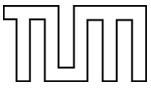

Technical University of Munich

Department of Mathematics

Master's Thesis

Mathematical Analysis of Neural Networks

Alina Leidinger

Supervisor: Prof. Dr. Massimo Fornasier

Advisor: Prof. Dr. Massimo Fornasier

Submission Date: June 15, 2019

I assure the single handed composition of this master's thesis only supported by declared resources.

Garching,

Abstract

Neural networks have gained major popularity in recent years and celebrated great successes in practice. However, they are notoriously opaque and their inner workings are not yet fully understood on a mathematical level. This is most apparent, when neural networks exhibit very surprising behaviour. In object classification settings for example, imperceptibly perturbed input images, so called adversarial examples, are often classified very differently than the original image. This sheds major doubt on the long-held belief that neural networks are able to attain some human level understanding of underlying concepts. This master's thesis examines the status quo of the mathematical analysis on neural networks in a literature review. The focus is on three key themes.

Firstly, classical and fundamental results in **approximation theory** are considered pertaining to density of neural networks within different function spaces under different assumptions on the activation function. Then, key results on the degree or order of approximation with neural networks are discussed. The curse of dimensionality is rigorously examined and we see how a deep network architecture facilitates breaking the curse in the case of compositional target functions.

Secondly, we concentrate on the theme of **stability** of neural networks. Mathematical analysis of adversarial examples is presented which highlight the instability of neural networks. The main focus of the section is then on scattering networks. Their stability properties under translation and deformation shed light on the recent success of Convolutional Neural Networks (CNNs).

Thirdly, methods for tractable learning of a target function given by a shallow neural network are discussed. We explore when **exact identification** of the network weights is possible using a number of samples which scales possibly only polynomially in the input dimension and network size. We compare two different approaches which are based on unique decomposition of a symmetric orthogonal third order tensor and matrix subspace optimisation respectively.

Abstract

Neuronale Netze erfreuen sich seit einigen Jahren großer Beliebtheit und erzielen in der Praxis beeindruckende Erfolge. Gleichzeitig sind sie bekannt dafür, als Modell vergleichsweise undurchsichtig zu sein. Aus einer mathematischen Perspektive sind sie bisher noch nicht vollständig verstanden. Dies zeigt sich deutlich, wenn neuronale Netze sich überraschend anders verhalten als erwartet. Bei der Klassifizierung von Objekten in Bildern, können unmerklich veränderte Bilder, "adversarial examples", zu einer falschen Klassifizierung führen. Das zieht wiederum den lange gehegten Glauben in Zweifel, dass neuronale Netze zugrundeliegende Konzepte begreifen können wie Menschen. Diese Masterarbeit präsentiert den Status Quo der mathematischen Analyse von neuronalen Netzen im Rahmen einer Literaturstudie. Der Fokus liegt hierbei auf drei Hauptthemen.

Zunächst werden einige fundamentale Resultate der **Approximationstheorie** veranschaulicht. Unter verschiedenen Annahmen über die Aktivierungsfunktion bildet die Klasse der neuronalen Netze eine dichte Teilmenge in unterschiedlichen Funktionsräumen. Weiterhin stellen wir grundlegende Resultate zum möglichen Näherungsgrad mit neuronalen Netzen vor. Wir analysieren den "Curse of Dimensionality" (Fluch der Dimensionalität) und sehen wie ebendieser mit Hilfe mehrlagiger neuronaler Netze umgangen werden kann, wenn die Zielfunktion als Komposition konstruiert ist.

Zweitens widmen wir uns der **Stabilität** von neuronalen Netzen. Wir präsentieren einen Teil der mathematischen Analyse zu "adversarial examples" (Gegnerische Beispiele), die die allgemeine Instabilität von neuronalen Netzen aufzeigen. Der Hauptfokus des Kapitels sind "scattering networks" (Streunetzwerke). Ihre Stabilitätseigenschaften bei Translation und Deformation bieten Erklärungen für den Erfolg von Convolutional Neural Networks (CNNs).

Drittens wird das Lernen einer Zielfunktion behandelt, die als neuronales Netz mit einer Lage gegeben ist. Wir analysieren, wann eine **exakte Identifikation** der Parameter möglich ist unter Verwendung einer Anzahl an Funktionswerten, die nur polynomiell skaliert in Bezug auf relevante Parameter wie die Inputdimension oder die Größe des Netzwerks. Wir vergleichen zwei Ansätze, die einmal auf der eindeutigen Dekomposition eines symmetrischen, orthogonalen Tensors und einmal auf der Optimierung innerhalb eines Matrixunterraums beruhen.

Notation & Preliminaries

General

If not otherwise stated, the input to a neural network is denoted by $x \in \mathbb{R}^d$. Let $B_d(R) = \{x \in \mathbb{R}^d : \|x\|_2 < R\}$ be the open l_2-ball of radius R in $\mathbb{R}^d$. For ease of notation denote by B_d the ball of radius one. Let $\mathcal{S}^{d-1}$ be the surface of the unit norm ball in $\mathbb{R}^d$. Let $\mu_{\mathcal{S}^{d-1}}$ be the uniform measure on the sphere $\mathcal{S}^{d-1}$. Let the boundary of a set K be ∂K. Let e_j be the jth element of the canonical basis in $\mathbb{R}^d$. For a complex number $z = re^{i\theta}$ with $r \geq 0$, $\angle z := \theta$ and is referred to as the *phase* of z. Denote the support of a function f by $\text{supp}(f)$. Let $\nabla_x f(x)$ or simply $\nabla f(x)$ be the gradient of the function f with respect to x. Let $\partial^\alpha f(x)$ be the partial derivative where the multi-index α determines with respect to which variables the derivative is taken. Let $\delta(x)$ be the Dirac function. Unless otherwise defined, denote by $\hat{f}$ the Fourier transform of function f. Let A^* be the adjoint operator of an operator A. For $N \in \mathbb{N}$ let $[N] = \{1, \cdots, N\}$. Abbreviate orthonormal system or orthonormal basis as ONS and ONB respectively and *almost everywhere* and *almost surely* as a.e. and a.s.

Vectors & Matrices

Let the inner product of $a, b \in \mathbb{R}^d$ be $\langle a, b \rangle = a \cdot b = a^T b$. Denote by $\otimes$ the vector outer product i.e. $a \otimes b = ab^T$ for column vectors a and b. Denote by $\text{vec}(A) \in \mathbb{R}^{m^2}$ the vectorisation of a matrix $A \in \mathbb{R}^{m \times m}$ defined by

$$\text{vec}(A)_k := a_{\lfloor \frac{k-1}{m} \rfloor + 1, (k-1 \bmod m)+1} \quad \text{for } k \in [m^2]. \tag{0.1}$$

Denote the singular value decomposition (SVD) of a matrix A by $A = USV^T$. Denote in general the jth largest singular value of A by s_j and collect the singular values in the diagonal matrix S in descending order. We also sometimes denote the largest and smallest singular values of A by $s_{\max(A)}$ and $s_{\min(A)}$ respectively. Let $\text{Sym}(\mathbb{R}^d)$ be the space of symmetric matrices in $\mathbb{R}^d \times \mathbb{R}^d$. Denote the p-(quasi)norm of $x \in \mathbb{R}^d$ for $0 < p < \infty$ by $\|x\|_p = \left(\sum_{j=1}^d |x_j|^p\right)^{1/p}$. This is a quasi-norm for $p \in (0,1)$ and a norm otherwise. The infinity norm of a vector is $\|x\|_\infty = \max_{j \in [d]} |x_j|$. Denote by $\|x\|_0$ the number of non-zero entries of x (note that this is not a true norm). The Schatten-p norm and the infinity norm of matrix $A \in \mathbb{R}^{m \times d}$ are the p-norm and the infinity norm applied to the vector of singular values of A. In particular, the Schatten-2 and the infinity norm are also referred to as the Frobenius norm and the spectral norm respectively, i.e. $\|A\|_F = \|A\|_2$ and $\|A\| = \|A\|_\infty$.

Background on Tensors used in (4.1.2)

Section (4.1.2) makes heavy use of tensors and tensor decomposition that is why we briefly introduce some basic notions pertaining to tensors. $\bigotimes^m \mathbb{R}^d$ is the m-fold outer product of $\mathbb{R}^d$. We say that $T \in \bigotimes^m \mathbb{R}^d$ is an mth order tensor which has m modes or dimensions. A tensor fibre is a vector obtained by fixing all but one index of a tensor eg. $T(i,j,:)$. The conversion of a tensor $T \in \mathbb{R}^{d\times d\times d}$ to a matrix $M \in \mathbb{R}^{d\times d^2}$ is defined as $T(i,j,l) = M(i, l+(j-1)d)$ for $i,j,l \in [d]$. A third order tensor T is of CP (CanDecomp/Parafac) *rank* 1 if

$$T = w \cdot a \otimes b \otimes c$$

for unit vectors $a,b,c, \in \mathbb{R}^d$ and $w \in \mathbb{R}$. Equivalently we can write $T(i,j,l) = w \cdot a(i)b(j)c(l)$. Third order tensor T has *rank* k if k is the smallest integer such that T can be written as the sum of k rank 1 tensors i.e.

$$T = \sum_{i\in[k]} w_i a_i \otimes b_i \otimes c_i$$

for unit vectors $a_i, b_i, c_i \in \mathbb{R}^d$ and $w_i \in \mathbb{R}$. The CP (CanDecomp/Parafac) decomposition of a tensor finds a sum of rank one components which approximates the input tensor. For matrices $M_i \in \mathbb{R}^{d\times d_i}, i \in [3]$ define the *multilinear form* $T(M_1, M_2, M_3) \in \mathbb{R}^{d_1\times d_2\times d_3}$ as

$$T(M_1, M_2, M_3)_{i_1,i_2,i_3} := \sum_{j_1,j_2,j_3\in[d]} T_{j_1,j_2,j_3} \cdot M_1(j_1,i_1) \cdot M_2(j_2,i_2) \cdot M_3(j_3,i_3). \tag{0.2}$$

This notation offers a succinct way of expressing linear combinations of tensor fibres as $T(Id, v, w) = \sum_{i,j\in[d]} v_i w_j T(:,i,j) \in \mathbb{R}^d$ for $v, w \in \mathbb{R}^d$.
The *Khatri-Rao product* of $A, B \in \mathbb{R}^{d\times k}$ is denoted by $A \odot B \in \mathbb{R}^{d^2\times k}$ and defined for $C := A \odot B$ by $C(l+(i-1)d, j) = A(i,j) \cdot B(l,j)$ for $i,l \in [d], j \in [k]$.

Function spaces

Let $L^p(X,\mu)$, $L^p(\mu)$ or simply L^p be the space of pth power integrable functions. L^p contains all measurable functions $f : X \to \mathbb{R}$ for $X \subset \mathbb{R}^d$ such that

$$\|f\|_p = \Big(\int_X |f(x)|^p d\mu(x) \Big)^{\frac{1}{p}} < \infty.$$

Let $L^p(K,\mu)$ be the same space defined on some compact set K. Similarly, let $L^\infty(X,\mu)$ be the measurable functions on X that are bounded a.e. with respect to μ. Let $L^\infty_{\text{loc}}(\mathbb{R}^d)$ be the locally essentially bounded functions. A function f is locally essentially bounded if f is in $L^\infty(K)$ for every compact set $K \subset \mathbb{R}^d$. Let $C(X)$, $C(K)$ be the spaces of continuous functions defined on the space X or the compact set K respectively. Let $C^m(X)$ be the space of functions that are continuous and have continuous partial derivatives up to order m. Let $C^\infty(X)$ be the smooth functions. $C^m_C(X)$ and $C^\infty_C(X)$ are the subsets of $C^m(X)$ and $C^\infty(X)$ containing functions of compact support.
Define the *weak derivative* as follows: Let $\Omega \subset \mathbb{R}^d$ be open, $1 \leq p \leq \infty$, $f \in L^p(\Omega)$ and α a multi-index. Then $g \in L^p(\Omega)$ is the weak α-th derivative of f if it fulfils:

$$\int_\Omega g(x)\phi(x)dx = (-1)^{|\alpha|}\int_\Omega f(x)\partial^\alpha\phi(x)dx \quad \forall \phi \in C^\infty_C(\Omega). \tag{0.3}$$

Then the weak α-th derivative of f is denoted by $\partial^\alpha f$.
For $1 \leq p \leq \infty$, $s \in \mathbb{N}$ and $\Omega \subset \mathbb{R}^d$, define the *Sobolev space* $W^p_s(\Omega)$ to be the class of functions in $L^p(\Omega)$ that have weak partial derivatives at almost all points of Ω up to order s such that those derivatives are in $L^p(\Omega)$ i.e.

$$W^p_s(\Omega) := \{f \in L^p(\Omega) : \partial^\alpha f \in L^p(\Omega) \quad \forall\ |\alpha| \leq s\}.$$

The Sobolev space $W^p_s(\Omega)$ is a Banach space when endowed with the norm

$$\|f\|_{s,p,\mu} := \left[\sum_{|\alpha|\leq s}\int_\Omega |\partial^\alpha f|^p d\mu\right]^{\frac{1}{p}} \quad 1 \leq p < \infty$$

or

$$\|f\|_{s,\infty,\mu} := \sum_{|\alpha|\leq s}\|\partial^\alpha f\|_\infty \quad p = \infty.$$

Let

$$\mathcal{B}^s_p(B_d) := \{f : f \in W^p_s(B_d), \|f\|_{s,p,\mu} \leq 1\}$$

be the Sobolev norm balls. Let $\mathcal{M}$ be the space of measurable functions. Let X^* or also $L(X;\mathbb{K})$ be the dual space of the space X which contains linear and continuous functionals from X to some field $\mathbb{K}$ i.e.

$$X^* = L(X;\mathbb{K}) = \{T : X \to \mathbb{K} \text{ s.t. } T \text{ is linear and continuous}\}.$$

Let P_k be the linear space of polynomials of degree at most k.

Contents

1 Introduction

Neural networks have gained major popularity in recent years and celebrated great successes in practice. However, they are notoriously opaque. In general, their inner workings are not fully understood on a mathematical level. This is most apparent, when neural networks exhibit very surprising behaviour [Sze+13]. In object classification settings for example, imperceptible perturbations in input images can lead to objects in images being classified incorrectly. Such input images that fool a neural network are commonly referred to as adversarial examples. The phenomenon of adversarial examples sheds major doubt on the long-held belief that neural networks are able to attain some human level understanding of underlying concepts. Because neural networks are applied so widely, there is a strong need for a more solid mathematical understanding.
Mathematicians have been working on the analysis of neural networks for over 30 years now. They have examined different theoretical aspects using tools and concepts from many different areas within mathematics. Indeed, the opacity of neural networks and the links to many different areas of mathematics make the analysis of neural networks into an exciting challenge. This thesis seeks to present the status quo of mathematical analysis on neural networks and takes the form of a literature review. We hope to provide the reader with some insights into the design choices that enable the success of networks today. The focus of the thesis will be on three key themes.

In chapter (2) on **approximation theory**, we examine classical and fundamental results on **density** of neural networks within different function spaces under different assumptions on the activation function. Notably, we look at the universal approximation results from the 90s of Cybenko and Hornik et al. Further, literature on the degree or **order of approximation** (2.2) is discussed and we learn, when neural networks suffer from the curse of dimensionality (2.18). Finally, we see how **breaking the curse of dimensionality** is possible by exploiting a deep network architecture and compositionality of the target function (2.3). This gives some insights on why a multi-layer structure is necessary for success.

In chapter (3) on **stability** of neural networks we present some of the mathematical analysis that has been done on the phenomenon of **adversarial examples** (3.1) which highlights the instability of neural networks in general. We compare two exemplary defence and regularisation strategies against adversarial attacks looking at the problem also from a game theory perspective (3.1.2).

The main focus of the chapter are then **scattering networks** (3.2) which were first introduced by Mallat et al. The central question here is whether it is possible for neural networks to be stable. Scattering networks iterate on convolution operations with wavelets and non-linearities. We present an analysis of the stability under translation or deformation drawing on tools from signal processing. Parallels between scattering networks and Convolutional Neural Networks (CNNs) give insights into the recent success of neural networks and offer in particular an explanation on why a non-linearity and a deep architecture are indeed necessary.

Chapter (4) discusses **tractable learning** of neural networks. The problem at hand will be to approximate a target function which is a one hidden layer neural network. Here, not only an arbitrarily close approximation, but the **exact identification** of the network weights is facilitated with a sample size that possibly scales only polynomially in the input dimension and network size. Anandkumar et al. (4.1) and Fornasier et al. (4.2) have both addressed this problem in their work and we compare their different approaches (4.3). Anandkumar et al. achieve exact weight identification by whitening and uniquely decomposing a symmetric orthogonal third order tensor. Fornasier et al. use Hessian evaluations and matrix subspace optimisation.

2 Approximation Theory

In this chapter we discuss classical and fundamental results in approximation theory. The chapter is structured into three parts examining results on density, order of approximation and strategies for beating the curse of dimensionality with deep neural networks.

In section (2.1) on **density**, we ask the question under which conditions a set of functions consisting of neural networks is a dense set within some function space. In other words, we want to investigate whether the closure of the set of neural networks gives the entire function space. For analogy we can think of the rational numbers which are dense within the reals. If neural networks are dense within some function space, we can theoretically approximate any function in that space by a neural network to any arbitrary degree of accuracy. Note however that density of neural networks in some space does not give any information on whether an efficient approximation is possible. In general, learning even an infinitely differentiable function requires exponentially many sampling points and is thus intractable [NW10]. Nevertheless without density, good approximation is impossible. Hence density results form a natural starting point for the analysis of neural networks.

In general, the approximation error by neural networks with one hidden layer of r neurons will be bounded above and below as we shall see in section (2.2). In this section on the degree or **order of approximation** we examine the theoretical error that is possible with neural networks and the complexity of the network required to achieve this. As we will discover, error bounds depend on parameters such as the number of neurons or the input dimension. We will see when shallow neural networks suffer from the famous **curse of dimensionality** (2.18).

In section (2.3) we take a look at the advantages that **deep networks** offer over shallow networks. In section (2.1) we learn that shallow neural networks are universal approximators. As shallow networks can be viewed as a special case of general neural networks, both shallow and deep nets are universal approximators. The accuracy that is theoretically achievable is thus the same in both cases. Then why are deep networks preferred over shallow ones in practice, even though their parameter tuning is much more difficult? In this section, we will seek to answer this question and see how **breaking the curse of dimensionality** is possible with deep neural networks, when target functions have a compositional structure.

2.1 Universal Approximation or Density in $C(X)$

In this section we examine results on density of neural network classes within function spaces. They are also famously known as *universal approximation* results. Concretely, we consider a set of one layer neural networks $\Sigma^d(\sigma)$ or short $\Sigma(\sigma)$,

$$\Sigma(\sigma) = \text{span}\Big\{\sigma(w \cdot x + \theta) : w \in \mathbb{R}^d, \theta \in \mathbb{R}\Big\}$$

with activation function $\sigma : \mathbb{R} \to \mathbb{R}$, weights $w \in \mathbb{R}^d$ and bias $\theta \in \mathbb{R}$. The central question of this chapter is under which conditions $\Sigma(\sigma)$ is a dense set in the space of continuous functions on some compact set K, $C(K)$. Here density is with respect to the uniform norm $\|f\|_{K,\infty} = \sup_{x\in K}|f(x)|$ for $f \in C(K)$ for every compact $K \subset \mathbb{R}^d$. Along the way, density will be considered also in other function spaces like the well known L^p spaces or the space of measurable functions, $\mathcal{M}$.
As an example, $L^p(K, \mu)$ - the p-integrable functions with respect to some non-negative finite Borel measure μ - are dense in $C(K)$ in the p-norm for $1 \leq p < \infty$. Hence, we will often extend results on density in $C(K)$ to results on $L^p(K, \mu)$.
From a practical point of view the reader should be aware that density of $\Sigma(\sigma)$ in $C(\mathbb{R}^d)$ does not guarantee an arbitrarily accurate approximation of $f \in C(\mathbb{R}^d)$ by members of

$$\Sigma_r(\sigma) = \Big\{ \sum_{i=1}^{r} c_i\sigma(w_i \cdot x + \theta_i) : c_i, \theta_i \in \mathbb{R}, w_i \in \mathbb{R}^d \Big\}$$

for fixed r. The shallow neural networks in $\Sigma_r(\sigma)$ are restricted to have not more than r units in the hidden layer.

A variety of different mathematical tools is used to prove density results. Here we will expose some of the most prominent theorems and the proof strategies taken. Cybenko [Cyb89], Hornik et al. [HSW89] and Funahashi [Fun89] prove very similar results on density of one layer neural networks in the space of continuous functions under slightly different assumptions on the activation function.

Key Concepts: Hahn-Banach Theorem, Riesz-Markov-Kakutani Representation Theorem, Lebesgue Bounded Convergence, Stone-Weierstrass Theorem

Cybenko [Cyb89] presents the most succinct proof. Key tools are the Hahn-Banach Theorem the Riesz-Markov-Kakutani Representation Theorem and the Lebesgue Bounded Convergence Theorem. He proves density in $C([0, 1]^d)$ for one layer nets with any sig-

moidal activation function (2.1) with limits of 0 and 1 at $\pm\infty$. Hornik et al. [HSW89] assume σ to have the same limit behaviour, but to be monotonous, taking values only in $[0,1]$ and potentially discontinuous at countably many points. They proceed differently by proving density in $C(X)$ for sums of products of activation functions by applying the Stone-Weierstrass Theorem (2.6). They then use the fact that we can approximate $\cos(x)$ arbitrarily well from $\Sigma_1(\sigma)$ and the product to sum formula for the cosine function to obtain density of $\Sigma_r(\sigma)$ in $C(X)$. Funahashi [Fun89] proves density in $C(K)$ for non-constant, bounded and monotonously increasing σ using Fourier analysis and mollifiers.
Due to the greater prominence of the results in [Cyb89] and [HSW89] we will focus on those proofs and compare the key ideas in more detail in (2.1.1.1) and (2.1.1.2). In both (2.1.1.1) and (2.1.1.2) first the key result of the authors on density in $C(K)$ is presented. Then related results by the authors on density for example in L^p and in the measurable functions $\mathcal{M}$ are presented.

2.1.1 Density in $C(X)$ with Continuous Activation Functions

2.1.1.1 Continuous Discriminatory Activation Functions

As a first result on density of the set of neural networks within the class of continuous functions we present the classic universal approximation result of Cybenko [Cyb89, Theorem 1]. Here the activation function σ is assumed to be continuous and sigmoidal (2.1).

Key Concepts: Riesz-Markov-Kakutani Representation Theorem, Lebesgue Bounded Convergence Theorem, Hahn-Banach Theorem, Hölder's Inequality

Due to the omnipresence of the concept of sigmoidality and the importance of the property of being discriminatory also to the work of Hornik [Hor91] we state the two definitions here.

Definition 2.1 (sigmoidal activation function)**.** *[Cyb89, p. 306]*
$\sigma : \mathbb{R} \to \mathbb{R}$ *is sigmoidal if*

$$\lim_{x\to\infty} \sigma(x) = 1 \quad \textit{and} \quad \lim_{x\to-\infty} \sigma(x) = 0.$$

Definition 2.2 (discriminatory activation function)**.** *[Cyb89, p. 306]*
$\sigma : \mathbb{R}^d \to \mathbb{R}$ is discriminatory if for μ a finite signed measure on $\mathbb{R}^d$,

$$\int_{\mathbb{R}^d} \sigma(w \cdot x + \theta) d\mu(x) = 0 \qquad \forall w \in \mathbb{R}^d, \theta \in \mathbb{R}$$

implies $\mu \equiv 0$.

Theorem 2.3 (Density in $C(K)$)**.** *[Cyb89, Theorem 1]*
If σ is continuous and sigmoidal, $\Sigma(\sigma)$ is dense in $C(K)$.

Proof:
The proof proceeds in two steps by proving two claims.
Claim 1: If σ is discriminatory, then $\Sigma(\sigma)$ is dense in $C([0,1]^d)$ [Cyb89, Theorem 1].
Claim 2: Continuous sigmoidal functions are discriminatory [Cyb89, Lemma 1].
Proof of Claim 1: The proof proceeds by contradiction. For $\Sigma_r(\sigma)$ not dense in $C(K)$ then the closure of $\Sigma(\sigma)$, $\overline{\Sigma(\sigma)}$, is a closed linear subspace of $C(K)$. Thus by Hahn-Banach there exists a functional Λ in the dual space of $C(K)$, $C(K)^*$, such that Λ vanishes on $\overline{\Sigma(\sigma)}$, but is not identically zero. By the Riesz-Markov-Kakutani Representation Theorem for compactly supported continuous functions we can write the functional Λ as

$$\Lambda(f) = \int_K f(x) d\mu(x) \qquad \forall f \in C(K)$$

for some finite signed regular Borel measure μ on K. Hence it holds that

$$0 = \Lambda(\sigma(w \cdot x + \theta)) = \int_K \sigma(w \cdot x + \theta) d\mu(x) \quad \forall w \in \mathbb{R}^d, \theta \in \mathbb{R}.$$

This implies $\mu \equiv 0$ and hence $\Lambda \equiv 0$ which gives a contradiction.
Proof of Claim 2: It remains to show that continuous sigmoidal functions are discriminatory. Assume

$$0 = \int_K \sigma(\lambda(w \cdot x + \theta) + \phi) d\mu(x)$$

for any w, x, θ, ϕ and finite, signed, regular Borel measure μ. By inspecting the limit behaviour as $\lambda \to \infty$ and using the Lebesgue Bounded Convergence Theorem we can show that

$$\begin{aligned} 0 &= \int_K \sigma(\lambda(w \cdot x + \theta) + \phi) d\mu(x) \\ &= \sigma(\phi)\mu(\{x | w \cdot x + \theta = 0\}) + \mu(\{x | w \cdot x + \theta > 0\}). \end{aligned}$$

We want μ to be identically zero. Define the functional F in the dual space of L^∞, $L^{\infty *}$, by $F : f \to \int_K f(w \cdot x) d\mu(x)$. We can show that $F(f) = 0$ if f is the indicator function

of intervals of the form $[c, \infty)$ and (c, ∞) and hence for indicator functions of any intervals. Hence F vanishes on the set of simple functions (functions given by a linear combination of indicator function of disjoint measurable sets). But since the simple functions are dense in $L^\infty(\mathbb{R})$, F is identically zero. In particular F vanishes at $e^{iw\cdot x}$, hence the Fourier transform of μ is zero. Hence μ is identically zero. □

Hornik [Hor91] takes an approach very similar to Cybenko and also extends the results in [Cyb89]. He obtains density of $\Sigma(\sigma)$ in $C(K)$ for σ continuous, bounded and non-constant. In fact σ being a continuous squashing function in [Cyb89], also implies it being bounded and non-constant [Hor91, Theorem 2]. The continuity assumption here can in fact be relaxed [HSW89] (2.1.1.2) or dropped entirely [Les+93] (2.1.2) as we shall see later.

Density in L_p

Hornik [Hor91, Theorem 1] relaxes the continuity assumption on σ and proves density in $L_p(\mu)$ for finite μ. We have seen that for density in $C(K)$ the proof boils down to proving σ is discriminatory. The same occurs for density in L_p. In fact similar to before, for a contradiction we assume that $\Sigma(\sigma)$ is not relatively closed in L^p. Hence by Hahn-Banach, there is a functional $\Lambda \in L^{p^*}$ that vanishes on $\Sigma(\sigma)$, but is not identically zero. Using the isometric isomorphism between L^p and L^{q^*} with q being the conjugate exponent we can write

$$\Lambda(f) = \int_{\mathbb{R}^d} fg d\mu$$

for some $g \in L^q(\mu)$. Now we can define the measure $\tilde{\mu}(B) := \int_B g d\mu$ and see by Hölder's inequality and the fact that μ is finite that $\tilde{\mu}$ is a non-zero finite signed measure.
Hence now in the case of both $C(K)$ and L^p we are facing the question whether there can exist a non-zero finite signed measure μ such that

$$\int_{\mathbb{R}^d} \sigma(w \cdot x + \theta) d\mu(x) = 0 \qquad \forall w \in \mathbb{R}^d, \theta \in \mathbb{R}$$

or in other words whether σ is discriminatory (2.2). Remember, in the case of $C(K)$ we have the additional continuity assumption on σ.
At this point Hornik [Hor91, Theorem 5] proves that any bounded and non-constant σ is discriminatory using concepts of Fourier analysis on groups [Rud17]. This is a stronger result than the one in [Cyb89] which assumes distinct limits at $\pm\infty$ (see (2.1)) instead.

Density in C^m

Using again that bounded and non-constant functions are discriminatory, it is also proved that for m times continuously differentiable $\sigma \in C^m(\mathbb{R}^d)$, $\Sigma(\sigma)$ is dense in $C^m(\mathbb{R}^d)$ with respect to the norm

$$\|f\|_{m,K} := \max_{|\alpha| \leq m} \sup_{x \in K} |\partial^\alpha f(x)|$$

for compact K [Hor91, Theorem 3]. $\Sigma(\sigma)$ is also dense in the space

$$C^{m,p}(\mu) = \left\{ f \in C^m(\mathbb{R}^d) : \|f\|_{m,p,\mu} < \infty \right\}$$

with

$$\|f\|_{m,p,\mu} := \left[\sum_{|\alpha| \leq m} \int_{\mathbb{R}^d} |\partial^\alpha f|^p d\mu \right]^{\frac{1}{p}} \quad 1 \leq p < \infty$$

for finite compactly supported measures μ [Hor91, Theorem 4]. As a generalisation we get also density of $\Sigma(\sigma)$ in $C^{m,p}(\mu)$ for all finite not necessarily compactly supported measures μ under the stronger assumption that $\sigma \in C^m(\mathbb{R}^d)$ is non-constant and all the derivatives up to order m are bounded.

Density in $\mathcal{M}$

As convergence in L^p implies convergence in μ measure, from the above we obtain also density of $\Sigma(\sigma)$ in the measurable functions $\mathcal{M}$ for σ bounded and non-constant [Hor91, p. 253]. It follows also that any $f \in \mathcal{M}$ can be approximated up to an accuracy of ϵ on a compact $K \subset \mathbb{R}^d$ s.t. $\mu(\mathbb{R}^d \backslash K) < \epsilon$.

2.1.1.2 Squashing Activation Functions

Hornik et al. [HSW89] prove a result on density in $C(K)$ which is similar to the result in [Cyb89], [Hor91] detailed in the previous section (2.1.1.1). In (2.1.1.1), σ was assumed to be continuous and discriminatory. Now in the following density in $C(K)$ will be proved when σ is a squashing function (2.4) which comprises it being continuous a.e. While the statements of the key results are quite similar, the proof presented in the following is very different and somewhat more straightforward, as it is only based on one deep result, the Stone-Weierstrass Theorem.

Key Concepts: Stone-Weierstrass Theorem

Definition 2.4 (Squashing function)**.** *The function $\sigma : \mathbb{R} \to [0,1]$ is referred to as squashing function if it has countably many discontinuities and the following limits at $\pm\infty$:*

$$\lim_{x\to\infty} \sigma(x) = 1 \qquad \textit{and} \qquad \lim_{x\to-\infty} \sigma(x) = 0.$$

Theorem 2.5 (Density in $C(K)$ for σ a squashing function)**.** *[HSW89, Theorem 2.3-2.4] If σ is an arbitrary squashing function, then $\Sigma(\sigma)$ is dense in $C(K)$.*

Proof outline:
Hornik et al. [HSW89] take an approach to proving density in $C(K)$ which is notably different to (2.1.1.1). They consider first a broader class of networks

$$\Sigma\Pi(\sigma) := \left\{ f : \mathbb{R}^d \to \mathbb{R} | f(x) = \sum_{i=1}^{q} \beta_i \prod_{j=1}^{l} \sigma(w_{ij} \cdot x + \theta_{ij}); \beta_i, \theta_{ij} \in \mathbb{R}; w_{ij} \in \mathbb{R}^d; l_i, q \in \mathbb{N} \right\}$$

for σ being continuous and non-constant. This enables them to directly apply the Stone-Weierstrass Theorem.

Theorem 2.6 (Stone-Weierstrass Theorem)**.** *Assume A is an algebra of continuous real functions on some compact set K which separates points on K, i.e. for all $x, y \in K$ such that $x \neq y$, there is some function $f \in A$ such that $f(x) \neq f(y)$. Assume further that A vanishes nowhere on K, i.e. that for all $x \in K$ there is some $f \in A$ such that $f(x) \neq 0$. Then A is dense in $C(K)$ in the supremum norm $\|f\|$.*

The theorem is applied with $A = \Sigma\Pi(\sigma)$ and the assumptions for the Stone-Weierstrass Theorem can be checked quite easily. This readily yields density in $C(K)$.
They then extend this result to activation functions σ that are squashing functions. A constructive approach is used to show that we can approximate arbitrarily well any continuous squashing function by members of $\Sigma^1(\phi)$ $(d = 1)$ for some arbitrary squashing function ϕ [HSW89, Lemma A.2.]. Hence it is not hard to show that $\Sigma\Pi(\sigma)$ is dense in $\Sigma\Pi(\phi)$ which is in turn dense in $C(K)$.
To move from $\Sigma\Pi(\sigma)$ to $\Sigma(\sigma)$ they use the fact that the cosine function can be approximated arbitrarily well on a bounded set by linear combinations of the cosine squasher function [GW88]. The approximator is a member of $\Sigma^1(\sigma)$ for σ being any squashing function. Hence we can approximate linear combinations of $cos(w_i \cdot x)$ by members of $\Sigma\Pi(\sigma)$. Now we know trigonometric polynomials which are of the form

$$\sum_{i=1}^{q} \beta_i \prod_{j=1}^{l_i} \cos(w_{ij} \cdot x + \theta_{ij})$$

are dense in $C(K)$. Using the product to sum identity for the cosine function we can write this as a linear combination of the cosine function. Hence we have density of $\Sigma(\sigma)$ in $C(X)$ for arbitrary squashing functions σ. □

Density in L_p

Hornik et al. [HSW89, Corollary 2.2] also obtained results on $L_p(\mathbb{R}^d, \mu)$ for $1 \leq p < \infty$ where now μ is such that there exists a compact $K \subset \mathbb{R}^d$ s.t. $\mu(K) = 1$ (in [Hor91, Theorem 1] μ is assumed to be finite on $\mathbb{R}^d$). Then density holds in the p-norm independently of the input dimension, the activation function or μ. This result hinges on the fact that we can approximate any function in L_p by a bounded function in L_p (by setting function values over some bound M to zero for M sufficiently large) and that we can approximate any bounded function in L_p by a continuous function. As we already have uniform density on compacta of $\Sigma(\sigma)$ in $C(\mathbb{R}^d)$ and as K has measure 1, density of $\Sigma(\sigma)$ in L_p follows.

Density in $\mathcal{M}$

Hornik et al. [HSW89, Theorem 2.2] go further than Cybenko [Cyb89] not only in allowing for countably many discontinuities, they also prove density in a different norm in the space of measurable functions $\mathcal{M}$. They use the norm

$$\|f\|_\mu := \inf\left\{\epsilon > 0 : \mu\{x : |f(x)| > \epsilon\} < \epsilon\right\}$$

and show that convergence in the uniform norm on compacta implies convergence in $\|\cdot\|_\mu$ and hence $C(\mathbb{R}^d)$ is dense in the measurable functions $\mathcal{M}$ under $\|\cdot\|_\mu$. Thus we also have density in $\mathcal{M}$ under $\|\cdot\|_\mu$ of one layer neural nets with any squashing function as activation.

2.1.2 Density in $C(K)$ with discontinuous activation functions

In the previous section some degree of continuity was assumed on the activation function σ. Either it was continuous everywhere (2.1.1.1) or only countably many discontinuities were permitted (2.1.1.2). Leshno et al. [Les+93] show us how we can drop the continuity assumption on σ entirely. Only non-polynomiality is required. The reader will note that dropping the continuity assumption on σ results in a proof which is comparatively more involved than the proofs presented so far. As an intermediate step interestingly it is proved that we can always reduce the discussion from $\mathbb{R}^d$ to $\mathbb{R}$.

Key Concepts: Weierstrass Approximation Theorem, Lusin's Theorem, Baire's Category Theorem

For the following theorem, denote by $L^\infty_{loc}(\mathbb{R})$ the locally essentially bounded functions on $\mathbb{R}$. A function f is locally essentially bounded if f restricted to some set K is in $L^\infty(K)$ for every compact $K \subset \mathbb{R}^d$.

Theorem 2.7 (Density in $C(\mathbb{R}^d)$ for possibly discontinuous σ)**.** *[Les+93, Theorem 1]*
$\Sigma(\sigma)$ is dense in $C(\mathbb{R}^d)$ iff $\sigma \in L^\infty_{loc}(\mathbb{R})$ is not a polynomial (a.e.) and the closure of its points of discontinuity has zero Lebesgue measure.
Also we can restrict the weights w to lie in some $W \subseteq \mathbb{R}^d$ such that there exists no non-trivial homogeneous polynomial vanishing on W [Les+93, Proposition 2].

Proof outline:
Clearly, if σ is a polynomial, then $\Sigma(\sigma)$ is not dense in $C(\mathbb{R}^d)$ as it consists of polynomials of bounded degree which are rather sparse in $C(\mathbb{R}^d)$.
The reverse implication requires more work and takes a number of steps that are instructive in themselves and presented as lemmas in the following.
First, we can reduce the problem from $\mathbb{R}^d$ to $\mathbb{R}$. Hence we are not dealing with sums of ridge functions anymore, but with linear combinations of scales and shifts of σ.

Lemma 2.8 (Dimensionality reduction from d to 1)**.** *[Les+93, p. 4 Step 2]*
If $\Sigma^1(\sigma)$ is dense in $C(\mathbb{R})$, then $\Sigma^d(\sigma)$ is dense in $C(\mathbb{R}^d)$.

By [VK61] the set of ridge functions

$$\operatorname{span}\Big\{\phi(w \cdot x) | w \in \mathbb{R}^d, \phi \in C(\mathbb{R})\Big\}$$

is dense in $C(\mathbb{R}^d)$. Thus there exists a $\phi_i \in C(\mathbb{R})$ such that we can approximate any $g(x) \in C(\mathbb{R}^n)$ arbitrarily well by $\sum_{i=1}^r \phi_i(w_i \cdot x)$ for some collection of $w_i \in \mathbb{R}^d$. As $\Sigma^1(\sigma)$ is dense in $C(\mathbb{R})$ by assumptions, we can approximate the ϕ_i by elements of

$$\operatorname{span}\{\sigma(\lambda x + \theta) : \theta, \lambda \in \mathbb{R}\}.$$

Using a very similar argument, Pinkus [Pin99, p. 155] shows that we can restrict w to lie in some $\mathbb{R} \times W$ for some $W \subseteq S^{n-1}$. W must have the property that no nontrivial homogeneous polynomial vanishes on W [VK61].

Proposition 2.9 (Density for smooth σ)**.** *[Les+93, p. 4 Step 3]*
If $\sigma \in C^\infty$ is not a polynomial, then $\Sigma^1(\sigma)$ is dense in $C(\mathbb{R})$.

Considering the definition of the derivative we see that derivatives of all orders of $\sigma(\lambda x + \theta)$ with respect to λ are in the closure of $\Sigma^1(\sigma)$, $\overline{\Sigma^1(\sigma)}$. Since σ is not a polynomial for all $k \in \mathbb{N}$ there exists a θ_k such that $\sigma(\theta_k) \neq 0$. Hence $\overline{\Sigma^1(\sigma)}$ contains all monomials

$$x^k \sigma^{(k)}(\theta_k) = \frac{\partial^k}{\partial \lambda^k} \sigma(\lambda x + \theta)\big|_{\lambda=0,\theta=\theta_k}.$$

Hence $\overline{\Sigma^1(\sigma)}$ contains all polynomials, hence by the Weierstrass Approximation Theorem $\Sigma^1(\sigma)$ is dense in $C(\mathbb{R})$.
In fact we can restrict θ to lie in some open set and λ to lie in a set that contains a sequence which converges to 0.

Remark: For $\sigma \in C(\mathbb{R})$ we can finish the prove at this point [Les+93, p. 6] using the theory of mean-periodic functions [Sch47]. It tells us that if $\sigma \in C(\mathbb{R})$ is not a polynomial then $\overline{\Sigma^1(\sigma)}$ contains a function of the form $e^{\lambda x}\cos(\nu x) \in C^\infty$ for some $\lambda, \nu \in \mathbb{R}\backslash\{0\}$.

The next lemma contains the crucial generalisation step from $\sigma \in C^\infty$ to $\sigma \in L^\infty_{\text{loc}}$.

Lemma 2.10 (Dropping the smoothness assumption on σ)**.** *[Les+93, p. 4 Step 4]*
For each σ locally essentially bounded, Riemann-integrable, not a polynomial and $\psi \in C^\infty$ with compact support it holds that the convolution $\sigma * \psi$ is in $\overline{\Sigma^1(\sigma)}$.

For continuous σ this would be easy [Pin99, Proposition 3.7]. If we don't have continuity, Leshno et al. [Les+93, p. 4 Step 4] use a constructive proof to show that we can approximate

$$\int \sigma(x - y)\psi(y)dy$$

uniformly by a sum of the form

$$\sum_{i=1}^{k} \sigma(x - y_i)\psi(y_i)\Delta y_i$$

on any compact K for adequately chosen function evaluation points y_i and distance between them Δy_i. Here Lusin's theorem is used to get uniform continuity of σ on $\text{supp}(\psi)\backslash U$ where U is a finite union of intervals that has small measure. On U itself they exploit the Riemann integrability of σ.

From this follows that if for some $\psi \in C^\infty$ with compact support, $\sigma * \psi$ is not a polynomial then $\Sigma^1(\sigma)$ is dense in $C(\mathbb{R})$ [Les+93, p. 5 Step 5]. It remains to exclude the possibility that the convolution $\sigma * \psi$ is a polynomial for all $\psi \in C^\infty$.

Lemma 2.11. *[Les+93, p. 5 Step 6, p. 6 Step 7]*
For σ bounded, Riemann integrable and not a polynomial assume that the convolution $\sigma * \psi$ is a polynomial for all $\psi \in C_C^\infty$ (compact support), then we can bound the degree of $\sigma * \psi$ for all ψ.
It follows that σ itself is a polynomial a.e. of the same bounded degree.

By Baire's Category Theorem there exists an $n \in \mathbb{N}$ such that

$$C_C^\infty([a,b]) = \left\{ \psi \in C_C^\infty([a,b]) | \text{ degree}(\sigma * \psi) \leq n \right\}.$$

If $\sigma * \psi$ is a polynomial of bounded degree for all $\psi \in C^\infty$, then in particular this is true for mollifiers. Hence we can have a sequence $(\psi_n)_{n \in N}$ such that $\sigma * \psi_n \to \sigma$ in L^p for $1 \leq p < \infty$. Hence if $\sigma * \psi$ is a polynomial of bounded degree so is σ (a.e.). This contradicts our assumption and finishes the proof. □

For completeness we mention also [Pin99, Proposition 3.8], another result on density in $C(\mathbb{R}^d)$ with discontinuous activation functions. The proof is very similar to the one we have just described. The assumption on the activation function differs slightly in that Pinkus [Pin99] assumes σ to be bounded and locally Riemann-integrable. We have concentrated on the result of Leshno et al. [Les+93] as presented above, as it is the earlier result.
Having now proved density in $C(\mathbb{R}^d)$ for non-polynomial activation functions, we can easily extend this result to obtain density in $L^p(\mu)$:

Theorem 2.12 (Density in $L^p(\mu)$ for possibly discontinuous σ)**.** *[Les+93, Proposition 1]*
$\Sigma(\sigma)$ is dense in $L^p(\mu)$ for $1 \leq p < \infty$ iff σ is not a polynomial (a.e.) where μ is a non-negative finite measure on $\mathbb{R}^n$ with compact support which is absolutely continuous with respect to the Lebesgue measure.

Proof:
Should σ be a polynomial, then $\Sigma(\sigma)$ contains only polynomials of bounded degree and is hence not dense in $L^p(\mu)$ for $1 \leq p < \infty$. For the other implication use the previous result that $\Sigma(\sigma)$ is dense in $C(K)$ in the uniform norm for non-polynomial σ. The claim then follows from the fact that $C(K)$ is dense in $L^p(\mu)$ in the p-norm [AF03]. □

Conclusion

In this section we have started our survey on the mathematical analysis of neural networks with fundamental results on density. We have examined two different strategies for proving density of shallow neural networks within the continuous functions ([Cyb89]/[Hor91] and [HSW89]), when the activation function is continuous (a.e.). The work of Leshno et al. [Les+93] has allowed us to drop the continuity assumption on the activation function entirely and replace it with the very weak assumption of non-polynomiality. As the activation functions commonly used in practice (like the ReLU, hyperbolic tangent or the sigmoid function) are continuous, the density result on discontinuous activation functions is instructive, but more of theoretical interest. Hence, we have learnt that shallow neural networks have the theoretical ability to approximate any continuous function to an arbitrary degree of accuracy. We have also examined some related results on density within the integrable and measurable functions. The density results show that neural networks are a powerful class of models that can approximate any reasonable function arbitrarily well. The question remains at which cost a very close approximation might come. Does the achievable error scale favourably in the relevant parameters such as the input dimension? In other words, is the close approximation which is possible in theory actually realistic in practice? We will turn to this question in the next section on order of approximation and the curse of dimensionality.

2.2 Order of Approximation

In this section we collect results on the order or the degree of approximation. In contrast to the last section that was examining a theoretical capability of neural networks to approximate a given function, we now examine the theoretical approximation error that is possible with neural networks. The error will depend on relevant parameters like the input dimension d, the number of neurons r and a parameter quantifying how smooth the target function is. In particular, the dependency on r highlights how complex a shallow neural networks needs to be to guarantee some specified error.

Definition 2.13 (Order of Approximation)**.** *[Pin99, p. 168]*
For function f in some space X denote the order of approximation by

$$E(f; X; \Sigma_r(\sigma)) = \inf_{g \in \Sigma_r(\sigma)} \|f - g\|_X.$$

Ultimately, we would obviously like to find upper and lower bounds on

$$E(X; \Sigma_r(\sigma)) := \sup_{f \in X} \inf_{g \in \Sigma_r(\sigma)} \|f - g\|_X.$$

These bounds will depend on parameters such the dimension d and the number of neurons r, thus indicating how many neurons are required for a prescribed accuracy.

Throughout this section we will consider as target functions members of the *Sobolev spaces* $W_s^p(B_d)$ that have weak partial derivatives up to order s at almost all point of B_d:

$$W_s^p(B_d) := \{f \in L^p(B_d) : \partial^\alpha f \in L^p(B_d) \quad \forall\, |\alpha| \leq s\}$$

for $1 \leq p \leq \infty$ and $s \in \mathbb{N}$. Sobolev spaces offer a natural way of characterising how smooth a target function is. We can think of target functions being categorised into smoothness classes according to the parameter s. In particular, we consider the Sobolev classes

$$\mathcal{B}_p^s(B_d) := \{f : f \in W_s^p(B_d), \|f\|_{s,p,\mu} \leq 1\}$$

of functions that have Sobolev norm 1 (Note that any function in $W_p^s([-1,1]^d)$ can be scaled to have norm 1. See also preliminaries).
One might expect that more favourable approximation properties hold if the target function displays a greater degree of smoothness. This idea is explored in this section.

Most order of approximation results do not consider $\Sigma_r(\sigma)$ specifically, but a more

general class of functions, namely the *ridge functions:*

$$\mathcal{R}_r = \left\{ \sum_{i=1}^{r} \sigma_i(w_i \cdot x) : w_i \in \mathbb{R}^d, \sigma_i \in C(\mathbb{R}), i \in [r] \right\}.$$

We observe that $\Sigma_r(\sigma) \subset \mathcal{R}_r$ for $\sigma \in C(\mathbb{R})$. Hence for the order of approximation from both the two sets, the following relation holds:

$$E(f; X; \Sigma_r(\sigma)) = \inf_{g \in \Sigma_r(\sigma)} \|f - g\|_X \geq \inf_{g \in \mathcal{R}_r} \|f - g\|_X = E(f; X; \mathcal{R}_r). \tag{2.1}$$

The following theorem shows that both the upper and the lower bound on the degree of approximation from $\mathcal{R}_r$ will be in the order of $r^{-s/(d-1)}$.

Theorem 2.14. *[Mai99, Theorem 1]*
For each $d \geq 2$, $s \geq 1$ it holds that

$$E(\mathcal{B}_2^s; \mathcal{R}_r; L_2) = sup_{f \in \mathcal{B}_2^s} E(f; \mathcal{R}_r; L_2) \asymp r^{-s/(d-1)}$$

where d is the input dimension for f and s is the number of continuous partial derivatives.

Comment: The theorem confirms our intuition that good approximation is more difficult in high dimensions and that favourable smoothness properties can improve approximation properties. The lower bound here is indeed relevant and more than a negligible worst case, as it is attained for a set of functions of large measure [MMR99]. For tighter bounds in the case of specific σ see [MM00].

The following result by [Pin99, Theorem 6.2] proves the upper bound in the previous theorem considering the more general $\mathcal{B}_p^s$ rather than $\mathcal{B}_2^s$.

Theorem 2.15. *[Pin99, Theorem 6.2]*
For each $p \in [1, \infty]$, $d \geq 2$, $s \geq 1$ it holds that

$$E(\mathcal{B}_p^s; \mathcal{R}_r; L_p) = sup_{f \in \mathcal{B}_p^s} E(f; \mathcal{R}_r; L_p) \leq r^{-s/(d-1)}$$

Proof outline:
Let P_k be the linear space of polynomials of degree at most k and H_k the space of homogeneous polynomials of degree k. The proof relies on the fact that we can find $r = \dim(H_k)$ ridge directions in order to span H_k. We have

$$H_k = \text{span}\{(w_i \cdot x)^k : i \in [r]\}$$

and thus

$$\begin{aligned} P_k &= \operatorname{span}\Big\{(w_i \cdot x)^j : i \in [r], j \in [k]\Big\} \\ &= \Big\{\sum_{i=1}^{r} p_i(w_i \cdot x) : p_i \text{ a univariate polynomial of degree at most k }, i \in [r]\Big\}. \end{aligned} \tag{2.2}$$

It follows that $P_k \subseteq \mathcal{R}_r$, hence

$$E(\mathcal{B}_p^s; \mathcal{R}_r; L_p) \leq E(\mathcal{B}_p^s; P_k; L_p) \leq Ck^{-s}.$$

Note at this point also the interesting link to (2.4). Now $\dim(H_k) = r = \binom{d-1+k}{k}$ and as $r \asymp k^{d-1}$, the result follows. □

The previous theorem does not give particularly deep insights as it merely states that we can approximate at least as well using ridge functions as we could with polynomials. The question remains why in that case it would be worth using neural networks. Pinkus et al. [MP99, Proposition 1], [Pin99, Theorem 6.4] find that in fact the lower bound of the previous theorem can be attained using activation functions that are sigmoidal (2.1), strictly increasing and in C^∞. They go about this by first constructing a specific σ, in order to approximate any member of $\mathcal{R}_r$ by a neural network. The upper bound then follows in conjunction with theorem (2.15).

Proposition 2.16. *[MP99, Proposition 1]*
There exist $\sigma \in C^\infty$ which is sigmoidal (2.1) and strictly increasing such that for every $f \in \mathcal{R}_r$ and $\epsilon > 0$ there exists $g \in \Sigma_{r+d+1}(\sigma)$ such that

$$|f(x) - g(x)| < \epsilon \qquad \forall x \in B_d.$$

Corollary 2.17. *[Pin99, Theorem 6.4]*
There exists $\sigma \in C^\infty$, sigmoidal and strictly increasing such that

$$E(\mathcal{B}_p^s; \Sigma_r(\sigma); L_p) \leq Cr^{-s/(d-1)}$$

for each $p \in [1, \infty]$, $s \geq 1$, $d \geq 2$.

We want to point out that the activation function used above has desirable properties, but is artificially constructed and only of theoretical interest, so we abstain from presenting the proof.

2.2.1 Continuous Methods of Approximation

So far we have analysed the order of approximation for different smoothness classes given by the $\mathcal{B}_p^s$. This tells us which errors we can theoretically achieve in the best case. However, the approximation errors in practice rely not only on the order of approximation, but also on other factors for example the method by which the approximation is done. Next, we consider the case in which the network parameters depend continuously on the target function. We will see that both the upper and the lower bound on the order of approximation is in the order of $r^{-s/d}$. This exponential dependence on the dimension d is famously known as the **curse of dimensionality**.

Key Concepts: Polynomial Approximation, Chebychev Polynomial, de la Vallée Poussin Operator, Baire's Category Theorem, Continuous Non-Linear t-Width, Bernstein t-Width, Borsuk-Ulam Theorem

We proceed by proving the upper bound. For this, Mhaskar [Mha96] considers generalised translation networks which are a generalisation of neural networks. We denote $\Sigma_{r,t}^d(\sigma)$ or short

$$\Sigma_{r,t}(\sigma) = \left\{ \sum_{i=1}^{r} c_i \sigma(W_i x - \theta_i) : c_i, \in \mathbb{R}, \theta_i \in \mathbb{R}^t, W_i \in \mathbb{R}^{t \times d} \right\} \tag{2.3}$$

for $\sigma : \mathbb{R}^t \to \mathbb{R}$. Remarkably, weights and biases in the approximation method in [Mha96, Theorem 2.1] are determined independently of the target function. The coefficients are given by linear continuous functionals evaluated at the target function. The conditions of the following theorem are in particular fulfilled by the logistic sigmoid activation function.

Theorem 2.18. *[Mha96, Theorem 2.1]*
Let $p \in [1, \infty]$, $1 \leq t \leq d$, $s \geq 1$, $r \geq 1$.
Assume the following assumptions hold:

- *The activation function is smooth on a ball of some radius in $\mathbb{R}^t$: $\sigma \in C^\infty(B(\rho))$ for $B(\rho) \subseteq \mathbb{R}^t$ and $\rho > 0$.*
- *There exists $\theta \in B(\rho)$ s.t. $\partial^k \sigma(\theta) \neq 0 \quad k \in \mathbb{Z}^d, k \geq 0$. Here k is a multi-integer of non-negative entries. Equivalently we could impose that σ is not a polynomial [CB54].*
- *Non-noisy samples of the target function $f \in W_p^s([-1,1]^d)$ are available.*

Then there exist matrices $\{W_1, \cdots, W_r\}$ *in* $\mathbb{R}^{t\times d}$ *such that for any* $f \in W_p^s([-1,1]^d)$ *there exist coefficients* $\{c_1(f), \cdots, c_r(f)\}$ *in* $\mathbb{R}$ *such that*

$$E(f; W_p^s([-1,1]^d); \Sigma_{r,t}(\sigma)) = \|f - \sum_{i=1}^{r} c_i(f)\sigma(W_i \cdot +\theta)\|_p \leq cr^{-s/d}\|f\|_{s,p}$$

where the c_i *are in the dual space of* $W_p^s([-1,1]^d)$.
Hence,

$$E(\mathcal{B}_p^s; \Sigma_r(\sigma); L_p) \leq Cr^{-s/d}.$$

Proof outline:
The proof relies on first constructing a polynomial $P_m(f)$ of bounded coefficients and degree $m \geq s$ s.t.

$$\|f - P_m(f)\|_p \leq cm^{-s}\|f\|_{s,p}.$$

The construction relies on using the de la Vallée Poussin operator and its properties along with Chebychev polynomials. For more details see [MM95], [Tim14]. The authors then proceed to approximate $P_m(f)$ by a neural network g. Using a similar reasoning as we have seen in the proof of proposition (2.9), we can obtain monomials from differentiating $\sigma(w_i \cdot x + \theta)$ provided σ is not a polynomial. The derivatives and hence the monomials in $P_m(f)$ are approximated by divided differences. Thus weights and biases are fixed independently of the target function f. The $c_i \in W_p^s([-1,1])^*$ are then the coefficients occurring in $P_m(f)$ that are given in terms of the de la Vallée Poussin operator. Summing up the approximating network g is in $\Sigma_r(\sigma)$ for $(12m+1)^d \leq r$ and

$$\|P_m(f) - g\|_\infty \leq cm^{-s}\|f\|_{s,p} \tag{2.4}$$

and using $\|\cdot\|_p \leq \|\cdot\|_\infty 2^{d/p}$ the authors arrive at the bound

$$\|f - g\|_p \leq cr^{-s/d}\|f\|_{s,p}$$

which gives the statement of the proof as $f \in \mathcal{B}_p^s$. □

For the interested reader we mention also [Pin99, Theorem 6.8] which states theorem (2.18) for $t = 1$ and gives a less technical proof.

Next, we turn to the problem of finding a lower bound on the order of approximation in case weights, thresholds and coefficients depend continuously on the target function. Here, a slightly larger lower bound in the order of $r^{-s/d}$ instead of $r^{-s/(d-1)}$ in the previous section holds. The continuity assumption enters in facilitating the appli-

cation of the Borsuk-Ulam Theorem. The following theorem by DeVore et al. [DHM89] relies on the concepts of continuous nonlinear t-width and Bernstein t-width which we briefly describe.

Definition 2.19 (non-linear t-width)**.** *[DHM89]*
For K compact in a linear space X, let $P_t : K \to \mathbb{R}^t$ continuous, $M_t : \mathbb{R}^t \to X$ arbitrary. For each P_t, M_t set

$$E(K; P_t, M_t; X) := \sup_{f \in K} \|f - M_t(P_t(f))\|_X$$

and define the continuous non-linear t-width as

$$h_t(K; X) = \inf_{P_t, M_t} E(K; P_t, M_t; X).$$

Definition 2.20 (Bernstein t-width)**.** *For a subset K of a normed linear space X, X_{t+1} a $t+1$-dimensional subspace of X and B_{d+1} the unit ball therein define the Bernstein t-width as*

$$b_t(K; X) = \sup_{X_{t+1}} \sup\{\lambda \in \mathbb{R} : B_{d+1}(\lambda) \subseteq X\}$$

Theorem 2.21. *[Pin99, Theorem 6.5], [DHM89]*
For each fixed $p \in [1, \infty]$, $s \geq 1$, $d \geq 1$ it holds that

$$h_t(\mathcal{B}_p^s; L_p) = \inf_{P_t, M_t} \sup_{f \in \mathcal{B}_p^s} \|f - M_t(P_t(f))\|_{L_p} \asymp t^{-s/d}$$

Proof outline:
We point out some instructive aspects of the proof of the lower bound. First, $h_t(K; X)$ is bounded from below by $b_t(K; X)$ for general K, X. For $P_t : K \to \mathbb{R}^d$ define

$$\tilde{P}_t(f) = P_t(f) - P_t(-f)$$

which is continuous and odd. Take some radius $\lambda \in \mathbb{R}$ such that $B_{d+1}(\lambda) \subseteq K$. Then $\tilde{P}_t$ is defined on the boundary of $B_{d+1}(\lambda)$, $\partial B_{d+1}(\lambda)$, and by continuity we can conclude from the Borsuk Antipodality Theorem (or Borsuk-Ulam Theorem) that there exists $f_0 \in \partial B_{d+1}(\lambda)$ such that $\tilde{P}_t(-f_0) = \tilde{P}_t(f_0)$ or by oddness of $\tilde{P}_t$ that $P_t(f_0) = P_t(-f_0)$. Thus for arbitrary $M_t : \mathbb{R}^d \to X$ we have

$$2f_0 = [f_0 - M_t(P_t(f_0))] - [-f_0 - M_t(P_t) - f_0))]$$

and

$$\max\{\|f_0 - M_t(P_t(f_0))\|_X, \|-f_0 - M_t(P_t(-f_0))\|_X\} \geq \|f_0\|_X = \lambda.$$

Because $f_0 \in K$

$$\sup_{f \in K} \|f - M_t(P_t(f))\|_X \geq \lambda,$$

passing to the infimum over P_t and M_t we get $h_t(K; X) \geq \lambda$ and thus passing to the supremum over λ we get

$$h_t(K; X) \geq b_t(K; X).$$

The remainder of the proof is then constructive and finally yields

$$h_t(\mathcal{B}_p^s; L_p) \geq b_t(\mathcal{B}_p^s; L_p) \geq Ct^{-s/d}.$$

□

The lower bound we just proved holds for any method of approximating $f \in \mathcal{B}_p^s$ by a member of $\Sigma_r(\sigma)$ where the network parameters depend continuously on f:

Theorem 2.22. *[Pin99, Theorem 6.6]*
Let $Q_r : L_p \to \Sigma_r(\sigma)$ be a an approximating map where the network parameters c_i, θ_i and w_i depend continuously on the target function $f \in L_p$. Then

$$\sup_{f \in \mathcal{B}_p^s} \|f - Q_r(f)\|_p \geq Cr^{-s/d}.$$

Here we point also to the work of Maiorov et al. [MM00] who drop the continuity assumption on the approximation procedure for specific σ. We mention an exemplary result using the sigmoid or logistic function:

Theorem 2.23. *[MM00, Theorem 4-5]*
Let $p \in [1, \infty]$, $s \geq 1$, $d \geq 2$ and σ be the sigmoid or logistic function

$$\sigma(t) = \frac{1}{1 + e^{-t}}.$$

Alternatively, let σ be a piecewise polynomial of bounded degree defined on a finite number of intervals whose union is $\mathbb{R}$. Then

$$E(\mathcal{B}_p^s; \Sigma_r(\sigma); L_p) \geq C(r \log r)^{-s/d}.$$

Generalisations of the aforementioned result can be found in [Pet98] and comments thereon in [Pin99]. However the authors only consider the case $p = 2$. The proof takes an approach similar to (2.18). In particular in [Pin99, Corollary 6.11] and [Pet98,

Corollary 9.2] activation functions of the type

$$\sigma_k(t) = \begin{cases} t^k & \text{if } t \geq 0 \\ 0 & \text{if } t < 0 \end{cases}$$

are considered. Setting $k = 1$ we obtain the following result about approximation in L_2 with the ReLU activation function.

Theorem 2.24. *[Pet98, Corollary 9.2], [Pin99, Corollary 6.11]*
For σ the ReLU function it holds that

$$E(\mathcal{B}_p^s; \Sigma_r(\sigma); L_2) \leq Cr^{-s/d}$$

for $s = 1, \cdots, \frac{d+3}{2}$.

Comment: Note that modern methods for learning neural networks do not in general depend continuously on the target function and hence the applicability of the results in this section is in some sense restricted. Poggio et al. [Pog+17, Theorem 1] point out that the bounds on the order of optimisation are optimal due to results on nonlinear t-width (2.19) by DeVore et al. [DHM89] on $W_p^s([-1,1]^d)$. Throughout this section we have discussed approaches of bounding the degree of approximation when estimating a member of a given Sobolev space by a neural network. We were considering worst case errors. Another angle of attack might be to try to identify sets of functions which are approximated well by networks in $\Sigma_r(\sigma)$. For a treatment of this problem we refer the interested reader in particular to [Mak96], [Bar93].

Conclusion

All the results we discussed in this section feature an upper or lower bound on the order of approximation which is exponential in the input dimension d. This phenomenon is commonly referred to as the **curse of dimensionality**. Considering the error bounds (2.18), (2.22) and switching view point, we observe that for a desired accuracy ϵ we need a number of neurons in the hidden layer that is in the order $\epsilon^{-d/s}$. This number grows exponentially as d grows which poses a great obstacle to high dimensional learning problems.
In the next section, we collect some results that seek to explain why deep neural networks work so much better in practice when it comes to countering the curse of dimensionality.

2.3 Breaking the Curse of Dimensionality with Deep Neural Networks

In the last two sections we have seen:

+ Shallow neural networks are universal approximators.

- Shallow neural networks suffer from the curse of dimensionality.

Because deep neural networks can be viewed as a generalisation of shallow neural networks, naturally also deep neural networks are universal approximators. The accuracy that is theoretically achievable with deep or shallow networks is thus the same. The results discussed so far and the widespread and successful usage of deep networks in practice today naturally raise some questions:

Q: Why are deep neural networks preferred over shallow neural networks in practice, if they are both universal approximators, but the depth of deep nets makes them much harder to train?

Q: Are deep neural networks so successful, because their multi-layer structure helps in combatting the curse of dimensionality?

In the following we seek to formalise these questions and find some answers in the work of Poggio et al. [Pog+17]. We would like to point out that the approach in this section differs somewhat from the approach in the previous section. In the last section we considered worst case approximation errors for a given function (2.2). In this section we take the opposite approach and identify classes of functions for which a good order of approximation is possible. In particular, Poggio et al. [Pog+17] identify functions that facilitate a linear order of approximation by deep neural network which is evidently better than an exponential order (2.25) also known as the curse of dimensionality mentioned in the section (2.2.1).

Key Concepts: Compositional Functions, Hierarchically Local Functions, Directed Acyclic Graphs, Effective Dimensionality

For context we remind the reader of the discussion on the curse of dimensionality in (2.2.1). Poggio et al. [Pog+17, Theorem 1] extend this key result to vector valued functions.

Theorem 2.25 (Curse of Dimensionality). *[Pog+17, Theorem 1]*
To approximate $f : \mathbb{R}^d \to \mathbb{R}^q$ with $f_i : \mathbb{R}^d \to \mathbb{R}$ in $\mathcal{B}_p^s(\mathbb{R}^d)$, $i = 1, \cdots, q$ up to accuracy ϵ in each of the components, a shallow network with d inputs and q outputs needs to have

$$r = \mathcal{O}\left(\epsilon^{-d/s}\right)$$

neurons in the hidden layer.

Poggio et al. [Pog+17] emphasise the importance of formulating error bounds in terms of the L^∞ and not the L^2 norm. This guarantees independence from the underlying distribution of the inputs and limits error propagation from layer to layer.

As in the theorem above it is often assumed that the target function exhibits some degree of smoothness, so that the curse of dimensionality can be to some extent counteracted. Indeed, the theorem above shows that a greater degree of smoothness represented by the parameter s can somewhat balance out a large input dimension d. The question remains whether assuming a large degree of smoothness accurately reflects approximation problems encountered in reality. One wonders whether the exponential order of approximation could be avoided entirely.
Poggio et al. [Pog+17] succeed in doing this by assuming not a large degree of smoothness, but by assuming that the target function is *compositional*. By this they refer to a function which is constructed through composition of a number of constituent functions. An example compositional function $f : \mathbb{R}^4 \to \mathbb{R}$ could be given by

$$f(x_1, x_2, x_3, x_4) = h_2(h_{11}(x_1, x_2), h_{12}(x_3, x_4))$$

where h_2, h_{11}, h_{12} are the *constituent functions*. The following two definitions formalise this idea.

Definition 2.26 ($\mathcal{G}$-function). *[Pog+17, p. 6]*
For $\mathcal{G}$ a directed acyclic graph (DAG) with nodes V define a $\mathcal{G}$-function $f : \mathbb{R}^d \to \mathbb{R}$ with a structure corresponding to $\mathcal{G}$: Every one of the d source nodes of $\mathcal{G}$ represents a one dimensional input of f. Inner nodes of $\mathcal{G}$ represent constituent functions which get one real one-dimensional input from every incoming edge. The outgoing edges feed the one dimensional function value to the next node. There is only one sink node so overall the $\mathcal{G}$-function assumes values in $\mathbb{R}$.

Definition 2.27 (Compositional functions with binary tree structure). *[Pog+17, p. 5]*
Define $\mathcal{B}_p^{s,2}$ to be the class of compositional functions $f : \mathbb{R}^d \to \mathbb{R}$ whose corresponding DAG $\mathcal{G}$ has a binary tree structure. The constituent functions h are in $\mathcal{B}_p^s(\mathbb{R}^2)$.

Returning to the previous example, we refer to a function of the form

$$f(x_1, x_2, x_3, x_4) = h_2(h_{11}(x_1, x_2), h_{12}(x_3, x_4))$$

as a compositional function with a binary tree structure, because each constituent function takes two inputs. We think of the dimension of the constituent function as arbitrary, but fixed (in this case 2), while the overall dimension of the function increases, when more layers of function composition are added.

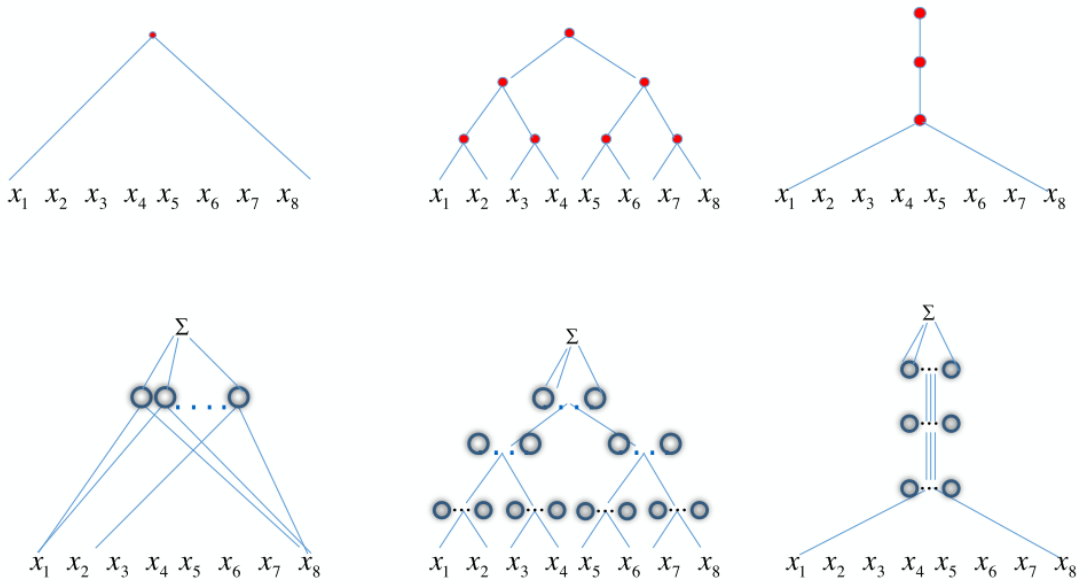

Figure 1: This figure has been taken from [Pog+17, Figure 1]. The graphs in the top row represent $\mathcal{G}$-functions of 8 variables. Each graph on the bottom row shows the optimal neural network architecture approximating the function above.

To illustrate the two previous definitions we consider the example functions represented in figure (1) taken from [Pog+17, Figure 1]. In the top row of the figure a $\mathcal{G}-$function in $\mathbb{R}^8$ is represented by the graph $\mathcal{G}$ describing its structure. The bottom row shows the optimal network architecture (compare (2.29, 2.31)) for approximating the respective function in the top row.
In the first column a generic function of the form $f(x_1, \cdots, x_8)$ (top image) is approximated by a one layer neural network with 8 inputs and r neurons (bottom image). In the second column a function of the form

$$f(x_1, \cdots, x_8) = h_3(h_{21}(h_{11}(x_1, x_2), h_{12}(x_3, x_4)), h_{22}(h_{13}(x_5, x_6), h_{14}(x_7, x_8)))$$

(top image) is approximated by a binary tree network (bottom image). Each red node in the graph of the function corresponds to a set of neurons symbolised by the blue circles

in the bottom image. Finally, in the third column a function of the form $f(x_1, \cdots, x_8) = h_3(h_2(h_1(x_1, \cdots, x_8)))$ (top image) is approximated by a deep neural network with 8 inputs and 3 hidden layers (bottom image).

We point out that the constituent functions making up a compositional function have an input dimensionality which is potentially much smaller than the overall input dimension d. This also holds true for neural networks used in practice. Convolutional Neural Networks (CNNs) [LKF10] for example use convolutional kernels as compositional functions that typically have a size of 3×3 or 5×5 [KSH12]. To formalise this notion of small dimension in the constituent functions, Poggio et al. [Pog+17, Definition 1] define the *effective dimension*.

Definition 2.28 (Effective Dimension)**.** *[Pog+17, Definition 1]*
Let k be the smallest positive integer such that for any $\epsilon > 0$ any function in a class X can be approximated up to accuracy ϵ by a neural network of ϵ^{-k} parameters. Then k is the effective dimension of the function class X.

In the light of the curse of dimensionality (2.2.1) $\mathcal{B}_p^s(\mathbb{R}^d)$ has effective dimension $\frac{d}{s}$.

From (2) we know that we can theoretically achieve the same accuracy with a shallow or a deep network. Crucially however we will see next that the number of parameters necessary for a fixed accuracy is much smaller for deep networks. The exponential dependence on the dimension can be avoided assuming compositionality.

Theorem 2.29 (Linear order of approximation for compositional functions)**.** *[Pog+17, Theorem 2]*
Assume $f \in \mathcal{B}_p^{s,2}$ and $\sigma \in C^\infty$ is not a polynomial. Then accuracy at least ϵ in the supremum norm can be achieved with a deep network with the same compositional structure and

$$r = \mathcal{O}\Big((d-1)\epsilon^{-2/s}\Big)$$

neurons.

Proof:
Because any constituent function of the compositional function f is in $\mathcal{B}_p^s(\mathbb{R}^2)$ we can apply (2.25) to deduce that it can be approximated by an element of $\Sigma_r(\sigma)$ with accuracy of the order $r^{-s/2}$. Seeing as f is in $\mathcal{B}_p^{s,2}$, each of the constituent functions has to be Lipschitz continuous. Using the approximators of the constituent functions in conjunction with the Minkowski inequality, lets us approximate f. For example for

$$f(x_1, x_2, x_3, x_4) = h_2(h_{11}(x_1, x_2), h_{12}(x_3, x_4))$$

and approximators to level ϵ p_2, p_{11}, p_{12} of h_2, h_{11}, h_{12} we can calculate

$$\begin{aligned}
&\|h_2(h_{11}, h_{12}) - p_2(p_{11}, p_{12})\| \\
&= \|h_2(h_{11}, h_{12}) - h_2(p_{11}, p_{12}) + h_2(p_{11}, p_{12}) - p_2(p_{11}, p_{12})\| \\
&\leq \|h_2(h_{11}, h_{12}) - h_2(p_{11}, p_{12})\| + \|h_2(p_{11}, p_{12}) - p_2(h_{11}, p_{12})\| \\
&\leq c\epsilon.
\end{aligned}$$

An analogous reasoning works for higher dimensional cases and is not stated explicitly for ease of notation. The fact that there are $d-1$ nodes then yields the result. □

The effective dimension introduced before offers a convenient way of comparing function classes. From the previous two theorems we see that $\mathcal{B}_p^s$ and $\mathcal{B}_p^{s,2}$ have effective dimensionality $\frac{d}{s}$ and $\frac{2}{s}$ respectively.

Using the same line of proof as theorem (2.29) in combination with [Bac17] we can extend this result to the non differentiable ReLU activation function. Importantly again, approximation is done in the supremum norm which unlike the L_2 norm limits error propagation through the layers.

Theorem 2.30 (Order of approximation for Lipschitz continuous functions)**.** *[Pog+17, Theorem 4]*
Choose the ReLU function as activation function. Then any $f : \mathbb{R}^d \to \mathbb{R}$ which is Lipschitz continuous with Lipschitz constant L can be approximated in the supremum norm up to order ϵ by a shallow network of complexity

$$r_{shallow} = \mathcal{O}\left(\left(\frac{\epsilon}{L}\right)^{-d}\right)$$

or a deep network of complexity

$$r_{deep} = \mathcal{O}\left(\left(d-1\right)\left(\frac{\epsilon}{L}\right)^{-2}\right).$$

Using the representation of compositional functions by DAGs we can generalise the previous theorem and formally state how deep neural networks avoid the curse of dimensionality.

Theorem 2.31 (Breaking the Curse of Dimensionality). *[Pog+17, Theorem 3]*
Consider a compositional $\mathcal{G}$-function $f : \mathbb{R}^d \to \mathbb{R}$ (see definition (2.26)) corresponding to a DAG $\mathcal{G}$ with nodes V where each constituent function represented by node $v \in V$ is in $\mathcal{B}_p^{s_v}(\mathbb{R}^{d_v})$ for d_v the number of incoming edges of v. Then for $\sigma \in C^\infty$, f can be approximated up to order ϵ by a shallow network with complexity

$$r_{shallow} = \mathcal{O}\left(\epsilon^{-d/\min_{v\in V} s_v}\right)$$

or by a deep network represented by $\mathcal{G}$ also with complexity

$$r_{deep} = \mathcal{O}\Big(\sum_{v\in V} \epsilon^{-d_v/s_v}\Big).$$

The theorem shows that if the constituent functions have an effective dimensionality which is smaller than the effective dimensionality of the shallow network

$$\frac{d}{\min_{v\in V} s_v},$$

then in this case deep networks are able to avoid the curse of dimensionality. Note that in this theorem there is an exact correspondence between the architecture of the deep neural network and the compositional structure of the target function (visualised also in (1)). Of course this assumption is utopian in practice. Poggio et al. [Pog+17] point out that the same favourable order of approximation with a deep network holds if the graph representing the neural network structure contains a subgraph which describes the compositional structure. A more elaborate network architecture hence increases the likelihood that that the structure of the target function is in a sense matched by the network. This however comes at a cost of having a larger number of parameters to optimise.

By our results on the order of approximation (2.2) we know that upper and lower approximation bounds for shallow networks are both exponential. In the next theorem, Poggio et al. [Pog+17] also give a concrete example of a gap between the complexity of a shallow network and a deep network for a C^∞ activation function. This demonstrates how deep neural networks offer an advantage in practice.

Theorem 2.32. *[Pog+17, Theorem 5]*
Let the activation function σ be in C^∞ and not a polynomial. Assume the class of target functions consists of functions of the form $f = f_2 \circ f_1$ for $f_1 : \mathbb{R}^d \to \mathbb{R}$ and $f_2 : \mathbb{R} \to \mathbb{R}$ with $\sum_{|\alpha|\leq 4} \|\partial^\alpha f_1\|_\infty \leq 1$ and $\|f_2^{(4)}\|_\infty \leq 1$. For any such f the optimal approximation error in the supremum norm achievable by a shallow network with one layer of r neurons is at least in the order of

$$\lceil 2^{-r/(d-1)} \rceil$$

and for deep networks is at most in the order

$$r^{-4/d}.$$

The asymptotic constants possibly depend d.

For the construction of the example see [Pog+17, Theorem 5]. Here the network architecture matches the structure of f exactly.

Insights:

1. Poggio et al. [Pog+17] point out that an optimal deep network architecture mirrors the compositional structure of the function to be approximated. However, a network whose graph representation contains as a subgraph the graph representation of the target function also take advantage of compositionality. As the exact compositional structure of the target function is in practice unknown the authors explain the apparent success of densely connected networks by network graph representations containing function graph representations of interest. Thus the networks have enough representative power to make use of compositionality while keeping a sub-exponential number of parameters. The authors highlight this trade-off situation between minimising the number of parameters for ease of optimisation and maintaining the capability to approximate well.

2. The proof of theorem (2.29) exploits only the locality of the constituent functions. No weight sharing (constituent functions at the same level being the same e.g. in the convolutional kernels of CNNs [LKF10]) is used. Poggio et al. [Pog+17] conduct experiments with CNNs with and without weight sharing (same or different filter weights at different locations). The numerical results lead the authors to believe that weight sharing in convolutional layers decreases the number of network parameters in practice, but plays only a subordinate role in facilitating a good order of approximation.

3. Poggio et al. [Pog+17] also link their results to multi-class learning, an approach in which almost the same model is trained for different task with a common architecture and weights, but only a differing final layer. A stronger generalisation is achieved as universal constituent functions have to be discovered. Compositionality in the constituent functions helps decrease complexity [Mau06].

4. Compositional functions do not in general have favourable approximation properties. By pointing out that linear operators can be decomposed into constituent calculations that are spread over different network layers without any computational advantage, Poggio et al. [Pog+17] underpin their claim that compositionality is not sufficient for efficient approximation. They claim that sub-classes of compositional functions like the hierarchically local functions are necessary to avoid the curse of dimensionality.

5. In the context of [Pog+17], one might ask how relevant their results are seeing as compositionality seems quite restrictive as an assumption. The authors conjecture that compositional functions are common in nature and importantly describe the structure of the brain in particular the visual cortex. They thus argue that compositional neural networks mirror the way humans perceive for instance images.

Conclusion

In chapter (2) we have thoroughly analysed the representative power of neural networks. We have considered 3 key aspects.

1. **Density (2.1):** We have seen that shallow neural networks and hence general neural networks are universal approximators for different function classes under the assumption of a variety of activation functions.

2. **Order of Approximation (2.2):** We have considered upper and lower bounds on the order of approximation. We have seen when approximation problems suffer from the **curse of dimensionality.**

3. **Breaking the Curse of Dimensionality with Deep Networks (2.3):** We have seen how deep neural networks can circumvent the curse of dimensionality by exploiting compositionality of the target functions. Deep networks have the representative power to reflect the structure of the target function in their own structure. Locality of constituent functions is the key rather than weight sharing.

3 Stability

The aim of this chapter is to highlight mathematical analysis that has been done on the stability of neural networks. First we outline some of the recent work on the phenomenon of adversarial examples (3.1) and their surprising robustness properties. The existence of adversarial examples highlights how unstable neural networks are in general.
The main focus of the chapter will be on the extensive work of Mallat et al. [Mal10], [BM10], [Mal12a], [BM13], [Mal16], [AM18b] on scattering networks (3.2). Mallat et al. have developed the first rigorous framework for analysing the stability of neural networks. They approach stability from a signal processing viewpoint and construct a scattering network using wavelets and non-linearities which helps to shed light on the inner workings of convolutional neural networks (CNNs) [LKF10]. In particular, it reveals how the specific architecture of neural networks is necessary for their success.

3.1 Adversarial Examples

Adversarial examples are generally understood to be input samples that differ only slightly from the true input samples, but are still misclassified quite spectacularly by the neural network. In the context of object classification in images for example, it is possible to perturb input images, such that the perturbation remains imperceptible to the human eye, but the label assigned by the network changes completely (see (2)).

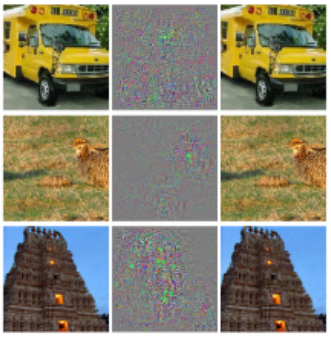

Figure 2: This image has been taken from [Sze+13, p. 6] and shows adversarial examples for AlexNet [KSH12]. The left column shows the true training images. The middle column shows a magnified version of the difference between the original image and the adversarial example. The right column shows the adversarial example corresponding to the true input which is misclassified.

Of course, this casts major doubt on the belief that neural networks are in some sense able to generalise or learn underlying concepts. While adversarial examples are un-

likely to occur naturally [Sze+13], [Dhi+18], their existence demonstrates that good test set performance does not mean models have really understood concepts in the way humans would. Szegedy et al. [Sze+13] have been the first to show instability of neural networks by constructing adversarial examples through corruption of training samples with additive noise. Much research has since gone into the topic [GSS14], [KGB16], [Hua+17], [Mad+17], [MDFF16], [CW17] . For recent surveys see [Pap+16], [AM18a]. Adversarial examples are not only of theoretical interest. They can also be fabricated in real world settings where there is no access to the network [AM18a, Section 4]. Hence research into adversarial attack and defence strategies is very relevant today, especially in safety critical application areas.

Szegedy et al. [Sze+13] show instability of neural networks by constructing adversarial examples (see for example (2)) through corruption of training samples with additive noise. In the image classification setting a neural network computes an input-output mapping $\mathcal{K} : \mathbb{R}^d \to \{1, \cdots, C\}$ mapping a pixel vector to one of C integer classes. Specifically adversarial examples are found by minimising the l^2 norm:

$$\begin{aligned} &\min \|r\|_2 \\ &\mathcal{K}(x+r) = l \\ &x + r \in [0,1]^d \end{aligned}$$

where $l \in [C]$ is some false label for input x. The non-convex problem is approximately solved using box-constrained L-BFGS. The resulting deformed images are indistinguishable from the originals to the human eye.

Goodfellow et al. [GSS14] linearise the network cost to find adversarial examples. Let $J(\theta, x, y)$ be the loss of a neural network with parameters θ evaluated at a pair of input and output (x, y). The original problem is formulated as

$$\Delta x = \underset{\|r\|_\infty \leq \lambda}{\arg\max}\, J(\theta, x+r, y) \tag{3.1}$$

where the perturbation r represents a step in the direction of increasing cross-entropy loss. The first order approximation to this non-convex problem (3.1) is given by [GSS14], [Mad+17]

$$\Delta x = \underset{\|r\|_\infty \leq \lambda}{\arg\max}\, J(\theta, x, y) + r^T \nabla_x J(\theta, x, y) = \underset{\|r\|_\infty \leq \lambda}{\arg\max}\, r^T \nabla_x J(\theta, x, y). \tag{3.2}$$

The resulting maximal perturbation is

$$r = \lambda \text{sign}(\nabla_x J(\theta, x, y)).$$

This method of generating adversarial perturbations is referred to as **Fast Gradient Sign Method (FGSM)** in [GSS14]. The gradient is computed with backpropagation. This requires access to the model which is unrealistic in real world settings. Dhillon et al. [Dhi+18] (see 3.1.2) use Monte Carlo sampling instead. The most recent past has seen a lot of research into adversarial attacks [AM18a, Section 3]. At this point we restrict ourselves to comparing the adversarial attack strategy presented so far to the more recent one by Alaifari et al. [AAG18].

Iterative Construction of Adversarial Examples with Gradient Descent

Instead of using an additive perturbation, Alaifari et al. [AAG18] take inspiration from the Deepfool method by Moosavi et al. [MDFF16] and iteratively deform an input image using gradient descent steps.

Key Concepts: Gradient Descent, Discretisation

Alaifari et al. [AAG18] model a deformation of an input image $f : [0,1]^2 \to \mathbb{R}^c$ with respect to vector field $\tau : [0,1]^2 \to \mathbb{R}^2$ as

$$L_\tau f(x) = f(x - \tau(x)) \quad \forall x \in [0,1]^2$$

where $f : [0,1]^2 \to \mathbb{R}^c$ ($c = 1$ or $c = 3$ for greyscale or colour images) is set to zero beyond $[0,1]^2$. In particular, translations are deformations with $\tau = c \in \mathbb{R}$. In general, the deformation $r = f - L_\tau f$ might be unbounded in the L^p norm even if the deformation is not visible for the human eye [AAG18]. This sheds doubt on whether the L^p norm on additive perturbations in [Sze+13], [GSS14] is an adequate design choice for measuring similarity between images as a human might perceive it.

Alaifari et al. [AAG18] proceed in the discrete setting and model an image $y \in \mathbb{R}^c$ to be function values of $f = (f_1, \cdots, f_c) : [0,1]^2 \to \mathbb{R}^c$ evaluated on the pixel grid

$$\{1/(W+1), \cdots, W/(W+1)\}^2 \subseteq [0,1]^2 \quad \text{i.e.} \quad y_{i,s,t} = f_i(s/(W+1), t/(W+1))$$

for $i \in [c]; s, t \in [W]$. Also $\tau : [0,1]^2 \to \mathbb{R}^2$ is evaluated on the same grid. Only vector

fields τ that map grid points again onto the grid are considered:

$$T := \left\{ \tau : \{1, \cdots, W\}^2 \to \mathbb{R}^2 \text{ s.t. } \tau(s,t) + (s,t) \in [1, W]^2 \quad \forall s, t \in \{1, \cdots, W\} \right\}.$$

The deformation of y with respect to τ is defined as

$$y_i^\tau = f \circ ((\mathrm{Id} + \tau)/(W+1)) \quad \text{or} \quad y_{i,s,t}^\tau = f_i \left(\frac{(s,t) + \tau(s,t)}{W+1} \right)$$

for all $i \in [c]$ and $s, t \in [W]$. The size of the deformation is measured as

$$\|\tau\|_T := \max_{s,t \in [W]} \|\tau(s,t)\|_2 .$$

For some space of images Y let $\mathcal{K} : Y \to \{1, \cdots, C\}$ be a classifier assigning to images one of C classes, let $F = (F_1, \cdots, F_C) : Y \to \mathbb{R}^C$ be the score functions such that

$$\mathcal{K}(y) = \arg\max_{k=1,\cdots,C} F_k(y).$$

Fix $y \in Y$ to be some image, $f : [0,1]^2 \to \mathbb{R}^c$ to be the interpolation obtained when taking y to be the function values of f on the pixel grid, $l = \mathcal{K}(y)$ to be the true label, $k \in [C]$ some target label and $F^* = F_k - F_l$. If y is not located on a decision boundary then $F^*(y) < 0$. Define $g : T \to \mathbb{R}$ to be the map $\tau \mapsto F^*(y^\tau)$. By construction,

$$g(0) = F^*(y) < 0.$$

Alaifari et al. [AAG18] seek to find a small $\tau \in T$ such that $g(\tau) \geq 0$. Using the Taylor expansion we can approximate g in a neighbourhood of 0 as

$$g(\tau) \approx g(0) + (D_0 g)\tau. \tag{3.3}$$

where $D_0 g$ is the derivative of g evaluated at zero. Now if τ were to be small in norm and such that $(D_0 g)\tau = -g(0)$, then y^τ would be approximately on a decision boundary between labels l and k. Least squares are employed to find a solution to

$$(D_0 g)\tau = -g(0)$$

which will depend on the image y. Thus gradient descent steps are taken using the deformed image of the previous iteration at each step until an adversarial example is found. For a rigorous derivation of the model we refer to the to date unpublished

[Ala18].
For optimal approximation properties in (3.3) a target label is chosen at every step such that τ is minimal in norm. However, targeted attacks are also possible that result in a chosen false label. Interestingly, Alaifari et al. [AAG18] observe that the deforming vector field τ tends to focus on edges in images. The adversarial attack developed in [AAG18] very successfully fools a CNN [LKF10] on MNIST [LeC98] and an Inception-v3 and a ResNet-101 [Sze+16], [He+16] on ImageNet [Rus+15]. [AAG18] report that deformations tend to be larger in $\|\cdot\|_T$ than common perturbations in the $\|\cdot\|_\infty$ norm, though they are imperceptible to the human eye.

Robustness of Adversarial Examples

So far we have seen how adversarial examples reveal a large degree of instability in neural networks. We will briefly point out how universal they are to different experimental setups.
Adversarial examples exhibit a notion of robustness in the sense that they fool networks independent of parameters, architectures and training sets [Sze+13] and make them report incorrect labels with high confidence [GSS14]. Autoencoder based models are observed to be more stable, but not completely resilient [Sze+13]. All of this indicates that adversarial examples are not merely a phenomenon related to overfitting, but in some sense intrinsic to neural networks themselves. The fact that there is no lack of supply of adversarial examples and that they are easily constructed, but do not commonly occur in practice raises an important question. How are adversarial examples distributed throughout the input space?

3.1.1 Distribution of Adversarial Examples in Input Space

Szegedy et al and Dhillon et al. [Sze+13], [Dhi+18] point out that training on randomly deformed examples or sampling around true training images is of limited use and seldom leads to adversarial examples. Szegedy et al. [Sze+13] hypothesise that adversarial examples have low probability and thus hardly occur in the test data, but are dense in the test set. For analogy it is often referred to the **rational numbers** that are sparsely distributed, but dense within the reals. This interpretation would explain good generalisation on the one hand, but vulnerability to adversarial examples on the other hand.
Goodfellow et al. [GSS14] argue that neural networks are "too linear" by design and hence vulnerable to adversarial examples. They point out that linear models are also sensitive to adversarial examples if the input dimension is large enough. Minor per-

turbations of a large number of entries can add up to a large perturbation of the inner product with the weights and thus to a large perturbation of the activation. Goodfellow et al. [GSS14] conjecture that popular networks like Long-Short Term Memory (LSTM) [HS97], ReLU networks [JKL+09], [GBB11] and maxout networks [Goo+13] are made to behave more linearly for easier parameter tuning and thus overwhelmingly misclassify adversarial examples with high confidence. Goodfellow et all. [GSS14] observe that a radial basis function (RBF) model offers more robustness at the cost of losing accuracy and reports very low confidence on false predictions. They claim that this is because RBF models behave less linearly.
Like Szegedy et al. [Sze+13], Goodfellow et al. [GSS14] find that adversarial examples generalise to different models and training sets. Surprisingly, often even the same false label is assigned by different models. This provides counter-evidence to the interpretation that adversarial examples behave like the rationals within the reals. It indicates that adversarial examples do not occur isolated in space at specific locations. Goodfellow et al. [GSS14] observe that it is enough for the perturbation to have a positive inner product with the gradient of the loss. Hence they deduce that adversarial examples make up connected regions, not isolated pockets and that it is the direction rather than location of a perturbation which is important. They see this also as an explanation of the ubiquity of adversarial examples and transferability of adversarial perturbations between data sets and models. For a collection also on other hypotheses on the existence of adversarial examples see [AM18a, Section 5].

3.1.2 Regularisation and Adversarial Training

The fact that adversarial examples can fool neural networks so successfully obviously calls for a defence strategy. For a recent survey on defence strategies see [AM18a, Section 6]. We only take a brief look at this point at an early regularisation method for better robustness against adversarial examples [Sze+13] and at a defence strategy inspired by game theory [Dhi+18].

Szegedy et al. [Sze+13] suggest the operator norm of network layers as a measure of the instability of the layers and deduce from that a method of regularisation. Representing the output of a network composed of L layers by $\phi(x)$, so

$$\phi(x) = \phi_L(\phi_{L-1}(\cdots \phi_1(x; W_1); W_2) \cdots ; W_L)$$

the authors consider then the Lipschitz constant for each of the layers i.e. the minimal

$L_k > 0$ s.t.

$$\|\phi_k(x; W_k) - \phi_k(x + r; W_k)\| \leq L_k \|r\|.$$

For a fully connected layer $\phi_k(x; W_k) = \sigma(W_k x + b_k)$ and assuming that σ is contractive (e.g. a ReLU function) we obtain

$$\|\phi_k(x; W_k) - \phi_k(x + r; W_k)\| \leq \|W_k\| \|r\|.$$

Hence, $L_k \leq \|W_k\|$. A conservative estimate for the operator norm of a fully connected layer is hence the spectral norm of the weight matrix, $\|W_k\|$. Szegedy et al. [Sze+13] also give an estimate for the operator norm of a convolutional layer.
Szegedy et al. [Sze+13] note that large bounds on the operator norm do not prove the existence of adversarial examples. However small bounds enforce their non-existence. Thus incorporating a regularisation strategy helps to guarantee small Lipschitz constants and hence lower the generalisation error. Looking at Lipschitz bounds of ImageNet [KSH12], Szegedy et al. [Sze+13] demonstrate that instabilities can already show up in the first layer.

In accordance with their hypothesis on the linear behaviour of networks (3.1.1), Goodfellow et al. [GSS14] remark that linear models cannot be constant near input samples and assign different labels to different inputs at the same time. They thus justify tweaking the training rather than the architecture of existing models for better robustness. They introduce the following loss:

$$\tilde{J}(\theta, x, y) = \alpha J(\theta, x, y) + (1 - \alpha) J(\theta, x + \epsilon \operatorname{sign}(\nabla_x J(\theta, x, y)), y) \tag{3.4}$$

for $\alpha \in (0, 1)$. Training based on this loss works as a defence mechanism against adversary examples, but does not offer complete immunity. Error rates on adversarial examples that were generated from the same model are now lower, but still in the double digits. Adversarial training results in much more localised and interpretable weights. Goodfellow et al. [GSS14] observe the regularisation effect to be superior to the one of dropout.
Goodfellow [GSS14] point out that adversarial training is only effective when the network has enough representative power to learn robustness to adversarial examples. In other words, universal approximation results (2) have to hold. This explains why perturbing the last hidden layer is ineffective in practice, as the final (linear) layer is not a universal approximator.

A game theoretic perspective on Adversarial Training

Adversarial training has a natural link to game theory. Indeed, a setting in which a neural network tries to correctly classify the input and an adversary tries to perturb the input to trick the network is a classic zero-sum game. In this short section we explore which insights we can draw from game theory, when we apply it to adversarial training.

Key Concepts: Minimax Zero-Sum Game, Mixed Strategy

Dhillon et al. [Dhi+18] consider the optimisation problem (3.1)

$$\Delta x = \arg\max_{\|r\|_\infty \leq \lambda} J(\theta, x + r, y) \tag{3.5}$$

which leads to the Fast Gradient Sign Method (FGSM) in [GSS14] (3.2) and cast it into the framework of game theory:

$$\pi^*, \rho^* := \arg\min_{\pi} \arg\max_{\rho} \mathbb{E}_{p\sim\pi, r\sim\rho} \left[J(M_p(\theta), x + r, y) \right]. \tag{3.6}$$

Here p and r are instances of the defensive, respectively attack strategy that are given by the distributions π and ρ and $\|r\|_\infty \leq \lambda$. The optimisation problem (3.6) can thus be seen as a **minimax zero sum game** between the adversary and the network. The adversary tries to perturb the input for maximal loss, while the defender tries to adapt the network parameters to keep the loss minimal. The optimal pair (π^*, ρ^*) is a Nash equilibrium, the optimal strategies are in general randomised and referred to as mixed strategies.
The defender uses **Stochastic Activation Pruning (SAP)** as its strategy and picks the weights $M_p(\theta)$ by dropping activations in each layer on the forward pass with probability proportional to their absolute value. The remaining activations are rescaled to compensate. SAP is applied post-hoc to the pretrained model. Empirically its performance is comparable to that of the unpruned model if the number of weights sampled to be retained is sufficiently large. For less samples accuracy suffers, but the model is stochastic and thus more flexible to evade the adversary's attacks. SAP significantly improves error rates on adversarial examples generated using FGSM. If the adversary knew that the defence strategy was SAP, it could combat SAP by finding

$$\rho^* = \arg\max_{\rho} \mathbb{E}_{p\sim\pi, r\sim\rho} \left[J(M_p(\theta), x + r, y) \right].$$

Using again a linearisation (3.1) of the loss as in [GSS14] the optimal perturbation is given by

$$r = \lambda \text{sign}(\mathbb{E}_{p\sim\pi}\left[J(M_p(\theta), x, y)\right]).$$

The expectation is approximated by Monte Carlo sampling, so no access to the model is needed.

Conclusion

In this section we have seen that neural networks can be fooled quite easily with adversarial examples. We have seen two example methods for constructing them, the original method [Sze+13] which uses additive perturbations and the more recent method which models perturbations with vector fields [AAG18]. We have seen that adversarial examples are easily transferable between different models and contrasted some different hypotheses on the distribution of adversarial examples in the input space. In [Sze+13] and [Dhi+18] we have looked at a regularisation method and a defence strategy against adversarial attacks.
Adversarial example show how unstable neural networks are in general. We will now reach the core of this chapter on the scattering networks introduced by Mallat et al. which show in which ways networks can be stable.

3.2 Scattering Networks

Thinking of neural network use cases like object classification, it is natural to assume that class labels do not change when objects are translated, scaled or slightly deformed within an image. In supervised learning, the task is to learn a function $f(x)$ from a sample set $\{(x_i, f(x_i)) : i \in [N]\}$. As we have seen in section (2.2), this problem suffers from the curse of dimensionality as dimension d of the x_i grows. In [Mal12a], [BM13] and [Mal16], Mallat et al. are aiming at finding an embedding function Φ that is applied to f. It should be translation invariant and stable to deformations. For stability they enforce Φ to be contractive in the L^2 norm and Lipschitz continuous with respect to any deformation L_τ.

Definition 3.1 (Lipschitz continuity for translation invariant operators)**.** *[Mal12a, p. 2] Let Φ be a translation invariant operator. If for compact $\Omega \subset \mathbb{R}^d$, there exists $C > 0$ with*

$$\|\Phi(L_\tau f) - \Phi(f)\| \leq C\|f\|(\sup_{x\in\mathbb{R}^d} |\nabla\tau(x)| + \sup_{x\in\mathbb{R}^d} |H\tau(x)|)$$

for all $f \in L^2(\mathbb{R}^d)$ with support in Ω and for all $\tau \in C^2(\mathbb{R}^d)$ where $L_\tau f(x) = f(x - \tau(x))$ is the deformed signal, then Φ is said to be Lipschitz continuous to C^2 diffeomorphisms.

In particular for $\tau(x) = c \in \mathbb{R}$, this implies global translation invariance of Φ [Mal12a]. See (3.2.3.1) for the connection between the embedding Φ to Convolutional Neural Networks (CNNs) [LKF10].
Due to Lipschitz continuity in the embedding or scattering domain the deformation has almost become **linearised**. So after applying Φ it is possible to use a linear classifier despite the deformation τ being potentially very non-linear. Stability in the scattering domain is preserved and the distance to the linear spaces obtained through the linear classifier will be discriminative for classification. In this sense linear spaces offer a characterisation of the class in classification, but also of the within class deformations. Bruna et al. [BM13] bridge the gap to neural neworks by saying that, because the final layer in the classification procedure is linear, it is necessary to have a non-linear operator which transforms signal variablility in such a way that a linear classifier can capture it. They **describe stability as** an operator that maps a non-linear deformation to a linear movement in a linear space that can be captured by a linear classifier.
This section first presents background information on wavelets (3.2.1) and constructs the scattering transform (3.2.2) which will take the role of the embedding Φ. The remainder of the section is dedicated to proving its favourable properties (3.2.3), (3.2.4), (3.2.5) which we motivated above.

3.2.1 Background on Wavelets and the Wavelet Transform

The translation invariant and stable embedding Φ that Mallat [Mal12a] seeks to find is constructed using wavelets as building blocks. Φ is closely related also to the Littlewood-Paley wavelet transform. Indeed, wavelets and wavelet transforms are of central importance to this section. We briefly provide the reader with some background information. In particular, we clarify some of the signal processing concepts and language used in this section.

The Fourier transform $\hat{f}(\omega)$ of a function $f(x)$ is given by [Mal99, p. 2]

$$\hat{f}(\omega) = \int_{\mathbb{R}} f(x) e^{-i\omega x} dx. \tag{3.7}$$

The back transformation from $\hat{f}(\omega)$ to $f(x)$ is given by

$$f(x) = \frac{1}{2\pi} \int_{\mathbb{R}} \hat{f}(\omega) e^{i\omega x} d\omega. \tag{3.8}$$

Hence the Fourier transform $\hat{f}$ offers a representation of the function or *signal* f as a sum of waves $e^{i\omega x}$. When f is supported on some compact set K, then the relationship between f and $\hat{f}$ stands for a representation of f in the orthonormal Fourier basis of $L^2(K)$, $\{e^{i2\pi mx}\}_{m\in\mathbb{Z}}$ [Mal99, p. 2]. When referring to f or $\hat{f}$, we often speak of the *spatial domain/time domain* or the *frequency domain/Fourier domain* respectively. With the *energy* of a signal f we mean its squared L^2 norm: $\|f\|_2^2$.
The Fourier transform is built using the waveform $e^{-i\omega t}$ which is supported on all of $\mathbb{R}$ [Mal99, p. 2]. Because of this, $\hat{f}$ is dependent on all values that f takes on the real line. It is thus dependent on the global behaviour of f. Local behaviour of f is not reflected well in the behaviour of $\hat{f}$.
The idea has hence been proposed to replace $e^{-i\omega t}$ by a localised waveform with bounded support. This was first done by Haar [Haa10] who introduced the following wavelet:

$$\psi(t) = \begin{cases} 1 & 0 \leq t < 1/2 \\ -1 & 1/2 \leq x < 1 \\ 0 & \text{otherwise.} \end{cases}$$

The wavelet ψ serves as the basic building block for constructing a *wavelet basis* for $L^2(\mathbb{R})$. In this context ψ is referred to as the *mother wavelet*. The basis elements are given by translation and dilations of ψ. Many different types of wavelets exist today with different properties. One example is the Meyer wavelet [Mey86] (see figure (3)).

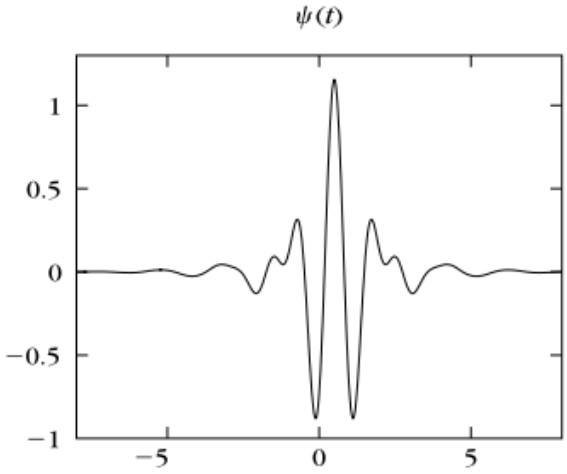

Figure 3: This figure has been taken from [Mal99, p. 291] and depicts a Meyer wavelet [Mey86].

3.2.1.1 The Littlewood-Paley Wavelet Transform

The Wavelet transform is constructed from a *mother wavelet* $\psi \in L^2(\mathbb{R}^d)$ [Fra+91], [Mey95] which has bounded support and zero average

$$\int \psi(x)dx = 0.$$

For the wavelet transform, the mother wavelet is then scaled and rotated according to a scale sequence $\{a^j\}_{j\in\mathbb{Z}}$ for $a > 1$ and a finite rotation group G. For images a is chosen to be 2 [BM13], [BM10]. The so-called child wavelets are derived from the mother wavelet through scaling and rotation and make up the *wavelet family*. They are given by

$$\psi_{2^j r}(x) = 2^{dj}\psi(2^j r^{-1}x)$$

where $r \in G$. For ease of notation it is also sometimes written $\lambda = 2^j r \in 2^{\mathbb{Z}} \times G$ and $|\lambda| = 2^j$. 2^j is referred to as the *scale* in this context. The child wavelets also average to zero: $\int \psi_{2^j r}(x)dx = 0$. The Fourier transform of the mother and child wavelets are related as

$$\hat{\psi}_{2^j r}(\omega) = \hat{\psi}(2^{-j} r^{-1}\omega).$$

As a mother wavelet, we use a complex wavelet of the form

$$\psi(x) = e^{i\eta x}\theta(x)$$

where $\hat{\theta}(\omega)$ is a real function of the complex frequency ω. Hence by properties of the Fourier transform

$$\hat{\psi}(\omega) = \hat{\theta}(\omega - \eta).$$

$\hat{\theta}$ is supported on a ball in the complex frequency plane which is centred at zero. Hence $\hat{\psi}$ is centred at η. For the child wavelets it holds that $\hat{\psi}_\lambda = \hat{\theta}(\lambda^{-1}\omega - \eta)$. We observe that for increasing j the child wavelet becomes increasingly localised in the time domain and covers a ball of larger radius in the complex frequency domain. To visualise this consider figure (4). Here we can see members of a wavelet family visualised in the frequency domain. The circles in red, purple, blue and yellow correspond each to a different scale 2^j. All circles of the same colour, red say, represent child wavelets constructed with the same scale 2^j, but with a different member r of the rotation group G for the rotating. The wavelet family covers the entire frequency plane except for a small ball centered at the origin depicted in green.

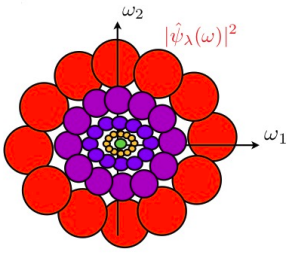

Figure 4: This figure has been taken from [Mal12b]. It depicts a covering of the complex frequency plane by a family of wavelets and an averaging filter at the origin (in green).

The Littlewood-Paley wavelet transform [Fra+91], [Mey95] is a representation of the function f. Specifically, it computes convolutions

$$\forall x \in \mathbb{R}^d, \quad W[\lambda]f(x) = f * \psi_\lambda(x) = \int f(u)\psi_\lambda(x-u)du$$

for different values of λ. As we have seen before, the various ψ_λ are supported on some ball in the frequency plane depending on their scale and rotation. Through the convolution they hence filter the signal f at different scales and in different directions. Because the mother wavelet is centred at a frequency η in the frequency plane, scaling and rotating it about the origin results in a covering of the frequency plane except for the low frequencies at the origin. For a mother wavelet of scale 2^J we only capture frequencies of modulus $|\lambda| = 2^j > 2^{-J}$.
To capture also smaller frequencies the signal f is averaged using convolution with a

low-pass filter ϕ_{2^J}:

$$A_J f = f * \phi_{2^J} \quad \text{with} \quad \phi_{2^J}(x) = 2^{-dJ}\phi(2^{-J}x) \text{ and } \int \phi(x)dx = 1.$$

Here ϕ is taken to be a Gaussian [BM13] and 2^J is the local frequency range over which the averaging is done. A low-pass filter keeps only frequency information at frequencies which are smaller than a certain threshold. In the spatial domain all variability at scales smaller than 2^J are erased by ϕ_{2^J}, variability at larger scales is kept. Hypothetically, when the window of the averaging ϕ_{2^J} increases and the averaging becomes global, complete translation invariance is obtained.
The **wavelet transform** is then a function representation given by

$$W_J f := \left\{ A_J f, (W[\lambda]f)_{\lambda \in \Lambda_J} \right\}$$

where indices are given by

$$\Lambda_J = \{\lambda = 2^j r : r \in G^+, 2^j > 2^{-J}\}.$$

Contrary to a basis representation this signal representation is characterised by great redundancy. The norm on W_J is defined as

$$\|W_J f\|^2 = \|A_J f\|^2 + \sum_{\lambda \in \Lambda_J} \|W[\lambda]f\|^2.$$

For ease of notation in all of the following we will denote the norm of a function g in a Hilbert space by $\|g\|$, unless otherwise stated. Note that for real f rotation by r or $-r$ of the wavelet leads to the same result. Thus we quotient out $\{-1, 1\}$ of G and consider only the resulting G^+ as our rotation group.

The following proposition gives a necessary and sufficient condition for W_J to be unitary.

Definition 3.2 (Unitary operator)**.** *[Mal12a, p. 6]*
A unitary operator is a linear operator U from a normed space X into a normed space Y which preserves the norm i.e.

$$\|Ux\|_Y = \|x\|_X \quad \forall x \in X.$$

Proposition 3.3 (Unitary wavelet transform). *[Mal12a, Proposition 2.1], [Fra+91]*
For any $J \in \mathbb{Z}$ or $J = \infty$, W_J is a unitary operator on $L^2(\mathbb{R}^d)$ for functions taking real or complex values iff at almost all points $\omega \in \mathbb{R}^d$

$$\beta \sum_{j=-\infty}^{\infty} \sum_{r \in G} |\hat{\psi}(2^{-j} r^{-1} \omega)|^2 = 1 \text{ and } |\hat{\phi}(\omega)|^2 = \beta \sum_{j=-\infty}^{0} \sum_{r \in G} |\hat{\psi}(2^{-j} r^{-1} \omega)|^2 \tag{3.9}$$

for $\beta = 1$ and $\beta = \frac{1}{2}$ in the complex and real case respectivly.

Proof:
For the if statement, by applying some algebraic manipulations it can be shown that (3.9) is equivalent to

$$\forall J \in \mathbb{Z} \qquad \left|\hat{\phi}(2^J \omega)\right|^2 + \sum_{\substack{j > -J \\ r \in G}} \left|\hat{\psi}(2^{-j} r^{-1} \omega)\right|^2 = 1 \tag{3.10}$$

in the complex case. Using properties of the Fourier transform of convolutions we have $\widehat{W[2^j r] f}(\omega) = \hat{f}(\omega) \widehat{\psi_{2^j r}}(\omega)$. Now multiplying (3.10) by $|\hat{f}(\omega)|^2$ and integrating over ω gives $\|\widehat{W_J f}\| = \|\hat{f}\|$. The Plancherel formula

$$\int |f(x)|^2 dx = \int |\hat{f}(\omega)|^2 d\omega \tag{3.11}$$

yields $\|W_J f\|^2 = \|f\|^2$ i.e. f being unitary.
For the only if statement, Mallat [Mal12a] proceeds by contradiction. Assume there exists a non-trivial f such that $\|W_J f\| = \|f\|$ but (3.10) holds. By the Plancherel formula (3.11) and the reasoning above this is not possible.
For the real case, $\beta = \frac{1}{2}$ is due to $|\hat{f}(\omega)| = |\hat{f}(-\omega)|$ and the restriction of r to lie in G^+. □

The following **assumptions** are be made throughout this section [Mal12a, p. 7]:

- $\hat{\psi}$ is real and satisfies the condition (3.9), so W_J is unitary. Applying a few algebraic manipulations, this implies also that $\hat{\psi}(0) = \int \psi(x) dx = 0$ and $\hat{\phi}(r\omega) = \hat{\phi}(\omega)$.
- $\hat{\phi}(\omega)$ is real and symmetric. This implies that ϕ is real, symmetric and rotation invariant: $\phi(rx) = \phi(x)\ \forall r \in G$.
- The decay of $\phi, \psi \in C^2(\mathbb{R}^d)$ and of their partial derivatives up to order 2 is in $\mathcal{O}\left((1 + |x|)^{-d-2}\right)$.

For the construction of wavelets that satisfies (3.9) in practice see [Mal12a, pp. 7-8], [Mey95].

3.2.1.2 The Advantage of Using Wavelets

We give a brief motivation for using wavelets to construct the translation invariant and stable embedding Φ.
The modulus of the Fourier transform of f described above, $|\hat{f}|$ is translation invariant [Mal12a], [BM13] and could be used as an embedding Φ. However it is not Lipschitz continuous to deformations, as it is in general unstable at high frequencies [Hör71]. Bruna et al. [BM13] demonstrate the instability of $|\hat{f}|$ at high frequencies with a small example. Let us consider a signal given by a complex function $f(x) = e^{i\omega x}\theta(x)$ where ω is the phase of the signal and $\theta(x)$ the (real) amplitude. We consider also a small scaling operator

$$L_\tau f(x) = f(x - \tau(x)) = f((1-s)x).$$

When we compare $\hat{f}$ and $\widehat{L_\tau f}$ for $f(x) = e^{i\omega x}\theta(x)$, the application of L_τ shows itself as a frequency shift from ω to $(1-s)\omega$ in the Fourier domain. So

$$\| |\widehat{L_\tau f}| - |\hat{f}| \|$$

is proportional to $|s||\omega|$. Then even if s is small $\| |\widehat{L_\tau f}| - |\hat{f}| \|$ can grow arbitrarily large by letting ω tend to infinity. Hence the Lipschitz continuity (3.1) fails, if we use the modulus of the Fourier transform as our embedding Φ [BM13], [Mal12a].
In the Fourier transform, the signal f is convolved with the wave form $e^{i\omega t}$ which is supported on all of $\mathbb{R}$. The frequency shift in the Fourier domain causes less instability if the signal is instead convolved with a localised wave whose frequency support is wider at large frequencies (see also (4)). Wavelets will be these localised waves. Mallat et al. [Mal12a], [BM13] claim the assumption of deformation stability necessitates separating different scales with wavelets. Mallat et al. [Mal12a], [BM13] hence motivate the construction of Φ from wavelets as those offer more stablility at high frequencies.

3.2.2 The Scattering Transform

As mentioned in the beginning, the main aim of Mallat [Mal12a] is to find a representation Φ which is both translation invariant and stable to deformations i.e. Lipschitz continuous to diffeomorphisms. As motivated before (3.2.1.2) convolutions with wavelets are used as building blocks as they are stable. The goal is hence to obtain translation invariance while keeping stability and avoiding any loss of information.

3.2.2.1 The Necessity for a Non-Linearity

Using a non-linear activation function in neural networks results in the parameter learning being a highly non-linear optimisation problem and hence very difficult. It is thus reasonable to make sure that a non-linear activation is indeed necessary. Mallat [Mal12a] argues it is in the following way: One way of building a translation invariant operator from some operator commuting with translations $U[\lambda] \in L^2(\mathbb{R}^d)$ is through integration: $\int U[\lambda]f(x)dx$ is invariant to translations, if $U[\lambda]$ commutes with translations and if the integral exists. Although $W[\lambda] = f * \psi_\lambda$ commutes with translations, integration is not helpful as

$$\int \psi(x)dx = 0 \implies \int W[\lambda]f(x)dx = \int (f * \psi_\lambda)(x)dx = 0.$$

Further applying any linear translation invariant operator $M[\lambda]$ to $W[\lambda]$ and integrating

$$U[\lambda]f := M[\lambda]W[\lambda]f$$

still yields zero. Hence, Mallat [Mal12a] argues that it is necessary to choose a non-linear $M[\lambda]$ to obtain a non-zero invariant.

3.2.2.2 Assumptions on the Non-Linearity

Mallat [Mal12a] thus justifies choosing $M[\lambda]$ to be a non-linear operator. For stability they also enforce $M[\lambda]$ to be *Lipschitz continuous to diffeomorphisms*. One way of retaining Lipschitz continuity to diffeomorphisms is by imposing that $M[\lambda]$ commutes with diffeomorphisms. Further, $M[\lambda]$ being *non-expansive* or *contracting* i.e.

$$\|M[\lambda]f - M[\lambda]g\| \leq \|f - g\| \quad \text{for all } f, g \in L^2(\mathbb{R}^d)$$

ensures stability in L^2. In particular, this also enforces that M preserves the L^2 norm,

$$\|M[\lambda]f\| = \|f\|.$$

$\|M[\lambda]f\| = \|f\|$ for all $f \in L^2(\mathbb{R}^d)$ implies $|M[\lambda]f| = |f|$. Mallat [Mal12a] now simply chooses $M[\lambda]$ to be the absolute value: $M[\lambda]f := |f|$.
Staying in the same notation as in section (3.2.1) we remind the reader that $\psi(x) = e^{i\eta x}\theta(x)$, $\psi_\lambda(x) = e^{i\lambda\eta x}\theta_\lambda(x)$, $\hat{\theta}_\lambda(\omega) = \hat{\theta}(\lambda^{-1}\omega)$ and that

$$W[\lambda]f(x) = f * \psi_\lambda(x) = e^{i\lambda\eta x}\Big(f^\lambda * \theta_\lambda(x)\Big)$$

with $f^\lambda = e^{-i\lambda\eta x} f(x)$. Applying $M[\lambda]$, the absolute value, to $W[\lambda]f$ results in a cancelling of the modulation term $e^{i\lambda\eta x}$. Hence, we have constructed the signal representation

$$M[\lambda]W[\lambda]f = |W[\lambda]f| = |f * \psi_\lambda| = |f^\lambda * \theta_\lambda|$$

which will lead to a Lipschitz continuous representation with respect to diffeomorphisms as we will formally prove in section (3.2.5).

3.2.2.3 The Necessity for a Deep Network Structure

Applying the absolute value operator $M[\lambda]$ results in information loss. In particular, cancelling the modulation term $e^{i\lambda\eta x}$ results in the phase informatin $\lambda\eta$ being lost. Mallat [Mal12a] argues that a multi-layer structure mitigates this information loss.
He illustrates the effects of $M[\lambda]$ with the following example. Let $f(x) = \cos(\omega_1 x) + a\cos(\omega_2 x)$ be an example signal. Then Mallat [Mal12a] calculates

$$M[\lambda]W[\lambda]f = \frac{1}{2}\left|\hat{\psi}_\lambda(\omega_1) + a\hat{\psi}_\lambda(\omega_2)e^{i(\omega_2-\omega_1)x}\right|.$$

Hence, oscillations happen at frequency $|\omega_2 - \omega_1|$ rather than at ω_1 or ω_2. Thus $M[\lambda]W[\lambda]$ carries information on interactions of frequencies rather than the frequencies themselves. $M[\lambda]W[\lambda]$ constitutes a map from one function to a function of lower frequencies. Indeed,

$$|\omega_2 - \omega_1| \leq |\omega_1| \qquad \text{and} \qquad |\omega_2 - \omega_1| \leq |\omega_2|. \tag{3.12}$$

Energy (see (3.2.1)) from high frequencies is pushed to lower frequencies and distributed accross different frequency bands or intervals (see also theorem (3.9)).
Define

$$U[\lambda] := M[\lambda]W[\lambda].$$

The *coefficients*

$$\int U[\lambda]f(x)dx = \int |f * \psi_\lambda(x)|dx$$

are now invariant to translations, but high frequencies are lost through integration [Mal12a]. The wavelet coefficients extract information at some scale depending on λ, say 2^{j_1}. Mallat solves the issue of high frequency information being lost by applying $U[\lambda']$ to $U[\lambda]f$. Then taking the modulus and integrating gives a class of translation invariant second order coefficients indexed by λ and λ'. Considering again Mallat's [Mal12a] example from before $U[\lambda']U[\lambda]$ turns out to be proportional to a factor $|\psi_{\lambda'}(|\omega_2 - \omega_1|)|$ for a suitable $\psi_{\lambda'}$. So second order coefficients are capturing interac-

tions between the scale 2^{j_1} and scales a maximum distance 2^{j_2} away for some j_2. By cascading $U[\lambda]$ and building a deep structure we recover information on frequencies larger than 2^J that is lost due to the integration at the previous layer. Because the modulus has a strong contraction effect (compare 3.12)), the norm of higher order coefficients goes to zero quite quickly. This makes it possible to restrict the depth. See (3.9) for the formal result and the necessary number of layers in practice.

3.2.2.4 Defining the Scattering Transform

Let us first define the scattering propagator which formalises the notion of iteratively applying $U[\lambda]$.

Definition 3.4 (Path, Scattering Propagator). *[Mal12a, Definition 2.2]*
Define a path to be a sequence $p = (\lambda_1, \cdots, \lambda_m)$ for $\lambda_k \in \Lambda_\infty = 2^{\mathbb{Z}} \times G^+$. Define

$$U[\lambda]f = |f * \psi_\lambda|$$

*for $f \in L^2(\mathbb{R}^d)$. Define a **Scattering Propagator** as the following product of operators*

$$U[p] = U[\lambda_m] \cdots U[\lambda_2]U[\lambda_1]$$

for given p. Let $U[\emptyset] = Id$. The scattering propagator applied to $f \in L^2$ is

$$U[p]f = \big|\cdots\big|f * \psi_{\lambda_1}\big| * \psi_{\lambda_2}\big| \cdots |*\psi_{\lambda_m}|.$$

$U[\lambda]$ and hence $U[p]$ are well defined for $f \in L^2$ as $\|U[\lambda]f\| \leq \|\psi_\lambda\|_1\|f\|$ for all $\lambda \in \Lambda_\infty$. With the scattering propagator we can now define the scattering transform first on L^1.

Definition 3.5 (P_∞, Scattering Transform on L^1). *[Mal12a, Definition 2.3]*
For $f \in L^1$ and $p \in \mathcal{P}_\infty$ the set of finite paths let

$$\bar{S}f(p) = \frac{1}{\mu_p}\int U[p]f(x)dx \quad \textit{with } \mu_p = \int U[p]\delta(x)dx$$

be the scattering transform of f.

The scattering transform for L^2 functions is redefined in [Mal12a, Section 3] as the limit of the windowed scattering transform presented below. It is proven to be non-expansive and pointwise convergent to the scattering transform defined on L^1 (3.5).

Definition 3.6 ($\mathcal{P}_J$, Windowed Scattering Transform). *[Mal12a, Definition 2.4] Define the windowed scattering transform for all $p \in \mathcal{P}_J$ as*

$$S_J[p]f(x) = U[p]f * \phi_{2^J}(x) = \int U[p]f(u)\phi_{2^J}(x-u)du$$

where $J \in \mathbb{Z}$. $\mathcal{P}_J$ is the set of finite paths $p = (\lambda_1, \cdots, \lambda_m)$ such that $\lambda_k \in \Lambda_J$ i.e. $|\lambda_k|2^{j_k} > 2^{-J}$. As before, $\phi_{2^J}(x) = 2^{-dJ}\phi(2^{-J}x)$. We have $S_J[\mathcal{P}_J]f = \{S_J[p]f\}_{p\in\mathcal{P}_J}$ and

$$S_J[p]f(x) = \big|\cdots\big|f * \psi_{\lambda_1}\big| * \psi_{\lambda_2}\big|\cdots|*\psi_{\lambda_m}| * \phi_{2^J}(x).$$

$S_J[\mathcal{P}_J]$ is the desired embedding function Φ. Results on its advantageous properties are presented in the next few sections.

- norm preservation (3.2.3)
- translation invariance (3.2.4)
- stability to diffeomorphisms (3.2.5)

3.2.3 Norm Preservation

In this section, we show that the scattering transform is norm preserving. As an intermediate step we also show that it is non-expansive. First, however we wish to highlight the connection of the scattering transform to neural networks, in particular to Convolutional Neural Networks (CNNs) [LKF10], as this will show the relevance of Mallat's work [Mal12a], [Mal16] and put the results of this section into better context.

3.2.3.1 Connection to CNNs

In order to compute the windowed scattering transform as defined above we can iteratively apply the operator

$$U_J f = \{A_J f, (U[\lambda]f)_{\lambda\in\Lambda_J}\} \quad \text{where} \quad A_J f = f * \phi_{2^J},\ U[\lambda]f = |W[\lambda]f| = |f * \psi_\lambda|.$$

After one iteration we obtain

$$U_J U[p]f = \{S_J[p]f, (U[p+\lambda]f)_{\lambda\in\Lambda_J}\}$$

where $p+\lambda$ denotes the concatenation of λ to the end of the path p. Let us denote Λ_J^m to be the set containing paths of length $m < \infty$, $\Lambda_J^0 = \{\emptyset\}$, $U[\Lambda_J^m] = \{U[p]\}_{p\in\Lambda_J^m}$ and $S_J[\Lambda_J^m] = \{S_J[p]\}_{p\in\Lambda_J^m}$. Notice also that $\mathcal{P}_J = \cup_{m\in\mathbb{N}}\Lambda_J^m$. Hence if one starts with

$f = U[\emptyset]f$ and iteratively calculates $U_J U[\Lambda_J^m]f$, one obtains $S_J[\mathcal{P}_J]f$ as m tends to infinity.

The structure of the scattering transform which iterates on a contractive operator can

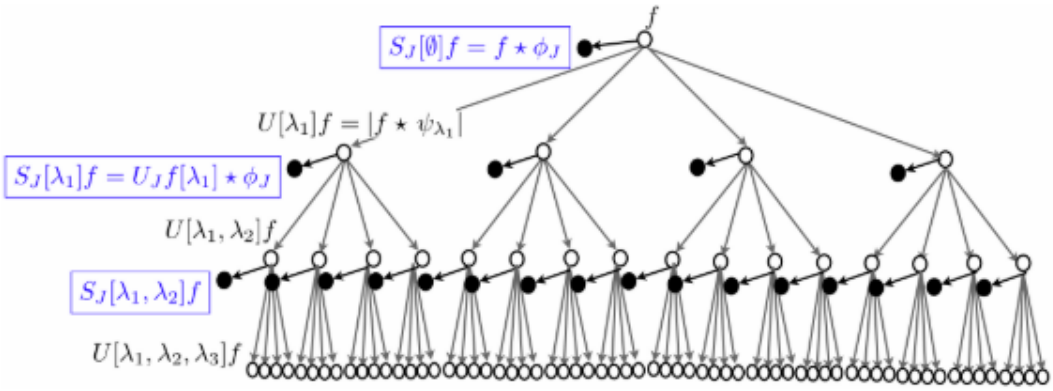

Figure 5: This figure has been taken from [Mal12a] and shows the architecture of a scattering networks. The black nodes symbolise the averaging operation. The white nodes represent the application of the operator $U[\lambda_1], U[\lambda_1, \lambda_2]$ etc.

be seen as a type of convolution neural network (CNN) architecture [Mal12a]. As in a CNN, at each layer a convolution and a pooling operation are applied. Here the pooling operation is the modulus of a complex quantity. Also the averaging can be seen as a type of pooling. Differences to typical CNNs are that CNN weights are learnt through backpropagation, while in the scattering transform weights are chosen to be wavelets. After applying Φ, we train a linear discriminator on the embedding Φ. Only here weights are actually learnt. Instead of the scattering transform also other multi-scale signal representations as mentioned in [AM18b] are possible choices of weights. Another difference to standard CNNs is that no pooling is done on the channels in each layer and that output occurs at all levels in the form of the $S_J[p]f = U[\lambda]f * \phi_{2^J}$ [Mal12a]. A CNN architecture only outputs at the last layer.

3.2.3.2 Non-Expansive Scattering Transform

Mallat [Mal12a, Proposition 2.5] proves that the scattering transform $S_J[\mathcal{P}_J]$ is non-expansive in the L^2 norm. For this, define the norms

$$\|S_J[\Omega]f\|^2 = \sum_{p\in\Omega} \|S_J[p]f\|^2 \qquad \text{and} \qquad \|U[\Omega]f\|^2 = \sum_{p\in\Omega} \|U[p]f\|^2.$$

for an arbitrary set of paths Ω. The norm of $U_J f$ is given by

$$\|U_J f\|^2 = \|A_J f\|^2 + \sum_{\lambda \in \Lambda_J} \||W[\lambda] f|\|.$$

Proposition 3.7 (Non-expansive scattering transform $S_J[\mathcal{P}_J]$)**.** *[Mal12a, Proposition 2.5]*

$$\|S_J[\mathcal{P}_J] f - S_J[\mathcal{P}_J] h\| \leq \|f - h\| \quad \forall f, h \in L^2(\mathbb{R}^d).$$

Proof:
Remembering $W_J f = \{A_J f, (W[\lambda] f)_{\lambda \in \Lambda_J}\}$, $U_J f = \{A_J f, (|W[\lambda] f|)_{\lambda \in \Lambda_J}\}$ and the fact that $U_J f$ is unitary, it follows that $W_J f$ is non-expansive, simply because the modulus operation is. Indeed, $||x| - |y|| \leq |x - y|$ for x, y in $\mathbb{R}$ or $\mathbb{C}$ and

$$\begin{aligned} \|U_J f - U_J h\|_2^2 &= \|A_J f - A_J h\|_2^2 + \sum_{\lambda \in \Lambda_J} \||W[\lambda] f| - |W[\lambda] h|\|_2^2 \\ &\leq \|W_J f - W_J h\|_2^2 \leq \|f - h\|_2^2. \end{aligned}$$

Because $\||W[\lambda] f|\| = \|W[\lambda] f\|$, putting $h = 0$ yields

$$\|U_J f\| = \|f\|.$$

U_J is hence norm preserving, while W_J is norm preserving and linear, so unitary. Iterating on the non-expansive U_J gives

$$\begin{aligned} \|U[\Lambda_J^m] f - U[\Lambda_J^m] h\|_2^2 &\geq \|U_J U[\Lambda_J^m] f - U_J U[\Lambda_J^m] h\|_2^2 \\ &= \|S_J[\Lambda_J^m] f - S_J[\Lambda_J^m] h\| + \|U[\Lambda_J^{m+1}] f - U[\Lambda_J^{m+1}] h\|. \end{aligned}$$

Now summing over m from 0 to ∞ allows us to go from Λ_J^m to $\mathcal{P}_J$ and yields

$$\|S_J[\mathcal{P}_J] f - S_J[\mathcal{P}_J] h\|_2^2 = \sum_{m=0}^{\infty} \|S_J[\Lambda_J^m] f - S_J[\Lambda_J^m] h\|_2^2 \leq \|f - h\|_2^2.$$

□

We introduce the technical condition of *admissibility* which will enable us to prove the next theorem. The main role of the admissibility condition is in guaranteeing norm preservation under $S_J[\mathcal{P}_J]$ (3.9). We will also see that scattering energy decreases, when the depth of the scattering network m increases.

Definition 3.8 (Admissibility). *[Mal12a, p. 15]*
Let ψ be a scattering wavelet such that there exists $\eta \in \mathbb{R}^d$, $\rho \geq 0$ s.t. $|\hat{\rho}(\omega)| \leq |\hat{\phi}(2\omega)|$ and $\hat{\rho}(0) = 1$ such that the quantity

$$\hat{\Psi}(\omega) := |\hat{\rho}(\omega - \eta)|^2 - \sum_{k=1}^{\infty} k(1 - |\hat{\rho}(2^{-k}(\omega - \eta))|^2)$$

fulfills the following property:

$$\inf_{1 \leq |\omega| \leq 2} \sum_{j=-\infty}^{\infty} \sum_{r \in G} \hat{\Psi}(2^{-j} r^{-1} \omega) |\hat{\psi}(2^{-j} r^{-1} \omega)|^2 =: \alpha > 0.$$

Theorem 3.9 (Decreasing scattering energy, norm preservation). *[Mal12a, Theorem 2.6]*
For an admissible (3.8) scattering wavelet the following holds:

$$\forall f \in L^2(\mathbb{R}^d) \quad \lim_{m \to \infty} \|U[\Lambda_J^m] f\|^2 = \lim_{m \to \infty} \sum_{n=m}^{\infty} \|S_J[\Lambda_J^n]\|^2 = 0 \tag{3.13}$$

and

$$\|S_J[\mathcal{P}_J] f\| = \|f\|. \tag{3.14}$$

Proof outline:
As the proof in [Mal12a] relies almost exclusively on algebraic manipulations, we omit technicalities and state only the more interpretable intermediate results. Mallat [Mal12a] first shows the equivalence of

$$\lim_{m \to \infty} \|U[\Lambda_J^m] f\|^2 = 0$$

to

$$\lim_{m \to \infty} \sum_{n=m}^{\infty} \|S_J[\Lambda_J^n]\|^2 = 0 \qquad \text{and} \qquad \|S_J[\mathcal{P}_J] f\| = \|f\|$$

by applying algebraic manipulations to the definitions.
Then he focuses on proving that the admissibility condition (3.8) implies

$$\lim_{m \to \infty} \|U[\Lambda_J^m] f\|^2 = 0.$$

The result uses an auxiliary lemma proving that the admissibility condition (3.8) implies that the quantity

$$a_m = \frac{\sum_{p \in \Lambda_J^m} j_m \|U[p] f\|^2}{\sum_{p \in \Lambda_J^m} \|U[p] f\|^2} \tag{3.15}$$

for $p = (\lambda_1, \cdots, \lambda_m)$ decreases almost by $\frac{\alpha}{2}$ (see (3.8) for definition) at every increase of m by 1. This is interpreted in [Mal12a] as the scattering energy being pushed to lower frequencies. Finally, the energy is absorbed by the averaging with ϕ_{2^J}. Next it is proved that

$$\frac{\alpha}{2}\|U[\mathcal{P}_J]f\|^2 \leq \max(J+1,1)\|f\|^2 + \|f\|_w^2 \tag{3.16}$$

where

$$\|f\|_w^2 := \sum_{j=0}^{\infty} \sum_{r\in G^+} j\|W[2^j r]f\|^2 < \infty.$$

If $\|f\|_w^2 < \infty$, then equation (3.16) and

$$\|U[\mathcal{P}_J]f\|^2 = \sum_{m=0}^{\infty} \|U[\Lambda_J^m]f\|^2$$

necessitate that $\lim_{m\to\infty}\|U[\Lambda_J^m]f\|^2 = 0$.
Mallat [Mal12a] uses a density argument to extend the results from functions that are bounded in $\|\cdot\|_w$ to $f \in L^2(\mathbb{R}^d)$. For $f_J = f * \phi_{2^J}$ with $\phi_{2^J}(x) = 2^{-dJ}\phi(2^{-d}x)$ it holds that $\lim_{J\to-\infty}\|f - f_J\| = 0$, because ϕ integrates to one. As in the L^2 case, the strategy is to prove $\|f_J\|_w < \infty$ which then implies $\lim_{m\to\infty}\|U[\Lambda_J^m]f_J\|^2 = 0$. It can be shown that

$$\begin{aligned}\|W[2^j r]f\|_2^2 &= \int |\hat{f}(\omega)|^2 |\hat{\phi}(2^J\omega)|^2 |\hat{\psi}(2^{-j}r^{-1}\omega|^2 d\omega \\ &\leq C2^{-2J-2j}\int |\hat{f}(\omega)|^2 d\omega\end{aligned} \tag{3.17}$$

for some $C > 0$. Here the equality is due to an application of the Plancherel formula (3.11). The inequality follows from two assumptions that are commonly made on the wavelets. Firstly it as assumed that ψ has a vanishing moment i.e. that $\int x\psi(x)dx = 0$. This implies by some algebraic calculations that $\hat{\psi}(\omega) = \mathcal{O}(|\omega|)$. By our initial assumptions on ϕ, derivatives of ϕ are integrable and hence $|\omega||\hat{\phi}(\omega)|$ is bounded.
Overall, (3.17) implies that $\|f_J\|_w^2 < \infty$ which in turn gives $\lim_{m\to\infty}\|U[\Lambda_J^m]f_J\|^2 = 0$. Using non-expansiveness of $U[\Lambda_J^m]$ and the triangle inequality gives

$$\|U[\Lambda_J^m]f\| \leq \|f - f_J\| + \|U[\Lambda_J^m]f_J\|.$$

Taking the limit as J tends to $-\infty$ and m tends to ∞ gives $\lim_{m\to\infty}\|U[\Lambda_J^m]f\|^2 = 0$ for all $f \in L^2(\mathbb{R}^d)$. □

Interpretation of the Result

Scattering energy is pushed to lower frequencies at every iteration, finally captured by the low pass filter ϕ_{2^J} and output. Mallat [Mal12a] highlights that calculations should in practice focus on frequency decreasing paths where $|\lambda_{k+1}| < |\lambda_k|$ as scattering energy is almost exclusively found there. In a sense, by choosing wavelets as filters an architecture has been designed which has sparse coefficients. Numerical implementations in [Mal10] are mentioned that calculate a scattering transform using $\mathcal{O}(d \log d)$ operations for a signal of size d, as it is sufficient to carry out calculations for frequency decreasing paths.
The fact that

$$\lim_{m\to\infty} \sum_{n=m}^{\infty} \|S_J[\Lambda_J^n]\|^2 = 0$$

also shows that we can restrict the path length to some m without significant loss of information. Mallat et al. [Mal12a], [BM10] conduct numerical experiments which show that paths of length 3 or longer hardly contribute to the overall scattering energy anymore in applications. Mallat [Mal12a] also points out possible choices of wavelets in practice that satisfy the admissibility condition (3.8) such that theorem (3.9) holds.

3.2.4 Translation Invariance

In this section, we analyse the properties of the scattering transform, when the signal f is subject to a translation. We will see that the transform is translation invariant in the limit. For this, Mallat [Mal12a] first examines the distance

$$\|S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]h\|$$

and proves that it is non-increasing for increasing J and thus converges in the limit.

Key Concepts: Schur's Lemma

Proposition 3.10 (Non-increasing scattering distance)**.** *[Mal16, Proposition 2.9]*

$$\|S_{J+1}[\mathcal{P}_{J+1}]f - S_{J+1}[\mathcal{P}_{J+1}]h\| \leq \|S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]h\| \quad \forall f, h \in L^2(\mathbb{R}^d), J \in \mathbb{Z}$$

Proof outline:
As $\mathcal{P}_J \subset \mathcal{P}_{J+1}$ the proof goes about extending $\mathcal{P}_J$ to $\mathcal{P}_{J+1}$. Considering $p' \in \mathcal{P}_{J+1}$ pick the prefix of p' of maximal length that is in $\mathcal{P}_J$ and denote it by p. The scale of the element of p' immediately following prefix p fulfils $2^{-J} \geq 2^j > 2^{-J+1}$. Thus the elemet

is necessarily $2^J r$. The set of all paths in $\mathcal{P}_{J+1}$ that have longest prefix p' is thus

$$\mathcal{P}_{J+1}^p := \{p\} \cup \{p + 2^J r + p''\}_{r \in G^+, p'' \in \mathcal{P}_{J+1}}$$

and gives a partition of $\mathcal{P}_{J+1} = \cup_{p \in \mathcal{P}_J} \mathcal{P}_{J+1}^p$. It can be proven from the unitary property of wavelets that

$$\forall f \in L^2(\mathbb{R}^d), q \in \mathbb{Z} \qquad \sum_{-q \geq l > -J} \sum_{r \in G^+} \|f * \psi_{2^l r}\|^2 + \|f * \phi_{2^J}\|^2 = \|f * \phi_{2^q}\|.$$

Applying the definitions and a number of algebraic manipulations we arrive at the desired result. As a side product it is also proved that

$$\sum_{p' \in \mathcal{P}_{J+1}^p} \|S_{J+1}[p']f\|^2 = \|S_J[p]f\|^2.$$

□

The scattering distance $\|S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]h\|$ converges for increasing J, as it is positive and we have shown that it is non-increasing for increasing J. As $\|S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]h\| \leq \|f - h\|$ also

$$\lim_{J \to \infty} \|S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]h\| \leq \|f - h\| \quad \forall f, h \in L^2(\mathbb{R}^d).$$

If wavelets are admissible (3.8), (3.9) shows norm preservation, $\|S_J[\mathcal{P}_J]f\| = \|f\|$. Then norm preservation holds also in the limit:

$$\lim_{J \to \infty} \|S_J[\mathcal{P}_J]f\| = \|f\|.$$

More so, the limit metric is invariant to translations L_c:

Theorem 3.11 (Translation invariance in the limit). *[Mal12a, Theorem 2.10]*
If wavelets are admissible (3.8), then

$$\forall f \in L^2(\mathbb{R}^d), \forall c \in \mathbb{R}^d \quad \lim_{J \to \infty} \|S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]L_c f\| = 0.$$

Proof:
Using commutative properties with translations

$$S_J[\mathcal{P}_J]L_c = L_c S_J[\mathcal{P}_J] \qquad \text{and} \qquad S_J[\mathcal{P}_J]f = A_J U[\mathcal{P}_J]f,$$

Mallat [Mal12a] obtains

$$\begin{aligned} \|S_J[\mathcal{P}_J]L_c f - S_J[\mathcal{P}_J]f\| &= \|L_c A_J U[\mathcal{P}_J]f - A_J U[\mathcal{P}_J]f\| \\ &\leq \|L_c A_J - A_J\| \|U[\mathcal{P}_J]\|. \end{aligned} \tag{3.18}$$

At this point Mallat [Mal12a, p. 60] applies Schur's Lemma on operators of the form $Kf(x) = \int f(u)k(x,u)du$ to bound the operator norm of $L_\tau A_J - A_J$:

$$\|L_\tau A_J - A_J\| \leq 2^{-J+d} \|\nabla \phi\|_1 \|\tau\|_\infty.$$

Setting $\tau(x) = c$ and collecting constants gives

$$\|L_c A_J - A_J\| \leq C 2^{-J} |c|.$$

Plugging this into equation (3.18) and using the bound on $\|U[\mathcal{P}_J]f\|$ (3.16) which follows from the admissibility condition (3.8) yields

$$\|L_c S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]f\|^2 \leq ((J+1)\|f\|^2 + \|f\|_w^2) C^2 \frac{2}{\alpha} 2^{-2J} |c|^2.$$

If $\|f\|_w < \infty$, then this implies that

$$\lim_{J \to \infty} \|S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]L_c f\| = 0.$$

To extend the result to L^2, Mallat [Mal12a] uses a density argument similar as in the proof of theorem (3.9) along with the non-expansive property of $S_J[\mathcal{P}_J]$ and the unitary property of L_c. □

Interpretation of the Result

Convolutions commute with translations. To have local invariance to translations, the scattering network does an averaging with the averaging kernel ϕ_{2^J}. Here 2^J is the local range over which translations are eliminated by the convolution. However also all variations beyond frequency 2^J are erased. When $J = \infty$ we have complete translation invariance, but almost all information is lost.

3.2.5 Lipschitz Continuity to Diffeomorphisms

So far we have shown translation invariance (in the limit) of the scattering transform. However, many more general perturbations occur in real world images. These are modelled by [Mal12a] using diffeomorphisms. In this final section, it is demonstrated that $S_J[\mathcal{P}_J]$ is Lipschitz continuous to diffeomorphisms, when $S_J[\mathcal{P}_J]$ is a windowed scattering transform based on an admissible (3.8) wavelet. We consider diffeomorphisms of the form [Mal12a, p. 21]

$$L_\tau f(x) = f(x - \tau(x)) \qquad \text{such that} \quad \|\nabla\tau\|_\infty < 1 \tag{3.19}$$

for $f \in L^2(\mathbb{R}^d)$. Define also the maximal increment in τ as $\|\Delta\tau\|_\infty := \sup_{x,u\in\mathbb{R}^d}|\tau(x) - \tau(u)|$ and denote the Hessian of τ by $H\tau$. For the following theorem, Mallat [Mal12a] introduces also the *mixed scattering norm*

$$\|U[\mathcal{P}_J]f\|_1 := \sum_{m=0}^{\infty} \|U[\Lambda_J^m]f\|.$$

Theorem 3.12 (Lipschitz continuity to diffeomorphisms)**.** *[Mal12a, Theorem 2.12] There exists $C > 0$ such that for all $f \in L^2(\mathbb{R}^d)$ with $\|U[\mathcal{P}_J]f\|_1 < \infty$ and all $\tau \in C^2(\mathbb{R}^d)$ with $\|\nabla\tau\|_\infty \leq 1/2$ the following statement holds:*

$$\|S_J[\mathcal{P}_J]L_\tau f - S_J[\mathcal{P}_J]f\| \leq C\|U[\mathcal{P}_J]f\|_1 K(\tau) \tag{3.20}$$

where

$$K(\tau) = 2^{-J}\|\tau\|_\infty + \|\nabla\tau\|_\infty \max\left(\log\frac{\|\Delta\tau\|_\infty}{\|\nabla\tau\|_\infty}, 1\right) + \|H\tau\|_\infty.$$

Furthermore, for all $m \geq 0$ it holds that

$$\|S_J[\mathcal{P}_{J,m}]L_\tau f - S_J[\mathcal{P}_{J,m}]f\| \leq Cm\|f\|K(\tau) \tag{3.21}$$

where $\mathcal{P}_{J,m} \subset \mathcal{P}_J$ contains paths of length less than m.

In the case that $2^{-J}\|\tau\|_\infty \leq \|\nabla\tau\|_\infty$ the term in equation (3.20) above accounting for translation is negligible and full Lipschitz continuity to deformation holds in the sense of definition (3.1).

Proof outline:
Using again equation (3.18) and the definition of the scattering norm gives

$$\|L_\tau S_J[\mathcal{P}_J]f - S_J[\mathcal{P}_J]f\| \leq \|L_\tau A_J - A_J\|\|U[\mathcal{P}_J]f\|_1.$$

Mallat [Mal12a, Lemma 2.13] uses a technical lemma that shows that for all operators L_τ on $L^2(\mathbb{R}^d)$

$$\|S_J[\mathcal{P}_J]L_\tau f - L_\tau S_J[\mathcal{P}_J]f\| \leq \|U[\mathcal{P}_J]f\|_1 \|U_J L - L U_J\|.$$

Further manipulations lead to

$$\|S_J[\mathcal{P}_J]L_\tau f - S_J[\mathcal{P}_J]f\| \leq C\|U[\mathcal{P}_J]\|(2^{-J}\|\tau\|_\infty + \|[W_J, L_\tau]\|) \tag{3.22}$$

where $[W_J, L_\tau] = W_J L_\tau - L_\tau W_J$. It remains to bound $\|[W_J, L_\tau]\|^2 = \|[W_J, L_\tau]^*[W_J, L_\tau]\|$ which is done through another technical lemma [Mal12a, Lemma 2.14]. The lemma proves that $\exists C > 0$ such that $\forall J \in \mathbb{Z}, \forall \tau \in C^2(\mathbb{R}^d)$ with $\|\Delta\tau\|_\infty \leq 1/2$

$$\|[W_J, L_\tau]\| \leq C\left(\|\Delta\tau\|_\infty \max\left(\log \frac{\|\Delta\tau\|_\infty}{\|\nabla\tau\|_\infty}, 1\right) + \|H\tau\|_\infty\right) \tag{3.23}$$

where $H\tau$ is the Hessian of τ. Inserting this above gives the first part of the proof, statement (3.20). The second statement of the theorem follows by exchanging $\|U[\mathcal{P}_J]\|_1$ for $\|U[\mathcal{P}_{J,m}]f\|_1$ and remembering the norm preservation property of U_J. □

Interpretation: Progressive Linearisation

1. Intermediate result (3.16) gives conditions under which $\|U[\mathcal{P}_J]f\|$ is finite. Mallat [Mal12a] conducts numerical experiments which show that paths of length 3 or longer hardly contribute to the overall scattering energy anymore. Thus Mallat [Mal12a] applies (3.21) with $m = 4$.

2. From the theorem we can see that the error $\|S_J[\mathcal{P}_J]L_\tau f - S_J[\mathcal{P}_J]f\|$ induced by the deformation L_τ is bounded by a multiple of $2^{-J}\|\tau\|_\infty$ and $\|\nabla\tau\|_\infty$ which quantify the extent to which τ translates and deforms the input respectively. The factor 2^J in claim (3.20) can be seen as the scale on which deformations have locally become linearised [Mal16].

3. The change of variables from f to $S_J[\mathcal{P}_J]f$ has locally linearised L_τ. Mallat [Mal16] states that CNNs [LKF10] linearise the target function f going from one layer to the next. The endeavour to linearise translations and diffeomorphisms in high dimension is based on the assumption that they are local symmetries i.e. that the target function does not vary locally under them. This makes sense intuitively as for example hand written digits still represent the same number if they are translated or deformed locally. The invariance is not global however as some

transformations can turn one digit into another. This is why Lipschitz continuity to diffeomorphisms as just proved is preferred over invariance.

4. The purpose of linearising local symmetries is that we can then project linearly, while keeping the value of f [Mal16]. Dimensionality has thus been reduced and the problem now lives in a linear space. So finally it is possible to do classification using a linear classifier as is usually done in neural networks in the last network layer.

Conclusion

In this section we have seen that neural networks can be fooled quite easily with adversarial examples. We have seen two example methods for constructing adversarial examples. We have considered the original reference on adversarial examples [Sze+13] which uses additive perturbations and a more recent method which models perturbations with vector fields [AAG18]. We have contrasted different hypotheses on the distribution of adversarial examples in the input space and have seen a regularisation procedure and a game theory inspired defence strategy for more robustness against adversarial examples.

Furthermore, we have looked at scattering networks and drawn some valuable insights on CNNs. From the work of Mallat et al. we have learnt that wavelets constitute powerful building blocks in a network, as they are localised and hence the wavelet transform is stable. Non-linear activation functions are necessary for scattering networks to build non-zero coefficients with wavelet transforms. The deep structure makes sure that information lost through averaging is recovered. Mallat et al. demonstrate that an embedding function for signals can be found which preserves the L^2 norm, does not expand distances, is translation invariant and stable to diffeomorphisms. In a scattering network, energy is pushed to lower frequencies at every iteration and output through an averaging at every layer. Hence, network depth can be restricted. Coefficients in a scattering network are sparse, because energy concentrates on frequency decreasing paths.
Mallat et al. have shown how an embedding applied to a signal linearises a potentially very non-linear deformation L_τ. They propose the interpretation that CNNs linearise the target function f going from one layer to the next. A linearisation of translations and diffeomorphisms is preferable over invariance, because labels are locally invariant under transformations, but not globally. After progressively linearising layer by layer it is possible to do classification using a linear classifier at the last network layer.

4 Exact Weight Identification

Approximating even a smooth function requires in general exponentially many samples [NW10]. Under which conditions is approximation possible using only polynomially many samples? In other words, when do learning problems become tractable? This chapter contrasts two different approaches to tractably learning a target function given by a one hidden layer network

$$f(x) = \sigma(Ax).$$

Going even further, the goal will be to uniquely identify the weight vectors of the network which are the rows of A. Anandkumar et al. (4.1) and Fornasier et al. (4.2) both put forward a method for tackling this problem. They make use of the fact that differentiation of a function given by a neural network exposes the weights due to the chain rule. Anandkumar et al. use cross moment tensors which are equivalent in expectation to a derivative of the neural network. They utilise tensor decomposition to access the weights. Fornasier et al. use a weak differentiation approach and the singular value decomposition (SVD) to approximate the span of the weight vectors. They prove that quasi-orthonormality of the weights can be assumed without loss of generality which leads to a spectral norm optimisation problem yielding the exact weight vectors. A comparison of the works of Anandkumar et al. and Fornasier et al. is presented in section (4.3).

4.1 Exact Weight Identification with Tensor Decomposition

In this section we focus on the work of Anandkumar et al. in [SA14], [JSA14], [JSA15] who learn neural networks using tensor methods with a number of samples that scales polynomially in the input dimension and network complexity. The authors uniquely identify the network weights.
A brief discussion of [SA14] serves as an introduction into the use of tensors in network learning (4.1.1). Here a random sparsity assumption on the weights is used to uniquely identify the weights (compare also the role of sparsity in [FSV12] (4.2.1)).
The main focus of this section will be on the work of Janzamin et al. [JSA14], [JSA15] (4.1.2) who refine the method of Sedghi et al. [SA14] by dropping the sparsity assumption and extending the method to the overcomplete case.

4.1.1 Introduction: Tensor Decomposition for Weight Identification

Sedghi et al. [SA14] make the problem of learning a neural network tractable by assuming knowledge of the density function of the input. Further they assume network weights to be sparse and the input dimension to be large enough which will facilitate exact recovery.

Key Concepts: Method-of-Moments, Tensor Decomposition, Stein's Lemma, l_1 minimisation

Method of Moments for Subspace Approximation

Sedghi et al. [SA14] consider the problem in which the target function is given by a one layer neural network whose weights need to be learnt. Let $x \in \mathbb{R}^d$ be the input and $y \in \mathbb{R}^{d_y}$ the output vector (or label) of the network given by

$$y = \sigma_2(A_2\sigma_1(A_1x))$$

for $A_1 \in \mathbb{R}^{k\times d}$ and $A_2 \in \mathbb{R}^{d_y\times k}$. We can also describe the network by $\mathbb{E}[y \mid h] = \sigma_2(A_2h)$ and $\mathbb{E}[h \mid x] = \sigma_1(A_1x)$. The aim in [SA14] is to recover A_1 uniquely. Having learnt A_1 the parameters of the last layers can be learnt using regression.

Definition 4.1 (Score function)**.** *[SA14, p. 4]*
For a random vector x with density $p(x)$ the score function is defined as

$$\nabla_x \log(p(x)).$$

In the following it is assumed that the probability density function of x, $p(x)$ is continuous such that the score function $\nabla_x \log p(x)$ exists [SA14]. It is also assumed that $p(x)$ is known or sufficiently well approximated. For this the authors suggest to use methods based on auto-encoders or score matching as proposed in [Ana+14], [Sri+13], [Dev85]. Let the *moment matrix* be

$$M := \mathbb{E}_{x,y}\left[y(\nabla_x \log p(x))^T\right].$$

Theorem 4.2. *[SA14, Theorem 1]*
For the one hidden layer neural network introduced above it holds

$$M := \mathbb{E}\left[y\nabla \log(p(x))^T\right] = -\mathbb{E}\left[\sigma_2'(\tilde{x}_2)A_2Diag(\sigma_1'(\tilde{x}_1))\right]A_1$$

with $\tilde{x}_2 = A_2\sigma_1(A_1x)$ and $\tilde{x}_1 = A_1x$.

The proof makes use of Stein's Lemma [Ste86], [Ste+04] which follows from integration by parts [NPS14].

Lemma 4.3 (Stein's Lemma). *[Ste+04, Theorem 1.1], [SA14, Proposition 1]*
For $x \in \mathbb{R}^d$ with density $p(x)$ and existing score function $\nabla_x \log(p(x))$ it holds that

$$\mathbb{E}\Big[g(x)(\nabla_x \log p(x))^T\Big] = -\mathbb{E}\Big[\nabla_x g(x)\Big]$$

if the expectations exist, $g : \mathbb{R}^d \to \mathbb{R}^{d_y}$ is differentiable and the elements of $g(x)p(x)^T$ tend to 0 at the boundaries of the support of $p(x)$.

Proof of theorem (4.2):
Then by applying consecutively the law of total expectation, the definition of y, Stein's Lemma and the Chain Rule it follows:

$$\begin{aligned} M &:= \mathbb{E}_{x,y}\Big[y(\nabla_x \log p(x))^T\Big] \\ &= \mathbb{E}_x\Big[\mathbb{E}_y\Big[y(\nabla_x \log p(x))^T \Big| x\Big]\Big] \\ &= \mathbb{E}_x\Big[\sigma_2(A_2(\sigma_1(A_1 x)))(\nabla_x \log p(x))^T\Big] \\ &= -\mathbb{E}_x\Big[\sigma_2'(\tilde{x}_2)A_2 \mathrm{Diag}(\sigma_1'(\tilde{x}_1))\Big]A_1 \end{aligned}$$

with $\tilde{x}_2 = A_2\sigma_1(A_1 x)$ and $\tilde{x}_1 = A_1 x$. □

For ease of notation define

$$B := -\mathbb{E}\Big[\sigma_2'(\tilde{x}_2)A_2 \mathrm{Diag}(\sigma_1'(\tilde{x}_1))\Big].$$

From the last theorem we can see that the spans of the moment matrix and the first layer weights are identical (if B is full rank) [SA14].

4.1.1.1 Random Sparsity for Exact Weight Identification

The previous theorem shows that A can be learnt if the unknown factorisation of M, $M = BA_1$, can be recovered. In practice an approximation of M via the sample mean will be given. Identifying the weight matrix A_1 uniquely from M is in general not possible, but the next theorem shows that it can be done if A_1 is sparse. Assume A_1 has *Bernoulli-Gaussian*(θ) entries i.e. every entry of A_1 is the product of a Bernoulli(θ) and an zero-mean Gaussian random variable. The Bernoulli variable takes the value 0 with probability $1-\theta$, hence θ controls the degree of sparsity of A_1. Under these

conditions the theorem presented below shows that the sparsest vectors in the row span of M are precisely the rows of A_1.

Theorem 4.4. *[SWW12, Theorem 1]*
Assume that A_1 is of full rank, has Bernoulli-Gaussian(θ) entries and for $C >$ it holds that

$$1/k \leq \theta \leq 1/C \qquad \text{and} \qquad d > Ck \log k.$$

Then for any alternative factorisation $M = B'A_1'$ such that

$$\max_i \|e_i^T A_1'\|_0 \leq \max_i \|e_i^T A_1\|_0$$

it holds that A_1' and A_1 are identical up to a scaling and permutation of the rows with probability at least $1 - \exp(-c'd)$ for some constant $c' > 0$.

Idea of the proof:
For $M = B'A_1' = BA_1$ the row spaces of M, A_1, A_1' are the same, if B and A_1 are full rank. Picking the sparsest elements of the row span of M will likely give the rows of X. Indeed, any row of A_1' is in the row span of M. If it is sparse, then with high probability it is one of the rows of A_1, because the rows of A_1 are sparse, thus likely the supports of the rows do not intersect and taking linear combinations increases the support size. This argument is formalised in the following lemma.

Lemma 4.5. *[SWW12, Lemma 6]*
If $A_1 \in \mathbb{R}^{k \times d}$ is a Bernoulli(θ)-Gaussian matrix with $1/k < \theta < 1/C$ and $d > Ck \log k$, then the probability that there exists a linear combination of more than 1 row of A_1 which has l_0-norm less than $11/9\theta d$ is at most $\exp(-c\theta d)$ for some $c, C > 0$.

This ensures the recovery of the rows of A_1 with exponentially small failure probability and finishes the proof. □

Sedghi et al. [SA14] base themselves on the results in [SWW12] and recover the sparsest vectors in the row space of M by solving the following optimisation problem:

$$\min_{w \in \mathbb{R}^{d_y}} \|w^T M\|_0 \text{ such that } w \neq 0. \tag{4.1}$$

As commonly done, a convex relaxation of the l_0 norm to the l_1 norm is undertaken. Also the constraint $w \neq 0$ is replaced by the constraint $r^T w = 1$ which yield the optimisation problem

$$\min_{w \in \mathbb{R}^{d_y}} \|w^T M\|_1 \quad \text{such that } r^T w = 1. \tag{4.2}$$

The optimisation problem is solved using the following routine:

Algorithm 1 Sparse Dictionary Learning, [SWW12, Section 5.2]

input M

1: **for** $j \in [d]$ **do**
2: $\quad \hat{w} := \arg\min_w \|w^T M\|_1$ s.t. $(Me_j)^T w = 1$
3: $\quad s_j := \hat{w}^T M$
4: **end for**
5: $\mathcal{S} = \{s_1, \cdots, s_d\}$
6: **for** $i \in [k]$ **do**
7: $\quad$ **repeat**
8: $\quad\quad l := \arg\min_{s_l \in \mathcal{S}} \|s_l\|_0$
9: $\quad\quad v_i := s_l$
10: $\quad\quad \mathcal{S} = \mathcal{S} \backslash \{s_l\}$
11: $\quad$ **until** $\operatorname{rank}([v_1, \cdots, v_i]) = i$
12: **end for**

output $\hat{A}_1 = \left(\frac{v_1}{\|v_1\|}, \cdots, \frac{v_k}{\|v_k\|}\right)^T$

It can be proven [SWW12] that the algorithm above outputs rows of A_1 with high probability, if r in equation (4.2) is fixed to be a column of M. This fact has been used in line 2 of algorithm (1). The algorithm above represents the core of the learning algorithm for A_1.

Algorithm 2 Learning A_1, [SA14, Algorithm 1]

input samples $\{(x_i, y_i) : i \in [n]\}$

1: Approximate the score function $\nabla \log p(x)$ as in [Ana+14], [Sri+13], [Dev85]
2: Approximate M as $\hat{M} = \frac{1}{n} \sum_{i \in [n]} y_i (\nabla_x \log p(x_i))^T$
3: $\hat{A}_1 =$ Sparse Dictionary Learning$(\hat{M})$ (1)

We briefly state the main result of [SA14, Theorem 2], an adaptation of [SWW12, Theorem 7, 8], which guarantees the recovery of A_1 with high probability.

Theorem 4.6. *[SA14, Theorem 2], [SWW12, Theorem 7, 8]*
Assume the following assumptions hold:

1. *The score function $\nabla_x \log p(x)$ is known or sufficiently well approximated by using an auto-encoder or score matching [Ana+14], [Sri+13], [Dev85].*
2. $\mathbb{E}_x \left[\sigma_2'(A_2 \sigma_1(A_1 x)) A_2 Diag(\sigma_1'(A_1 x)\right]$ *is full rank.*
3. *A_1 is a Bernoulli(θ)-Gaussian matrix of full rank with normalised rows and is sparse with*

$$\frac{2}{k} \leq \theta \leq \frac{\alpha}{\sqrt{k}}.$$

4. *The input dimension is large enough:* $d > \mathcal{O}(k \log^4 k)$.

Then A_1 *can be recovered uniquely by algorithm (1) with exponentially high probability.*

The work of Sedghi et al. [SA14] has served as an introduction to the use of tensors in network learning and exemplified how sparsity facilitates exact recovery. We omit the proof of this result and turn straight to the more advanced and universal method of Anandkumar et al., the NN-LIFT algorithm [JSA15].

4.1.2 Neural Network LearnIng using Feature Tensors (NN-LIFT)

The previously introduced method of Sedghi et al. [SA14] is refined in [JSA15] by dropping the sparsity assumption and extending the method to the overcomplete case. Janzamin et al. [JSA15] develop Neural Network LearnIng using Feature Tensors (NN-LIFT) which is a tractable learning method with bounded expected error and polynomial sample complexity.
Similarly to before, Janzamin et al. [JSA15] look at neural networks from a probabilistic point of view. In their work, input x and output $y \in \{0, 1\}$ are distributed according to some distributions with probability density functions $p(x)$ and $p(y \mid x)$. $p(x)$ is assumed to be known (Janzamin et al. [JSA15, Section 3.1] describe a number of ways of approximating it [Ana+14], [Sri+13], [Dev85]) and $p(y \mid x)$ needs to be approximated. Hence we are in a generative setting. The goal is to learn a function f

$$f(x) := \mathbb{E}[y \mid x] = \langle a_2, \sigma_1(A_1^T x + b_1) \rangle + b_2 \qquad A_1 \in \mathbb{R}^{d \times k}, b_1, a_2 \in \mathbb{R}^k, b_2 \in \mathbb{R} \tag{4.3}$$

from n samples $(x_i, y_i) \overset{\text{iid}}{\sim} p(x, y)$.
Contrary to Sedghi et al. [SA14], Janzamin et al. [JSA15] model the bias terms explicitly and only consider the binary label case. They state that their method can be extended also to the multi-label case. In this setting the architecture and dimensions of the weights are known. The goal is a procedure that first identifies the unknown, fixed parameters A_1, a_2, b_1, b_2 uniquely. Then using the estimated parameters, f can be approximated. In our discussion we focus on the parameter identification.

Key Concepts: Method-of-Moments, Score Functions, CP Tensor Decomposition, Whitening, Bernstein Inequality

The parameters of a one hidden layer neural network A_1, b_1 and a_2, b_2 are identified uniquely using CP (CanDecomp/Parafac) tensor decomposition (see preliminaries), a Fourier based method and ridge regression respectively. Overall, the approximation

procedure which we will explain in detail can be summarised in the following algorithm.

Algorithm 3 NN-LIFT, [JSA15, Algorithm 1]

input samples $\{(x_i, y_i) : i \in [n]\}$, $\mathcal{S}_3(x)$, $\tilde{\epsilon}_1$ (desired accuracy for $(A_1)_j$)

1: $\hat{T} := \frac{1}{2}\sum_{i\in[n]} y_i \cdot \mathcal{S}_3(x_i)$ (see (4.8))
2: $\{(\hat{A}_1)_j\}_{j\in[k]} :=$ tensor decomposition$(\hat{T})$ as in (4)
3: $\hat{b}_1 :=$ Fourier method$(\{(x_i, y_i) : i \in [n]\}, \hat{A}_1, \tilde{\epsilon}_1)$ as in (7)
4: $(\hat{a}_2, \hat{b}_2)$ = Linear Regression$(\{(x_i, y_i) : i \in [n]\}, \hat{A}_1, \hat{b}_1)$

output $\hat{A}_1, \hat{a}_2, \hat{b}_1, \hat{b}_2$

Given then an approximate network $\hat{f}(x) = \langle \hat{a}_2, \sigma_1(\hat{A}_1^T x + \hat{b}_1)\rangle + \hat{b}_2$, Janzamin et al. [JSA15] also bound the estimation error

$$\mathbb{E}_x\Big[|f(x) - \hat{f}(x)|^2\Big].$$

They consider two different cases: The undercomplete setting in which $k \leq d$ and the overcomplete case in which $k > d$, but $k \in \mathcal{O}(d^2)$. Next, we briefly state the main result in [JSA15] which bounds the error above. For the rest of the section we detail the parameter identification methods in algorithm (3) and prove the main result below.

4.1.3 Main result: Polynomial Sample Complexity

Janzamin et al. [JSA15] uniquely identify the network parameters of the one hidden layer network (4.3) with sample complexity that depends only polynomially on the number of neurons k and the input dimension d.

Theorem 4.7 (Polynomial sample complexity of NN-LIFT). *[JSA15, Theorem 3]*
Under the assumptions (4.1.3.1) the estimated weights $\hat{A}_1, \hat{b}_1, \hat{a}_2, \hat{b}_2$ obtained as output of (3) facilitate polynomial approximation for the function $f := \langle a_2, \sigma(A_1^T x + b_1)\rangle + b_2$.
For $\hat{f}(x) := \langle \hat{a}_2, \sigma(\hat{A}_1^T x + \hat{b}_1)\rangle + \hat{b}_2$ it holds

$$\mathbb{E}_x[|\hat{f}(x) - f(x)|^2] \leq \tilde{\mathcal{O}}(\epsilon^2)$$

under the condition that the number of samples n satisfies

$$n \geq \tilde{\mathcal{O}}\left(\frac{k}{\epsilon^2} \cdot \mathbb{E}\left[\|M_3(x)M_3^T(x)\|\right]\right.$$
$$\left. \cdot \, poly\left(y_{max}, \frac{\mathbb{E}[\|\mathcal{S}_2(x)\mathcal{S}_2^T(x)\|]}{\mathbb{E}[\|\mathcal{M}_3(x)\mathcal{M}_3^T(x)\|]}, \frac{\tilde{\xi}_f}{\psi}, \frac{\tilde{\lambda}_{max}}{\tilde{\lambda}_{min}}, \frac{1}{\lambda_{min}}, \frac{s_{max}(A_1)}{s_{min}(A_1)}, |\Omega_l|, L, \frac{\|a_2\|}{(a_2)_{min}}, |b_2|, \sigma_{noise}, \rho_\lambda\right)\right) \tag{4.4}$$

where $\lambda_{min} := \min_{j\in[k]}|\lambda_j|$, $\tilde{\lambda}_{min} := \min_{j\in[k]}|\tilde{\lambda}_j|$, $\tilde{\lambda}_{max} := \max_{j\in[k]}|\tilde{\lambda}_j|$, $(a_2)_{min} := \min_{j\in[k]}|a_2(j)|$ *and* $|f(x)| \leq y_{max}$ *for* $x \in B_d(r)$.

4.1.3.1 Assumptions for the Main Result (4.7)

1. General assumptions
   (a) The density function $p(x)$ of input x is continuous, differentiable and known up to normalisation.
   (b) The score functions (4.8) of order 2 and 3, $\mathcal{S}_2$, $\mathcal{S}_3$, exist and are bounded. Janzamin et al. [JSA15, Section 3.1.] point to estimation methods of these quantities as in [Ana+14], [Sri+13], [Dev85].
   (c) A number n of iid samples (x_i, y_i) are available.
   (d) The activation function σ is Lipschitz continuous with Lipschitz constant L. This ensures that error propagation from layer to layer is limited.
   (e) Mild regularity conditions: $f(x) \cdot \nabla^{(2)}p(x)$, $\nabla f(x) \cdot \nabla p(x)^T$ and $\nabla^{(2)}f(x) \cdot p(x)$ are tending to 0 on the edges of support of $p(x)$. This conditions is necessary for the results involving the score function $\mathcal{S}_m(x)$ (see [JSA14] for details).
2. Assumptions used in the estimation of A_1 using tensor decomposition (4.1.4)
   (a) For whitening procedure (5): If $k \leq d$, then $A_1 \in \mathbb{R}^{d\times k}$ is assumed to have full column rank and the minimum singular value of A_1, $s_{\min}(A_1)$ is bounded away from zero by $\epsilon > 0$. If $k \in \mathcal{O}(d^2)$, it is assumed that the Khatri-Rao product $A_1 \odot A_1$ (see preliminaries) has full column rank (see (4.1.4.1) for motivation) and that $s_{min}(A_1 \odot A_1)$ is bounded away from zero by ϵ.
   (b) The coefficients

$$\lambda_j := \mathbb{E}[\sigma'''(z_j)] \cdot a_2(j) \qquad \tilde{\lambda}_j := \mathbb{E}[\sigma''(z_j)] \cdot a_2(j)$$

for $j \in [k]$ are not zero (used in (4.10) and (4.5)).

(c) $\sigma(z) = 1 - \sigma(-z)$ resolves the ambiguity about the sign of the columns of A_1 (see [JSA15] for details).

3. Assumptions used for the estimation of b_1 by the Fourier method (4.1.7)

   (a) A_1 is column-normalised. This is needed as otherwise the columns $(A_1)_j$ are only identifiable up to norm, for example for σ being the Heaviside function [JSA15, p. 17].

   (b) The entries of b_1 are bounded. This is needed as also for identifiability as b_1 is derived from the phase (or argument) of a complex number which remains the same if one adds multiples of 2π and is hence not unique.

   (c) x is bounded and $p(x)$ is bounded below by some $\psi > 0$.

4. Assumptions used in the regression estimation of a_2, b_2

   (a) It is imposed that statistical leverage is bounded and noise is subgaussian (see [JSA15, p. 16] for details).

For the rest of this section on the work of Anandkumar et al., we work towards proving the main result (4.7). We proceed by outlining the approximation method for A_1. We first motivate the method with some results on score functions (4.1.4), then describe intermediate steps of the method, namely the whitening step (4.1.5), the tensor power iteration (4.1.6) and the un-whitening (4.1.5.1). The necessary sample complexity for the estimation of A_1 will be discussed in (4.1.6).
The estimation method for b_1 is presented in section (4.1.7) and its sample complexity analysed in section (4.1.7.1).

4.1.4 Approximation of A_1 via Tensor Decomposition

We have already introduced score functions in our introduction (4.1.1) and they continue to be a crucial tool to learn A_1. Continuing the line of approach used in [SA14], Janzamin et al. [JSA15] use higher order score functions.

Definition 4.8 (score function of order m)**.** *[JSA14, Defintion 3]*
Let the score function of order m be

$$\mathcal{S}_m(x) := (-1)^m \frac{\nabla_x^{(m)} p(x)}{p(x)} \in \bigotimes^m \mathbb{R}^d$$

where $\bigotimes^m \mathbb{R}^d$ is the m-fold outer product of $\mathbb{R}^d$.

Note that for $m = 1$ the score function as defined in (4.1) is recovered. In the following we assume that a sufficiently good approximation of $\mathcal{S}_2(x)$ and $\mathcal{S}_3(x)$ is available. The following theorem is an extension of Stein's Lemma (4.3) that enables us to access higher order derivatives through higher order score functions.

Theorem 4.9. *[JSA14, Theorem 1, 6]*
For $g(x) : \mathbb{R}^d \to \bigotimes^r \mathbb{R}^{d_y}$ such that all entries in

$$\nabla_x^{(i)} g(x) \otimes \mathcal{S}_{m-i-1}(x) \otimes p(x)$$

for $i \in \{0, \cdots, m-1\}$ vanish on the boundaries of the support of $p(x)$, then

$$\mathbb{E}\Big[g(x) \otimes \mathcal{S}_m(x)\Big] = \mathbb{E}\Big[\nabla_x^{(m)} g(x)\Big]$$

assuming that $\mathcal{S}_m(x)$ exists.

Idea of the proof:
Janzamin et al. [JSA14] use Stein's Lemma (4.3), the recursion formula

$$\mathcal{S}_m(x) = -\mathcal{S}_{m-1}(x) \otimes \nabla_x \log p(x) - \nabla_x \mathcal{S}_{m-1}(x)$$

and the product rule for differentiation with tensors. □

For $f(x) := \mathbb{E}[y \mid x]$ it therefore follows [JSA14] that

$$\mathbb{E}\Big[y \cdot \mathcal{S}_3(x)\Big] = \mathbb{E}\Big[\nabla_x^{(3)} f(x)\Big].$$

Hence by the chain rule multiples of A_1 come up on the right hand side when differentiating and thus the score functions can be used to get access to A_1. Janzamin et al. [JSA15] interpret the differentiation as having a linearisation effect. The use the following lemma to support their claim that tensor decomposition is necessary.

Lemma 4.10. *[JSA15, Lemma 6]*
For the one hidden layer network (4.3)

$$\mathbb{E}\Big[y \cdot \mathcal{S}_3(x)\Big] = \sum_{j \in [k]} \lambda_j (A_1)_j \otimes (A_1)_j \otimes (A_1)_j \in \mathbb{R}^{d \times d \times d}$$

for

$$\lambda_j = \mathbb{E}[\sigma'''(z_j)] \cdot a_2(j)$$

where $z_j = A_1^T x + b_1$ and $(A_1)_j$ is the jth column of A_1.

Idea of the proof:
Applying the theorem above to label function (4.3) gives [JSA14]

$$\mathbb{E}\Big[y \cdot \mathcal{S}_3(x)\Big] = \mathbb{E}\Big[\nabla_x^{(3)} f(x)\Big].$$

The three factors $(A_1)_j$ result from applying the chain rule three times. Cross terms cancel because $\sigma'''(z_j) \in \mathbb{R}^{k\times k\times k\times k}$ is a diagonal tensor with entries given by

$$\frac{\partial^3 \sigma(z_j)}{\partial z_j}.$$

By assumptions (2) the rank 1 components do not vanish. □

Tensor Decomposition Algorithm

The strategy in approximating A_1 is hence to recover the rank 1 components from the moment tensor $\mathbb{E}[y \cdot \mathcal{S}_3(x)]$ (in practice the sample mean is used as an approximation). This is done via the tensor decomposition method introduced in [Ana+14], [AGJ14] for orthogonal tensors. To apply the method also to non-orthogonal tensors, Janzamin et al. [JSA15] extend it by a **whitening** procedure (4.1.5) which orthogonalises the tensor. The core of the decomposition method then relies on a **tensor power iteration** (6).

Algorithm 4 Tensor Decomposition [JSA15, Algorithm 4]

input symmetric tensor T
1: **if** $k \in \mathcal{O}(d^2)$ **then**
2: tensorize T (4.1.4.1)
3: **end if**
4: $T = \text{Whiten}(T)$ (5)
5: **for** $j = 1$ **to** k **do**
6: $(v_j, \mu_j, T) = \text{Tensor Power Decomposition}(T)$ (6)
7: **end for**
8: $A_1(j) := \text{Un-Whiten}(v_j), j \in [k]$ (4.1.5.1)

4.1.4.1 The Overcomplete Case $k \in \mathcal{O}(d^2)$

To decompose tensors for which $k \in \mathcal{O}(d^2)$ i.e. when the rank is larger than the dimension, Janzamin et al. [JSA15] consider the order 6 tensor

$$T = \sum_{j\in[k]} \lambda_j a_j^{\otimes 6} \in \bigotimes^6 \mathbb{R}^d.$$

They apply a tensorisation step to convert it into an order 3 tensor $\tilde{T} \in \bigotimes^3 \mathbb{R}^{d^2}$ by calculating

$$\tilde{T}(i_2 + d(i_1 - 1), j_2 + d(j_1 - 1), l_2 + d(l_1 - 1)) := T(i_1, i_2, j_1, j_2, l_1, l_2).$$

Then $\tilde{T}$ can be composed as

$$\tilde{T} = \sum_{j \in [k]} \lambda_j (a_j \odot a_j)^{\otimes 3}$$

where $\odot$ denotes the Khatri-Rao product. The tensor decomposition then proceeds with this new tensor $\tilde{T}$ with the full rank condition (2) now imposed on the $A_1 \odot A_1$.

4.1.5 Whitening

To decompose a tensor into rank 1 components it is necessary to whiten the tensor or in other words to orthogonalise its components first [AGJ14]. Orthogonalisation is possible as the columns of A_1 are linearly independent by assumption (2) in the case that $k \leq d$ and as $A_1 \odot A_1$ is full rank in the case $k \in \mathcal{O}(d^2)$. Janzamin et al. [JSA15] use the following procedure for whitening.

Algorithm 5 Whitening [JSA15, Procedure 5]

input tensor $T \in \mathbb{R}^{d \times d \times d}$

1: Set $\tilde{M}_2 := \mathbb{E}[y \cdot \mathcal{S}_2(x)] = \sum_{j \in [k]} \tilde{\lambda}_j \cdot (A_1)_j \otimes (A_1)_j \in \mathbb{R}^{d \times d}$ with $\tilde{\lambda}_j = \mathbb{E}[\sigma''(z_j) \cdot a_2(j)]$

2: Compute SVD of M_2, $M_2 = U \mathrm{Diag}(\gamma) U^T$ for $U \in \mathbb{R}^{d \times k}$ (rank k), $\gamma \in \mathbb{R}^k$

3: Compute whitening matrix $W := U \mathrm{Diag}(\gamma^{-1/2}) \in \mathbb{R}^{d \times k}$

return $T(W, W, W) \in \mathbb{R}^{k \times k \times k}$

We briefly outline why and in what sense the output $T(W, W, W)$ has been whitened. To whiten some third order tensor

$$\tilde{T} = \sum_{j \in [k]} \lambda_j \cdot (A_1)_j \otimes (A_1)_j \otimes (A_1)_j$$

as described above, Janzamin et al. [JSA15] construct moment tensor

$$\tilde{M}_2 := \mathbb{E}\Big[y \cdot \mathcal{S}_2(x)\Big] = \sum_{j \in [k]} \tilde{\lambda}_j \cdot (A_1)_j \otimes (A_1)_j \in \mathbb{R}^{d \times d} \tag{4.5}$$

with $\tilde{\lambda}_j = \mathbb{E}[\sigma''(z_j) \cdot a_2(j)]$, $z = A_1^T x + b_1$ and $(A_1)_j$ being the weight vectors i.e. in this setting the columns of A_1. The coefficients do not vanish due to assumption (2). For W defined in algorithm (5) we demonstrate next that $W^T \tilde{M}_2 W = I_k$ where I_k is the

k-dimensional identity matrix.
Using the multilinear form (see preliminaries) it holds that

$$\tilde{T}(W,W,W) = \sum_{j\in[k]} \lambda_j (W^T(A_1)_j)^{\otimes 3} = \sum_{j\in[k]} \frac{\lambda_j}{\tilde{\lambda}_j^{3/2}} (W^T(A_1)_j \sqrt{\tilde{\lambda}_j})^{\otimes 3} = \sum_{j\in[k]} \mu_j v_j^{\otimes 3}$$

for $\mu_j := \lambda_j \tilde{\lambda}_j^{-3/2}$, $v_j := W^T(A_1)_j \sqrt{\tilde{\lambda}_j}, j \in [k]$. The vectors v_j are collected as columns in a matrix V. Then

$$V := W^T A_1 \mathrm{Diag}(\tilde{\lambda}^{1/2})$$

and by the definition of W,

$$VV^T = W^T A_1 \mathrm{Diag}(\tilde{\lambda}) A_1^T W = W^T \tilde{M}_2 W = I_k.$$

Because $V \in \mathbb{R}^{k\times k}$, also $V^T V = I_k$. Janzamin et al. [JSA15] conclude that $\tilde{T}(W,W,W)$ is whitened. $\{v_j\}_{j\in[k]}$ gives an ONB and hence the algorithm succeeds in doing the whitening.

4.1.5.1 Un-whitening

At the end of the tensor decomposition there is an un-whitening step [JSA15, Procedure 6] i.e. a back transformation to the original domain. This is done by calculating (using the notation from before)

$$(A_1)_j = \frac{1}{\sqrt{\tilde{\lambda}_j}} U \mathrm{Diag}(\gamma^{1/2}) v_j \quad j \in [k].$$

The reasoning behind this step is the following: Combining the definitions

$$W := U\mathrm{Diag}(\gamma^{-1/2}) \qquad \text{and} \qquad V := W^T A_1 \mathrm{Diag}(\tilde{\lambda}^{1/2})$$

yield

$$U\mathrm{Diag}(\gamma^{1/2}) V = UU^T A_1 \mathrm{Diag}(\tilde{\lambda}^{1/2}).$$

Remembering that U contains the singular vectors of A_1, by properties of the SVD it follows that UU^T is the projection onto the row space of A_1 and hence $UU^T A_1 = A_1$. Thus, $A_1 = U\mathrm{Diag}(\gamma^{1/2}) V \mathrm{Diag}(\tilde{\gamma}^{-1/2})$ which is the same as the column-wise definition of $(A_1)_j$ stated above.

4.1.6 Tensor Power Method

The core of the tensor decomposition method is a tensor power iteration which is a natural extension to the power iteration used in numerical calculations of the SVD of a matrix.

Algorithm 6 Tensor Power Method [Ana+14, Algorithm 7]

input $T \in \mathbb{R}^{d\times d\times d}$ symmetric; N and R number of iterations and initialisations
output $(\hat{\mu}, \hat{v})$ eigenvector-eigenvalue pair; $T - \hat{\mu}\hat{v}^{\otimes 3}$

1: **for** $\tau = 1$ **to** R **do**
2: Initialise $\hat{v}_0^\tau$ as in [AGJ14]
3: **for** $t = 1$ **to** N **do**
4: Power iteration update $\hat{v}_t^\tau := \frac{T(I,\hat{v}_{t-1}^\tau,\hat{v}_{t-1}^\tau)}{\|T(I,\hat{v}_{t-1}^\tau,\hat{v}_{t-1}^\tau)\|}$
5: **end for**
6: **end for**
7: $\tau^* := \arg\max_{\tau\in[R]}\{T(\hat{v}_N^\tau, \hat{v}_N^\tau, \hat{v}_N^\tau)\}$
8: N additional power iteration updates with $\hat{v}_N^{\tau^*}$ as starting point. Set the result to be $\hat{v}$.
9: Set $\hat{\mu} := T(\hat{v}, \hat{v}, \hat{v})$.
10: **return** $(\hat{v}, \hat{\mu})$; deflated tensor $T - \hat{\mu}\hat{v}^{\otimes 3}$

For the convergence guarantees and a stability analysis of the tensor power iteration for the decomposition of orthogonal tensors we point the interested reader to [Ana+14] and [AGJ14].

Sample Complexity in Estimating A_1

To prove the overall polynomial sample complexity in the main result (4.7) we need to analyse the sample complexity which is necessary for the estimation of A_1 and b_1. The case of A_1 is considered in this section. For the analysis of the sample complexity for b_1 see section (4.1.7.1).
The next result proves the required sample complexity for the estimation of A_1. From algorithm (3) we see that a noisy version of moment tensor $\mathbb{E}[y \cdot \mathcal{S}_3(x)]$ is decomposed. The theorem below builds on guarantees for noisy orthogonal tensor decomposition in [AGJ14] and incorporates the whitening procedure into the analysis.

Lemma 4.11 (Sample complexity for the identification of A_1)**.** *[JSA15, Lemma 9] The output $\hat{A}_1$ of the NN-LIFT algorithm (3) fulfils*

$$\min_{z\in\{1,-1\}} \|(A_1)_j - z\cdot(\hat{A}_1)_j\| \leq \tilde{\mathcal{O}}(\tilde{\epsilon}_1), \quad j \in [k]$$

with high probability under the assumptions (2) and with whitening done as in (5). Further, the number of samples n needs to fulfil

$$n \geq \left\{ \tilde{\mathcal{O}}\left(y_{max}^2 \mathbb{E}\left[\| M_3(x) M_3(x)^T \| \right] \frac{\tilde{\lambda}_{max}^4}{\tilde{\lambda}_{min}^4} \frac{s_{max}^2(A_1)}{\lambda_{min}^2 \cdot s_{min}^6(A_1)} \cdot \frac{1}{\tilde{\epsilon}_1^2} \right), \tag{4.6}$$

$$\tilde{\mathcal{O}}\left(y_{max}^2 \cdot \mathbb{E}\left[\| M_3(x) M_3(x)^T \| \right] \cdot \left(\frac{\tilde{\lambda}_{max}}{\tilde{\lambda}_{min}} \right)^3 \frac{1}{\lambda_{min}^2 \cdot s_{min}^6(A_1)} \cdot k \right), \tag{4.7}$$

$$\tilde{\mathcal{O}}\left(y_{max}^2 \cdot \frac{\mathbb{E}[\| \mathcal{S}_2(x) \mathcal{S}_2^T(x) \|]^{3/2}}{\mathbb{E}[\| \mathcal{M}_3(x) \mathcal{M}_3^T(x) \|]^{1/2}} \cdot \frac{1}{\tilde{\lambda}_{min}^2 \cdot s_{min}^3(A_1)} \right) \right\}. \tag{4.8}$$

Proof:
To estimate A_1 first an approximation to $\tilde{T} = \mathbb{E}\left[y \cdot \mathcal{S}_3(x) \right]$ is formed in step 1 of algorithm (3). Janzamin et al. [JSA15] bound this error using the Bernstein inequality. They consider

$$E := \tilde{T} - \hat{T} = \mathbb{E}\left[y \cdot \mathcal{S}_3(x) \right] - \frac{1}{n} \sum_{i \in [n]} y_i \cdot \mathcal{S}_3(x_i).$$

Similarly,

$$E_2 := \tilde{M}_2 - \hat{M}_2 = \mathbb{E}\left[y \cdot \mathcal{S}_2(x) \right] - \frac{1}{n} \sum_{i \in [n]} y_i \cdot \mathcal{S}_2(x_i).$$

To bound E with the Bernstein inequality, E is matricised giving

$$\tilde{E} := \mathbb{E}\left[y \cdot M_3(x) \right] - \frac{1}{n} \sum_{i \in [n]} y_i \cdot M_3(x_i) = \sum_{i \in [n]} \frac{1}{n} \Big(\mathbb{E}\left[y \cdot M_3(x) \right] - y_i \cdot M_3(x_i) \Big).$$

The summands can be bounded by $\frac{y_{max}}{n} \mathbb{E}\left[\| M_3(x) \| \right]$ for $|y| \leq y_{max}$. Bounding the variance using

$$\frac{1}{n^2} \Big\| \sum_{i \in [n]} \mathbb{E}\left[y_i^2 \cdot M_3(x_i) M_3^T(x_i) \right] \Big\| \leq \frac{1}{n} y_{max}^2 \mathbb{E}\left[\| M_3(x) M_3(x)^T \| \right]$$

gives

$$\| E \| \leq \| \tilde{E} \| \leq \tilde{\mathcal{O}}\Big(\frac{y_{max}}{\sqrt{n}} \sqrt{\mathbb{E}\left[\| M_3(x) M_3^T(x) \| \right]} \Big)$$

with high probability. Similarly, by Bernstein's inequality with high probability it holds that

$$\| E_2 \| \leq \tilde{\mathcal{O}}\Big(\frac{y_{max}}{\sqrt{n}} \sqrt{\mathbb{E}[\| \mathcal{S}_2(x) \mathcal{S}_2^T(x) \|]} \Big).$$

□

4.1.7 Learning the bias b_1 by a Fourier Method

So far we have uniquely identified the multiplicative weights A_1 and analysed the sample complexity necessary for the estimation. It remains to do the same for the bias vector b_1. Having identified all of the network parameters, it is possible to estimate f uniformly and finish the proof of the main result (4.7). The estimation of b_1 is done in [JSA15] by a Fourier method. Complex quantities are calculated using the labels or output y_i, the density function of the input and random frequencies. Their phase or argument gives access to b_1:

Lemma 4.12 (Fourier method). *[JSA15, Lemma 7]*
The quantity

$$v := \frac{1}{n} \sum_{i \in [n]} \frac{y_i}{p(x_i)} e^{-i\langle \omega_i, x_i \rangle}$$

for ω_i drawn uniformly iid from the manifold

$$\Omega_l := \left\{ \omega \in \mathbb{R}^d : \|\omega\| = \frac{1}{2}, |\langle \omega, (\hat{A}_1)_l \rangle| \geq \frac{1 - \tilde{\epsilon}_1^2/2}{2} \right\}$$

has expectation

$$\mathbb{E}_{x,y,\omega}[v] = \frac{1}{|\Omega_l|} \hat{\sigma}(1/2) a_2(l) e^{i\pi b_1(l)}$$

where $\hat{\sigma}$ is the Fourier transform of σ.

Idea of the proof:
Using the law of expectation and keeping in mind equation (4.3) gives

$$\begin{aligned}
\mathbb{E}[v] &= \mathbb{E}_{x,y,\omega}\left[\frac{y}{p(x)} e^{-i\langle \omega, x \rangle} \right] \\
&= \mathbb{E}_{x,\omega}\left[\mathbb{E}_{y|x,\omega}\left[\frac{y}{p(x)} e^{-i\langle \omega, x \rangle} \middle| x, \omega \right] \right] = \mathbb{E}_{x,\omega}\left[\frac{f(x)}{p(x)} e^{-i\langle \omega, x \rangle} \right] \\
&= \int_{\Omega_l} \int f(x) e^{-i\langle \omega, x \rangle} p(\omega) dx d\omega = \int_{\Omega_l} \hat{f}(\omega) p(\omega) d\omega.
\end{aligned}$$

The proof then utilises an algebraic expression for the Fourier transform of f, $\hat{f}$, found in [MIA94] which is plugged in and yields the desired result after some calculations. □

Exploiting the previous lemma, Janzamin et al. [JSA15] use the argument or phase of v, $\angle v$, to arrive at the following algorithm for estimating b_1.

Algorithm 7 Fourier method [JSA15, Procedure 2]

input samples $\{(x_i, y_i)\} : i \in [n]\}$, $\hat{A}_1$, $\tilde{\epsilon}$, pdf $p(x)$ of x

1: **for** $l = 1$ **to** k **do**
2: Draw $\{\omega_i : i \in [n]\}$ uniformly on Ω_l (iid)
3: $v := \frac{1}{n} \sum_{i \in [n]} \frac{y_i}{p(x_i)} e^{-i\langle \omega_i, x_i \rangle} \in \mathbb{C}$
4: $\hat{b}_1(l) := \frac{1}{\pi}(\angle v - \angle \hat{\sigma}(1/2))$
5: **end for**
6: **return** $\hat{b}_1$

Comment: Lemma (4.12) visibly also gives a way of estimating a_2 by taking the absolute value. However, Janzamin et al. [JSA15] estimate a_2 and b_2 together using ridge regression.

4.1.7.1 Sample Complexity in estimating the Bias b_1

As a final step towards proving the sample complexity in the main result (4.7) we need to assess the sample complexity needed to approximate b_1. To this end, Janzamin et al. [JSA15] analyse the concentration of quantity v (as defined in algorithm (7) above) around its mean.

Lemma 4.13 (Sample complexity for the approximation of b_1)**.** *[JSA15, Lemma 11] Under conditions (3), the estimate $\hat{b}_1 := \frac{1}{\pi}(\angle v - \angle \tilde{\sigma}(1/2))$ in algorithm (7) satisfies*

$$|b_1(l) - \hat{b}_1(l)| \leq \frac{|\Omega_l|}{\pi |\tilde{\sigma}(1/2)||a_2(l)|} \mathcal{O}(\tilde{\epsilon}_2)$$

with probability as least $1 - \delta$ if the number of samples n is at least

$$\mathcal{O}\left(\frac{\tilde{\xi}_f}{\psi \tilde{\epsilon}_2^2} \log \frac{k}{\delta} \right)$$

for small enough $\tilde{\epsilon}_2 \leq \tilde{\xi}_f =: \int_{\mathbb{R}^d} f(x)dx$ and $p(x) \geq \psi > 0$.

Proof:
The proof hinges on an application of the Bernstein inequality. Define

$$v_i := \frac{1}{n} \frac{y_i}{p(x_i)} e^{-i\langle \omega_i, x_i \rangle}$$

so that $v = \sum_{i \in [n]} v_i$. Then $v_i - \mathbb{E}[v_i]$ is almost surely bounded by $\mathcal{O}(\frac{1}{\psi n})$, because $p(x) \geq \psi$ by conditions (3) and $y_i \in \{0, 1\}$. The same two assumptions are used to

prove that the variance is also bounded:

$$\begin{aligned}\sigma^2 &:= \sum_{i\in[n]} \mathbb{E}\Big[(v_i - \mathbb{E}[v_i])\overline{(v_i - \mathbb{E}[v_i])}\Big] \\ &\leq \sum_{i\in[n]} \mathbb{E}\Big[v_i\overline{v_i}\Big] = \sum_{i\in[n]} \frac{1}{n^2}\mathbb{E}\Big[\frac{y_i^2}{p(x_i)^2}\Big] \\ &= \sum_{i\in[n]} \frac{1}{n^2}\mathbb{E}\left[\mathbb{E}\left[\frac{y_i^2}{p(x_i)^2}\middle| x\right]\right] = \sum_{i\in[n]} \frac{1}{n^2}\mathbb{E}\left[\frac{f(x)}{p(x)^2}\right] \\ &\leq \frac{1}{n\psi}\int_{\mathbb{R}^d} f(x)dx = \frac{\tilde{\xi}_f}{n\psi}\end{aligned}$$

where we keep in mind the relation $f(x) = \mathbb{E}[y \mid x]$. Now Bernstein's inequality can be applied to the zero-mean random variables $v_i - \mathbb{E}[v_i]$ in combination with the assumption on the sample complexity to get

$$|v - \mathbb{E}[v]| \leq \mathcal{O}\left(\frac{1}{\psi n}\log\frac{1}{n} + \sqrt{\frac{\tilde{\xi}_f}{\psi n}\log\frac{1}{\delta}}\right) \leq \mathcal{O}(\tilde{\epsilon}_2).$$

Because

$$\mathbb{E}_{x,y,\omega}[v] = \frac{1}{|\Omega_l|}\hat{\sigma}(1/2)a_2(l)e^{i\pi b_1(l)},$$

we can now estimate $|\hat{b}_1(l) - b_1(l)|$. Let

$$\phi := \angle v - \angle\mathbb{E}[v] = \pi(\hat{b}_1(l) - b_1(l)).$$

Now if $\tilde{\epsilon}_2$ and thus also ϕ are small, Janzamin et al. [JSA15] use a small angle approximation on ϕ:

$$\phi \leq \tan(\phi) \sim \frac{|v - \mathbb{E}[v]|}{|\mathbb{E}[v]|}.$$

Then

$$|\hat{b}_1(l) - b_1(l)| \leq \frac{\phi}{\pi} \leq \frac{\tan(\phi)}{\pi} \leq \frac{|v - \mathbb{E}[v]|}{\pi|\mathbb{E}[v]|} \leq \frac{\mathcal{O}(\tilde{\epsilon}_2)}{\pi|\mathbb{E}[v]|} \leq \frac{|\Omega_l|}{\pi|\tilde{\sigma}(1/2)||a_2(l)|}\mathcal{O}(\tilde{\epsilon}_2)$$

where the last inequality follows from lemma (4.12). □

Conclusion

Finally, we can combine the results on the sample complexity for the approximation of A_1 (4.11) and b_1 (4.13) together with [JSA15, Lemma 12, 14] which give guarantees on identifying a_2 and b_2 via a ridge regression. This yields the overall bound on the sample complexity for the NN-LIFT algorithm which we stated in the beginning in theorem (4.7).
With the work of Anandkumar et al. we have examined a method which identifies the parameters of a shallow neural network exactly. The sample complexity scales only polynomially in d and k. Key tools used in the approximation are the decomposition of a moment tensor based on whitening and a tensor power iteration to obtain A_1 as well as a Fourier based method to estimate b_1.
In the next section (4.2) we will consider an alternative method of uniquely identifying the network weights with polynomial sample complexity. We will compare the two methods in (4.3).

4.2 Exact Weight Identification with Matrix Subspace Optimisation

This section presents an alternative approach to uniquely identifying neural network weights introduced by Fornasier et al. [FVD19], [FVD18]. They use only polynomially many samples. The identified weights serve in approximating the target function uniformly and hence the curse of dimensionality is avoided. Key themes and tools are numerical differentiation, stability properties of the SVD, whitening and spectral norm optimisation. The authors impose only mild regularity conditions on the weights and activation function.
As an introduction and as it is instructive in itself, we first briefly present an earlier result by Fornasier et al. [FSV12] in section (4.2.1). Their method introduces us to an approximation method based on numerical differentiation and *active sampling* which we will encounter again as part of the discussion of [FVD19] in section (4.2.5.1). In [FSV12] also *compressed sensing* techniques and a sparseness condition are used to learn the span of the weight vectors.
This approach is refined in [FVD19]. Here Fornasier et al. take the crucial step from approximating the span of the weights to identifying them uniquely and drop the sparseness condition. The main focus of this section will therefore be on [FVD19] starting in section (4.2.2). The following brief outline of [FSV12] serves as a motivation.

4.2.1 Introduction: Active Sampling for Weight Identification

Fornasier et al. [FSV12] make use of the numerical differentiation technique active sampling and the theory of compressed sensing to prove tractability of learning a one layer neural network with high probability.

Key Concepts: Numerical Differentiation, Compressed Sensing, Stability of the Singular Value Decomposition, Wedin's Bound

Their overall aim is to approximate a function given by a one layer neural network

$$f(x) = \sum_{i=1}^{k} c_i \sigma(a_i \cdot x + b).$$

Instead of considering this specific case, Fornasier et al. [FSV12] choose to consider the more general problem of approximating a sum of ridge functions which can be seen as

a generalisation of a shallow neural network. The target function hence takes the form

$$f(x) = \sum_{i=1}^{k} \sigma_i(a_i \cdot x) = \sigma(Ax) \tag{4.9}$$

where $f : B_d(1+\bar{\epsilon}) \to \mathbb{R}$, $\sigma \in C^2(AB_d(1+\bar{\epsilon}))$ and $A \in \mathbb{R}^{k\times d}$ contains the $a_i \in \mathbb{R}^d$ as rows and $x \in \mathbb{R}^d$ for d large. A *ridge function* is a function in $\mathbb{R}^d$ of the type $\sigma(a_i \cdot x)$. It is the most simple form of a multidimensional function and constant along the direction a_i. The a_i are referred to as *ridge directions* and $a_i \cdot x$ is seen as a projection. Traditionally the study of ridge functions is often referred to as projection pursuit in the literature [Hub85].
In our setting, both σ and the non-zero ridge directions are unknown. To make use of compressed sensing techniques, Fornasier et al. [FSV12] assume the weights are approximately sparse.

We briefly summarise the **assumptions** under which the problem becomes tractable.

1. The weight vectors $\{a_i : i \in [k]\}$ are orthonormal and approximately sparse:

$$\|a_i\|_q \leq C_1 \quad \text{for} \quad 0 < q \leq 1, i \in [k]. \tag{4.10}$$

2. $\sigma : \mathbb{R}^k \to \mathbb{R}$ is twice continuously differentiable: $\sup_{|\alpha|\leq 2} \|\partial^\alpha \sigma\|_\infty \leq C_2$

3. H_f is positive, semi-definite and has rank k.

$$\begin{aligned} H_f &:= \int_{\mathcal{S}^{d-1}} \nabla f(x) \nabla f(x)^T d\mu_{\mathcal{S}^{d-1}}(x) \\ s_1(H_f) &\geq \cdots \geq s_k(H_f) \geq \alpha > 0 \end{aligned} \tag{4.11}$$

for $\{s_i(H_f) : i \in [k]\}$ being the singular values of H_f in descending order.

Active Sampling

Fornasier et al. [FSV12] approximate the row space of A by using numerical differentiation and active sampling which gives access to the a_i by the chain rule. The same approach is later taken in [FVD19] as part of the unique identification of the weights and a more elaborate analysis is presented in section (4.2.5.1). Consider the Taylor expansion of the directional derivative of f at points x_j in directions ψ_i

$$\nabla\sigma(Ax_j)^T A\psi_i = \frac{\partial f}{\partial \psi_i}(x_j) = \frac{f(x_j + \epsilon\psi_i) - f(x_j)}{\epsilon} - \frac{\epsilon}{2}\psi_i^T \nabla^2 f(\xi_{ij})\psi_i$$

for $i \in [m_\Psi], j \in [m_X]$ and some appropriate $\xi_{ij} \in B_d(1+\bar{\epsilon})$. The derivative directions ψ_i are given by Bernoulli vectors. The evaluation points x_j are sampled actively, randomly and uniformly on the unit sphere in $\mathbb{R}^d$. Succinctly, we write

$$X := (A^T \nabla\sigma(Ax_1), \cdots, A^T \nabla\sigma(Ax_{m_x})) = A^T G^T$$

for appropriately defined G. We also write

$$\Psi X = Y - \mathcal{E}$$

where $\mathcal{E}_{ij} = \frac{\epsilon}{2}\psi_i^T \nabla^2 f(\xi_{ij})\psi_i$, $Y_{ij} = \frac{f(x_j+\epsilon\psi_i)-f(x_j)}{\epsilon}$ and the derivative directions collected in the matrix $\Psi \in \mathbb{R}^{m_\Psi \times d}$. We observe that the columns of X are linear combinations of the rows of A. By compressed sensing techniques the rows of X can under certain conditions be recovered through l_1 minimisation as solutions of

$$\hat{x}_j = \underset{y_j=\Psi z}{\arg\min} \|z\|_1$$

where y_j are the columns of Y. As conditions we enforce the approximate sparsity condition on a_i (4.10). Secondly, Ψ must fulfil the Restricted Isometry Property (RIP).

Definition 4.14 (Restricted Isometry Property)**.** *A matrix $\Psi \in R^{m_\Psi \times d}$ has the Restricted Isometry Property of order t with constant $\delta_t \in (0,1)$ if δ_t is the smallest constant such that*

$$(1-\delta_t)\|z\|_2^2 \leq \|\Psi z\|_2^2 \leq (1+\delta_t)\|z\|_2^2 \tag{4.12}$$

for all $z \in \mathbb{R}^d$ s.t. $\|z\|_0 \leq t$.

Indeed if Ψ is a Bernoulli matrix (i.e. a matrix of random entries that take on the values $\pm\sqrt{m_\Psi^{-1}}$ with equal probability) it fulfils the RIP with exponentially small failure probability [FSV12, Theorem 3.2]. It is also possible to choose other discrete distributions for the entries of Ψ which result in a sparse matrix. We point out that both Fornasier et al. [FSV12] and Sedghi et al. [SA14] (4.1.1) have proposed a method for tractable learning under sparsity assumptions.
If G^T as defined above is full rank, the row span of A is equal to the column span of the right singular vectors of X^T. Hence an approximation of the row span of A, but not the individual rows of A can be made [FSV12, Algorithm 2]. Ultimately an approximating function $\hat{f}$ is defined pointwise [FSV12]. A uniform bound on $|f-\hat{f}|$ which holds with high probability is found using Chernoff's bound for matrices (4.20), Wedin's bound (4.18) for the stability of singular spaces and the Lipschitz continuity of

σ [FSV12, Theorem 4.1]. A detailed account of the exact results is not given here, as the approach taken in [FVD19] is very similar, more universal and will be presented next.

4.2.2 Identification of Shallow Neural Networks by Fewest Samples

The previously discussed approach of Fornasier et al. in [FSV12] is refined in [FVD18]. Again the aim is to approximate a function given by a sum of ridge functions which can be seen as a generalisation of a one layer neural net

$$f(x) = \sigma(Ax) = \sum_{i=1}^{k} \sigma_i(a_i \cdot x) \tag{4.13}$$

where σ_i, a_i are unknown. Conditions are imposed to not only approximate f up to any accuracy, but to uniquely identify the weights and the activation function. Approximation is done in the uniform norm.

Key Concepts: Weak Differentiation, Passive Sampling, Stability of the Singular Value Decomposition, Wedin's Bound, Weyl's Inequality, Chernoff Bound for Random Matrices, Whitening, Quasi-Orthonormality, Spectral Norm Optimisation

The authors propose a **multi-step procedure:** First they approximate the row space of A (equivalently the span of the a_i) by numerical differentiation (4.2.5). Then they reduce the problem from input dimension d to k (4.2.4), the number of hidden units which can be detected if unknown. Further they approximate $\text{span}\{a_1 a_1^T, \cdots, a_k a_k^T\}$ (4.2.6.1) and prove that the a_i are unit length and quasi-orthonormal (see (4.23) for the definition) without loss of generality (4.2.7). Lastly they formulate and solve a suitable spectral optimisation problem which recovers the matrices $a_i a_i^T$ uniquely (4.2.6) and hence by SVD also the a_i.

4.2.3 Main result: Polynomial Sample Complexity

We now present the main result which states that f can be approximated uniformly using a number of samples which scales only polynomially in d and k. The theorem covers the case in which the active sampling procedure (4.2.5.1) is used instead of the alternative passive sampling (4.2.5.2). Throughout this section we prove the main result below, while detailing the necessary approximation steps.

Theorem 4.15 (Polynomial sample complexity). *[FVD19, Theorem 2.1]*
Under the assumptions (4.2.3.1) algorithms and methods (8), (4.2.6.1), (4.2.7.2), (9) and (10) approximate weight vectors $\{a_1, \cdots, a_k\}$ *up to a sign ambiguity by* $\{\hat{a}_1, \cdots, \hat{a}_k\}$ *from a maximum of* $m_X[(d+1)+(k+1)(k+2)/2]$ *random function evaluations*

$$\Big(\sum_{i=1}^{k} \|\hat{a}_i - a_i\|_2^2\Big)^{1/2} \lesssim \epsilon$$

with probability

$$1 - k \exp\left(-\frac{m_x c}{2\ max\{C_1, C_2\}^2 k^2}\right)$$

where $\epsilon, c > 0$ *and the asymptotic constant above contains* c *and some power of* k*. This facilitates a uniform approximation of* f *(see [FVD19] for details) in the order of* ϵ*:*

$$\|f - \hat{f}\|_{L^\infty(B_d} \lesssim \epsilon.$$

Comment: In the absence of noise, increasing m_x the number of function evaluations in active sampling (4.2.5.1), does not improve the quality of approximation but only decreases the failure probability. The approximation error is only determined by the active sampling method which approximates derivatives with classical finite differences.

4.2.3.1 Assumptions for the Main Result (4.15) with Active Sampling (4.2.5.1)

1. f is a sum of k ridge functions as in (4.13) with input $x \in B_d(1+\epsilon)$. The ridge direction vectors are linearly independent, so $k \leq d$. Without loss of generality they are assumed to be unit length. $\sigma_i : [-1,1] \to \mathbb{R}$ for $i \in [k]$.

2. $C_j := \max_{i\in[k]} \max_{-1\leq t\leq 1} |\sigma_i^{(j)}(t)| < \infty$ for $j \in \{0,1,2\}$.

3. Define the matrices

$$J[f] := \int_{\mathcal{S}^{d-1}} \nabla f(x) \nabla f(x)^T d\mu_{\mathcal{S}^{d-1}}(x) \tag{4.14}$$

and

$$H_2[f] := \int_{\mathcal{S}^{k-1}} \mathrm{vec}(\nabla^2 f(x)) \otimes \mathrm{vec}(\nabla^2 f(x)) d\mu_{\mathcal{S}^{k-1}}(x). \tag{4.15}$$

For both of them, the smallest singular value is bounded away from zero: $s_k(J[f]) \geq \alpha > 0$, $s_k(H_2[f]) \geq \alpha_2 > 0$. Hence $J[f]$ and $H_2[f]$ have full rank k. See preliminaries for a definition of the vectorisation operation.

We proceed by presenting the approximation steps taken in [FVD19] that we mentioned in the beginning.

4.2.4 Dimensionality Reduction d to k

With the next theorem, Fornasier et al. [FVD19] reduce the dimension of the problem from d to k which constitutes a potentially drastic improvement for high dimensional functions. As the theorem shows, this will necessitate an approximation of

$$A := \text{span}\{a_1, \cdots, a_k\}$$

$\tilde{A}$ which is derived in (4.2.5). A slight abuse of notation is committed here by denoting by A the weight matrix of the neural network, but also its row span.

Theorem 4.16 (Dimensionality reduction d to k)**.** *[FVD19, Theorem 1.1]*
For $k \leq d$ and input $x \in B_d$ define

$$\tilde{f}(y) := f(By) = \sum_{i=1}^{k} \sigma_i(a_i \cdot By) = \sum_{i=1}^{k} \sigma_i(\alpha_i \cdot y) \qquad y \in B_k \subset \mathbb{R}^k, \quad \alpha_i = B^T a_i \tag{4.16}$$

where $B = (b_1|\cdots|b_k) \in \mathbb{R}^{d\times k}$ for $\{b_i : i \in [k]\}$ an ONB spanning some k-dimensional subspace $\tilde{A} \subset \mathbb{R}^d$ is used to transform the input between $\mathbb{R}^d$ and $\mathbb{R}^k$. Then for any $\hat{f} : \mathbb{R}^k \to \mathbb{R}$ and any $\{\hat{\alpha}_1, \cdots, \hat{\alpha}_k\}$

$$\|f - \hat{f}(B^T\cdot)\|_\infty \leq \|f\|_{Lip}\|P_A - P_{\tilde{A}}\|_F + \|\tilde{f} - \hat{f}\|_\infty$$

and

$$\|a_i - B\hat{\alpha}_i\|_2 \leq \|P_A - P_{\tilde{A}}\|_F + \|\alpha_i - \hat{\alpha}_i\|_2$$

where $\|f\|_{Lip}$ denotes the Lipschitz constant of f.

Proof:
Assume $\hat{f}$ approximates $\tilde{f}$ (4.16) uniformly on B_k.
Claim 1: $\hat{f}(B^T\cdot)$ approximates f uniformly on B_d.
Proof:

$$\begin{aligned}
|f(x) - \hat{f}(B^Tx)| &\leq |f(x) - \tilde{f}(B^Tx)| + |\tilde{f}(B^Tx) - \hat{f}(B^Tx)| \\
&\leq |f(x) - f(BB^Tx)| + \|\tilde{f} - \hat{f}\|_\infty \\
&= |f(P_Ax) - f(P_{\tilde{A}}x)| + \|\tilde{f} - \hat{f}\|_\infty \\
&\leq \|f\|_{Lip} \cdot \|P_Ax - P_{\tilde{A}}(x)\|_2 + \|\tilde{f} - \hat{f}\|_\infty.
\end{aligned}$$

Taking the supremum on B_d gives

$$\|f - \hat{f}(B^T\cdot)\| \leq \|f\|_{Lip} \cdot \|P_A - P_{\tilde{A}}\|_\infty + \|\tilde{f} - \hat{f}\|_\infty.$$

Claim 2: Assuming good quality approximations $\hat{\alpha}_i \in \mathbb{R}^k$ of the $\alpha_i \in \mathbb{R}^k$ are available, the $a_i \in \mathbb{R}^d$ will be well approximated by the $B\hat{\alpha}_i$.
Proof: Using $B\alpha_i = BB^T a_i = P_{\tilde{A}} a_i$,

$$\begin{aligned}\|a_i - B\hat{\alpha}_i\|_2 &\leq \|a_i - B\alpha_i\| + \|B(\alpha_i - \hat{\alpha}_i)\|_2 \\ &= \|(P_A - P_{\tilde{A}})a_i\|_2 + \|B(\alpha_i - \hat{\alpha}_i)\|_2 \\ &\leq \|P_A - P_{\tilde{A}}\|_\infty + \|\alpha_i - \hat{\alpha}_i\|_2.\end{aligned}$$

□

This result can be exploited as for an approximation $\hat{f}$ of $\tilde{f}$ (in $\mathbb{R}^k$) and a good approximation of A by $\tilde{A}$ (as found in (4.2.5)) also the original high dimensional f is approximated well by by $\hat{f}(B^T\cdot)$ in $\mathbb{R}^d$. The original ridge directions are then well approximated if we can approximate the $\alpha_i \in \mathbb{R}^k$.

4.2.5 Approximation of $A = \mathbf{span}\{a_i : i \in [k]\}$

The dimensionality reduction procedure described above requires a good approximation of $A = \text{span}\{a_1, \cdots, a_k\}$. This can be achieved via active or passive sampling.

4.2.5.1 Active Sampling

As in [FSV12], Fornasier et al. make use of numerical differentiation which exposes the weights. By the chain rule,

$$\nabla f(x) = \sum_{i=1}^{k} \sigma_i(a_i \cdot x)a_i \in \text{span}\{a_i : i \in [k]\}, \tag{4.17}$$

so if the derivative is considered at sufficiently many points we can exhaust the space A. The derivative is evaluated at m_x number of points which are actively chosen. Hence, Fornasier et al. [FVD19] coin the phrase *active sampling*. Points are sampled uniformly on $\mathcal{S}^{d-1}$. Finite differences are used to construct an approximation of directional derivatives. The directions are taken to be the elements of the canonical basis e_j, contrary to the earlier work of Fornasier et al. [FSV12] (4.2.1) where the directions are

random Bernoulli vectors.

$$X_{j,l} := \frac{\partial}{\partial e_j} f(x_l) = \frac{f(x_l + \epsilon e_j) - f(x_l)}{\epsilon} - \left[\frac{\partial}{\partial e_j} f(x_l + \eta_{j,l} e_j) - \frac{\partial}{\partial e_j} f(x_l) \right] =: Y_{j,l} - \mathcal{E}_{j,l} \tag{4.18}$$

for some $\eta_{j,l} \in [0, \epsilon]$. The directional derivatives are collected in matrix $X \in \mathbb{R}^{d \times m_X}$, the approximate derivatives in $Y \in \mathbb{R}^{d \times m_X}$ and the error terms in $\mathcal{E} \in \mathbb{R}^{d \times m_X}$. The blockwise Singular Value Decomposition of X^T and Y^T plays an important role.

$$X^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} S_1 & 0 \\ 0 & S_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} \qquad Y^T = \begin{pmatrix} \tilde{U}_1 & \tilde{U}_2 \end{pmatrix} \begin{pmatrix} \tilde{S}_1 & 0 \\ 0 & \tilde{S}_2 \end{pmatrix} \begin{pmatrix} \tilde{V}_1^T \\ \tilde{V}_2^T \end{pmatrix} \tag{4.19}$$

where the k largest singular values are collected in S_1 and S_2 respectively. We observe that because of (4.17) A and the column span of X are identical. X is approximated by Y. It is hence natural to define $\tilde{A}$ using the singular vectors contained in $\tilde{V}_1^T$ that are associated with the k largest singular values.

Algorithm 8 Approximating A by active sampling, [FVD19, Algorithm 2.1]

1: Sample $\{x_i : i \in [m_x]\}$ uniformly on $\mathcal{S}^{d-1}$
2: Construct Y as in (4.18) using $m_x(d+1)$ function evaluations
3: Compute the SVD of Y^T as in (4.19)
4: Define $\tilde{A}$ as the row space of $\tilde{V}_1^T$

The space $\tilde{A}$ is a good approximation of the space A in the sense that $\|P_A - P_{\tilde{A}}\|$ is small. The authors proceed to bound this quantity.

Theorem 4.17 (Approximation of A by active sampling). *[FVD19, Theorem 2.2] Under the assumptions (4.2.3.1) the approximation of A by $\tilde{A}$ as in (8) satisfies*

$$\|P_A - P_{\tilde{A}}\|_F \leq \frac{2C_2 \epsilon k}{\sqrt{\alpha(1-s)} - C_2 \epsilon k}$$

with probability at least $1 - k \exp(-\frac{m_x \alpha s^2}{2k^2 C_1^2})$ for $s \in (0,1)$, C_2 bounding the Lipschitz constants of all the σ_i' for $i \in [k]$ and α, C_1 as in (4.2.3.1).

The proof utilises some deep results namely Wedin's Bound, Weyl's inequality and Chernoff's Bound.

Theorem 4.18 (Wedin's bound). *[Wed72] Consider the SVDs as in (4.19). If there is an $\bar{\alpha}$ such that*

$$\min_{l,\tilde{l}} |s_{\tilde{l}}(\tilde{S}_1) - s(S_2)| \geq \bar{\alpha} \qquad \text{and} \qquad \min_{\tilde{l}} |s_{\tilde{l}}(\tilde{S}_1)| \geq \bar{\alpha},$$

then

$$\|V_1V_1 - \tilde{V}_1\tilde{V}_1^T\| \leq \frac{\sqrt{2}}{\bar{\alpha}}\|X^T - Y^T\|_F.$$

Theorem 4.19 (Weyl's inequality). *[Wey12]*
Consider the SVDs as in (4.19).

$$|s_k(X^T) - s_k(Y^T)| \leq \|X - Y\|_F.$$

Theorem 4.20 (Chernoff's Bound). *[Tro12, Corollary 5.2, Remark 5.3]*
For matrices $X_1, \cdots, X_n \in \mathbb{R}^{k\times k}$ that are random, independent and positive semidefinite with $s_1(X_j) \leq C$ a.s. for $j \in [n]$ it holds that

$$P\left\{s_k\Big(\sum_{j=1}^n X_j\Big) - s_k\Big(\sum_{j=1}^n \mathbb{E}X_j\Big) \leq -ts_k\Big(\sum_{j=1}^n \mathbb{E}X_j\Big)\right\} \leq k\exp\left(-\frac{s_k\Big(\sum_{j=1}^n \mathbb{E}X_j\Big)t^2}{2C}\right)$$

for all $t \in (0,1)$ with s_l denoting the lth largest singular value.

Proof of theorem (4.17):
Applying the results above it holds that

$$\|P_A - P_{\tilde{A}}\| \overset{SVD}{=} \|V_1V_1 - \tilde{V}_1\tilde{V}_1^T\| \overset{(4.18)}{\leq} \sqrt{2}\frac{\|X-Y\|_F}{s_k(Y^T)} \overset{(4.19)}{\leq} \frac{\sqrt{2}\|X-Y\|_F}{s_k(X^T) - \|X-Y\|_F}.$$

Fornasier et al. [FVD19] estimate the terms in the upper bound. Using some algebraic manipulations and the Lipschitz continuity of σ', $\|X - Y\|_F = \|\mathcal{E}\|_F$ can be bounded by $C_2\epsilon\sqrt{m_x}k$.
Claim: With probability at least $1 - k\exp(-\frac{m_x\alpha s^2}{2k^2C_1^2})$ it holds that [FVD19, Lemma 2.3]

$$s_k(X^T) \geq \sqrt{m_x\alpha(1-s)}.$$

Proof of the Claim: The aim here is to apply the Chernoff bound (4.20) in a constructive way. Let $\{w_1, \cdots, w_k\}$ be an ONB for A, so $P_A = (w_1^T \cdots w_k^T)^T$. By properties of the SVD,

$$s_j(X) = s_j(P_AX) = \sqrt{s_j(P_AXX^TP_A^T)}.$$

Now we can calculate

$$XX^T = \sum_{i=1}^{m_x} \nabla f(x_i)\nabla f(x_i)^T \qquad \text{and} \qquad P_AXX^TP_A^T = \sum_{i=1}^{m_x} P_A\nabla f(x_i)\nabla f(x_i)^TP_A^T.$$

Furthermore,

$$\begin{aligned} s_1(P_A \nabla f(x) \nabla f(x)^T P_A^T) &= s_1(\nabla f(x) \nabla f(x)^T) \overset{\text{rank } 1}{=} \|\nabla f(x) \nabla f(x)^T\|_F \\ &= \|\nabla f(x)\|_2^2 = \sum_{l=1}^{d} \Big(\sum_{i=1}^{k} \sigma_i'(a_i \cdot x) a_{i,l} \Big)^2 \\ &\overset{(4.2.3.1)}{\leq} C_1^2 \sum_{l=1}^{d} \Big(\sum_{i=1}^{k} |a_{i,l}| \Big)^2 \leq C_1^2 \Big(\sum_{i=1}^{k} \sum_{l=1}^{d} |a_{i,l}|^2 \Big)^2 \leq C_1^2 k^2 \end{aligned}$$

where we have used in the second equality that $\nabla f(x) \nabla f(x)^T$ has rank 1. Hence the Chernoff Bound (4.20) can be applied to $X_j = P_A \nabla f(x_j) \nabla f(x_j)^T P_A^T$ with $C = C_1^2 k^2$. By definition $\mathbb{E} X_j = P_A J[f] P_A^T$, so $s_k(\sum_{j=1}^{m_x} \mathbb{E} X_j) \geq \alpha m_x$. Finally,

$$s_k(X) = \sqrt{s_k(P_A X X^T P_A^T)} \geq \sqrt{s_k(\sum_{j=1}^{m_x} \mathbb{E} X_j)(1-s)} \geq \sqrt{m_x \alpha (1-s)}$$

with probability at least

$$1 - k \exp\Big(- \frac{s_k(\sum_{j=1}^{m_x} \mathbb{E} X_j) s^2}{2 C_1^2 k^2} \Big) \geq 1 - k \exp\Big(- \frac{m_x \alpha s^2}{2 C_1^2 k^2} \Big).$$

Plugging in the upper bound of $s_k(X)$ and $\|X - Y\|_F$ yields the statement of the theorem. □

Comment: For ϵ small enough and m_X large enough there is a gap in the spectrum of Y somewhere with high probability [FVD19]. This can be used to detect k.

4.2.5.2 Passive Sampling

The alternative approach to approximating A is passive sampling [FVD19, Section 2.2.2]. Here samples are assumed to be available, but not actively chosen according to some distribution. They are also corrupted by noise. This setting is a more realistic reflection of reality than the active sampling approach and avoids instabilities due to the numerical differentiation in active sampling [FVD19]. As in [SA14], [JSA15], we assume that $p(x)$ the density of the input x is known.

Assumptions

1. Points $\{x_1, \cdots, x_{m_x}\}$ are given to us that are sampled independently and corrupted by noise.

2. The pdf $p(x)$ of input x is smooth and known, $\text{supp}(p) \subset B_d$.

3. There is a probability space $(\mathcal{V}, \pi)$ and a family of C_c^1 compactly supported functions $\phi_\nu : \mathbb{R}^d \to \mathbb{R}$ for $\nu \in \mathcal{V}$ such that

$$\max_{\nu \in \mathcal{V}} \max_{x \in B_d} \| \frac{\nabla \phi_\nu(x)}{p(x)} \| \leq C_\nu$$

and

$$J_\nu[f] := \int_{\mathcal{V}} \Big(\int_{\mathbb{R}^d} \nabla f(x) \phi_\nu(x) dx \Big) \Big(\int_{\mathbb{R}^d} \nabla f(x) \phi_\nu(x) dx \Big)^T d\pi(\nu)$$

has full rank. Let the smallest singular value of $J_{\mathcal{V}}[f]$ be bounded below by $s_k(J_{\mathcal{V}}[f]) \geq \alpha > 0$.

4. The $\{a_i : i \in [k]\}$ are linearly independent and unit length.

5. $C_j := \max_{i \in [k]} \max_{-1 \leq t \leq 1} |\sigma_i^{(j)}(t)| < \infty$ for $j \in \{0, 1\}$.

Assume the conditions above hold. One way of constructing the function family ϕ_ν as described in the conditions and at the same time creating a link between $p(x)$ and the ϕ_ν is by picking a resolution of the identity which is bounded. Fornasier et al. [FVD19, Remark 4] pick a family of non-negative, smooth and compactly supported functions $\psi_\nu \geq 0$ with $\int_{\mathcal{V}} \psi_\nu(x) d\pi(\nu) = 1$ for all $x \in B_d$ and $\max_{x \in B_d} \{|\psi_\nu(x)|, \|\nabla \psi_\nu(x)\|_2\} \leq C_\Psi$. For discrete $\mathcal{V}$ the reader will recognise the family ψ_ν as a partition of the unity. The authors pick also some bounded, non-negative and smooth function $q(x)$ with

$$\max_{x \in B_d} \Big\{ \frac{q(x)}{p(x)}, \| \frac{\nabla q(x)}{p(x)} \|_2 \Big\} \leq C_q$$

and define

$$\phi_\nu(x) = \psi_\nu(x) q(x)$$

which fulfil the assumptions detailed above.
Fornasier et al. [FVD19] consider the quantity

$$Y_j = -\frac{1}{N} \sum_{l=1}^{N} (f(x_l) + n_l) \frac{\nabla \phi_{\nu_j}(x_l)}{p(x_l)} \approx -\int_{\mathbb{R}^d} f(x) \frac{\nabla \phi_{\nu_j}(x)}{p(x)} p(x) dx \tag{4.20}$$

$$= \int_{\mathbb{R}^d} \nabla f(x) \phi_{\nu_j}(x) dx \tag{4.21}$$

$$= \sum_{i=1}^{k} \Big(\int_{\mathbb{R}^d} \sigma'(a_i \cdot x) \phi_{\nu_j}(x) dx \Big) a_i \tag{4.22}$$

where ν_j for $j \in [m_x]$ have been sampled iid according to π. The noise in the point evaluations of f at the x_l is denoted by n_l which are independent, bounded and zero-mean. The penultimate step in the derivation above is due to an application of integration by parts and offers a link to weak differentiation. We want to point out here that the passive sampling approach by Fornasier et al. [FVD19] shows very interesting parallels to the application of Stein's Lemma in theorem (4.9) by Janzamin et al. [JSA15] as discussed in the previous section (for a more detailed discussion on this parallel see (4.3)).

From the derivation above it can be seen that the Y_j are approximately in the span of the a_i. By the conditions imposed on ϕ, the span of the a_i can be exhausted if Y_j is considered for different j corresponding to different samples ν_j. Now quite similar to the active sampling case, vectors Y_j for $j \in [m_x]$ are collected as columns in a matrix $Y_\mathcal{V} \in \mathbb{R}^{d \times m_x}$. $X_\mathcal{V} \in \mathbb{R}^{d \times m_x}$ has as columns the vectors

$$X_j = \int_{\mathbb{R}^d} \nabla f(x) \phi_{\nu_j}(x) dx.$$

Now A can be approximated as in the case of active sampling by replacing Y by $Y_\mathcal{V}$ in the algorithm (8). Fornasier et al. [FVD19, Theorem 2.4] derive bounds on $\|P_A - P_{\tilde{A}}\|_F$. As the bounds and and the tools used to prove them are very similar to the ones in theorem (4.17) we do not go into detail here.

Passive sampling avoids instabilities that arise as a result of numerical differentiation in active sampling. Utilising given sampling points rather than actively choosing the samples also reflects more closely reality.

4.2.6 Exact Identification via Spectral Norm Optimisation

How could one possibly go from approximating the span of the a_i to approximating the individual a_i? Fornasier et al. [FVD19, Section 3] facilitate this by considering not just $A = \text{span}\{a_i : i \in [k]\}$, but

$$\mathcal{A} = \text{span}\{a_i a_i^T : i \in [k]\}.$$

The other key to exact weight identification is proving that without loss of generality the a_i can be assumed to be quasi-orthonormal (4.2.7) in the sense of definition (4.23). Then also the $a_i a_i^T$ are quasi-orthonormal. The authors show that this is enough for unique identification of the $a_i a_i^T$ and thus of the a_i via SVD. The exact $\mathcal{A}$ is not available and will have to be approximated by $\tilde{\mathcal{A}}$ as in section (4.2.6.1). Assuming quasi-orthonormality results in a geometric separation of the $a_i a_i^T$. Indeed, searching within

$\{M \in \mathcal{A} : \|M\|_F \leq 1\}$ for matrices of maximal spectral norm will yield the exact $a_i a_i^T$. Consider the following optimisation problem:

$$\arg\max \|M\|_\infty \quad \text{such that} \quad M \in \tilde{\mathcal{A}}, \|M\|_F \leq 1. \tag{4.23}$$

Notice that any solution will lie on the boundary of the problem, so it will have Frobenius norm 1 (because a convex function is maximised over a compact set). Denoting for the moment by s_i the singular values of some $M \in \mathcal{A}$ it holds that

$$\|M\|_F = \Big(\sum_{i=1}^{k} s_i^2\Big)^{\frac{1}{2}} \quad \text{and} \quad \|M\|_\infty = \max_{i\in[k]} |s_i|.$$

So $\|M\|_\infty \leq \|M\|_F \leq 1$ on $\{M \in \mathcal{A} : \|A\|_F \leq 1\}$. Hence for any maximiser to the optimisation problem (4.23),

$$1 = \|M\|_F = \Big(\sum_{i=1}^{k} s_i^2\Big)^{\frac{1}{2}} \simeq \max_{i\in[k]} |s_i| = \|M\|_\infty.$$

M will be close to 1 in spectral norm, hence close to rank 1. By linear independence of the a_i, the only elements of rank 1 in the span of the $\{a_i a_i^T : i \in [k]\}$ will be the individual $a_i a_i^T$. So any maximiser will be necessarily close to one of the $a_i a_i^T$:

$$\arg\max_{\substack{\|M\|_F\leq 1 \\ M\in\tilde{\mathcal{A}}}} \|M\|_\infty \simeq \{a_i a_i^T\} \in \{\|M\|_F = \|M\|_\infty = 1\}. \tag{4.24}$$

Spectral Gap for Stability of the SVD

Because there will be a gap between the first singular value and the others, the singular value decomposition is stable. Indeed, stability of the singular values is guaranteed by Weyl's Theorem (4.19) even if there is no spectral gap. Singular spaces which are characterised by the singular vectors are in general unstable unless there is a spectral gap as is proved in Wedin's Theorem (4.18). This is in fact the reason why formulating an optimisation problem such as the problem (4.23) is indeed necessary for the identification of the a_i [FVD19]. Simply calculating the SVD of some $M \in \tilde{\mathcal{A}}$ will not be enough. While

$$M \simeq P_{\mathcal{A}}(M) = \sum_{i=1}^{k} \lambda_i a_i a_i^T \simeq \sum_{i=1}^{k} b_i b_i^T,$$

for some $\lambda_i \in \mathbb{R}$ and $\{b_i : i \in [k]\}$ an ONB close to the quasi-orthonormal set $\{a_i : i \in [k]\}$ (4.23), calculating the SVD of M will not be sufficient, as it is not in general close

to the SVD stated on the right hand side. Stability of the SVD only holds if there is a gap in the spectrum of M. This is being enforced by Fornasier et al. [FVD19] through the optimisation problem (4.23).

Outline of the Strategy

The strategy pursued by Fornasier et al. [FVD19] for exact weight identification is to first approximate $\mathcal{A}$ as done in section (4.2.6.1). Then they prove quasi-orthonormality (see definition (4.23)) of the a_i without loss of generality in section (4.2.7) and hence also the quasi-orthonormality of the $a_i a_i^T$. This holds in the case that only approximations of A and $\mathcal{A}$ are available. Then the authors solve the optimisation problem (4.23) with algorithm (9).

4.2.6.1 Approximation of $\mathcal{A} = \mathbf{span}\{a_i a_i^T : i \in [k]\}$

This section details how to approximate $\mathcal{A} = \text{span}\{a_i a_i^T : i \in [k]\}$. To this end, Fornasier et al. [FVD19] extend ideas from the approximation of A in the previous section (4.2.5). Only the case of active sampling is presented here. The extension of passive sampling from approximating A to approximating $\mathcal{A}$ follows an analogous line of thought (see [FVD19] for details). Having approximated A and used this to reduce the dimension of the problem from d to k, we can restrict the discussion to the case $d = k$.

Active Sampling

Analogous to the approximation of A by active sampling, $\mathcal{A}$ can be similarly approximated using second order derivatives and finite differences [FVD19, Section 3.1.2]. Due to the twofold application of the chain rule $\nabla^2 f(x)$ lies in the span of the $a_i a_i^T$. Y is constructed similarly to before using a vectorisation (see preliminaries for the definition) of the matrix containing the second order finite difference. Hence, $H_2[f]$ is replaced by $J[f]$. The authors arrive at an analogous algorithm to approximate $\mathcal{A}$ as algorithm (8) (see [FVD19, Algorithm 3.1] for details). As the analysis of this is very similar to section (4.2.5.1) and uses the same mathematical tools, we only state the final result here.

Theorem 4.21 (Approximation of $\mathcal{A}$). *[FVD19, Theorem 3.2]*
Under the assumptions (4.2.3.1) construct $\mathcal{A}$ as described above using $m_x[(k+1)(k+2)/2]$ point evaluations of f. Then

$$\|P_{\mathcal{A}} - P_{\tilde{\mathcal{A}}}\|_{F\to F} \leq \frac{4C_3 k\epsilon}{\sqrt{\alpha_2(1-s)} - 2C_3 k\epsilon}$$

for $s \in (0,1)$ with probability at least $1 - k\exp\left(-\frac{m_x \alpha_2 s^2}{2k^2 C_2^2}\right)$. Here $\|\cdot\|_{F\to F}$ is the operator norm associated to the Frobenius norm and C_3 bounds the Lipschitz constant of σ_i'' for $i \in [k]$.

4.2.7 Whitening

To achieve a geometric separation of the a_i which we can exploit in optimisation problem (4.23), the a_i need to be quasi-orthonormal (4.23). Fornasier et al. [FVD19] prove that we can assume quasi-orthonormality of the a_i without loss of any generality by designing a whitening procedure. For this, the authors have drawn inspiration also from the whitening procedure by Janzamin et al. [JSA15] which we discussed before in section (4.1.5).

Motivation

Let us assume that we are in the idealised case where $\mathcal{A} = \tilde{\mathcal{A}}$. Then $G \in \mathcal{A}$ is strictly positive i.e.

$$G = \sum_{i=1}^{k} \lambda_i a_i a_i^T \succ 0 \quad \text{or} \quad \langle Gv, v\rangle > 0 \qquad \forall v \neq 0$$

iff the a_i are linearly independent and the λ_i are strictly positive [FVD19, Lemma 3.5].

Proposition 4.22 (Whitening matrix). *[FVD19, Proposition 3.6]*
Consider some symmetric and positive definite $G \in \mathcal{A}$

$$G = \sum_{i=1}^{k} \lambda_i a_i a_i^T$$

i.e. the a_i are linearly independent and $\lambda_i > 0$ for $i \in [k]$ [FVD19, Lemma 3.5]. Consider the SVD of G, $G = UDU^T$ where U is orthogonal and D is diagonal. Then

$$\left\{\sqrt{\lambda_i} D^{-\frac{1}{2}} U^T a_i : i = 1, \cdots, k\right\}$$

is an ONB. Call

$$W := D^{-\frac{1}{2}} U$$

*the **whitening matrix**. Then we have an orthogonal resolution of the identity:*

$$I_k = WG^TW = \sum_{i=1}^{k} Wa_i \otimes Wa_i.$$

Proof:
$G = UDU^T = \sum_{i=1}^{k} \lambda_i a_i a_i^T = AD_\lambda A^T$ for $A = (a_1 \mid \cdots \mid a_k)$, $D_\lambda = \mathrm{Diag}(\lambda_i : i \in [k])$ and $\lambda_i > 0$ for $i \in [k]$. Collecting the $\sqrt{\lambda_i} Wa_i$ as columns in a matrix results in a matrix $WAD_{\sqrt{\lambda}}$ for $W := D^{-\frac{1}{2}}U^T$. $WAD_{\sqrt{\lambda}}$ is orthonormal, because

$$\begin{aligned}(WAD_{\sqrt{\lambda}})(WAD_{\sqrt{\lambda}})^T &= WAD_{\sqrt{\lambda}}D_{\sqrt{\lambda}}A^TW^T = WAD_\lambda A^TW^T \\ &= WGW^T = (D^{-1/2}U^T)(UDU^T)(D^{-1/2}U^T)^T = I_k.\end{aligned}$$

□

Then

$$f(W^Tx) = \sum_{i=1}^{k} \sigma_i(Wa_i \cdot x) = \sum_{i=1}^{k} \tilde{\sigma}_i(\sqrt{\lambda_i} Wa_i \cdot x)$$

for $\tilde{\sigma}_i(\cdot) = \sigma_i(\lambda_i^{-1/2}\ \cdot)$ and we can thus assume without loss of generality that the weight vectors form an orthonormal system though they are unknown. However this holds only if we can access an element of $\mathcal{A}$.

The perturbed case $\mathcal{A} \neq \tilde{\mathcal{A}}$

Even though an exact element of $\mathcal{A}$ is not available, we have a good approximation of $\mathcal{A}$, $\tilde{\mathcal{A}}$, as found in section (4.2.6.1). Fornasier et al. [FVD19] aim to **construct** $\tilde{G} \in \tilde{\mathcal{A}}$ s.t. $\tilde{G} \succeq \gamma I_k$ to use this for whitening instead of G. This is achieved in section (4.2.7.1). The resulting set of vectors will not be orthonormal as in the unperturbed case, but will be quasi-orthonormal.

Definition 4.23 (Quasi-orthonormality)**.** *[FVD19, Definition 1.1]*
Unit vectors $a_1, \cdots, a_k$ are quasi-orthonormal if

$$\mathcal{S}(a_1, \cdots, a_k) := \inf\left\{ \left(\sum_{i=1}^{k} \|a_i - b_i\|_2^2 \right)^{\frac{1}{2}} : b_1, \cdots, b_k \text{ is an ONB in } \mathbb{R}^k \right\} < \epsilon.$$

Theorem 4.24 (Quasi-orthonormality after perturbed whitening)**.** *[FVD19, Theorem 3.7]*
Assume some $\tilde{G} \in \tilde{\mathcal{A}}$ is known which is positive definite and let $G := P_{\mathcal{A}}(\tilde{G})$ with $\tilde{G} \succeq \gamma I_k$ for $\gamma > 0$. Let $G = \sum_{i=1}^{k} \lambda_i a_i \otimes a_i = UDU^T$ and $\tilde{G} = \tilde{U}\tilde{D}\tilde{U}^T$ be the singular value

decompositions. Assume $\|P_{\mathcal{A}} - P_{\tilde{\mathcal{A}}}\| \leq \eta$ *for* $\eta > 0$. *Then*

$$\mathcal{S}(\sqrt{\lambda_1}\tilde{W}a_1, \cdots, \sqrt{\lambda_k}\tilde{W}a_k) \leq \frac{\eta\|\tilde{G}\|_F}{\gamma}.$$

Furthermore,

$$\left\{\frac{\tilde{W}a_1}{\|\tilde{W}a_1\|_2}, \cdots, \frac{\tilde{W}a_k}{\|\tilde{W}a_k\|_2}\right\}$$

is ϵ*-quasi-orthonormal in the sense of (4.23) for* $\epsilon := \frac{\sqrt{2}\eta\|\tilde{G}\|_F}{\gamma}$.

Idea of the proof:
The proof relies on calculating $\mathcal{S}(\sqrt{\lambda_1}\tilde{W}a_1, \cdots, \sqrt{\lambda_k}\tilde{W}a_k) \leq \frac{\eta\|\tilde{G}\|_F}{\gamma}$ from which then the second assertion follows by algebraic calculations. The proof relies on a consequence of the SVD. For any matrix A with singular values $\{s_i : i \in [k]\}$ it holds that [FVD19, Theorem 6.2(i)]

$$\mathcal{S}(a_1, \cdots, a_k) = \left(\sum_{i=1}^{k}(s_i^2 - 1)^2\right)^{1/2}.$$

This can be thought of as comparing A to I_k or comparing $\{a_i : i \in [k]\}$ to the canonical basis. □

Simple algebraic manipulations now show that if $\{a_i : i \in [k]\}$ is quasi-orthonormal with quasi-orthonormality parameter ϵ, then $\{a_i a_i^T : i \in [k]\}$ is quasi-orthonormal with parameter 2ϵ [FVD19, Lemma 6.3(iv)].

4.2.7.1 Constructing positive definite $\tilde{G} \in \tilde{\mathcal{A}}$

Applying theorem (4.24) in practice necessitates finding a positive definite matrix $\tilde{G}$ within the the subspace $\tilde{\mathcal{A}} \subset \mathbb{R}^{k\times k}$. This $\tilde{G}$ should have minimal eigenvalue as large as possible, so that the whitening basis is as close to being orthonormal as possible by the form of theorem (4.24). Fornasier et al. [FVD19] phrase this search as a convex optimisation problem. They define the concave function $l(\tilde{A})$:

$$l(\tilde{A}) := \min_{\substack{x\in\mathbb{R}^k \\ \|x\|_2=1}} x^T\tilde{A}x. \tag{4.25}$$

Then $-l(\tilde{A})$ is a convex function and the authors formulate the following convex minimisation problem.

$$\alpha := \min_{\substack{\tilde{A}\in\tilde{\mathcal{A}} \\ \|\tilde{A}\|_F\leq 1}} -l(\tilde{A}) \tag{4.26}$$

The optimiser of this problem will be the element of $\tilde{\mathcal{A}}$ which has the largest minimal eigenvalue.

4.2.7.2 Iterative Whitening

If $\alpha < 0$ in the above optimisation problem, then there is a strictly positive matrix within $\tilde{\mathcal{A}}$ [FVD19, Theorem 3.8]. Fornasier et al. [FVD19, Remark 6] point to methods in the literature for solving the optimisation problem. Having found a positive definite $\tilde{G}$ of maximal minimal eigenvalue we can apply theorem (4.24) and define

$$f(\tilde{W}^T x) = \sum_{i=1}^{k} \sigma_i(\tilde{W}a_i \cdot x) = \sum_{i=1}^{k} \tilde{\sigma}_i(\sqrt{\tilde{\lambda}_i}\tilde{W}a_i \cdot x) = \tilde{f}(x)$$

for $\tilde{\sigma}_i(\cdot) = \sigma_i(\lambda_i^{-1/2}\cdot)$ and $\tilde{\lambda}_i = 1/\|\tilde{W}a_i\|_2^2$. Hence, without loss of any generality it can be assumed that the weight vectors of a given one layer neural network are quasi-orthonormal. This whitening procedure can be iteratively repeated, so that the ϵ parameter of quasi-orthonormality decreases more and more. This is reported to be particularly useful if η is comparatively large i.e. if the projections onto $\mathcal{A}$ and $\tilde{\mathcal{A}}$ are not too close. It η is very small, then Fornasier et al. [FVD19] report the unique identification of the weights to be rather robust and the role of ϵ to be less important. Then the repetition of the whitening process described above plays only a subordinate rule. Considering the dependence of the quasi-orthonormality constant in theorem (4.24) on η, it becomes clear that we need η to be small as a bound of $\|P_{\mathcal{A}} - P_{\tilde{\mathcal{A}}}\|$. $\tilde{G}$ needs to be well enough conditioned for a large γ.

4.2.7.3 The Optimisation Problem (4.23)

Let us state again the optimisation problem (4.23):

$$\arg\max \|M\|_\infty \quad \text{such that} \quad M \in \tilde{\mathcal{A}}, \|M\|_F \leq 1. \tag{4.27}$$

Fornasier et al. [FVD19, Theorem 3.16] show that any solution is necessarily very close to one of the elements of $\{a_i a_i^T : i \in [k]\}$.

An algorithmic solution to optimisation problem (4.23)

Note that any optimiser of problem (4.23) will have a *spectral gap* between the largest singular value and the remaining ones. It will be in $\tilde{\mathcal{A}}$ and have $\|M\|_F = 1$. Fornasier et al. [FVD19] construct an algorithm which enforces these three properties and solves (4.23):

Algorithm 9 Solving (4.23), [FVD19, Algorithm 3.4]

input $\gamma > 1$, random $X^{(0)} \in \tilde{\mathcal{A}}$ s.t. $\|X^0\|_F = 1$

1: **for** $l \geq 0$ **do**
2: $\quad X^{(l+1)} := P_{\tilde{\mathcal{A}}} \Pi_\gamma(X^{(l)})$
3: **end for**

The projection $P_{\tilde{\mathcal{A}}}$ enforces that the algorithm output will be in $\tilde{\mathcal{A}}$. The operator Π_γ iteratively creates a spectral gap and forces the Frobenius norm to be 1:

$$\Pi_\gamma(X) := \frac{1}{\sqrt{\gamma^2 s_1^2 + s_2^2 + \cdots + s_k^2}} U \begin{pmatrix} \gamma s_1 & 0 & \cdots & 0 \\ 0 & s_2 & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & s_k \end{pmatrix} V^T$$

for the SVD of X, $X = USV^T$ with $s_1 \geq \cdots \geq s_k$. The operator is well defined if its argument has the shape $\sum_{i=1}^k \lambda_i a_i a_i^T$ for linearly independent $\{a_i : i \in [k]\}$ and strictly positive λ_i [FVD19, p. 44]. If this is the case, Π_γ scales the first singular value by some $\gamma > 1$ and renormalises the matrix to Frobenius norm 1. Overall it increases the gap between the first and the second singular value.

Any solution to optimisation problem (4.23) found by algorithm (9) can be used to uniquely identify one of the a_i through the following algorithm.

Algorithm 10 Unique identification of a_i, [FVD19, Algorithm 3.3]

input M an optimiser of (4.23) found by (9)

1: **if** $\|M\|_\infty$ is not an eigenvalue of M **then**
2: $\quad$ replace M by $-M$
3: **end if**
4: Compute eigenvalue decomposition of $M = \sum_{j=1}^k s_j u_j u_j^T$ s.t. $s_1 \geq \cdots \geq s_k$

output $\hat{a} := u_1$

The iteration

$$X^{(l+1)} = P_{\tilde{\mathcal{A}}} \Pi_\gamma(X_l)$$

above converges to a solution of problem (4.23) as Π_γ represents a subgradient step towards maximising the spectral norm and $P_{\tilde{\mathcal{A}}}$ projects back onto $\tilde{\mathcal{A}}$ which is a condition on the solution of optimisation problem (4.23). For a rigorous analysis of the algorithms we refer the reader to [FVD19, Theorem 3.12, 3.16].

Analysis of the problem (4.23): First and second order optimality conditions

Although the infinity norm is not a smooth function, Fornasier et al. [FVD19] analyse the optimisation problem (4.23) using differentiation methods and consider first and second order optimality conditions.

Theorem 4.25 (Optimality conditions for (4.23))**.** *[FVD19, Theorem 3.4]*
For M a local maximiser of (4.23) we have

$$u_j^T X u_j = 0 \quad \forall X \in \mathcal{S}_{\tilde{A}}, X \perp M \quad \forall j \in [k] \text{ s.t. } |s_j| = \|M\|_\infty \tag{4.28}$$

for $\mathcal{S}_{\tilde{A}} := \{M \in \tilde{\mathcal{A}} : \|M\|_F = 1\}$.
If ϵ-quasi-orthonormality of the a_i holds, i.e. $\mathcal{S}(a_1, \cdots, a_k) \leq \epsilon$, and also $3m\|P_{\mathcal{A}} - P_{\tilde{\mathcal{A}}}\| < (1-\epsilon)^2$, then

$$|s_1| = \|M\|_\infty \quad \text{with} \quad s_1 \notin \{s_2, \cdots, s_k\}$$

and

$$2\sum_{i=2}^{k} \frac{(u_1^T X u_l)^2}{|s_1 - s_l|} \leq |s_1| \quad \forall X \in \tilde{\mathcal{A}}, \|X\|_F = 1, X \perp M, l \neq 1 \tag{4.29}$$

Comments on the proof:
It is enough to consider $X \perp M$ without loss of generality (see [FVD19, Remark 5] for details). The proof of the first order optimality condition in the theorem relies on considering the Taylor expansion of the map

$$\gamma \to \frac{\|M + \gamma X\|_\infty}{\|M + \gamma X\|_F}$$

for M some maximiser of optimisation problem (4.23), $X \in \tilde{\mathcal{A}}$, $X \perp M$. The proof exploits the eigenvalue decomposition of M and the fact that the map has a local maximum in $\gamma = 0$.
The second order optimality condition follows again from the Taylor expansion in combination with the Lagrangian formulation.

The second order optimality condition (4.29) indicates that any optimiser M will have a spectral gap between the first singular value and the smaller singular values. From the two optimality conditions it can be derived that any output of the algorithm (9) can indeed be used to approximate the a_i as in algorithm (10) (see [FVD19] for details). In practice we can test any output of the algorithm (9) against the optimality conditions to check if they are optimisers of problem (4.23). [FVD19, Section 4] discusses the approximation of the actual target function f using the identified network parameters.

Finally, combining all the approximation steps (8) (4.2.6.1) (4.2.7.2), (9), (10) we outlined and analysed in sections (4.2.4), (4.2.5), (4.2.6), (4.2.7), (4.2.7.3) yields the main theorem of Fornasier et al. [FVD19] (4.15) which we set out to prove at the beginning of the section.

4.3 Comparison

In this final section we highlight similarities and differences between the approaches of Anandkumar et al. (4.1) and Fornasier et al. (4.2). They both uniquely identify the parameters of a shallow neural network using polynomially many samples.

Unlike Fornasier et al. [FVD19], Janzamin et al. [JSA15] learn the first layer bias b_1 independently from the multiplicative weights A_1 using a Fourier method. Janzamin et al. [JSA15] learn the parameters of a second and final layer using regression. Both Janzamin et al. [JSA15] and Fornasier et al. [FVD19] assume that the input x is distributed according to some distribution whose density $p(x)$ is known or sufficiently well estimated (see for example [Ana+14], [Sri+13], [Dev85] for estimation methods). They show that this prior information along with some additional assumptions makes learning tractable.

Error Bounds

In terms of the choice of norm for bounding $\|f - \hat{f}\|$ the results in [FVD19] are stronger than the ones in [JSA15]. Janzamin et al. [JSA15] bound $\mathbb{E}_x[|\hat{f}(x) - f(x)|^2]$. Fornasier et al. [FVD19] approximate A with exponentially small failure probability. Then using $\hat{A}$ defining $\hat{f}$ is not a probabilistic procedure anymore and the authors continue to bound $\|f - \hat{f}\|_{L^\infty(B_d)}$. Bounding $f - \hat{f}$ in the uniform norm has the advantage that the bound is independent from the underlying data distribution.

Notably in [FSV12] (4.2.1) and [SA14] (4.1.1) sparsity assumptions are used to facilitate tractable learning. In section (4.2.1) matrix factorisation is enabled with high probability, as A_1 is assumed to be multiplied by a factor which is approximately sparse and has the Restricted Isometry Property. In [SA14] on the other hand the sparsity assumption is on the matrix A_1 which is chosen to be Bernoulli-Gaussian. Here factorisation is possible by the Sparse Dictionary Learning algorithm in [SWW12].

Of Tensors And Matrices

Weak Differentiation

The active sampling method in section (4.2.5.1) suffers from numerical instabilities when finite differences are used [FVD19]. Fornasier et al. [FVD19] point out that both their solution to this - passive sampling (4.2.5.2) - but also the method in [JSA15] exploit weak differentiation (see for instance (4.8) or lemma (4.10)). For x distributed according to some distribution μ,

$$\begin{aligned}
\Delta_N^m(f) :&= \frac{1}{N}\sum_{l=1}^{N}(-1)^m f(x_l)\frac{\nabla^m p(x_l)}{p(x_l)} \\
&\approx \int_{\mathbb{R}^d} f(x)(-1)^m \frac{\nabla^m p(x)}{p(x)} p(x)dx \\
&= \int_{\mathbb{R}^d} \nabla^m f(x) d\mu(x) \\
&= \mathbb{E}_{x\sim\mu}[\nabla^m f] \\
&= \sum_{i=1}^{k}\Big(\int_{\mathbb{R}^d} \sigma^{(m)}(a_i\cdot x)d\mu(x)\Big)\underbrace{a_i\otimes\cdots\otimes a_i}_{m \text{ times}}.
\end{aligned} \tag{4.30}$$

Tensors that represent higher order derivatives can thus be approximated using function evaluations of f and the known density $p(x)$ of the input x.

Non-Orthogonal Weights

If the $\{a_i : i \in [k]\}$ are orthogonal (so necessarily $k \leq d$), then the a_i can be recovered from Δ_N^2 using spectral decomposition [FVD19]. The problem becomes more tricky if the weight vectors are not orthogonal. This is anyway always the case if the problem is overdetermined ($k > d$).
To tackle this, Janzamin et al. [JSA15] consider Δ_N^3 instead o Δ_N^2 which is a tensor. They claim that working with tensors of the minimum order 3 instead of matrices is necessary. They give two reasons. Contrary to matrices, it is possible for tensors to have identifiable non-orthogonal components and it is possible for them to have a rank larger than their dimension. They claim it is hence natural to use tensors as orthogonality cannot in general be assumed and as tensors make it possible to deal with the overdetermined case, $d < k$.
Janzamin et al. [JSA15] build on [AGJ14] and use the weak differentiation approach (4.30). Fornasier et al. [FVD19] point to [Kol15], [Rob16] which show that decomposing a symmetric orthogonal tensor is tractable. Anandkumar et al. [AGJ14] use a whiten-

ing procedure to then show that the decomposition of a symmetric non-orthogonal tensor is also tractable.

Fornasier et al. [FVD19] consider the case $k \leq d$ and refute that turning to tensors is necessary in order to deal with non-orthogonal weights. With their whitening procedure (4.2.7), they achieve quasi-orthonormality and manage to stay in the realm of matrices. Fornasier et al. [FVD19] argue that this is preferable because in general tensor operations are more complex and often intractable [HL13], [Hås90].

In [JSA15] an approximation of the tensor

$$\mathbb{E}\left[y \cdot \mathcal{S}_3(x)\right] = \mathbb{E}\left[\nabla_x^{(3)} f(x)\right]$$

is decomposed (see algorithm (3)). In practice, the authors use the sample mean. Also their whitening procedure (4.1.5) is based on one instance of a tensor only. Fornasier et al. [FVD19] point out that using just one instance of a matrix or tensor is risky as its decomposition could be an ill-posed problem. Spectral and tensor decomposition suffer from instabilities if for instance there is no gap in the spectrum [Ste98]. Fornasier et al. [FVD19] draw inspiration for their whitening procedure (4.2.7) from the whitening procedure by Janzamin et al. [JSA15] (4.1.5) with the important difference that they search within the matrix subspace $\mathcal{A}$ for a well-conditioned matrix which can be stably decomposed (4.2.7.1). Also the key step of recovering the weight vectors by matrix decomposition is robust, as again they search for a suitable matrix (4.2.7.3).

Both Fornasier et al. [FVD19] and Janzamin et al. [JSA15] arrive at error bounds (see theorems (4.7), (4.11) and theorems (4.17), (4.21) respectively) that are inverse proportional to $s_{\min}(A_1)$ and α, α_2 respectively. These terms need to be controlled, as they are not in general small if no well-conditioning holds. In [JSA15] this is potentially more critical as here terms occur in theorems (4.7) and (4.11) even in large powers. It remains the question how likely it is that the one instance of a tensor is ill-conditioned.

5 Conclusion

Approximation Theory

We have seen that shallow neural networks and hence neural networks in general are universal approximators for different function classes under the assumption of a variety of activation functions. The class of shallow neural networks is dense within the continuous functions even under weak assumptions on the activation function like non-polynomiality and local essential boundedness. Density results for $C(X)$ can be extended quite straightforwardly to other function classes like the integrable or the measurable functions. Furthermore, we have considered upper and lower bounds on the order of approximation. We have learnt when approximation problems suffer from the curse of dimensionality. We have analysed how deep neural networks can circumvent the curse of dimensionality by exploiting compositionality of the target functions. Deep networks have the representative power to reflect the structure of the target function in their own structure. Locality of constituent functions is the key for success rather than weight sharing.

Stability

In the stability section we have observed that neural networks can be fooled quite easily with adversarial examples. We have contrasted different hypotheses on the distribution of adversarial examples in input space and examined two possible defence or regularisation mechanisms.
From the work of Mallat et al. we learnt that wavelets constitute powerful building blocks for a network as their localised nature offers stability. Mallat et al. demonstrate that an embedding for functions or signals can be found which preserves the L^2 norm, does not expand distances, is translation invariant and stable to diffeomorphisms. In a scattering network energy is pushed to lower frequencies at every iteration and output through an averaging filter at ever layer. Hence network depth can be restricted. Coefficients in a scattering network are sparse, because energy concentrates on frequency decreasing paths. The deep structure ensures that information lost through averaging can be recovered. Mallat et al. propose the interpretation that CNNs progressively linearise the target function or signal f from layer to layer. At the last layer the signal has been sufficiently linearised to then do classification using a linear classifier.

Tractable Learning with Exact Weight Identification

Anandkumar et al. and Fornasier et al. use knowledge on the input density $p(x)$ to facilitate tractable learning with exact weight identification. So given a shallow neural network as a target function, they not only approximate this function uniformly, but identify the network parameters uniquely circumventing at the same time the curse of dimensionality. In (4.1.1) and (4.2.1), they show how random sparsity facilitates unique identification of the weight. When no sparsity is given, they use approaches inspired by weak differentiation - moment tensor decomposition (4.8), (4.10) [JSA15] and passive sampling (4.2.5.2) [FVD19]. Matrices or tensors that represent higher order derivatives of the target function are decomposed to obtain the weight vectors. To enable this decomposition, the authors of both papers use whitening or orthogonalisation procedures on tensors/matrices. While Anandkumar et al. decompose one instance of a suitable tensor, Fornasier et al. ensure robustness by looking for a well-conditioned matrix within a relevant matrix subspace.

References

[AAG18] Rima Alaifari, Giovanni S. Alberti, and Tandri Gauksson. "ADef: an Iterative Algorithm to Construct Adversarial Deformations". In: (2018). DOI: 10.1111/j.1469-185X.1996.tb00746.x. arXiv: 1804.07729.

[AF03] Robert A Adams and John JF Fournier. *Sobolev spaces*. Vol. 140. Elsevier, 2003.

[AGJ14] Animashree Anandkumar, Rong Ge, and Majid Janzamin. "Guaranteed Non-Orthogonal Tensor Decomposition via Alternating Rank-1 Updates". In: *arXiv preprint arXiv:1402.5180* (2014).

[Ala18] Rima Alaifari. "Adversarial deformations for deep neural networks". In: *Time, frequency, and everything that follows. In celebration of the 64th birthday of Ingrid Daubechies 2018*. 2018.

[AM18a] Naveed Akhtar and Ajmal Mian. "Threat of adversarial attacks on deep learning in computer vision: A survey". In: *IEEE Access* 6 (2018), pp. 14410–14430.

[AM18b] Tomás Angles and Stéphane Mallat. "Generative networks as inverse problems with scattering transforms". In: *arXiv preprint arXiv:1805.06621* (2018).

[Ana+14] Animashree Anandkumar et al. "Tensor decompositions for learning latent variable models". In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 2773–2832.

[Bac17] Francis Bach. "Breaking the curse of dimensionality with convex neural networks". In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 629–681.

[Bar93] Andrew R Barron. "Universal approximation bounds for superpositions of a sigmoidal function". In: *IEEE Transactions on Information theory* 39.3 (1993), pp. 930–945.

[BM10] Joan Bruna and Stéphane Mallat. "Classification with scattering operators". In: *arXiv preprint arXiv:1011.3023* (2010).

[BM13] Joan Bruna and Stéphane Mallat. "Invariant scattering convolution networks". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1872–1886.

[CB54] Ernesto Corominas and Ferran Sunyer Balaguer. "Condiciones para que una funcion infinitamente derivable sea un polinomio". In: *Revista matemática hispanoamericana* 14.1 (1954), pp. 26–43.

[CW17] Nicholas Carlini and David Wagner. "Towards evaluating the robustness of neural networks". In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2017, pp. 39–57.

[Cyb89] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[Dev85] Luc Devroye. "Nonparametric density estimation". In: *The L_1 View* (1985).

[Dhi+18] Guneet S Dhillon et al. "Stochastic activation pruning for robust adversarial defense". In: *arXiv preprint arXiv:1803.01442* (2018).

[DHM89] Ronald A DeVore, Ralph Howard, and Charles Micchelli. "Optimal nonlinear approximation". In: *Manuscripta mathematica* 63.4 (1989), pp. 469–478.

[Fra+91] Michael Frazier et al. *Littlewood-Paley theory and the study of function spaces*. 79. American Mathematical Soc., 1991.

[FSV12] Massimo Fornasier, Karin Schnass, and Jan Vybíral. "Learning functions of few arbitrary linear parameters in high dimensions". In: *Foundations of Computational Mathematics* 12.2 (2012), pp. 229–262.

[Fun89] Ken-Ichi Funahashi. "On the approximate realization of continuous mappings by neural networks". In: *Neural networks* 2.3 (1989), pp. 183–192.

[FVD18] Massimo Fornasier, Jan Vybíral, and Ingrid Daubechies. "Identification of Shallow Neural Networks by Fewest Samples". In: *arXiv preprint arXiv:1804.01592* (2018).

[FVD19] Massimo Fornasier, Jan Vybíral, and Ingrid Daubechies. "Robust and Resource Efficient Identification of Shallow Neural Networks by Fewest Samples". In: (2019). arXiv: `arXiv:1804.01592v2`.

[GBB11] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Deep sparse rectifier neural networks". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 2011, pp. 315–323.

[Goo+13] Ian J Goodfellow et al. "Maxout networks". In: *arXiv preprint arXiv:1302.4389* (2013).

[GSS14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (2014).

[GW88] A Ronald Gallant and Halbert White. "There exists a neural network that does not make avoidable mistakes". In: *Proceedings of the Second Annual IEEE Conference on Neural Networks, San Diego, CA, I*. 1988.

[Haa10] Alfred Haar. "Zur theorie der orthogonalen funktionensysteme". In: *Mathematische Annalen* 69.3 (1910), pp. 331–371.

[He+16] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[HL13] Christopher J Hillar and Lek-Heng Lim. "Most tensor problems are NP-hard". In: *Journal of the ACM (JACM)* 60.6 (2013), p. 45.

[Hor91] Kurt Hornik. "Approximation capabilities of multilayer feedforward networks". In: *Neural networks* 4.2 (1991), pp. 251–257.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[HSW89] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural networks* 2.5 (1989), pp. 359–366.

[Hua+17] Xiaowei Huang et al. "Safety verification of deep neural networks". In: *International Conference on Computer Aided Verification*. Springer. 2017, pp. 3–29.

[Hub85] Peter J Huber. "Projection pursuit". In: *The annals of Statistics* (1985), pp. 435–475.

[Hör71] Lars Hörmander. "Fourier integral operators. I". In: *Acta mathematica* 127.1 (1971), pp. 79–183.

[Hås90] Johan Håstad. "Tensor rank is NP-complete". In: *Journal of Algorithms* 11.4 (1990), pp. 644–654.

[JKL+09] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. "What is the best multi-stage architecture for object recognition?" In: *2009 IEEE 12th international conference on computer vision*. IEEE. 2009, pp. 2146–2153.

[JSA14] Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. "Score function features for discriminative learning: Matrix and tensor framework". In: *arXiv preprint arXiv:1412.2863* (2014).

[JSA15] Majid Janzamin, Hanie Sedghi, and Anima Anandkumar. "Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods". In: *arXiv preprint arXiv:1506.08473* (2015).

[KGB16] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. "Adversarial examples in the physical world". In: *arXiv preprint arXiv:1607.02533* (2016).

[Kol15] Tamara G Kolda. "Symmetric orthogonal tensor decomposition is trivial". In: *arXiv preprint arXiv:1503.01375* (2015).

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[LeC98] Yann LeCun. "The MNIST database of handwritten digits". In: *http://yann. lecun. com/exdb/mnist/* (1998).

[Les+93] Moshe Leshno et al. "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". In: *Neural Networks* (1993). DOI: 10.1016/S0893-6080(05)80131-5.

[LKF10] Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. "Convolutional networks and applications in vision". In: *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*. IEEE. 2010, pp. 253–256.

[Mad+17] Aleksander Madry et al. "Towards deep learning models resistant to adversarial attacks". In: *arXiv preprint arXiv:1706.06083* (2017).

[Mai99] VE Maiorov. "On best approximation by ridge functions". In: *Journal of Approximation Theory* 99.1 (1999), pp. 68–94.

[Mak96] Yuly Makovoz. "Random approximants and neural networks". In: *Journal of Approximation Theory* 85.1 (1996), pp. 98–109.

[Mal10] Stéphane Mallat. "Recursive interferometric representation". In: *Proc. of EUSICO conference, Danemark*. 2010.

[Mal12a] Stéphane Mallat. "Group invariant scattering". In: *Communications on Pure and Applied Mathematics* 65.10 (2012), pp. 1331–1398.

[Mal12b] Stéphane Mallat. *Scattering Invariant Deep Networks for Classification*. 2012. URL: https://www.ipam.ucla.edu/programs/summer-schools/graduate-summer-school-deep-learning-feature-learning/?tab=schedule.

[Mal16] Stéphane Mallat. *Understanding deep convolutional networks*. 2016. DOI: 10.1098/rsta.2015.0203. arXiv: 1601.04920. URL: https://www.ipam.ucla.edu/programs/summer-schools/graduate-summer-school-deep-learning-feature-learning/?tab=schedule.

[Mal99] Stéphane Mallat. *A wavelet tour of signal processing*. Elsevier, 1999.

[Mau06] Andreas Maurer. "Bounds for linear multi-task learning". In: *Journal of Machine Learning Research* 7.Jan (2006), pp. 117–139.

[MDFF16] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. "Deepfool: a simple and accurate method to fool deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2574–2582.

[Mey86] Yves Meyer. "Principe d'incertitude, bases hilbertiennes et algebres d'operateurs". In: *Bourbaki Seminar, 1986*. 1986, pp. 209–223.

[Mey95] Yves Meyer. *Wavelets and operators*. Vol. 1. Cambridge university press, 1995.

[Mha96] H N Mhaskar. *NEURAL NETWORKS FOR OPTIMAL APPROXIMATION OF SMOOTH AND ANALYTIC FUNCTIONS*. Tech. rep. 1996, pp. 164–177.

[MIA94] Robert J Marks II and Payman Arabshahi. "Fourier analysis and filtering of a single hidden layer perceptron". In: *International Conference on Artificial Neural Networks (IEEE/ENNS), Sorrento, Italy*. 1994.

[MM00] VE Maiorov and Ron Meir. "On the near optimality of the stochastic approximation of smooth functions by neural networks". In: *Advances in Computational Mathematics* 13.1 (2000), pp. 79–103.

[MM95] HN Mhaskar and Charles A Micchelli. "Degree of approximation by neural and translation networks with a single hidden layer". In: *Advances in applied mathematics* 16.2 (1995), pp. 151–183.

[MMR99] Vitaly Maiorov, Ron Meir, and Joel Ratsaby. "On the approximation of functional classes equipped with a uniform measure using ridge functions". In: *Journal of approximation theory* 99.1 (1999), pp. 95–111.

[MP99] Vitaly Maiorov and Allan Pinkus. "Lower bounds for approximation by MLP neural networks". In: *Neurocomputing* 25.1-3 (1999), pp. 81–91.

[NPS14] Ivan Nourdin, Giovanni Peccati, and Yvik Swan. "Integration by parts and representation of information functionals". In: *2014 IEEE International Symposium on Information Theory*. IEEE. 2014, pp. 2217–2221.

[NW10] Erich Novak and Henryk Wozniakowski. "Tractability of multivariate problems". In: *Volume II: Standard Information for Functionals, EMS, Zürich* (2010).

[Pap+16] Nicolas Papernot et al. "The limitations of deep learning in adversarial settings". In: *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2016, pp. 372–387.

[Pet98] Pencho P Petrushev. "Approximation by ridge functions and neural networks". In: *SIAM Journal on Mathematical Analysis* 30.1 (1998), pp. 155–189.

[Pin99] Allan Pinkus. "Approximation theory of the MLP model in neural networks". In: *Acta Numerica* (1999). DOI: 10.1017/S0962492900002919.

[Pog+17] Tomaso Poggio et al. *Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review*. 2017. DOI: 10.1007/s11633-017-1054-2. arXiv: 1611.00740.

[Rob16] Elina Robeva. "Orthogonal decomposition of symmetric tensors". In: *SIAM Journal on Matrix Analysis and Applications* 37.1 (2016), pp. 86–102.

[Rud17] Walter Rudin. *Fourier analysis on groups*. Courier Dover Publications, 2017.

[Rus+15] Olga Russakovsky et al. "Imagenet large scale visual recognition challenge". In: *International journal of computer vision* 115.3 (2015), pp. 211–252.

[SA14] Hanie Sedghi and Anima Anandkumar. "Provable methods for training neural networks with sparse connectivity". In: *arXiv preprint arXiv:1412.2693* (2014).

[Sch47] Laurent Schwartz. "Théorie générale des fonctions moyenne-périodiques". In: *Annals of Mathematics* (1947), pp. 857–929.

[Sri+13] Bharath Sriperumbudur et al. "Density estimation in infinite dimensional exponential families". In: *arXiv preprint arXiv:1312.3516* (2013).

[Ste+04] Charles Stein et al. "Use of exchangeable pairs in the analysis of simulations". In: *Stein's Method*. Institute of Mathematical Statistics, 2004, pp. 1–25.

[Ste86] Charles Stein. "Approximate computation of expectations". In: IMS. 1986.

[Ste98] Gilbert W Stewart. *Perturbation theory for the singular value decomposition*. Tech. rep. 1998.

[SWW12] Daniel A Spielman, Huan Wang, and John Wright. "Exact recovery of sparsely-used dictionaries". In: *Conference on Learning Theory*. 2012, pp. 37–1.

[Sze+13] Christian Szegedy et al. "Intriguing properties of neural networks". In: *arXiv preprint arXiv:1312.6199* (2013).

[Sze+16] Christian Szegedy et al. "Rethinking the inception architecture for computer vision". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2818–2826.

[Tim14] Aleksandr Filippovich Timan. *Theory of approximation of functions of a real variable*. Vol. 34. Elsevier, 2014.

[Tro12] Joel A Tropp. "User-friendly tail bounds for sums of random matrices". In: *Foundations of computational mathematics* 12.4 (2012), pp. 389–434.

[VK61] BA Vostretsov and Mikhail Aleksandrovich Kreines. "Approximation of continuous functions by superpositions of plane waves". In: *Doklady Akademii Nauk*. Vol. 140. 6. Russian Academy of Sciences. 1961, pp. 1237–1240.

[Wed72] Per-Åke Wedin. "Perturbation bounds in connection with singular value decomposition". In: *BIT Numerical Mathematics* 12.1 (1972), pp. 99–111.

[Wey12] Hermann Weyl. "Das asymptotische Verteilungsgesetz der Eigenwerte linearer partieller Differentialgleichungen (mit einer Anwendung auf die Theorie der Hohlraumstrahlung)". In: *Mathematische Annalen* 71.4 (1912), pp. 441–479.