - B_0 \cdot \varrho\left(\sum_{i=1}^{d} x_i + y_2 - d\right)$$

mithilfe eines MLPs Ω mit zwei Schichten darstellen können. Da $\omega \circ \gamma = n_\varepsilon$, können wir also $\Lambda_\varepsilon = \Omega \odot \Gamma$ wählen und haben somit ein MLP, welches n_ε darstellt und vier Schichten hat, konstruiert. Dabei hat Λ_ε höchstens $c = c(d)$ nichtnull Gewichte, da die Anzahl der nichtnull Gewichte in Ω und Γ nur von d abhängt. Diese sind dank der Wahl von B_0 und voriger Erkenntnisse über die t_i alle (s_2, ε)-quantisiert, für ein $s_2 = s_2(d, B, p) \in \mathbb{N}$ (siehe auch Bemerkung 2.12). Die Abhängigkeit von d und p kommt hierbei durch $s_1 = s_1(d, B, p)$ in unserer Betrachtung der t_i zustande.

Zudem beobachten wir, dass in beiden Fällen ($\tilde{b}_i - \tilde{a}_i \geq 2\tilde{\varepsilon}$ für alle i und $\tilde{b}_i - \tilde{a}_i < 2\tilde{\varepsilon}$ für ein i) für alle $y \in [-B, B] \subset [-B_0, B_0]$ gilt:

- Falls $x \in \prod_{i=1}^d [\tilde{a}_i + \tilde{\varepsilon}, \tilde{b}_i - \tilde{\varepsilon}] (= \emptyset$ im zweiten Fall), dann ist $n_\varepsilon(x, y) = y$;
- Falls $x \in \mathbb{R}^d \setminus \prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i]$, dann ist $n_\varepsilon(x, y) = 0$.

Hierfür betrachten wir die Definition von n_ε und merken an, dass $|y| \leq B_0$ und damit $\varrho(\frac{y}{B_0}) \leq 1 \leq d$. Außerdem ist im ersten Fall $\sum_{i=1}^d t_i(x_i) = d$, und im zweiten Fall $\sum_{i=1}^d t_i(x_i) = 0$.

Des Weiteren ist mithilfe von 3.29 das Lebesgue-Maß von $\prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i] \setminus \prod_{i=1}^d [\tilde{a}_i + \tilde{\varepsilon}, \tilde{b}_i - \tilde{\varepsilon}]$ beschränkt durch $2d\tilde{\varepsilon} \leq (\frac{\varepsilon}{4B})^{p_0}$, was wir später benötigen werden. Sei zuletzt $S = \sum_{i=1}^d t_i(x_i) - d$. Dann ist, weil die ReLU Aktivierungsfunktion 1-Lipschitz ist,

$$\begin{aligned} |n_\varepsilon(x, y)| &= B_0 \cdot \left| \varrho\left(S + \varrho\left(\frac{y}{B_0}\right)\right) - \varrho\left(S + \varrho\left(-\frac{y}{B_0}\right)\right) \right| \\ &\leq B_0 \cdot 1 \cdot \left| \left(S + \varrho\left(\frac{y}{B_0}\right)\right) - \left(S + \varrho\left(-\frac{y}{B_0}\right)\right) \right| \\ &= B_0 \cdot \left| \varrho\left(\frac{y}{B_0}\right) - \varrho\left(-\frac{y}{B_0}\right) \right| = |y| \leq B. \end{aligned}$$

Damit ist n_ε integrierbar und für jedes integrierbare $f : [-\frac{1}{2}, \frac{1}{2}]^d \to [-B, B]$ ist mit der Dreiecksungleichung ($p_0 \geq 1$)

$$\begin{aligned} &\left\| n_\varepsilon(\bullet, f(\bullet)) - \chi_{\prod_{i=1}^d [a_i, b_i]} \cdot f \right\|_{L^{p_0}} \\ &\leq \left\| n_\varepsilon(\bullet, f(\bullet)) - \chi_{\prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i]} \cdot f \right\|_{L^{p_0}} + \left\| \chi_{\prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i]} \cdot f - \chi_{\prod_{i=1}^d [a_i, b_i]} \cdot f \right\|_{L^{p_0}}. \end{aligned}$$

Wir betrachten zunächst den ersten Term. Sei

$$\delta : \left[-\frac{1}{2}, \frac{1}{2}\right]^d \to \mathbb{R}, \quad \delta(x) := n_\varepsilon(x, f(x)) - \chi_{\prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i]}(x) \cdot f(x).$$

Zunächst ist mithilfe der obigen Überlegungen zu n_ε klar, dass $\delta(x) = 0$ gilt für alle $x \notin \prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i] \setminus \prod_{i=1}^d [\tilde{a}_i + \tilde{\varepsilon}, \tilde{b}_i - \tilde{\varepsilon}] =: E$. Außerdem wissen wir, dank $|f|, |n_\varepsilon| \leq B$, dass $|\delta(x)| \leq 2B$. Zudem haben wir gezeigt, dass das Lebesgue-Maß $\lambda(E)$ durch $2d\tilde{\varepsilon} \leq (\frac{\varepsilon}{4B})^{p_0}$ beschränkt ist. Zuletzt ist δ integrierbar, da f und offensichtlich auch die beschränkte

und stetige Funktion n_ε integrierbar sind. Damit ist nun

$$\begin{aligned}\|\delta\|_{L^{p_0}([-\frac{1}{2},\frac{1}{2}])} &= \left(\int_{[-\frac{1}{2},\frac{1}{2}]} |\delta(x)|^{p_0} \mathrm{d}x\right)^{1/p_0} \leq ((2B)^{p_0} \cdot \lambda(E))^{1/p_0} \\ &\leq 2B \cdot (2d\tilde{\varepsilon})^{1/p_0} \leq 2B \cdot ((\frac{\varepsilon}{4B})^{p_0})^{1/p_0} = \frac{\varepsilon}{2}.\end{aligned}$$

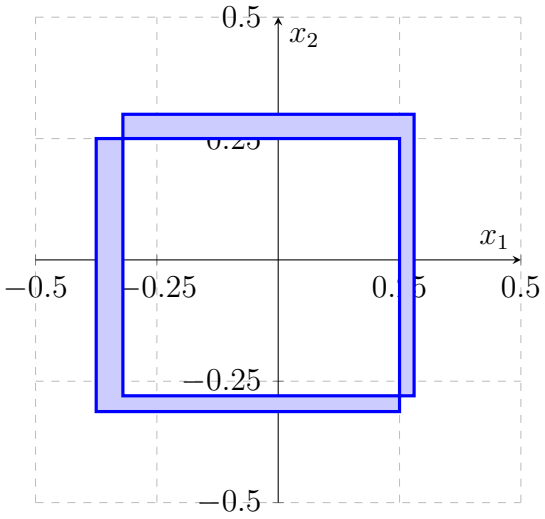

Abbildung 3.9: Beispiel für Menge D mit $d = 2$: $\left(\prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i]\right) \Delta \left(\prod_{i=1}^d [a_i, b_i]\right) = \left([-\frac{3}{8}, \frac{1}{4}] \times [-\frac{1}{16}, \frac{1}{4}]\right) \Delta \left([-0.32, 0.28] \times [-0.28, 0.3]\right)$.

Für den zweiten Term sei nun

$$\gamma : \left[-\frac{1}{2}, \frac{1}{2}\right]^d \to \mathbb{R}, \quad \gamma(x) := \chi_{\prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i]}(x) \cdot f(x) - \chi_{\prod_{i=1}^d [a_i, b_i]}(x) \cdot f(x).$$

Wieder ist γ integrierbar und $|\gamma(x)| \leq 2B$, und, da $a_i, \tilde{a}_i, b_i, \tilde{b}_i \in [-\frac{1}{2}, \frac{1}{2}]$ mit $|a_i - \tilde{a}_i| \leq \tilde{\varepsilon}$ und $|b_i - \tilde{b}_i| \leq \tilde{\varepsilon}$, gilt für $D := \left(\prod_{i=1}^d [\tilde{a}_i, \tilde{b}_i]\right) \Delta \left(\prod_{i=1}^d [a_i, b_i]\right)$, dass $\lambda(D) \leq 2d\tilde{\varepsilon}$ (siehe auch Abbildung 3.9). Dabei bezeichnet Δ die symmetrische Differenz. Offensichtlich ist $\gamma(x) = 0$ für $x \in [-\frac{1}{2}.\frac{1}{2}] \backslash D$, also ist, ähnlich wie zuvor,

$$\|\gamma\|_{L^{p_0}} \leq ((2B)^{p_0} \cdot \lambda(D))^{1/p_0} \leq 2B \cdot (2d\tilde{\varepsilon})^{1/p_0} \leq \frac{\varepsilon}{2}.$$

Mithilfe von Abschätzung 3.27 und $R_\varrho(\Lambda_\varepsilon) = n_\varepsilon$ gilt somit

$$\left\| R_\varrho(\Lambda_\varepsilon)(\bullet, f(\bullet)) - \chi_{\prod_{i=1}^d [a_i, b_i]} \cdot f \right\|_{L^p} \leq \left\| n_\varepsilon(\bullet, f(\bullet)) - \chi_{\prod_{i=1}^d [a_i, b_i]} \cdot f \right\|_{L^{p_0}} \leq \varepsilon,$$

insbesondere auch für $f = R_\varrho(\Phi)$ mit Φ wie im Lemma, wie erwünscht. Die Voraussetzung der Integrierbarkeit von f war dabei keine Einschränkung, da die Realisierung eines MLP Φ mit ReLU Aktivierungsfunktion auf einem kompakten Intervall als beschränkte und stetige Funktion immer integrierbar ist. □

Aus technischen Gründen benötigen wir die nachfolgende Verfeinerung des vorherigen Lemmas. Dieses wollen wir später verwenden, um, wie schon in im Beweis von Satz 3.3, Taylor-Polynome in lokalen Umgebungen zu approximieren.

Lemma 3.9. *Seien $d, m, s \in \mathbb{N}$, $p \in (0, \infty)$ und $\varepsilon \in \left(0, \frac{1}{2}\right)$. Sei Φ ein Multilayer Perceptron mit Eingabedimension d, Ausgabedimension m und (s, ε)-quantisierten Gewichten. Sei $B \geq 1$ so, dass für alle $\ell = 1, \dots, m$ gilt:*

$$\left\| [R_\varrho(\Phi)]_\ell \right\|_{L^\infty([-\frac{1}{2}, \frac{1}{2}]^d)} \leq B.$$

Weiterhin seien für jedes $\ell = 1, \dots, m$ und $i = 1, \dots, d$ Intervalle $[a_{i,\ell}, b_{i,\ell}] \subseteq [-\frac{1}{2}, \frac{1}{2}]$ gegeben mit $a_{i,\ell} \leq b_{i,\ell}$.

Dann existieren Konstanten $c = c(d) > 0$ und $s_0 = s_0(d, B, p) \in \mathbb{N}$ sowie ein Multilayer Perceptron Ψ_ε mit Eingabedimension d und Ausgabedimension 1, sodass gilt:

- *Ψ_ε hat höchstens $L(\Phi) + 6$ Schichten.*
- *Die Anzahl der nichtnull Gewichte von Ψ_ε ist:*

$$M(\Psi_\varepsilon) \leq c \cdot (m + L(\Phi) + M(\Phi)).$$

- *Alle Gewichte von Ψ_ε sind $(\max\{s, s_0\}, \frac{\varepsilon}{m})$-quantisiert.*
- *Für die Realisierung $R_\varrho(\Psi_\varepsilon)$ gilt:*

$$\left\| R_\varrho(\Psi_\varepsilon) - \sum_{\ell=1}^{m} \chi_{\prod_{i=1}^d [a_{i,\ell}, b_{i,\ell}]} \cdot [R_\varrho(\Phi)]_\ell \right\|_{L^p([-\frac{1}{2}, \frac{1}{2}]^d)} \leq \varepsilon.$$

Beweis. Sei zunächst $L := L(\Phi)$ und sei $\tilde{\Phi} := P(\Phi_{d,L}^{\mathrm{Id}}, \Phi)$, wobei $\Phi_{d,L}^{\mathrm{Id}}$ wie in Bemerkung 2.14 ist. Dann hat $\Phi_{d,L}^{\mathrm{Id}}$ genau $L = L(\Phi)$ Schichten, höchstens $2d \cdot L(\Phi)$ nichtnull, $(1, \varepsilon)$-quantisierte Gewichte und es gilt $R_\varrho(\Phi_{d,L}^{\mathrm{Id}}) = \mathrm{Id}_{\mathbb{R}^d}$. Damit hat $\tilde{\Phi}$ also $L = L(\Phi)$ Schichten und höchstens $M(\Phi) + 2d \cdot L(\Phi)$ nichtnull, (s, ε)-quantisierte Gewichte.

Als Nächstes wählen wir $p_0 := \max\{1, p\}$ und für jedes $\ell \in \{1, \dots, m\}$ sei Λ_ε^ℓ ein MLP wie in Lemma 3.8. Hierbei wählen wir $a_i = a_{i,\ell}$, $b_i = b_{i,\ell}$ sowie p_0 statt p und $\frac{\varepsilon}{m}$ statt ε. Dann gibt es Konstanten $c_0 = c_0(d) \in \mathbb{N}$ und $s_0 = s_0(d, B, p) \in \mathbb{N}$, sodass alle Λ_ε^ℓ höchstens c_0 nichtnull, $(s_0, \frac{\varepsilon}{m})$-quantisierte Gewichte haben (wähle zum Beispiel jeweils das Maximum der Konstanten über alle ℓ). Zudem haben alle Λ_ε^ℓ vier Schichten.

Nun sei $P_\ell \in \mathbb{R}^{(d+1)\times(d+m)}$ die Abbildungsmatrix (in Standardbasis) der linearen Abbildung

$$\mathbb{R}^d \times \mathbb{R}^m \ni (x, y) \mapsto (x, y_\ell) \in \mathbb{R}^d \times \mathbb{R}^1$$

und $\Phi_\ell := ((P_\ell, 0))$ das zugehörige MLP. Offensichtlich hat Φ_ℓ eine Schicht und genau $d + 1$ nichtnull, $(1, \varepsilon)$-quantisierte Gewichte.

Schließlich definieren wir $\Phi^{\text{sum}} := ((A^{\text{sum}}, b^{\text{sum}}))$, wobei

$$A^{\text{sum}} := (1, \dots, 1) \in \mathbb{R}^{1\times m}, \quad b^{\text{sum}} := 0.$$

Φ^{sum} hat genau m nichtnull, $(1, \varepsilon)$-quantisierte Gewichte und eine Schicht.

Mit all diesen Vorbereitungen definieren wir nun

$$\Psi_\varepsilon := \Phi^{\text{sum}} \odot P(\Lambda^1_\varepsilon \odot \Phi_1, P(\Lambda^2_\varepsilon \odot \Phi_2, \dots, P(\Lambda^{m-1}_\varepsilon \odot \Phi_{m-1}, \Lambda^m_\varepsilon \odot \Phi_m)\dots)) \odot \tilde{\Phi}.$$

Zunächst stellen wir mithilfe von Bemerkung 2.17 fest, dass $\Lambda^\ell_\varepsilon \odot \Phi_\ell$ fünf Schichten und höchstens $2 \cdot (c_0 + d + 1)$ Gewichte hat. Mit den Bemerkung 2.17 und 2.12 sehen wir, dass Ψ_ε genau $1 + 5 + L(\Phi)$ Schichten und höchstens

$$16 \cdot \max\{m, 2 \cdot (c_0 + d + 1), M(\Phi) + 2dL(\Phi)\} \leq c \cdot (m + L(\Phi) + M(\Phi))$$

nichtnull, $(\max\{s_0, s\}, \frac{\varepsilon}{m})$-quantisierte Gewichte hat für eine Konstante $c = c(d) > 0$.

Wir stellen fest, dass für alle $x \in \mathbb{R}^d$

$$R_\varrho(\Psi_\varepsilon)(x) = \sum_{\ell=1}^m R_\varrho(\Lambda^\ell_\varepsilon)(x, [R_\varrho(\Phi)(x)]_\ell).$$

Da $p_0 \geq 1$ gilt die Dreiecksungleichung und es ist

$$\begin{aligned}
&\left\| R_\varrho(\Psi_\varepsilon) - \sum_{\ell=1}^m \chi_{\prod_{i=1}^d [a_{i,\ell}, b_{i,\ell}]} \cdot [R_\varrho(\Phi)]_\ell \right\|_{L^{p_0}([-\frac{1}{2}, \frac{1}{2}]^d)} \\
&\leq \sum_{\ell=1}^m \left\| R_\varrho(\Lambda^\ell_\varepsilon)(\bullet, [R_\varrho(\Phi)(\bullet)]_\ell) - \chi_{\prod_{i=1}^d [a_{i,\ell}, b_{i,\ell}]} \cdot [R_\varrho(\Phi)]_\ell \right\|_{L^{p_0}([-\frac{1}{2}, \frac{1}{2}]^d)} \\
&\leq \sum_{\ell=1}^m \frac{\varepsilon}{m} = \varepsilon.
\end{aligned}$$

Im dritten Schritt haben wir hierbei die Eigenschaft der MLPs Λ^ℓ_ε nach Lemma 3.8 verwendet. Da $p_0 \geq p$ folgt die Aussage des Satzes mit Ungleichung 3.27. □

Schließlich benötigen wir noch ein allgemeines Resultat zu unserer Klasse glatter Funktionen; genauer zu deren lokaler Approximation durch Taylor-Polynome. Diese Taylor-Polynome werden wir im Anschluss verwenden, um die Approximation der glatten Funktionen durch MLPs zu realisieren.

Lemma 3.10. *Sei $\beta \in (0, \infty)$ und schreibe $\beta = n + \sigma$ mit $n \in \mathbb{N}_0$ und $\sigma \in (0, 1]$. Sei zudem $d \in \mathbb{N}$. Dann existiert eine Konstante $C = C(\beta, d) > 0$, sodass folgendes gilt:*

Für jede Funktion $f \in \mathcal{F}_{\beta,d,B}$ und jedes $x_0 \in (-\frac{1}{2}, \frac{1}{2})^d$ existiert ein Polynom $p(x) = \sum_{|\alpha|\leq n} c_\alpha (x - x_0)^\alpha$, wobei $c_\alpha \in [-C \cdot B, C \cdot B]$ für alle $\alpha \in \mathbb{N}^d_0$ mit $|\alpha| \leq n$, sodass

$$|f(x) - p(x)| \leq C \cdot B \cdot |x - x_0|^\beta \quad \text{für alle } x \in \left[-\frac{1}{2}, \frac{1}{2}\right]^d.$$

Das Polynom $p = p_{f,x_0}$ ist das Taylorpolynom von f vom Grad n.

Beweis. Sei zunächst $n = 0$, sodass $\beta = \sigma \leq 1$. Wir wählen dann $C = 1$ und $p(x) := f(x_0)$. Mit der Definition von $\mathcal{F}_{\beta,d,B}$ ist $|f(x_0)| \leq \|f\|_{C^{0,\beta}} \leq B$, also $f(x_0) \in [-B, B]$, und es gilt

$$|f(x) - p(x)| = |f(x) - f(x_0)| \leq \mathrm{Lip}_\sigma(f) \cdot |x - y|^\sigma \leq B \cdot |x - x_0|^\beta,$$

wie gewünscht. Wir können also $n \in \mathbb{N}$ annehmen.

Das Taylorpolynom von Grad $n-1$ von f am Stützpunkt x_0 ist gegeben durch

$$p_0(x) := f(x_0) + \sum_{|\alpha|=1}^{n-1} \frac{1}{\alpha!}(\partial^\alpha f)(x_0) \cdot (x - x_0)^\alpha \tag{3.30}$$

gegeben. Wir argumentieren kurz, inwiefern diese Notation äquivalent zu der im Satz von Taylor mit Integralrestglied in [6, Theorem C.15] ist und fahren in diesem Beweis dann mit letzterer fort. Hierfür sei $\underline{d} := \{1, \dots, d\}$. In [6] ist für $I = (i_1, \dots, i_m) \in \underline{d}^m$ mit $m \in \mathbb{N}$ dann $\partial_I f := \partial_{i_1} \cdots \partial_{i_m} f$ und $y^I = y^{i_1} \cdots y^{i_m}$. Das Taylorpolynom von Grad $n-1$ ist dort dann gegeben durch

$$p_0(x) = f(x_0) + \sum_{m=1}^{n-1} \frac{1}{m!} \sum_{I \in \underline{d}^m} \partial_I f(x_0)(x - x_0)^I. \tag{3.31}$$

Zunächst bemerken wir, dass die Reihenfolge der Differentiation keine Rolle spielt, da $f \in C^n([-\frac{1}{2}, \frac{1}{2}]^d)$. Für ein festes α gibt es in $\frac{1}{m!}\sum_{I \in \underline{d}^m} \partial_I f(x_o)(x - x_0)^I$ in 3.31 mithilfe des Multinomialkoeffizienten also genau $\binom{|\alpha|!}{\alpha} = \binom{m!}{\alpha}$ Summanden mit $\partial_I f(x_0)(x - x_0)^I = \partial^\alpha f(x_0)(x - x_0)^\alpha$. Mit dem gegebenen Vorfaktor lassen sich diese Terme zusammenfassen zu

$$\frac{1}{m!}\binom{m!}{\alpha}\partial^\alpha f(x_0)(x - x_0)^\alpha = \frac{1}{m!}\frac{m!}{\alpha!}\partial^\alpha f(x_0)(x - x_0)^\alpha = \frac{1}{\alpha!}\partial^\alpha f(x_0)(x - x_0)^\alpha.$$

Unsere Notationen in 3.30 und 3.31 sind also äquivalent.

Wir wenden nun den Satz von Taylor mit Integralrestglied aus [6] an und erhalten für $x \in (-\frac{1}{2}, \frac{1}{2})^d$:

$$\begin{aligned}
f(x) - p_0(x) =& \frac{1}{(n-1)!} \sum_{I \in \underline{d}^n} (x - x_0)^I \int_0^1 (1-t)^{n-1} \partial_I f(x_0 + t(x - x_0)) \mathrm{d}t \\
=& \frac{1}{(n-1)!} \Bigg(\sum_{I \in \underline{d}^n} (x - x_0)^I \int_0^1 (1-t)^{n-1} \partial_I f(x_0) \mathrm{d}t \\
&+ \sum_{I \in \underline{d}^n} (x - x_0)^I \int_0^1 (1-t)^{n-1} [\partial_I f(x_0 + t(x - x_0)) - \partial_I f(x_0)] \mathrm{d}t \Bigg) \\
=& \frac{1}{n!} \sum_{I \in \underline{d}^n} \partial_I f(x_0) \cdot (x - x_0)^I \\
&+ \frac{1}{(n-1)!} \sum_{I \in \underline{d}^n} (x - x_0)^I \int_0^1 (1-t)^{n-1} [\partial_I f(x_0 + t(x - x_0)) - \partial_I f(x_0)] \mathrm{d}t \\
=& : q(x) + R(x).
\end{aligned}$$

Dabei haben wir im zweiten Schritt eine nahrhafte Null eingefügt und im dritten Schritt verwendet, dass $\int_0^1 (1-t)^{n-1} \mathrm{d}t = \frac{1}{n}$. Nun ist $p := p_0 + q$ gerade das Taylor-Polynom von f von Grad n am Stützpunkt x_0. Wir schreiben in unserer üblichen Notation

$$p(x) = \sum_{|\alpha| \leq n} c_\alpha (x - x_0)^\alpha$$

mit Koeffizienten $c_\alpha \in \mathbb{R}$ und bemerken, mit $C = C(n,d) = C(\beta,d) := d^n$, dass

$$|c_\alpha| \leq \sum_{I \in \underline{d}^{|\alpha|} \text{ mit } \alpha = e_{i_1} + \ldots + e_{i_{|\alpha|}}} \frac{1}{|\alpha|!} \cdot |\partial_I f(x_0)| \leq d^{|\alpha|} \cdot B \leq d^n \cdot B = C \cdot B,$$

wobei $(e_1, \ldots, e_d)$ die Standardbasis von $\mathbb{R}^d$ ist, und verifizieren damit $c_\alpha \in [-C \cdot B, C \cdot B]$ für alle $\alpha \in \mathbb{N}_0^d$. Wir haben dabei verwendet, dass $|\underline{d}^{|\alpha|}| = d^{|\alpha|}$.

Da $\partial_I f$ für $I \in \underline{d}^n$ der Definition von $\mathcal{F}_{\beta,d,B}$ nach σ-hölderstetig ist mit $\mathrm{Lip}_\sigma(\partial_I f) \leq \|f\|_{C^{0,\beta}} \leq B$, gilt nun

$$\begin{aligned}
|f(x) - p(x)| &= |R(x)| \\
&= \left| \frac{1}{(n-1)!} \cdot \sum_{I \in \underline{d}^n} (x - x_0)^I \cdot \int_0^1 (1-t)^{n-1} \cdot [\partial_I f(x_0 + t(x - x_0)) - \partial_I f(x_0)] \mathrm{d}t \right| \\
&\leq \frac{1}{(n-1)!} \cdot \sum_{I \in \underline{d}^n} |(x - x_0)^I| \cdot \int_0^1 (1-t)^{n-1} \cdot B \cdot |t(x - x_0)|^\sigma \mathrm{d}t \\
&\leq \frac{B}{(n-1)!} \cdot \sum_{I \in \underline{d}^n} |x - x_0|^n \cdot |x - x_0|^\sigma \int_0^1 (1-t)^{n-1} t^\sigma \mathrm{d}t \\
&\leq \frac{B}{(n-1)!} \cdot d^n \cdot |x - x_0|^\beta \leq C \cdot B \cdot |x - x_0|^\beta.
\end{aligned}$$

Dabei haben wir im dritten Schritt die Hölderstetigkeit, im vierten Schritt $|x_i - (x_0)_i| \leq |x - x_0|$ und im fünften Schritt $|\underline{d}^{|\alpha|}| = d^{|\alpha|}$ und $(1-t)^{n-1} t^\sigma \leq 1$ für $t \in (0,1)$ verwendet. Durch Stetigkeit gilt dieses Ergebnis für alle $x \in [-\frac{1}{2}, \frac{1}{2}]^d$ und nicht nur für $x \in (-\frac{1}{2}, \frac{1}{2})^d$). □

Zuletzt kommen wir zu einer der Hauptaussagen von Petersen und Voigtlaender in [7] und dem zweiten der beiden zentralen Ergebnisse dieser Arbeit. Das nachfolgende Theorem liefert uns ein erstaunliches Ergebnis zur Expressivität von ReLU Multilayer Perceptrons: In der Approximation unserer glatter Funktionen bezüglich der L^p Norm wächst deren benötigte Tiefe, um eine Approximationsgenauigkeit ε zu erreichen, nicht mit ε sondern lediglich mit der Glattheit β der approximierten Funktionen.

Satz 3.11. *Für alle $d \in \mathbb{N}$ und $\beta, B, p > 0$ existieren Konstanten $s = s(d, \beta, B, p) \in \mathbb{N}$, $c = c(d, \beta, B) > 0$ und $L = L(\beta, d) \in \mathbb{N}$, wobei*

$$L \leq 11 + (1 + \lceil \log_2 \beta \rceil) \cdot (11 + \tfrac{\beta}{d}),$$

sodass für jede Funktion $f \in \mathcal{F}_{\beta,d,B}$ und jedes $\varepsilon \in (0, \frac{1}{2})$ ein Multilayer Perceptron Φ_ε^f existiert mit:

- *Φ_ε^f hat höchstens L Schichten.*
- *Die Anzahl der nichtnull Gewichte von Φ_ε^f ist höchstens $c \cdot \varepsilon^{-\frac{d}{\beta}}$.*
- *Alle Gewichte von Φ_ε^f sind (s, ε)-quantisiert.*
- *Für die Realisierung von Φ_ε^f gilt:*

$$\|R_\varrho(\Phi_\varepsilon^f) - f\|_{L^p([-\frac{1}{2},\frac{1}{2}]^d)} < \varepsilon \quad \text{und}$$
$$\|R_\varrho(\Phi_\varepsilon^f)\|_{\sup} \leq \lceil B \rceil.$$

Beweis. Wir gehen hier grundsätzlich ähnlich wie im Beweis von Satz 3.3 vor: Wir approximieren eine Funktion $f \in \mathcal{F}_{\beta,d,B}$ zunächst durch lokale Taylor-Polynome und nähern diese anschließend mithilfe unserer "Abschneidung" sowie der Approximation von Polynomen an.

Sei also $f \in \mathcal{F}_{\beta,d,B}$. Wie im Beweis von Lemma 3.8 wählen wir $p_0 := \lceil p \rceil \in \mathbb{N}$. Mithilfe von 3.27 reicht es dann, Approximation bezüglich der L^{p_0} Norm zu betrachten. Sei wieder $\beta = n + \sigma$ mit $n \in \mathbb{N}$ und $\sigma \in (0, 1]$. Sei zudem $C = C(d, \beta) > 0$ wie in Lemma 3.10 und sei

$$N := \left\lceil \left(\frac{\varepsilon}{4CBd^\beta} \right)^{-\frac{1}{\beta}} \right\rceil \in \mathbb{N}.$$

Seien außerdem für $\lambda \in \{1, \dots, N\}^d$

$$I_\lambda := \prod_{i=1}^d \left[\frac{\lambda_i - 1}{N} - \frac{1}{2}, \frac{\lambda_i}{N} - \frac{1}{2} \right].$$

Diese Intervalle ergeben nun eine bis auf Nullmengen disjunkte Zerlegung der Definitionsmenge:

$$\left[-\frac{1}{2}, \frac{1}{2} \right]^d = \bigcup_{\lambda \in \{1,\dots,N\}^d} I_\lambda. \tag{3.32}$$

Außerdem ist für alle $x \in I_\lambda$

$$I_\lambda \subset \overline{B}_{1/n}^{\|\cdot\|_\infty}(x) \subset \overline{B}_{d/n}^{|\cdot|}(x), \tag{3.33}$$

wobei $|\cdot| = \|\cdot\|_2$ und $\overline{B}_a^{\|\cdot\|}(x)$ wie üblicherweise der abgeschlossene a-Ball bezüglich einer Norm $\|\cdot\|$ um x ist. Die erste Teilmengenrelation folgt dabei direkt aus der Definition und die zweite folgt aus $|y - z| \leq \sqrt{d} \cdot \max_{i=1,\dots,d}\{|y_i - z_i|\}$. Wir geben diesen "Gitterpunkten" λ nun eine beliebige Nummerierung $\{1, \dots, N\}^d = \{\lambda_1, \dots, \lambda_{N^d}\}$ und wählen für jedes $i \in \{1, \dots, N^d\}$ einen Punkt x_i im Inneren des Intervalls I_{λ_i}. Für alle $\alpha \in \mathbb{N}_0^d$ mit $|\alpha| \leq n$ und jeden dieser Punkte x_i setzen wir $c_{i,\alpha} := \frac{\partial^\alpha f(x_i)}{\alpha!}$ und bemerken, dass $|c_{i,\alpha}| \leq B$, da $f \in \mathcal{F}_{\beta,d,B}$.

Damit ist das Taylorpolynom p_i von Grad n von f am Stützpunkt x_i gegeben durch

$$p_i(x) := \sum_{|\alpha|\leq n} c_{i,\alpha}(x-x_0)^\alpha = \sum_{|\alpha|\leq n} \frac{\partial^\alpha f(x_i)}{\alpha!}(x-x_0)^\alpha.$$

Mithilfe von Lemma 3.10 sowie 3.32 und 3.33 ist damit

$$\sup_{i\in\{1,\dots,N^d\},\, x\in I_{\lambda_i}} |f(x)-p_i(x)| \leq CB\left(\frac{d}{N}\right)^\beta. \tag{3.34}$$

Insbesondere folgt hieraus für festes i mit der Dreiecksungleichung und $f\in\mathcal{F}_{\beta,d,B}$, dass für alle $x\in I_{\lambda_i}$

$$|p_i(x)| \leq |f(x)| + Cd^\beta B \leq \left\lceil (1+Cd^\beta)\cdot B\right\rceil =: B_1. \tag{3.35}$$

Nun wählen wir mit den Stützpunkten $(x_i)_{i\in\{1,\dots,N^d\}}$ und den Koeffizienten $(c_{i,\alpha})$ das MLP $\Phi^{\mathrm{p}}_{\varepsilon/4}$ wie in Lemma 3.7. Hierbei verwenden wir also $m=N^d$ und $\varepsilon'=\frac{\varepsilon}{4}$. Dank Lemma 3.7 hat $\Phi^{\mathrm{p}}_{\varepsilon/4}$ höchstens $L_1=L_1(d,\beta)$ Schichten, wobei $L_1\leq 1+(1+\lceil\log_2\beta\rceil)\cdot(10+\frac{\beta}{d})$, und höchstens $c_1\cdot(\varepsilon^{-d/\beta}+N^d)$ nichtnull Gewichte, die $(s_1,\frac{\varepsilon}{4})$-quantisiert und damit auch (s_1,ε)-quantisiert sind (siehe Bemerkung 2.12). Dabei werden $s_1=s_1(d,\beta,B)\in\mathbb{N}$ und $c_1=c_1(d,\beta,B)>0$ ebenfalls wie in Lemma 3.7 gewählt.

Nun wenden wir Lemma 3.4 mit $B'=B_1$ an und erhalten ein Multilayer Perceptron $\Psi^{\mathrm{p}}_{\varepsilon/4}$ mit

$$R_\varrho(\Psi^{\mathrm{p}}_{\varepsilon/4}) = (\tau_{B_1}\times\cdots\times\tau_{B_1})\circ R_\varrho(\Phi^{\mathrm{p}}_{\varepsilon/4}),$$

wobei $\tau_{B_1}:\mathbb{R}\to[-B_1,B_1]$ eine 1-Lipschitz Funktion ist und $\tau_{B_1}(x)=x$ für alle $x\in[-B_1,B_1]$. Dank Lemma 3.4 wissen wir außerdem, dass $\Psi^{\mathrm{p}}_{\varepsilon/4}$ für eine universelle Konstante $c_2>0$ höchstens $2c_1\cdot(\varepsilon^{-d/\beta}+N^d)+c_2\cdot N^d$ nichtnull Gewichte und damit für ein $c_3=c_3(d,\beta,B)>0$ höchstens $c_3\cdot(\varepsilon^{-d/\beta}+N^d)$ nichtnull Gewichte hat, welche für ein passendes $s_2=s_2(d,\beta,B)\in\mathbb{N}$ außerdem (s_2,ε)-quantisiert sind. Schließlich gilt mit Lemma 3.4 für die Tiefe von $\Psi^{\mathrm{p}}_{\varepsilon/4}$

$$L_2=L_2(d,\beta) := L(\Psi^{\mathrm{p}}_{\varepsilon/4}) \leq 2+L_1 \leq 3+(1+\lceil\log_2\beta\rceil)\cdot(10+\frac{\beta}{d}).$$

Mithilfe von Ungleichung 3.35 sowie den Eigenschaften von τ_{B_1} und $\Phi^{\mathrm{p}}_{\varepsilon/4}$ können wir nun die folgende Abschätzung für alle $i\in\{1,\dots,N^d\}$ und alle $x\in I_{\lambda_i}$ machen:

$$\begin{aligned}\left|[R_\varrho(\Psi^{\mathrm{p}}_{\varepsilon/4})(x)]_i - p_i(x)\right| &= \left|\tau_{B_1}([R_\varrho(\Phi^{\mathrm{p}}_{\varepsilon/4})(x)]_i) - \tau_{B_1}(p_i(x))\right| \\ &\leq \left|[R_\varrho(\Phi^{\mathrm{p}}_{\varepsilon/4})(x)]_i - p_i(x)\right| \leq \frac{\varepsilon}{4}.\end{aligned}$$

Dabei haben wir im ersten Schritt die Definition von $\Psi^{\mathrm{p}}_{\varepsilon/4}$, Ungleichung 3.35 sowie $\tau_{B_1}(x)=x$ für $|x|\leq B_1$, im zweiten Schritt die Lipschitzstetigkeit von τ_{B_1}, und im

dritten Schritt die Approximationseigenschaften von $\Phi^{\mathrm{p}}_{\varepsilon/4}$ dank Lemma 3.7 verwendet. Mit diesem Ergebnis sowie Ungleichung 3.34 und unserer Wahl von N erhalten wir dann

$$\begin{aligned}\left\| f - \sum_{i=1}^{N^d} \chi_{I_{\lambda_i}} \cdot [R_\varrho(\Psi^{\mathrm{p}}_{\varepsilon/4})(x)]_i \right\|_{L^\infty} &\leq \sup_{i\in\{1,\dots,N^d\},\, x\in I_{\lambda_i}} \left| f(x) - [R_\varrho(\Psi^{\mathrm{p}}_{\varepsilon/4})(x)]_i \right| \\ &= \sup_{i\in\{1,\dots,N^d\},\, x\in I_{\lambda_i}} \left| f(x) - p_i(x) + p_i(x) - [R_\varrho(\Psi^{\mathrm{p}}_{\varepsilon/4})(x)]_i \right| \\ &\leq \frac{\varepsilon}{4} + \sup_{i\in\{1,\dots,N^d\},\, x\in I_{\lambda_i}} |f(x) - p_i(x)| \leq \frac{\varepsilon}{4} + CB\left(\frac{d}{N}\right)^\beta \leq \frac{\varepsilon}{2}, \end{aligned} \tag{3.36}$$

wobei wir im dritten Schritt die Dreiecksungleichung und das vorige Ergebnis, im vierten Schritt Ungleichung 3.34 und im fünften Schritt die Definition von N verwendet haben.

Wir möchten nun ein Multilayer Perceptron Ψ_ε finden, welches korrekt quantisierte Gewichte, höchstens $L_3 \leq 9 + (1 + \lceil \log_2 \beta \rceil) \cdot (10 + \frac{\beta}{d})$ Schichten und höchstens $c_8 \cdot \varepsilon^{-d/\beta}$ nichtnull Gewichte hat und die folgende Ungleichung erfüllt:

$$\left\| R_\varrho(\Psi_\varepsilon) - \sum_{i=1}^{N^d} \chi_{I_{\lambda_i}} \cdot [R_\varrho(\Psi^{\mathrm{p}}_{\varepsilon/4})]_i \right\|_{L^{p_0}} \leq \frac{\varepsilon}{2}. \tag{3.37}$$

Mit den Abschätzungen 3.27 und 3.36, der Dreiecksungleichung und $\| \bullet \|_{L^{p_0}([-1/2,1/2]^d)} \leq \| \bullet \|_{L^\infty([-1/2,1/2]^d)}$ folgt damit dann auch die Approximationsaussage des Satzes.

Um ein solches Multilayer Perceptron zu erhalten, wenden wir Lemma 3.9 mit $\varepsilon' = \frac{\varepsilon}{2}$, $p' = p_0$, $m' = N^d$, $\Phi' = \Psi^{\mathrm{p}}_{\varepsilon/4}$ und den Intervallen $I_{\lambda_i}, i \in \{1, \dots, N^d\}$ an. Das hierdurch gewonnene Multilayer Perceptron Ψ_ε erfüllt nach Lemma 3.9 die Abschätzung 3.37. Mit $R_i := \chi_{I_{\lambda_i}} \cdot [R_\varrho(\Psi^{\mathrm{p}}_{\varepsilon/4})]_i$ ist also

$$\begin{aligned}\left\| f - R_\varrho(\Psi_\varepsilon) \right\|_{L^p} &\leq \left\| f - R_\varrho(\Psi_\varepsilon) \right\|_{L^{p_0}} \\ &\leq \left\| f - \sum_{i=1}^{N^d} R_i \right\|_{L^{p_0}} + \left\| \sum_{i=1}^{N^d} R_i - R_\varrho(\Psi_\varepsilon) \right\|_{L^{p_0}} \\ &\leq \left\| f - \sum_{i=1}^{N^d} R_i \right\|_{L^\infty} + \frac{\varepsilon}{2} \leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon. \end{aligned} \tag{3.38}$$

Zudem hat Ψ_ε höchstens $L_3 := L_3(\beta, d) = 6 + L(\Psi^{\mathrm{p}}_{\varepsilon/4}) \leq 9 + (1 + \lceil \log_2 \beta \rceil) \cdot (10 + \frac{\beta}{d})$ Schichten und höchstens

$$\begin{aligned} c_4 \cdot (N^d + L(\Psi^{\mathrm{p}}_{\varepsilon/4}) + M(\Psi^{\mathrm{p}}_{\varepsilon/4})) &\leq c_4 \cdot (N^d + L_2 + M(\Psi^{\mathrm{p}}_{\varepsilon/4})) \\ &\leq c_5 \cdot (N^d + c_3 \cdot (\varepsilon^{-d/\beta} + N^d)) \end{aligned} \tag{3.39}$$

Gewichte, wobei $c_4 = c_4(d) > 0$ nach Lemma 3.9 und $c_5 = c_5(d, \beta) > 0$, da $L_2 = L_2(d, \beta)$. Diese Gewichte sind außerdem $(\max\{s_2, s_0\}, \frac{\varepsilon/2}{N^d})$-quantisiert für ein $s_0 = s_0(d, B, p_0) = s_0(d, B, p) \in \mathbb{N}$. Da

$$N = \left\lceil \left(\frac{\varepsilon}{4CBd^\beta}\right)^{-\frac{1}{\beta}} \right\rceil \leq \varepsilon^{-\frac{1}{\beta}} \cdot \left(\frac{1}{4C(d,\beta)Bd^\beta}\right)^{-\frac{1}{\beta}} + 1 \leq \varepsilon^{-\frac{1}{\beta}} \cdot c_7$$

für ein passendes $c_7 = c_7(d, \beta, B) > 1$, wobei wir nutzen, dass $\varepsilon^{-\frac{1}{\beta}} \geq 1$, folgt mit 3.39 und $\varepsilon^{-a} \leq \varepsilon^{-b}$ für $0 < a \leq b$ (weil $0 < \varepsilon < \frac{1}{2}$), dass Ψ_ε höchstens

$$c_5 \cdot \left(\varepsilon^{-1/\beta} \cdot c_7 + c_3 \cdot \left(\varepsilon^{-d/\beta} + \varepsilon^{-1/\beta} \cdot c_7\right)\right) \leq c_8 \cdot \varepsilon^{-d/\beta},$$

nichtnull Gewichte hat für ein $c_8 = c_8(d, \beta, B) > 0$, wie gewünscht. Außerdem gilt damit

$$\frac{\varepsilon}{2N^d} \geq \frac{\varepsilon}{2\varepsilon^{-d/\beta} c_7^d} = \frac{\varepsilon^{1+d/\beta}}{2c_7^d}. \tag{3.40}$$

Wir nutzen Bemerkung 2.12: Da die Gewichte $(\max\{s_2, s_0\}, \frac{\varepsilon}{2N^d})$-quantisiert sind, sind sie mit dem ersten Teil der Bemerkung und der obigen Ungleichung 3.2 auch $(\max\{s_2, s_0\}, \frac{\varepsilon^{1+d/\beta}}{2c_7^d})$-quantisiert. Wählen wir im dritten Teil der Bemerkung nun $q = 1 + \frac{d}{\beta}$ und $C' = 2c_7^d \geq 1$, so folgt, dass die Gewichte für ein $s_3 = s_3(s_0, s_2, q, c_7) = s_3(d, \beta, B, p) \in \mathbb{N}$ auch (s_3, ε)-quantisiert sind.

Schließlich müssen wir das Multilayer Perceptron Ψ_ε nur noch so erweitern, dass wir ein Multilayer Perceptron Φ_ε^f erhalten, welches $\|R_\varrho(\Phi_\varepsilon^f)\|_{\sup} \leq \lceil B \rceil$ erfüllt. Hierfür wenden wir Lemma 3.4 an. Nach diesem Lemma hat das neue MLP dann höchstens L Schichten für

$$L := L(\beta, d) \leq L(\Psi_\varepsilon) + 2 \leq 11 + (1 + \lceil \log_2 \beta \rceil) \cdot (10 + \frac{\beta}{d}).$$

Es gilt außerdem

$$M(\Phi_\varepsilon^f) \leq 2M(\Psi_\varepsilon) + c' \leq 2c_8(d, \beta, B) \cdot \varepsilon^{-d/\beta} + c' \leq c \cdot \varepsilon^{-d/\beta}$$

für ein $c := c(d, \beta, B)$. Für $s := s(d, \beta, B, p) = \max\{\lceil \log_2(\lceil B \rceil) \rceil, s_3(d, \beta, B, p)\}$ hat Φ_ε^f zudem (s, ε)-quantisierte Gewichte. Das MLP erfüllt nach Lemma 3.4 die Beziehung $R_\varrho(\Phi_\varepsilon^f) = \tau_B \circ R_\varrho(\Psi_\varepsilon)$ und, da mit den Eigenschaften von τ_B und $\|f\|_{\sup} \leq B$ gilt, dass $f = \tau_B \circ f$, erhalten wir mit der 1-Lipschitzstetigkeit von τ_B und 3.38

$$\|R_\varrho(\Phi_\varepsilon^f) - f\|_{L^p([-1/2,1/2]^d)} \leq \|R_\varrho(\Psi_\varepsilon) - f\|_{L^p([-1/2,1/2]^d)} \leq \varepsilon,$$

und sind damit fertig. □

4 Numerische Experimente

Im Folgenden werden wir einige numerische Untersuchungen zur praktischen Anwendbarkeit der oben formulierten Hauptergebnisse dieser Arbeit durchführen. Hierfür wählen wir Funktionen aus den jeweils gegebenen Funktionenklassen und trainieren neuronale Feedforward-Netze mit verschiedenen Architekturen und damit verschiedenen Anzahlen an Schichten, Neuronen und nichtnull Gewichten, um diese Funktionen zu approximieren.

Für jede Architektur führen wir das Training fünf Mal durch, um unsere Ergebnisse weniger abhängig von der Initialisierung der Gewichte und dem Trainingsverlauf zu machen. Da wir uns hier mit der Expressivität von neuronalen Netzen befassen, verwenden wir anschließend den besten erreichten Fehler über alle fünf Experimente und über alle durchlaufenen Epochen in der entsprechend relevanten Norm. Wir trainieren über bis zu 250,000 Epochen bei einem Trainingsset bestehend aus 100,000 gleichverteilten Punkten $(x_i, f(x_i))$. Das Testset besteht aus 10,000 weiterer solcher Punkte. Um das Training insbesondere bei kleineren Netzen, deren Training schnell abgeschlossen ist, abzukürzen, verwenden wir ein Early-Stopping-Verfahren: Wenn die Verlustfunktion innerhalb von 10,000 Epochen nicht um mindestens 5% reduziert werden konnte, beenden wir das Training. Durch Testtrainings mit verschiedenen Hyperparametern haben wir herausgefunden, dass der Adam-Optimierer mit einer Lernrate von $\alpha = 0.1$ für unsere Zwecke ein gutes Ergebnis liefert. Wir haben uns für den MSE (Mean Squared Error) als Verlustfunktion entschieden, welcher für eine Problemstellung dieser Art als eine natürliche Wahl scheint. Diese ist insbesondere sinnvoll, weil wir in Unterkapitel 4.2 den L^2 Fehler zur Bewertung der Approximationen verwenden, wofür sich der MSE hervorragend eignet.

Diese Vorgehensweise hat natürlich in gewisser Weise ihre Limitierungen. Die Ergebnisse der vorangehenden theoretischen Arbeiten treffen Aussagen zur Expressivität von Netzwerken, ohne dabei das Training eines Netzes zu berücksichtigen. Sie stellen lediglich fest, dass ein Netz mit einer entsprechenden Parameterwahl eine bestimmte Funktion gut approximieren *kann*.

Im Allgemeinen werden wir jedoch durch das Training eines neuronalen Netzes kein globales Minimum der Verlustfunktion finden und somit auch nicht die optimale Darstellungsfähigkeit einer Netzwerkarchitektur vollständig ausschöpfen können. Zudem entspricht ein Minimum unserer Verlustfunktion im Allgemeinen nicht zwingend einem Minimum des L^p oder L^∞ Fehlers. Besonders bei tiefen Netzwerken, die bekanntlich schwerer zu trainieren sind, wird das Training oft früh stagnieren und keine relevanten Ergebnisse liefern. Solche Ergebnisse, bei denen eine Netzwerkarchitektur offensichtlich nicht aufgrund ihrer Darstellungsfähigkeit, sondern aufgrund eines schlecht verlaufenen Trainings eine besonders schlechte Approximation liefert, werden wir in den nachfolgenden Untersuchungen ignorieren.

Eine weitere Einschränkung dieses Vorgehens liegt in der Betrachtung der nichtnull Gewichte. In unseren theoretischen Ergebnissen treffen wir ausschließlich Aussagen zu den nichtnull Gewichten eines Netzwerks und gehen davon aus, dass alle Gewichte, die nicht "benötigt" werden, gleich null sind. In der Realität des Trainings eines neuronalen Netzwerks sind jedoch praktisch alle Gewichte eines Netzwerks nichtnull. In den nachfolgenden Untersuchungen werden wir zwar strikt zwischen allgemeinen Feedforward-Netzen und Multilayer Perceptrons unterscheiden, jedoch keine weiteren Einschränkungen an die Gewichte vornehmen und die Anzahl der nichtnull Gewichte mit der Gesamtanzahl der Gewichte eines Netzwerks gleichsetzen. Es stellt sich heraus, dass diese Annahme legitim ist; in jedem unserer trainierten Netze sind alle Parameter nichtnull. Für ein allgemeines Feedforward-Netz Φ gilt also $M(\Phi) = G(\Phi)$, und für ein Multilayer Perceptron Ψ gilt $M(\Psi) = G_{\text{MLP}}(\Psi)$. Besonders bei allgemeinen Feedforward-Netzen, bei denen die Anzahl der Gewichte mit steigender Tiefe rasant anwächst, kann dies zu relevanten Diskrepanzen zwischen den theoretischen Ergebnissen und unseren praktischen Befunden führen. Da die theoretischen Ergebnisse des vorherigen Kapitels keine spezifischen Aussagen zur "Wahl" oder Verteilung der nichtnull Gewichte machen, sehen wir jedoch keinen sinnvollen Weg, dieses Problem zu umgehen.

Zuletzt sei noch erwähnt, dass wir für unsere Experimente ausschließlich "rechteckige" Netzwerkarchitekturen betrachten – also Architekturen, bei denen die Anzahl der Neuronen in den inneren Schichten konstant ist:

$$K := N_1 = N_\ell \quad \text{für} \quad \ell = 2, \dots, L-1.$$

Mit unseren Einschränkungen ist eine Architektur $\mathcal{A}$ durch das Parametertupel (L, K) sowie den Netzwerktyp (Feedforward oder MLP) und der Eingabe- und Ausgabedimension eindeutig definiert. Da letztere aus dem Kontext hervorgehen werden, werden wir Netzwerkarchitekturen im Folgenden häufig durch dieses Tupel bezeichnen.

Eine Untersuchung anderer "Formen" von Netzwerkarchitekturen haben wir zwar in Betracht gezogen, jedoch schnell festgestellt, dass die enorme Wahlfreiheit und die Vielzahl an zu untersuchenden Architekturen uns daran gehindert hätten, sinnvolle Ergebnisse zu produzieren. Im Kontext der Sätze 3.3 und 3.11 wäre es zwar wünschenswert, alle möglichen Architekturen zu untersuchen, da beide Sätze lediglich Aussagen zur *Existenz* einer Architektur bzw. eines Netzes mit bestimmten Eigenschaften machen. Eine solch umfassende Untersuchung ist jedoch für uns praktisch nicht durchführbar.

4.1 L^∞ Approximation in einem Sobolev Raum

Zunächst beschäftigen wir uns mit Satz 3.3. Dieser besagt, dass für gegebene n, d, ε eine ReLU Netzwerkarchitektur existiert, welche jede Funktion $f \in F_{n,d}$ mit einem L^∞ Fehler von höchstens ε approximieren kann. Diese Architektur besitzt höchstens $c \cdot \ln(\frac{1}{\varepsilon})$ Schichten sowie höchstens $c \cdot \varepsilon^{-d/n} \cdot \ln(\frac{1}{\varepsilon})$ Gewichte und Neuronen.

Eine mögliche Herangehensweise, um diese Aussage zu untersuchen, wäre, eine Vielzahl an Funktionen $f_1, \dots, f_m \in F_{d,n}$ zu betrachten und verschiedene Netzwerkarchitekturen $\mathcal{A}_1, \dots, \mathcal{A}_k$ auf diese Funktionen zu trainieren, wobei diese Architekturen eine

Approximationsgenauigkeit $\varepsilon_j^{\mathcal{A}_i}$ erreichen. Anschließend könnte man $\max_{j\in\{1,\dots,m\}} \varepsilon_j^{\mathcal{A}_i}$ mit verschiedenen Charakteristika der Architekturen, wie der Anzahl der Schichten oder der Anzahl der Gewichte, vergleichen. Solch umfangreiche Untersuchungen zu dieser Aussage wären jedoch äußerst rechen- und damit zeitintensiv und würden die für diese Arbeit gegebenen Kapazitäten überschreiten.

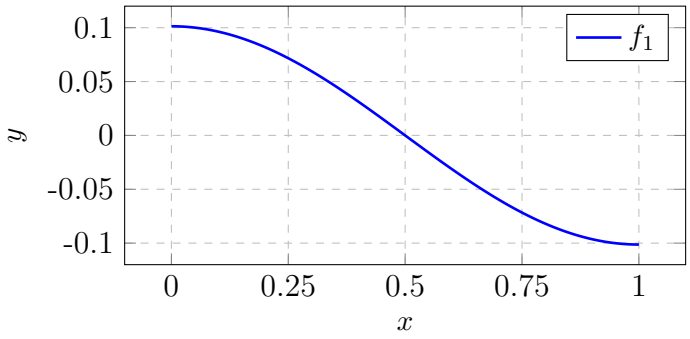

Abbildung 4.1: Funktion $f_1(x) = \frac{1}{\pi^2}\cos(\pi x)$.

Wir beschränken uns daher auf eine einzige Funktion f_1, die in gewisser Weise maximal in $F_{n,d}$ liegt. Sei $d = 1$, $n = 2$ und

$$f_1 : [0,1] \to \mathbb{R}, \quad f_1(x) := \frac{1}{2\pi^2} \cdot \cos(\pi x).$$

Dann ist $f_1 \in F_{2,1}$, aber $f_1 \notin F_{3,1}$, da $f_1''(x) = \cos(\pi x)$ und $f_1'''(x) = \pi \cdot \cos(\pi x)$, sodass $\|f_1\|_{\mathcal{W}^{2,\infty}} = 1$ und $\|f_1\|_{\mathcal{W}^{3,\infty}} = \pi > 1$.

Für verschiedene Netzwerkarchitekturen und Trainingsdurchläufe generieren wir jeweils 100,000 gleichverteilte x_i im Intervall $[0, 1]$ und trainieren diese mit den Trainingsdaten $(x_i, f_1(x_i))$. Die Wahl des MSE als Verlustfunktion optimiert zwar nicht direkt die L^∞ Norm, die wir hier als Maß für die Güte unserer Approximaton verwenden, ist aber trotzdem eine sinnvolle Wahl. So gewichtet der MSE größere Fehler stärker als kleinere Fehler und führt somit zu einer auch in der L^∞ Norm guten Approximation. Zudem führt der MSE zu einer glatten Optimierungslandschaft und eignet sich daher gut als Verlustfunktion. Die Netze, die wir hier trainieren, sind allgemeine Feedforward-Netzwerke, die also Verbindungen zwischen nicht-benachbarten Schichten zulassen. Für unsere Untersuchungen trainieren wir hauptsächlich mit Netzwerkarchitekturen (L, K), deren Breite $K := \max_{1\le j\le L-1} N_j = N_1$ nicht größer als 1,024 ist und deren Anzahl an Gewichten nicht größer als 4,000 ist. Dabei beginnen wir mit einer Breite von $K = 4$ und erhöhen diese mit einer Schrittweite ungefähr proportional zur Breite selbst, also $K \in \{4, 6, 8, 12, 16, \dots\}$. Für die Tiefe betrachten wir $L \in \{2, \dots, 7\}$, da sich das Training für tiefere Architekturen als extrem ineffizient herausstellt.

Dabei sei angemerkt, dass eine hohe Anzahl an Gewichten bereits für relativ kleine und "kurze" Feedforward-Netzwerkarchitekturen erreicht ist. Für die Anzahl der Gewichte

solcher Architekturen $\mathcal{A}_{L,K}$ mit konstanter Breite K und Tiefe L gilt nämlich:

$$G(\mathcal{A}_{L,K}) \geq \sum_{i=1}^{L-2} i \cdot K^2 = K^2 \sum_{i=1}^{L-2} i = K^2 \cdot \frac{(L-2)(L-1)}{2} \geq \frac{K^2(L^2-3L)}{2}, \tag{4.1}$$

wobei wir hier nur die Verbindungen zwischen den inneren Schichten abgeschätzt haben. Biases und Verbindungen mit den Eingabe- und Ausgabeschichten wurden dabei vernachlässigt. Dies bedeutet beispielsweise, dass eine Feedforward-Architektur mit Tiefe $L = 10$ und Breite $K = 25$, also mit 225 Neuronen in den inneren Schichten, bereits weit über 21,875 Gewichte hat; ein MLP dieser Größe hat nur ungefähr 5,000 Gewichte. Aus diesem Grund wird sich unsere Analyse der Tiefe solcher Architekturen auf ein Minimum beschränken.

4.1.1 Einfluss der Anzahl an Gewichten und Neuronen

Da uns interessiert, welche die "kleinste" Netzwerkarchitektur ist, welche die Funktion f_1 mit einer gewissen Genauigkeit ε approximieren kann, betrachten wir in Abbildung 4.2 (a) jeweils die Anzahl der Gewichte des Netzes mit der geringsten Anzahl an Gewichten, das einen Fehler von ε oder besser in der L^∞ Norm erreicht hat. Für unsere trainierten Netzwerke $\Phi_1, \ldots, \Phi_N$ ist also

$$M_{\min}(\varepsilon) := \min\{G(\Phi_i) : 1 \leq i \leq N \text{ und } \|R_\varrho(\Phi_i) - f_1\|_{L^\infty([0,1]^d)} \leq \varepsilon\}.$$

Ebenfalls abgebildet sind die Funktionen $c_{\text{Schranke}} \cdot \varepsilon^{-1/2} \cdot \ln(1/\varepsilon) =: g_S(\varepsilon)$ und $c_{\text{MSE}} \cdot \varepsilon^{-1/2} \cdot \ln(1/\varepsilon) =: g_M(\varepsilon)$. Der Parameter $c_{\text{Schranke}} = 2.540$ ist dabei im Sinne von Satz 3.3 so gewählt, dass g_S eine (minimale) obere Schranke für die benötigten Gewichte darstellt. Der Parameter $c_{\text{MSE}} = 1.285$ wurde so gewählt, dass der MSE zwischen $M_{\min}$ und g_M minimiert ist, wodurch wir das Verhalten der beiden Funktionen abseits von einzelnen Ausreißern besser vergleichen können.

An diesen Daten lässt sich erkennen, dass sich die benötigte Anzahl der Gewichte tatsächlich ähnlich der in Satz 3.3 aufgestellten oberen Schranke verhält. Dieses ähnliche Verhalten zeigt sich vor allem bei schlechteren Fehlerwerten, ab einem Wert von ungefähr $\varepsilon = 1.5 \cdot 10^{-3}$. Um bessere Approximationen zu erreichen, scheinen wir jedoch verhältnismäßig deutlich mehr Gewichte zu benötigen, weswegen $M_{\min}$ für abnehmendes ε schließlich deutlich steiler ansteigt als g_S. Für diese Diskrepanz gibt es mehrere mögliche Erklärungen:

- Die Architektur größerer Netze wird im Sinne der Konstruktionen in den Beweisen unserer theoretischen Ergebnisse immer "suboptimaler". In diesen haben wir häufig sorgfältig ausgewählt, welche Gewichte wir benötigen und welche der vielen Gewichte wir auf null setzen. Da in der Praxis jedoch alle Gewichte nichtnull sind und die Anzahl der Gewichte in allgemeinen Feedforward-Netzen extrem schnell wächst, werden auch viele weniger einflussreiche Gewichte hinzugefügt.

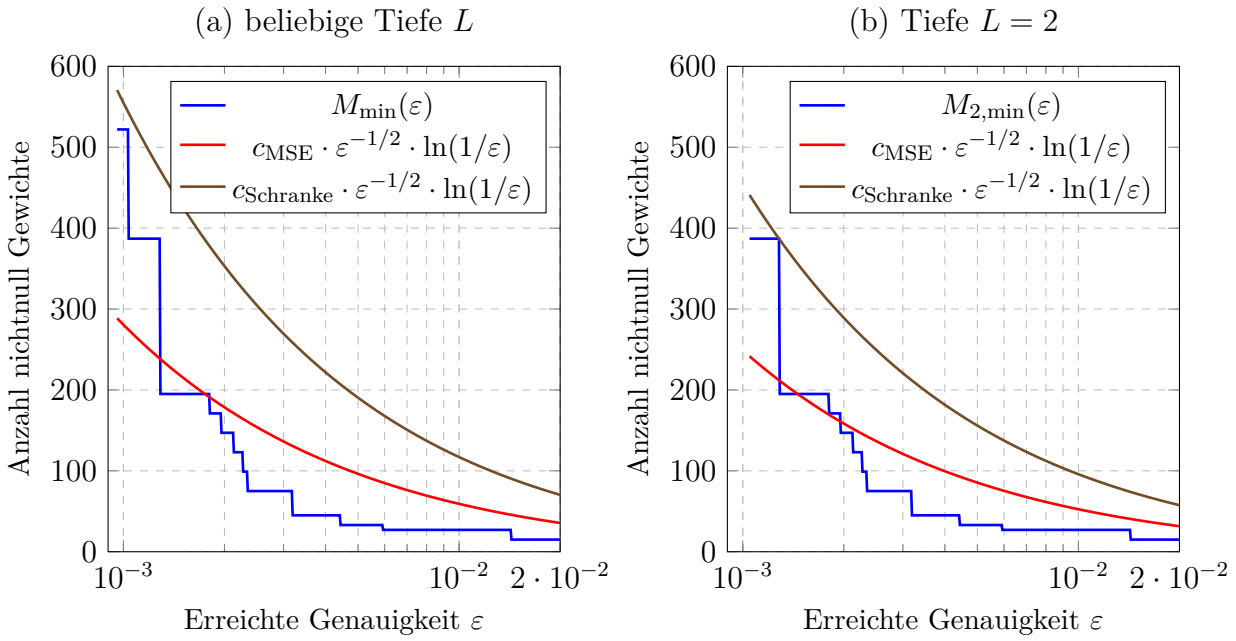

Abbildung 4.2: Vergleich der minimalen Anzahl der nichtnull Gewichte für eine Genauigkeit ε für (a) unbeschränkte Tiefe und (b) flache Netzwerke.

- Eine bessere Approximationsgenauigkeit erfordert offensichtlich ein Netzwerk mit einer höheren Anzahl an Gewichten. Mit zunehmender Anzahl der Gewichte gestaltet sich das Training jedoch schwieriger. Das Training wird, vor allem für tiefere Netze, empfindlicher gegenüber der Initialisierung der Gewichte, es wird anfällig für verschwindende und explodierende Gradienten, und es müssen schlichtweg mehr Parameter optimiert werden, was den Trainingsprozess erschwert. In allen Experimenten - sowohl in diesem Unterkapitel als auch im nachfolgenden - waren die Diskrepanzen in den erreichten Genauigkeiten zwischen den fünf trainierten Netzen pro Architektur für größere Architekturen deutlich höher als für kleinere Architekturen. Dies untermauert die obige These.

- Wir haben weniger große als kleine Netze trainiert. Der Grund dafür ist, dass die Verwendung von ausschließlich "rechteckigen" Netzen zusammen mit dem quadratischen Anstieg der Anzahl der Gewichte in Abhängigkeit von L und K (siehe 4.1) dazu führt, dass die Anzahl der Gewichte mit dem Anstieg dieser beiden Hyperparameter immer schneller wächst. Da das Training von neuronalen Netzwerke stark vom Zufall abhängt, ist es klar, dass die geringere Stichprobengröße für größere Netze auch zu schlechteren Ergebnissen für größere Netzwerke führt.

Des Weiteren sei angemerkt, dass unsere Ergebnisse auch abgesehen von den oben angesprochenen Punkten nicht den Aussagen von Satz 3.3 widersprechen. Dieser gibt schließlich nur eine obere Schranke und macht keine Aussagen darüber, wie sich die Anzahl der benötigten Gewichte für ein endliches Intervall $\varepsilon \in [a, b]$ zu verhalten hat.

Zuletzt bemerken wir, dass nach Abbildung 4.2 (a) das Netz, das den besten Fehler erreicht hat, lediglich ungefähr 520 Gewichte hat. Dies mag zunächst überraschend erscheinen, da wir Netzwerkarchitekturen mit bis zu 4,000 Gewichten trainiert haben, lässt sich jedoch ebenfalls durch die oben genannten Punkte zum Training großer Netzwerke erklären.

Vergleichen wir die Ergebnisse aus Abbildung 4.2 (a) nun mit denen aus Abbildung 4.2 (b), in der wir eine äquivalente Betrachtung durchgeführt haben, uns jedoch auf flache Netzwerke, also Netzwerke der Tiefe $L = 2$, beschränkt haben, so erkennen wir, dass flache Netzwerke fast immer bessere Ergebnisse erzielen als tiefere Netzwerke. Diese scheinbare Überlegenheit flacher Netzwerke werden wir in Abschnitt 4.1.2 noch einmal genauer untersuchen.

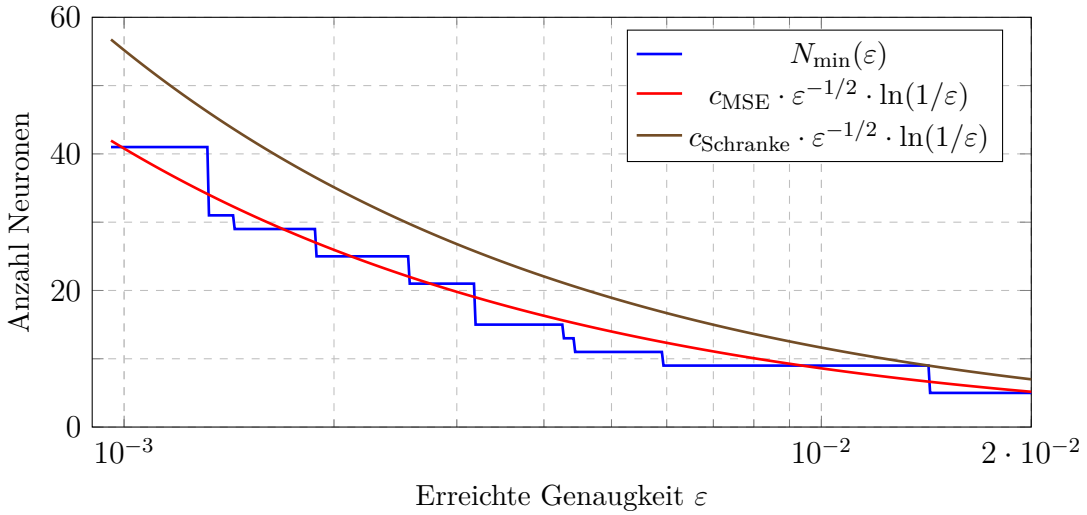

Abbildung 4.3: Minimale Anzahl der Neuronen für eine erreichte Approximationsgenauigkeit ε.

In Abbildung 4.3 führen wir ebenfalls eine analoge Untersuchung an, bilden hier jedoch die minimale Anzahl an Neuronen für eine erreichte Approximationsgenauigkeit ab:

$$N_{\min}(\varepsilon) := \min\{N(\Phi_i) : 1 \leq i \leq N \text{ und } \|R_\varrho(\Phi_i) - f_1\|_{L^\infty([0,1]^d)} \leq \varepsilon\}.$$

Interessant ist hier, dass sich $N_{\min}$ quasi exakt so verhält, wie es Satz 3.3 vermuten lässt - deutlich besser als dies für die Anzahl der Gewichte der Fall war. Ein Grund hierfür ist vermutlich, dass die Anzahl der benötigten Neuronen nicht, wie die Anzahl der nichtnull Gewichte, negativ davon beeinflusst wird, dass alle Gewichte nichtnull sind. Im Gegenteil sogar: In den Beweisen unserer theoretischen Ergebnisse war für ein Feedforward-Netz das Verhältnis zwischen Neuronen und nicht-null Gewichten, $\frac{N(\Phi)}{M(\Phi)}$, deutlich höher als bei unseren real trainierten Netzen, da wir hier, wie zuvor angesprochen, eine Vielzahl an Gewichten fest auf 0 gesetzt hatten. Dadurch haben diese real trainierten Netze im Vergleich zu den in den Beweisen konstruierten Netzen mehr Parameter bei gleicher Anzahl an Neuronen und damit eine höhere Expressivität. So hat die verhältnismäßig

kleine Architektur $(2, 20)$, welche den Bestwert aller Architekturen erzielen konnte, nur 42 Neuronen, aber 522 nichtnull Gewichte.

Es ist anzunehmen, dass dieses Verhältnis von Neuronen zu nichtnull Gewichten und die oben angesprochenen Effekte des Trainings für größere Netzwerke hier die Balance halten und damit für das "gute Verhalten" von $N_{\min}$ sorgen. Des Weiteren legt dieses Ergebnis nahe, dass unsere Vermutung in der Analyse von Abbildung 4.2 (a), dass die suboptimalen Architekturen einen Einfluss auf die Abweichungen bei $M_{\min}$ haben, tatsächlich ein relevanter Faktor ist.

4.1.2 Einfluss der Tiefe

Nun untersuchen wir den Einfluss der Tiefe unserer verschiedenen verwendeten Netzwerkarchitekturen auf die erreichte Genauigkeit der Approximation. In Abbildung 4.4 ist für verschiedene Spannen der Anzahl der nichtnull Gewichten die erreichte Genauigkeit und die Anzahl der Schichten der verschiedenen Architekturen abgebildet. Die Unterteilung in verschiedene Gewichtsspannen sorgt dabei für eine bessere Vergleichbarkeit der Architekturen.

Was diese Abbildung zunächst erkennen lässt, ist, dass Architekturen mit einer Tiefe von $L \geq 5$ scheinbar grundsätzlich schlechtere Ergebnisse als flachere Architekturen erzielen. Dies hängt sehr wahrscheinlich auch mit dem zuvor angesprochenen schlechter verlaufenden Training für tiefe Netzwerke zusammen. Gerade bei tiefen Netzwerken werden verschwindende und explodierende Gradienten sowie eine hohe Sensibilität gegenüber der Initialisierung sehr schnell zu Problemen. Zudem haben wir hier keine Optimierung der Parameterinitialisierung vorgenommen, um dies zu vermeiden. Vermutlich spielen hier aber auch strukturelle Eigenschaften der Architekturen eine Rolle. So ist in Abbildung 4.4 (a) beispielsweise ein Netzwerk mit 7 Schichten zu sehen, welches weniger als 500 Gewichte hat. Dies ist das Netz $(7, 4)$, das nur Breite $K = 4$ hat. Eine solch geringe Breite beschränkt die Expressivität einer Architektur immens und stellt auch für viele weitere der tiefen Architekturen eine Einschränkung dar.

Interessant ist in dieser Abbildung auch, dass Netze der Tiefe $L = 3$ und $L = 4$ mit wenigen Ausnahmen sehr gute Ergebnisse erzielen. Diese scheinen eine Balance aus einem gut verlaufenden Training und der höheren Expressivität tieferer Netze zu finden. Hierbei wird auch noch einmal deutlich, dass flache Netzwerke nicht zwingend aufgrund von besseren Approximationseigenschaften bessere Resultate erzielt haben, sondern dass die zuvor angesprochene Überzahl an Experimenten mit flachen Architekturen eine entscheidende Rolle spielt. Zwar liefern flache Netze in Abbildung 4.4 (a) die beste Approximation, dies liegt aber wohl vor allem daran, dass deutlich mehr flache Netzwerke mit 0 bis 500 Gewichten trainiert wurden. In allen anderen Fällen ist die Anzahl der Netze mit verschiedenen Tiefen gleichmäßiger verteilt, und tatsächlich erzielen hier konsistent Netze der Tiefen 3 oder 4 die besten Ergebnisse. Hinzu kommt, dass für die Spanne von 0 bis 500 Gewichten sogar bei Netzen der Tiefe 3 oder 4 die Breite des Netzwerks, wie zuvor angesprochen, schnell zum "Bottleneck" wird. Die breiteste Architektur der Tiefe 3, die weniger als 500 Gewichte hat, ist $(3, 19)$; für Tiefe 4 ist es sogar nur $(4, 11)$.

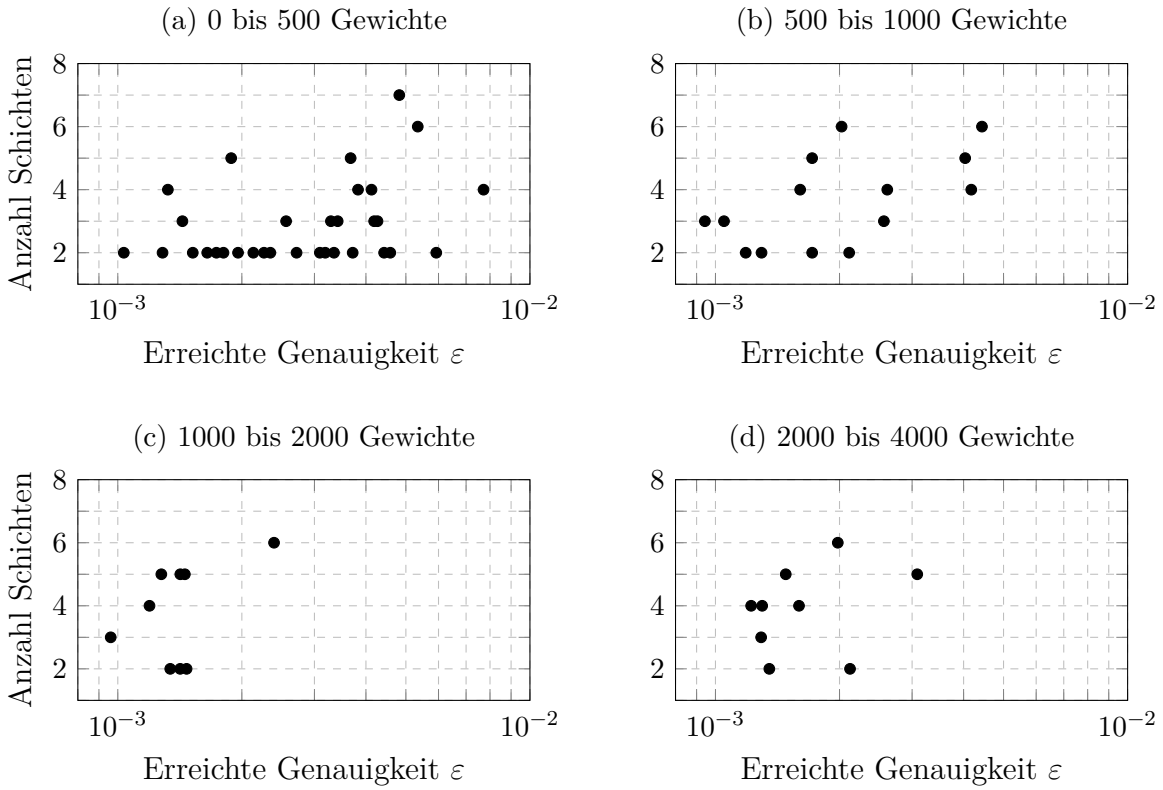

Abbildung 4.4: Anzahl Schichten $L(\Phi)$ vs. erreichte Genauigkeit ε für verschiedene Gewichtsspannen. Einige Architekturen haben sehr schlechte Genauigkeiten ($\gg 10^{-2}$) erreicht und werden für bessere Lesbarkeit der Abbildung nicht dargestellt.

4.2 L^p Approximation einer glatten Funktion

Nun stellen wir Untersuchungen zu Satz 3.11 an. Zunächst bemerken wir, dass wir die Quantisierung hier nicht weiter betrachten werden. Eine Implementierung der Quantisierung würde sich als sehr technisch herausstellen und das ohnehin schon zeitaufwändige Training nochmals drastisch verlängern. Hinzu kommt das Problem, dass erst nach dem Training klar ist, welchen Fehler eine gegebene Netzwerkarchitektur erreicht. Damit ist also auch erst nach dem Training ersichtlich, welche Quantisierungsstufe für das Netz hätte gewählt werden sollen.

Auch die Schranke für die Tiefe des Netzwerks können wir hier leider nicht weiter betrachten. Tiefe Netzwerke erfordern erheblichen Mehraufwand im Training, und da die in Satz 3.11 gegebene Schranke mindestens bei $L \leq 22$ liegt, sind uns solche Untersuchungen mit unseren Ressourcen leider nicht möglich; keines unserer Netze wird mehr als 22 Schichten umfassen.

Satz 3.11 besagt, dass für gegebene d, β, B, p eine Konstante c existiert, sodass für jede Funktion $f \in \mathcal{F}_{\beta,d,B}$ und jedes $\varepsilon \in (0, \frac{1}{2})$ ein Multilayer-Perceptron existiert, dessen L^p Fehler höchstens ε beträgt und dessen Realisierung durch $\lceil B \rceil$ beschränkt ist.

Wir wählen $d = 1$, $\beta = \frac{3}{2}$, $B = \frac{3}{2}$ und $p = 2$. Dann liegt die Funktion

$$f_2 : \left[-\frac{1}{2}, \frac{1}{2}\right] \to \mathbb{R}, \quad f_2(x) := |x|^{\frac{3}{2}}$$

maximal in $\mathcal{F}_{\beta,d,B}$. Dies verifizieren wir:

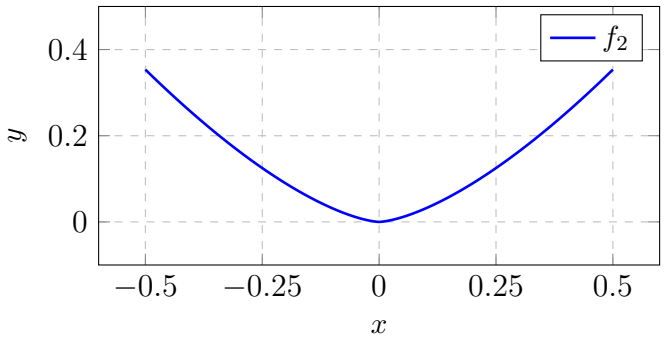

Abbildung 4.5: Funktion $f_2(x) = |x|^{\frac{3}{2}}$.

Die Funktion f_2 liegt in $C^1\left(\left[-\frac{1}{2}, \frac{1}{2}\right]\right)$. Für $x \in \left[-\frac{1}{2}, 0\right)$ und $x \in \left(0, \frac{1}{2}\right]$ ist die stetige Differenzierbarkeit offensichtlich, da f_2 hier als Verknüpfung stetig differenzierbarer Funktionen definiert ist. In $x_0 = 0$ gilt

$$\lim_{x \searrow 0} \frac{|x|^{3/2} - 0}{x - 0} = \lim_{x \searrow 0} x^{1/2} = 0 = \lim_{x \nearrow 0} (-x)^{1/2} = \lim_{x \nearrow 0} \frac{|x|^{3/2} - 0}{x - 0}$$

und somit ist f_2 auch in x_0 stetig differenzierbar, mit $f_2'(x) = \frac{3}{2} \cdot |x|^{\frac{1}{2}}$. Es ist leicht zu erkennen, dass $f_2 \notin C^2\left(\left[-\frac{1}{2}, \frac{1}{2}\right]\right)$.

Die Funktion f_2 wird im Betrag durch $|f_2\left(\frac{1}{2}\right)| = \frac{1}{2\sqrt{2}}$ maximiert, die Ableitung $|f_2'|$ durch $|f_2'\left(\frac{1}{2}\right)| = \frac{3}{2\sqrt{2}}$, und es gilt $\mathrm{Lip}_{\frac{1}{2}}(f_2') = \frac{3}{2}$. Wir zeigen,

$$\mathrm{Lip}_{\frac{1}{2}}(f_2') = \sup_{\substack{x,y \in \left[-\frac{1}{2}, \frac{1}{2}\right] \\ x \neq y}} \frac{\left|\frac{3}{2}|x|^{\frac{1}{2}} - \frac{3}{2}|y|^{\frac{1}{2}}\right|}{|x - y|^{\frac{1}{2}}} = \frac{3}{2} \cdot \sup_{\substack{x,y \in \left[-\frac{1}{2}, \frac{1}{2}\right] \\ x \neq y}} \frac{\left||x|^{\frac{1}{2}} - |y|^{\frac{1}{2}}\right|}{|x - y|^{\frac{1}{2}}} = \frac{3}{2}. \tag{4.2}$$

Sei $x \neq y$. Wir nehmen zunächst $x, y \leq 0$ an. Dann ist mit $a = (-x)^{\frac{1}{2}} \geq 0$ und $b = (-y)^{\frac{1}{2}} \geq 0$ der Zähler des obigen Ausdrucks gegeben durch $|a - b|$ und der Nenner durch $|x - y|^{\frac{1}{2}} = |(-x) - (-y)|^{\frac{1}{2}} = |a^2 - b^2|^{\frac{1}{2}}$. Der Ausdruck in (4.2) wird damit zu

$$\frac{|a - b|}{|a^2 - b^2|^{\frac{1}{2}}} = \frac{|a - b|}{|a - b|^{\frac{1}{2}} \cdot |a + b|^{\frac{1}{2}}} = \left(\frac{|a - b|}{|a + b|}\right)^{\frac{1}{2}} \leq 1,$$

wobei der letzte Schritt daraus folgt, dass a und b dasselbe Vorzeichen haben. Für den Fall $x, y \geq 0$ gehen wir analog vor. Nehmen wir nun an, dass x und y unterschiedliches

Vorzeichen haben; sei ohne Beschränkung der Allgemeinheit $x \geq 0$ und $y \leq 0$. Analog zu zuvor seien $a = x^{\frac{1}{2}} \geq 0$ und $b = (-y)^{\frac{1}{2}}$. Dann ist der Ausdruck in (4.2) gegeben durch $\frac{|a-b|}{|a^2+b^2|^{\frac{1}{2}}}$. Es gilt $\frac{|a-b|}{|a^2+b^2|^{\frac{1}{2}}} \leq 1$, da aufgrund von $a, b \geq 0$ gilt:

$$(|a-b|)^2 = a^2 - 2ab + b^2 \leq a^2 + b^2 = \left(|a^2+b^2|^{\frac{1}{2}}\right)^2.$$

Damit ist $\mathrm{Lip}_{\frac{1}{2}}(f_2') \leq \frac{3}{2}$. Um zu zeigen, dass $\mathrm{Lip}_{\frac{1}{2}}(f_2') = \frac{3}{2}$, wählen wir zum Beispiel $x = \frac{1}{2}$ und $y = 0$. Somit gilt $f_2 \in \mathcal{F}_{\beta,d,B}$.

Zuletzt merken wir an, dass f_2 neben B auch im Sinne von β maximal in $\mathcal{F}_{\beta,d,B}$ liegt, da für $\delta > 0$ gilt, dass $\mathrm{Lip}_{\frac{1}{2}+\delta}(f_2') = \infty$. Sei dafür $y = 0$ und $x > 0$. Dann ist

$$\frac{\left||x|^{\frac{1}{2}} - |y|^{\frac{1}{2}}\right|}{|x-y|^{\frac{1}{2}+\delta}} = \frac{x^{\frac{1}{2}}}{x^{\frac{1}{2}+\delta}} = x^{-\delta},$$

was unbeschränkt auf $\left[-\frac{1}{2}, \frac{1}{2}\right]$ ist.

Unsere Vorgehensweise ist analog zu der aus Unterkapitel 4.1: Wir generieren 100,000 Datenpunkte im Intervall $[-\frac{1}{2}, \frac{1}{2}]$ und trainieren darauf verschiedene Netzwerkarchitekturen. Es handelt sich hierbei ausschließlich um Multilayer Perceptron Architekturen, die also keine Verbindungen zwischen nicht-benachbarten Schichten zulassen. Als Verlustfunktion verwenden wir den MSE, der sich hervorragend für die Minimierung des L^2 Fehlers eignet. Wie zuvor trainieren wir hauptsächlich Architekturen mit nicht mehr als 4,000 Parametern und wählen die Breite K mit sich erhöhender Schrittweite, also $K \in \{4, 6, 8, 12, 16, \ldots\}$. Da die Anzahl der Gewichte für MLPs mit steigender Tiefe nicht so rasant wächst wie für allgemeine Feedforward-Netze, betrachten wir Tiefen $L \in \{2, \ldots, 20\}$. Wie sich jedoch herausstellen wird, erzielt keine Architektur mit einer Tiefe $L \geq 9$ sinnvolle Ergebnisse.

Zuletzt sei angemerkt, dass wir uns ausschließlich auf die Aussagen zur Anzahl der Gewichte in Abhängigkeit von der Approximationsgenauigkeit aus Satz 3.11 konzentrieren. Eine Untersuchung der Tiefe sowie der Quantisierung ist, wie bereits erwähnt, mit unseren Ressourcen nicht sinnvoll durchführbar. Die Behauptung $\|R_\varrho(\Phi_\varepsilon^f)\|_{\sup} \leq \lceil B \rceil$ ist für unsere Wahl von $B = \frac{3}{2}$ und damit $\lceil B \rceil = 2$ ebenfalls uninteressant. Da $\|f_2\|_{\sup} = \frac{1}{2\sqrt{2}}$ gilt, müsste ein Netz eine extrem schlechte Approximation liefern, um diese Bedingung nicht zu erfüllen, und wäre für unsere Betrachtung somit irrelevant.

4.2.1 Einfluss der Anzahl an Gewichten und Neuronen

Da Satz 3.11 ebenfalls nur eine Aussage zur Existenz eines Multilayer Perceptrons mit gewissen Eigenschaften macht, betrachten wir in Abbildung 4.6 wie zuvor die minimale Anzahl der Gewichte der MLPs, die einen L^2 Fehler ε unterschreiten konnten:

$$M_{\min}(\varepsilon) := \min\{G_{\mathrm{MLP}}(\Phi_i) : 1 \leq i \leq N \text{ und } \|R_\varrho(\Phi_i) - f_2\|_{L^2([-\frac{1}{2},\frac{1}{2}]^d)} \leq \varepsilon\}.$$

Dabei sind die $\{\Phi_1, \ldots, \Phi_N\}$ unsere trainierten MLPs, und $c_{MSE} \cdot \varepsilon^{-2/3}$ sowie $c_{\mathrm{Schranke}} \cdot \varepsilon^{-2/3}$ sind ebenfalls wie zuvor im Sinne der von Satz 3.11 gegebenen Schranke für die Anzahl der Gewichte gewählt.

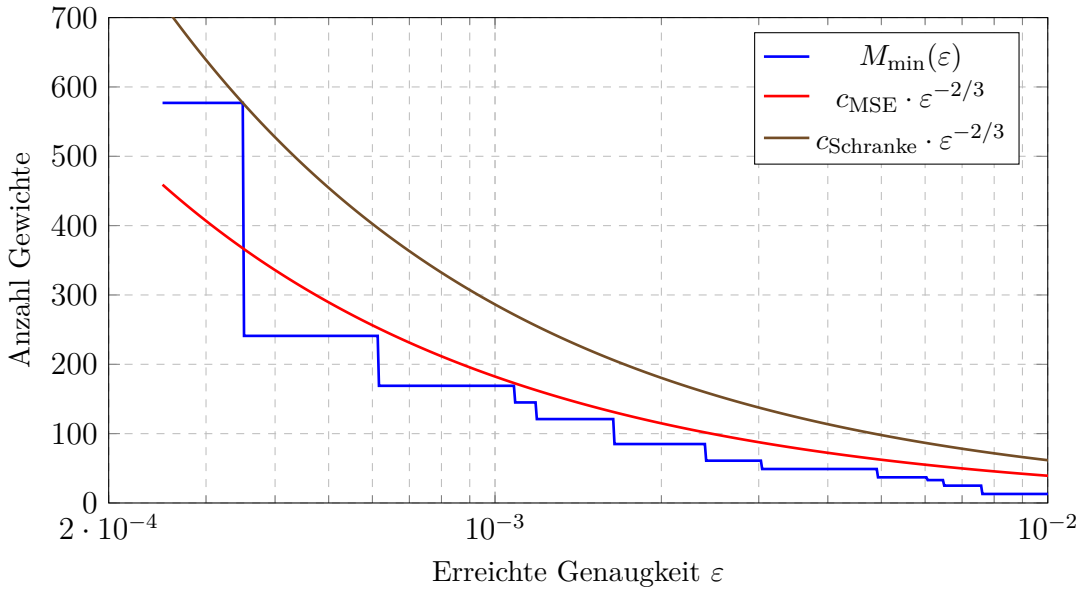

Abbildung 4.6: Minimale Anzahl der nichtnull Gewichte für eine Approximationsgenauigkeit ε.

Es zeichnet sich ein ähnliches Bild wie in Abbildung 4.2 (a) ab: Vor allem für größere ε verhält sich $M_{\min}$ so, wie Satz 3.11 vermuten lässt. Für kleinere ε steigt $M_{\min}$ jedoch deutlich an, wenn auch in einem geringeren Maße als in Abbildung 4.2 (a).

Die Gründe für dieses Verhalten sind vermutlich ähnlich den im vorigen Kapitel erwähnten: Wahrscheinlich spielen das schlechter verlaufende Training tieferer MLPs sowie die geringere Anzahl an trainierten Netzen für tiefere Architekturen eine entscheidende Rolle.

Dass die Abweichung von der im Theorem gegebenen Schranke hier geringer ausfällt, lässt sich damit begründen, dass die suboptimale Wahl der Architektur bei MLPs weniger stark ins Gewicht fällt als bei allgemeinen Feedforward-Netzen. Wie bereits diskutiert, werden in den Beweisen unserer theoretischen Ergebnisse die nichtnull Gewichte gezielt ausgewählt, während die real trainierten Netze viele nichtnull Gewichte enthalten, die einen vergleichsweise geringeren Einfluss auf die Expressivität haben. Bei MLPs ist die Anzahl solcher Gewichte jedoch aufgrund der geringeren Anzahl an möglichen Verbindungen deutlich kleiner, was die Diskrepanz zwischen den theoretisch konstruierten und den realen Netzen verringert. Insbesondere wächst die Anzahl dieser Gewichte bei MLPs nicht so rasant mit der Tiefe des Netzwerks, da die Anzahl der Gewichte eines MLPs nur linear und nicht quadratisch mit der Tiefe L skaliert.

Erneut sei hier jedoch angemerkt, dass Satz 3.11 lediglich eine obere Schranke für die Anzahl der Gewichte vorgibt und daher nicht spezifisch aussagt, dass sich die Anzahl der Gewichte der betrachteten endlichen Menge an Netzen zwingend wie $c \cdot \varepsilon^{-2/3}$ verhalten muss.

Ebenfalls bemerkenswert an diesen Ergebnissen ist, dass das Netz mit dem besten Fehler erneut weniger als 600 Gewichte besitzt, obwohl auch Netze mit mehr als 4,000

Parametern trainiert wurden. Ein naheliegender Gedanke wäre, dass Overfitting die Ursache sein könnte – dies lässt sich jedoch ausschließen. Alle Netze - sowohl in diesem Unterkapitel als auch in Unterkapitel 4.1 - erzielen auf den Trainings- und Testdaten nahezu identische Ergebnisse, was gegen Overfitting spricht. Stattdessen könnte das zuvor erwähnte suboptimal verlaufende Training für größere Netze eine entscheidende Rolle spielen. Besonders auffällig ist, dass bei vielen größeren Netzen kein Early Stopping ausgelöst wurde, was darauf hindeutet, dass das Training noch nicht vollständig abgeschlossen war. Erheblich längere Trainingszeiten wären jedoch mit unseren Ressourcen nicht realisierbar gewesen.

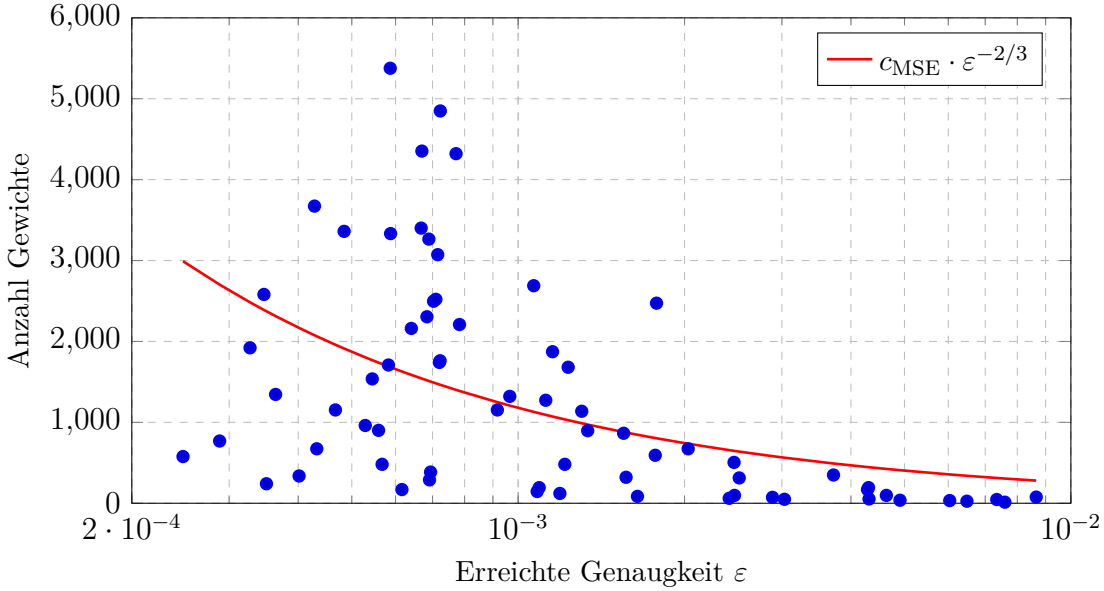

Abbildung 4.7: Erreichte Genuaigkeit vs. Anzahl der Gewichte für alle trainierten MLPs. Einige MLPs erreichten nur eine Genauigkeit $\gg 10^{-2}$ und sind aus Gründen der Lesbarkeit nicht abgebildet.

In Abbildung 4.7 sind die Ergebnisse aller trainierten MLPs abgebildet, mit Ausnahme der MLPs, die keinen relevanten Trainingsfortschritt erzielen konnten. Auffällig ist, dass alle Netze ein ähnliches Verhalten wie oben beschrieben aufweisen. Für größere Werte von ε zeigen die meisten Datenpunkte ein Verhalten, das gut mit $c \cdot \varepsilon^{-2/3}$ übereinstimmt. Für kleinere ε gibt es hingegen viele Ausreißer nach oben. Diese Ausreißer treten in diesem Fall jedoch etwas früher auf, was darauf zurückzuführen ist, dass das Training einiger großer Netze nicht optimal verlief.

In Abbildung 4.8 ist die gleiche Betrachtung für die Anzahl der Neuronen dargestellt. Zwar macht Satz 3.11 keine spezifischen Aussagen über diese, dennoch ist klar erkennbar, dass sie einer ähnlichen Schranke wie die Anzahl der Gewichte folgen.

Im Gegensatz zu Abbildung 4.3 in Abschnitt 4.1.1 passt N_{min} jedoch nicht so gut zur gegebenen Schranke. Ein möglicher Grund hierfür könnte erneut die Suboptimalität der Architekturen im Vergleich zu den in unseren Beweisen konstruierten Netzen sein. Während das Verhältnis der Neuronen zu den nichtnull Gewichten $\frac{N(\Phi)}{M(\Phi)}$ bei allgemeinen

Feedforward-Netzen zwischen den theoretisch konstruierten und den real trainierten Netzen stark abweicht und somit einen Ausgleich zu negativen Faktoren wie schlecht verlaufendem Training bildet, ist dieser Einfluss bei MLPs weniger ausgeprägt. Da bei MLPs weniger zusätzliche Gewichte eingeführt werden, wirkt sich dieser Effekt hier nicht so drastisch aus.

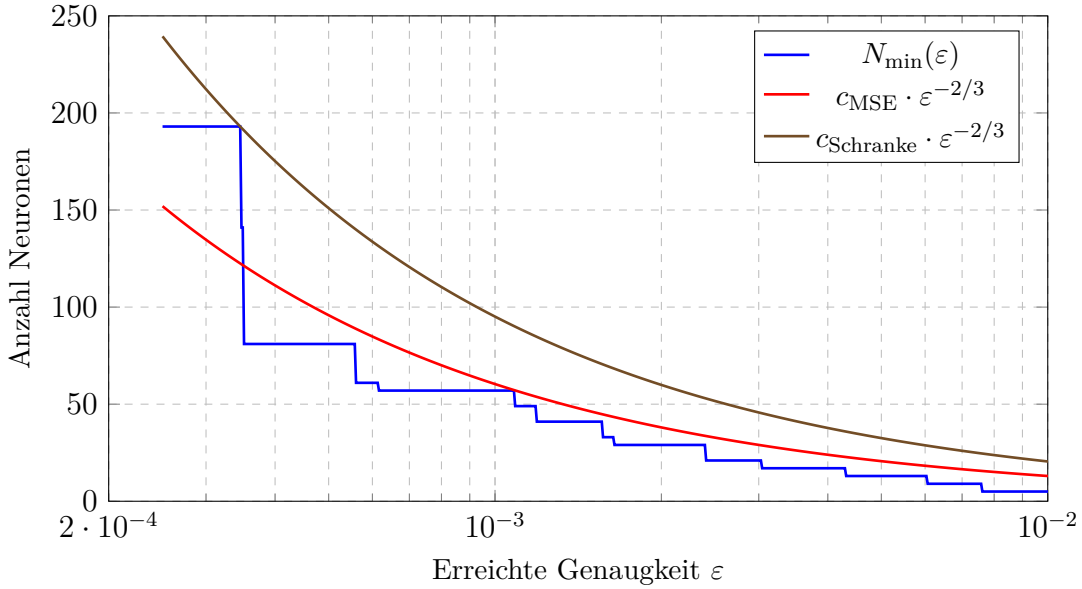

Abbildung 4.8: Minimale Anzahl der Neuronen für eine Approximationsgenauigkeit ε.

4.2.2 Einfluss der Tiefe

In Abbildung 4.9 betrachten wir erneut für verschiedene Spannen der Anzahl der Gewichte den erreichten Fehler im Vergleich zur Anzahl der Schichten der trainierten MLPs. Einige Ergebnisse wurden hier zur besseren Lesbarkeit ausgeschlossen. Dies betrifft MLPs, deren erreichter Fehler weit über 10^{-2} lag. Diese Netze sind hauptsächlich MLPs mit einer Tiefe von $L \geq 9$, die schlecht trainiert haben und bei denen, wie bereits in Abschnitt 4.1.2 angesprochen, die Breite des Netzwerks zum "Bottleneck" wurde.

Interessant ist hier, dass fast ausschließlich MLPs der Tiefe $L = 2$ die besten Ergebnisse erzielen. Im Gegensatz zu unseren Überlegungen in Abschnitt 4.1.2 scheint die höhere Anzahl an MLPs mit Tiefe 2 hierbei eine untergeordnete Rolle zu spielen. In den Gewichtsspannen von 500 bis 1.000 sowie von 1.000 bis 2.000 erreichen die MLPs mit Tiefe 2 bessere Genauigkeiten als alle anderen MLPs in diesen Gewichtsspannen. Für die Spanne von 0 bis 500 erzielen diese Netze ebenfalls die besten Ergebnisse, obwohl auch einige Netze schlechte Resultate liefern. Dies liegt vermutlich daran, dass hier Architekturen wie $(2, 4), (2, 6), \ldots$ implementiert wurden, die aufgrund ihrer sehr geringen Größe weniger leistungsfähig sind. Ein Grund für die guten Ergebnisse dieser flachen Netze könnte sein, dass die konvexe Funktion f_1 eine eher unkomplizierte Struktur aufweist

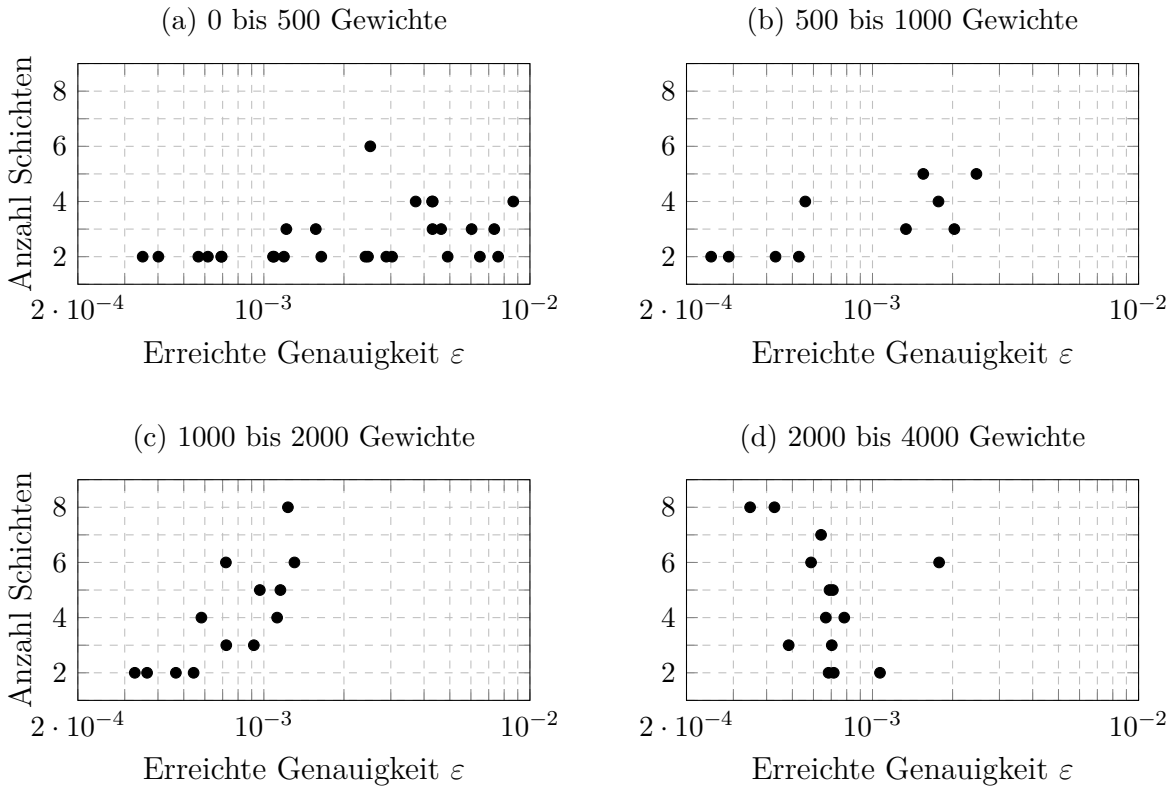

Abbildung 4.9: Anzahl Schichten $L(\Phi)$ vs. erreichte Genauigkeit ε für verschiedene Gewichtsspannen. Einige Architekturen haben sehr schlechte Genauigkeiten ($\gg 10^{-2}$) erreicht und werden für bessere Lesbarkeit der Abbildung nicht dargestellt.

und Tiefe hier möglicherweise keinen erheblichen Vorteil bietet; zumindest keinen Vorteil, der das erschwerte Training von tieferen Netzen hier aufwiegt.

Nur in Abbildung 4.9 (d) erzielen tiefere Netze bessere Ergebnisse – insbesondere zwei Netze der Tiefe 8. Dies zeigt, dass die Expressivität solcher tiefen Netze trotzdem hoch ist und untermauert die These, dass viele der Ergebnisse für tiefe Netze durch schlecht verlaufendes Training verfälscht sind.

4.3 Zusammenfassung der Ergebnisse

In diesem Kapitel haben wir numerische Experimente durchgeführt, um die praktische Anwendbarkeit der theoretischen Hauptergebnisse dieser Arbeit zu untersuchen. Dazu haben wir Funktionen aus den entsprechenden Funktionenklassen ausgewählt und neuronale Feedforward-Netze mit verschiedenen Architekturen hinsichtlich Tiefe, Neuronenanzahl und Anzahl der nichtnull Gewichte trainiert, um diese Funktionen zu approximieren.

Unsere Experimente haben gezeigt, dass die Anzahl der Gewichte und Neuronen, die für die Approximation mit Genauigkeit ε benötigt werden, tatsächlich ungefähr den

theoretischen Schranken entspricht, die in Satz 3.3 und Satz 3.11 angegeben sind. Für kleinere Werte von ε traten jedoch technische Probleme auf, die zu Abweichungen führten. Dennoch sollten die in diesen Sätzen formulierten oberen Schranken wirklich nur als solche verstanden werden: Da in den Beweisen dieser Sätze bei der Wahl der nichtnull Gewichte gezielt vorgegangen wird, ergeben sich in der Praxis Diskrepanzen zwischen den theoretischen Vorhersagen und den tatsächlich benötigten Ressourcen.

Wir haben festgestellt, dass flache Netzwerke oft bessere Ergebnisse erzielen als tiefere Netzwerke. Dies könnte auf die Herausforderungen beim Training tiefer Netzwerke zurückzuführen sein, wie zum Beispiel Probleme mit verschwindenden oder explodierenden Gradienten. Bei der Approximation beider untersuchter Funktionen konnten flache Netzwerke überraschend gute Ergebnisse erzielen. Dies könnte jedoch auch mit der relativ einfachen Struktur der von uns approximierten Funktionen sowie mit der begrenzten Größe und Dimensionalität unserer Experimente zusammenhängen.

Unsere Ergebnisse weisen darauf hin, dass praktische Einschränkungen beim Training neuronaler Netze, wie die Schwierigkeit, globale Minima der Verlustfunktion zu finden, einen erheblichen Einfluss auf die erzielten Approximationsergebnisse haben. Daher sollten solche Einschränkungen bei der Anwendung der theoretischen Ergebnisse in der Praxis berücksichtigt werden.

5 Fazit und Ausblick

In dieser Bachelorarbeit wurde die Beziehung zwischen Approximationsraten und Netzwerkkomplexität für neuronale Netzwerke mit ReLU Aktivierungsfunktionen untersucht. Ziel war es, theoretische Schranken für die Netzwerkkomplexität abzuleiten, um Funktionen verschiedener Klassen mit einer vorgegebenen Genauigkeit zu approximieren. Betrachtet wurden hierfür Funktionen aus Sobolev-Räumen mit Approximation in der L^∞ Norm sowie Funktionen mit bestimmten Glattheitseigenschaften mit Approximation in allgemeinen L^p Normen.

Die theoretischen Ergebnisse zeigen, dass die Approximationsfähigkeit von ReLU Netzwerken eng mit deren Tiefe sowie der Anzahl der Neuronen und der nichtnull Gewichte verknüpft ist. Es wurde nachgewiesen, dass:

- die benötigte Netzwerkkomplexität mit der gewünschten Approximationsgenauigkeit skaliert,
- für Funktionen mit höherer Regularität effizientere Approximationen möglich sind, was durch präzise Abhängigkeiten zwischen Regularität und Netzwerkkomplexität ausgedrückt wurde.

Durch numerische Experimente wurden die theoretischen Erkenntnisse validiert. Die Ergebnisse betonen die Bedeutung sorgfältig entworfener Netzwerkarchitekturen für die effiziente Approximation komplexer Funktionen.

Zusammenfassend leistet diese Arbeit einen Beitrag zur theoretischen Fundierung der Architekturwahl für neuronale Netzwerke in datenintensiven Anwendungsbereichen. Die formulierten Schranken bieten eine Grundlage für das Design von Netzwerken, die sowohl hohe Genauigkeit als auch geringe Ressourcenanforderungen vereinen.

Zukünftige Forschungen könnten darauf abzielen, die hier vorgestellten Konzepte auf andere Aktivierungsfunktionen oder Netzwerktopologien auszudehnen und deren praktische Anwendbarkeit in verschiedenen Bereichen wie Bildverarbeitung oder naturwissenschaftlicher Simulation weiter zu untersuchen.

Literaturverzeichnis

[1] A. R. Barron. “Universal approximation bounds for superpositions of a sigmoidal function”. In: *IEEE Transactions on Information Theory* 39.3 (1993), S. 930–945.

[2] G. Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of Control, Signals and Systems* 2.4 (1989), S. 303–314.

[3] B. Ding und H. Qian und J. Zhou. “Activation functions and their characteristics in deep neural networks”. In: *Proceedings of the Chinese Control and Decision Conference*. USA, 2018, S. 1836–1841.

[4] J. Elstrodt. *Maß- und Wahrscheinlichkeitstheorie*. 8th. Springer Spektrum, 2018.

[5] G. B. Folland. “Remainder Estimates in Taylor’s Theorem”. In: *The American Mathematical Monthly* 97.3 (1990), S. 233–235.

[6] J. M. Lee. *Introduction to Smooth Manifolds*. 2nd. Bd. 218. Graduate Texts in Mathematics. Springer, 2012.

[7] P. Petersen und F. Voigtlaender. “Optimal approximation of piecewise smooth functions using deep ReLU neural networks”. In: *Neural Networks* 108 (2018), S. 296–330.

[8] M. Telgarsky. “Representation benefits of deep feedforward networks”. In: *JMLR: Workshop and Conference Proceedings*. Bd. 49. 2016, S. 1–23.

[9] D. Yarotsky. “Error bounds for approximations with deep ReLU networks”. In: *Neural Networks* 94 (2017), S. 103–114.