

Toward Deeper Understanding of Neural Networks: The Power of Initialization and a Dual View on Expressivity

Amit Daniely* Roy Frostig[†] Yoram Singer[‡]

May 23, 2017

Abstract

We develop a general duality between neural networks and compositional kernels, striving towards a better understanding of deep learning. We show that initial representations generated by common random initializations are sufficiently rich to express all functions in the dual kernel space. Hence, though the training objective is hard to optimize in the worst case, the initial weights form a good starting point for optimization. Our dual view also reveals a pragmatic and aesthetic perspective of neural networks and underscores their expressive power.

*Email: amitdaniely@google.com

[†]Email: rf@cs.stanford.edu. Work performed at Google.

[‡]Email: singer@google.com

Contents

1	Introduction	1
2	Related work	2
3	Setting	3
4	Computation skeletons	4
4.1	From computation skeletons to neural networks	6
4.2	From computation skeletons to reproducing kernels	7
5	Main results	9
5.1	Incorporating bias terms	11
6	Mathematical background	11
7	Compositional kernel spaces	13
8	The dual activation function	15
9	Proofs	20
9.1	Well-behaved activations	20
9.2	Proofs of Thms. 2 and 3	23
9.3	Proofs of Thms. 4 and 5	26
10	Discussion	29

1 Introduction

Neural network (NN) learning has underpinned state of the art empirical results in numerous applied machine learning tasks (see for instance [31, 33]). Nonetheless, neural network learning remains rather poorly understood in several regards. Notably, it remains unclear why training algorithms find good weights, how learning is impacted by the network architecture and activations, what is the role of random weight initialization, and how to choose a concrete optimization procedure for a given architecture.

We start by analyzing the expressive power of NNs subsequent to the random weight initialization. The motivation is the empirical success of training algorithms despite inherent computational intractability, and the fact that they optimize highly non-convex objectives with potentially many local minima. Our key result shows that random initialization already positions learning algorithms at a good starting point. We define an object termed a *computation skeleton* that describes a distilled structure of feed-forward networks. A skeleton induces a family of network architectures along with a hypothesis class \mathcal{H} of functions obtained by certain non-linear compositions according to the skeleton’s structure. We show that the representation generated by random initialization is sufficiently rich to approximately express the functions in \mathcal{H} . Concretely, all functions in \mathcal{H} can be approximated by tuning the weights of the last layer, which is a convex optimization task.

In addition to explaining in part the success in finding good weights, our study provides an appealing perspective on neural network learning. We establish a tight connection between network architectures and their dual kernel spaces. This connection generalizes several previous constructions (see Sec 2). As we demonstrate, our dual view gives rise to design principles for NNs, supporting current practice and suggesting new ideas. We outline below a few points.

- Duals of convolutional networks appear a more suitable fit for vision and acoustic tasks than those of fully connected networks.
- Our framework surfaces a principled initialization scheme. It is very similar to common practice, but incorporates a small correction.
- By modifying the activation functions, two consecutive fully connected layers can be replaced with one while preserving the network’s dual kernel.
- The ReLU activation, i.e. $x \mapsto \max(x, 0)$, possesses favorable properties. Its dual kernel is expressive, and it can be well approximated by random initialization, even when the initialization’s scale is moderately changed.
- As the number of layers in a fully connected network becomes very large, its dual kernel converges to a degenerate form for any non-linear activation.
- Our result suggests that optimizing the weights of the last layer can serve as a convex proxy for choosing among different architectures prior to training. This idea was advocated and tested empirically in [49].

2 Related work

Current theoretical understanding of NN learning. Understanding neural network learning, particularly its recent successes, commonly decomposes into the following research questions.

- (i) What functions can be efficiently expressed by neural networks?
- (ii) When does a low empirical loss result in a low population loss?
- (iii) Why and when do efficient algorithms, such as stochastic gradient, find good weights?

Though still far from being complete, previous work provides some understanding of questions (i) and (ii). Standard results from complexity theory [28] imply that essentially all functions of interest (that is, any efficiently computable function) can be expressed by a network of moderate size. Biological phenomena show that many relevant functions can be expressed by even simpler networks, similar to convolutional neural networks [32] that are dominant in ML tasks today. Barron’s theorem [7] states that even two-layer networks can express a very rich set of functions. As for question (ii), both classical [10, 9, 3] and more recent [40, 22] results from statistical learning theory show that as the number of examples grows in comparison to the size of the network the empirical loss must be close to the population loss. In contrast to the first two, question (iii) is rather poorly understood. While learning algorithms succeed in practice, theoretical analysis is overly pessimistic. Direct interpretation of theoretical results suggests that when going slightly deeper beyond single layer networks, e.g. to depth two networks with very few hidden units, it is hard to predict even marginally better than random [29, 30, 17, 18, 16]. Finally, we note that the recent empirical successes of NNs have prompted a surge of theoretical work around NN learning [47, 1, 4, 12, 39, 35, 19, 52, 14].

Compositional kernels and connections to networks. The idea of composing kernels has repeatedly appeared throughout the machine learning literature, for instance in early work by Schölkopf et al. [51], Grauman and Darrell [21]. Inspired by deep networks’ success, researchers considered deep composition of kernels [36, 13, 11]. For fully connected two-layer networks, the correspondence between kernels and neural networks with random weights has been examined in [46, 45, 38, 56]. Notably, Rahimi and Recht [46] proved a formal connection (similar to ours) for the RBF kernel. Their work was extended to include polynomial kernels [27, 42] as well as other kernels [6, 5]. Several authors have further explored ways to extend this line of research to deeper, either fully-connected networks [13] or convolutional networks [24, 2, 36]. Our work sets a common foundation for and expands on these ideas. We extend the analysis from fully-connected and convolutional networks to a rather broad family of architectures. In addition, we prove approximation guarantees between a network and its corresponding kernel in our more general setting. We thus extend previous analyses that only applies to fully connected two-layer networks. Finally, we use the connection as an analytical tool to reason about architectural design choices.

3 Setting

Notation. We denote vectors by bold-face letters (e.g. \mathbf{x}), and matrices by upper case Greek letters (e.g. Σ). The 2-norm of $\mathbf{x} \in \mathbb{R}^d$ is denoted by $\|\mathbf{x}\|$. For functions $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ we let

$$\|\sigma\| := \sqrt{\mathbb{E}_{X \sim \mathcal{N}(0,1)} \sigma^2(X)} = \sqrt{\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \sigma^2(x) e^{-\frac{x^2}{2}} dx}.$$

Let $G = (V, E)$ be a directed acyclic graph. The set of neighbors incoming to a vertex v is denoted $\text{in}(v) := \{u \in V \mid uv \in E\}$. The $d - 1$ dimensional sphere is denoted $\mathbb{S}^{d-1} = \{\mathbf{x} \in \mathbb{R}^d \mid \|\mathbf{x}\| = 1\}$. We provide a brief overview of reproducing kernel Hilbert spaces in the sequel and merely introduce notation here. In a Hilbert space \mathcal{H} , we use a slightly non-standard notation \mathcal{H}^B for the ball of radius B , $\{\mathbf{x} \in \mathcal{H} \mid \|\mathbf{x}\|_{\mathcal{H}} \leq B\}$. We use $[x]_+$ to denote $\max(x, 0)$ and $\mathbf{1}[b]$ to denote the indicator function of a binary variable b .

Input space. Throughout the paper we assume that each example is a sequence of n elements, each of which is represented as a unit vector. Namely, we fix n and take the input space to be $\mathcal{X} = \mathcal{X}_{n,d} = (\mathbb{S}^{d-1})^n$. Each input example is denoted,

$$\mathbf{x} = (\mathbf{x}^1, \dots, \mathbf{x}^n), \text{ where } \mathbf{x}^i \in \mathbb{S}^{d-1}. \quad (1)$$

We refer to each vector \mathbf{x}^i as the input's i th *coordinate*, and use x_j^i to denote its j th scalar entry. Though this notation is slightly non-standard, it unifies input types seen in various domains. For example, binary features can be encoded by taking $d = 1$, in which case $\mathcal{X} = \{\pm 1\}^n$. Meanwhile, images and audio signals are often represented as bounded and continuous numerical values—we can assume in full generality that these values lie in $[-1, 1]$. To match the setup above, we embed $[-1, 1]$ into the circle \mathbb{S}^1 , e.g. via the map $x \mapsto (\sin(\frac{\pi x}{2}), \cos(\frac{\pi x}{2}))$. When each coordinate is categorical—taking one of d values—we can represent category $j \in [d]$ by the unit vector $\mathbf{e}_j \in \mathbb{S}^{d-1}$. When d may be very large or the basic units exhibits some structure, such as when the input is a sequence of words, a more concise encoding may be useful, e.g. as unit vectors in a low dimension space $\mathbb{S}^{d'}$ where $d' \ll d$ (see for instance Mikolov et al. [37], Levy and Goldberg [34]).

Supervised learning. The goal in supervised learning is to devise a mapping from the input space \mathcal{X} to an output space \mathcal{Y} based on a sample $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$, where $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, drawn i.i.d. from a distribution \mathcal{D} over $\mathcal{X} \times \mathcal{Y}$. A supervised learning problem is further specified by an output length k and a loss function $\ell : \mathbb{R}^k \times \mathcal{Y} \rightarrow [0, \infty)$, and the goal is to find a predictor $h : \mathcal{X} \rightarrow \mathbb{R}^k$ whose loss, $\mathcal{L}_{\mathcal{D}}(h) := \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}} \ell(h(\mathbf{x}), y)$, is small. The *empirical* loss $\mathcal{L}_S(h) := \frac{1}{m} \sum_{i=1}^m \ell(h(\mathbf{x}_i), y_i)$ is commonly used as a proxy for the loss $\mathcal{L}_{\mathcal{D}}$. Regression problems correspond to $\mathcal{Y} = \mathbb{R}$ and, for instance, the squared loss $\ell(\hat{y}, y) = (\hat{y} - y)^2$. Binary classification is captured by $\mathcal{Y} = \{\pm 1\}$ and, say, the zero-one loss $\ell(\hat{y}, y) = \mathbf{1}[\hat{y}y \leq 0]$ or the hinge loss $\ell(\hat{y}, y) = [1 - \hat{y}y]_+$, with standard extensions to the multiclass case. A loss ℓ is L -Lipschitz if $|\ell(y_1, y) - \ell(y_2, y)| \leq L|y_1 - y_2|$ for all $y_1, y_2 \in \mathbb{R}^k$, $y \in \mathcal{Y}$, and it is convex if $\ell(\cdot, y)$ is convex for every $y \in \mathcal{Y}$.

Neural network learning. We define a *neural network* \mathcal{N} to be a vertices weighted directed acyclic graph (DAG) whose nodes are denoted $V(\mathcal{N})$ and edges $E(\mathcal{N})$. The weight function will be denoted by $\delta : V(\mathcal{N}) \rightarrow [0, \infty)$, and its sole role would be to dictate the distribution of the initial weights (see definition 3). Each of its internal units, i.e. nodes with both incoming and outgoing edges, is associated with an *activation* function $\sigma_v : \mathbb{R} \rightarrow \mathbb{R}$. In this paper’s context, an activation can be any function that is square integrable with respect to the Gaussian measure on \mathbb{R} . We say that σ is *normalized* if $\|\sigma\| = 1$. The set of nodes having only incoming edges are called the output nodes. To match the setup of a supervised learning problem, a network \mathcal{N} has nd input nodes and k output nodes, denoted o_1, \dots, o_k . A network \mathcal{N} together with a weight vector $\mathbf{w} = \{w_{uv} \mid uv \in E\}$ defines a predictor $h_{\mathcal{N}, \mathbf{w}} : \mathcal{X} \rightarrow \mathbb{R}^k$ whose prediction is given by “propagating” \mathbf{x} forward through the network. Formally, we define $h_{v, \mathbf{w}}(\cdot)$ to be the output of the subgraph of the node v as follows: for an input node v , $h_{v, \mathbf{w}}$ outputs the corresponding coordinate in \mathbf{x} , and for all other nodes, we define $h_{v, \mathbf{w}}$ recursively as

$$h_{v, \mathbf{w}}(\mathbf{x}) = \sigma_v \left(\sum_{u \in \text{in}(v)} w_{uv} h_{u, \mathbf{w}}(\mathbf{x}) \right).$$

Finally, we let $h_{\mathcal{N}, \mathbf{w}}(\mathbf{x}) = (h_{o_1, \mathbf{w}}(\mathbf{x}), \dots, h_{o_k, \mathbf{w}}(\mathbf{x}))$. We also refer to internal nodes as *hidden units*. The *output layer* of \mathcal{N} is the sub-network consisting of all output neurons of \mathcal{N} along with their incoming edges. The *representation* induced by a network \mathcal{N} is the network $\text{rep}(\mathcal{N})$ obtained from \mathcal{N} by removing the output layer. The *representation* function induced by the weights \mathbf{w} is $\mathcal{R}_{\mathcal{N}, \mathbf{w}} := h_{\text{rep}(\mathcal{N}), \mathbf{w}}$. Given a sample S , a learning algorithm searches for weights \mathbf{w} having small empirical loss $\mathcal{L}_S(\mathbf{w}) = \frac{1}{m} \sum_{i=1}^m \ell(h_{\mathcal{N}, \mathbf{w}}(\mathbf{x}_i), y_i)$. A popular approach is to randomly initialize the weights and then use a variant of the stochastic gradient method to improve these weights in the direction of lower empirical loss.

Kernel learning. A function $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a *reproducing kernel*, or simply a kernel, if for every $\mathbf{x}_1, \dots, \mathbf{x}_r \in \mathcal{X}$, the $r \times r$ matrix $\Gamma_{i,j} = \{\kappa(\mathbf{x}_i, \mathbf{x}_j)\}$ is positive semi-definite. Each kernel induces a Hilbert space \mathcal{H}_κ of functions from \mathcal{X} to \mathbb{R} with a corresponding norm $\|\cdot\|_{\mathcal{H}_\kappa}$. A kernel and its corresponding space are *normalized* if $\forall \mathbf{x} \in \mathcal{X}, \kappa(\mathbf{x}, \mathbf{x}) = 1$. Given a convex loss function ℓ , a sample S , and a kernel κ , a kernel learning algorithm finds a function $f = (f_1, \dots, f_k) \in \mathcal{H}_\kappa^k$ whose empirical loss, $\mathcal{L}_S(f) = \frac{1}{m} \sum_i \ell(f(\mathbf{x}_i), y_i)$, is minimal among all functions with $\sum_i \|f_i\|_\kappa^2 \leq R^2$ for some $R > 0$. Alternatively, kernel algorithms minimize the *regularized loss*,

$$\mathcal{L}_S^R(f) = \frac{1}{m} \sum_{i=1}^m \ell(f(\mathbf{x}_i), y_i) + \frac{1}{R^2} \sum_{i=1}^k \|f_i\|_\kappa^2,$$

a convex objective that often can be efficiently minimized.

4 Computation skeletons

In this section we define a simple structure which we term a computation skeleton. The purpose of a computational skeleton is to compactly describe a feed-forward computation

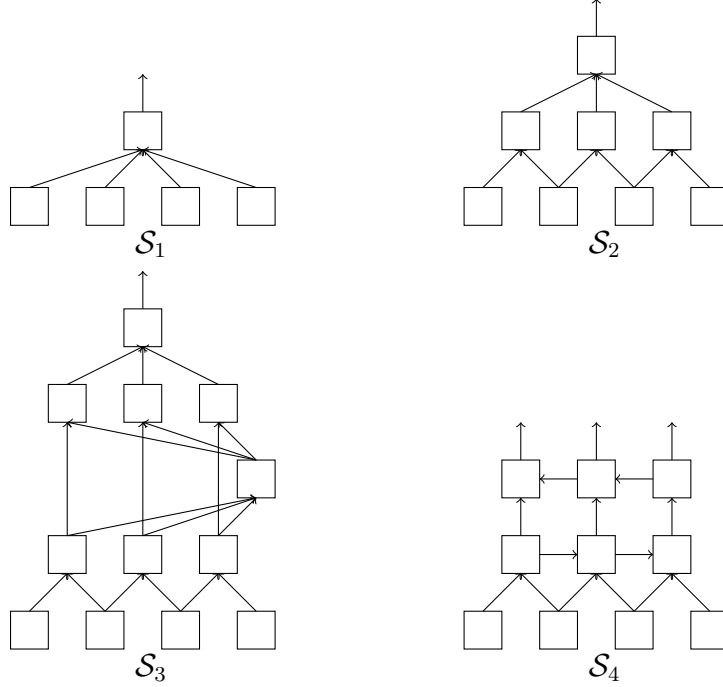


Figure 1: Examples of computation skeletons.

from an input to an output. A single skeleton encompasses a family of neural networks that share the same skeletal structure. Likewise, it defines a corresponding kernel space.

Definition 1. A computation skeleton \mathcal{S} is a DAG whose non-input nodes are labeled by activations.

Though the formal definition of neural networks and skeletons appear identical, we make a conceptual distinction between them as their role in our analysis is rather different. Accompanied by a set of weights, a neural network describes a concrete function, whereas the skeleton stands for a topology common to several networks as well as for a kernel. To further underscore the differences we note that skeletons are naturally more compact than networks. In particular, all examples of skeletons in this paper are *irreducible*, meaning that for each two nodes $v, u \in V(\mathcal{S})$, $\text{in}(v) \neq \text{in}(u)$. We further restrict our attention to skeletons with a single output node, showing later that single-output skeletons can capture supervised problems with outputs in \mathbb{R}^k . We denote by $|\mathcal{S}|$ the number of non-input nodes of \mathcal{S} .

Figure 1 shows four example skeletons, omitting the designation of the activation functions. The skeleton \mathcal{S}_1 is rather basic as it aggregates all the inputs in a single step. Such topology can be useful in the absence of any prior knowledge of how the output label may be computed from an input example, and it is commonly used in natural language processing where the input is represented as a bag-of-words [23]. The only structure in \mathcal{S}_1 is a single *fully connected* layer:

Terminology (Fully connected layer of a skeleton). *An induced subgraph of a skeleton with $r + 1$ nodes, u_1, \dots, u_r, v , is called a fully connected layer if its edges are u_1v, \dots, u_rv .*

The skeleton \mathcal{S}_2 is slightly more involved: it first processes consecutive (overlapping) parts of the input, and the next layer aggregates the partial results. Altogether, it corresponds to networks with a single one-dimensional convolutional layer, followed by a fully connected layer. The two-dimensional (and deeper) counterparts of such skeletons correspond to networks that are common in visual object recognition.

Terminology (Convolution layer of a skeleton). *Let s, w, q be positive integers and denote $n = s(q - 1) + w$. A subgraph of a skeleton is a one dimensional convolution layer of width w and stride s if it has $n + q$ nodes, $u_1, \dots, u_n, v_1, \dots, v_q$, and qw edges, $u_{s(i-1)+j}v_i$, for $1 \leq i \leq q, 1 \leq j \leq w$.*

The skeleton \mathcal{S}_3 is a somewhat more sophisticated version of \mathcal{S}_2 : the local computations are first aggregated, then reconsidered with the aggregate, and finally aggregated again. The last skeleton, \mathcal{S}_4 , corresponds to the networks that arise in learning sequence-to-sequence mappings as used in translation, speech recognition, and OCR tasks (see for example Sutskever et al. [55]).

4.1 From computation skeletons to neural networks

The following definition shows how a skeleton, accompanied with a replication parameter $r \geq 1$ and a number of output nodes k , induces a neural network architecture. Recall that inputs are ordered sets of vectors in \mathbb{S}^{d-1} .

Definition 2 (Realization of a skeleton). *Let \mathcal{S} be a computation skeleton and consider input coordinates in \mathbb{S}^{d-1} as in (1). For $r, k \geq 1$ we define the following neural network $\mathcal{N} = \mathcal{N}(\mathcal{S}, r, k)$. For each input node in \mathcal{S} , \mathcal{N} has d corresponding input neurons with weight $1/d$. For each internal node $v \in \mathcal{S}$ labeled by an activation σ , \mathcal{N} has r neurons v^1, \dots, v^r , each with an activation σ and weight $1/r$. In addition, \mathcal{N} has k output neurons o_1, \dots, o_k with the identity activation $\sigma(x) = x$ and weight 1. There is an edge $v^i u^j \in E(\mathcal{N})$ whenever $uv \in E(\mathcal{S})$. For every output node v in \mathcal{S} , each neuron v^j is connected to all output neurons o_1, \dots, o_k . We term \mathcal{N} the (r, k) -fold realization of \mathcal{S} . We also define the r -fold realization of \mathcal{S} as¹ $\mathcal{N}(\mathcal{S}, r) = \text{rep}(\mathcal{N}(\mathcal{S}, r, 1))$.*

Note that the notion of the replication parameter r corresponds, in the terminology of convolutional networks, to the number of channels taken in a convolutional layer and to the number of hidden units taken in a fully-connected layer.

Figure 2 illustrates a $(5, 4)$ - and 5-realizations of a skeleton with coordinate dimension $d = 2$. The $(5, 4)$ -realization is a network with a single (one dimensional) convolutional layer having 5 channels, stride of 2, and width of 4, followed by three fully-connected layers. The global replication parameter r in a realization is used for brevity; it is straightforward to extend results when the different nodes in \mathcal{S} are each replicated to a different extent.

¹Note that for every k , $\text{rep}(\mathcal{N}(\mathcal{S}, r, 1)) = \text{rep}(\mathcal{N}(\mathcal{S}, r, k))$.

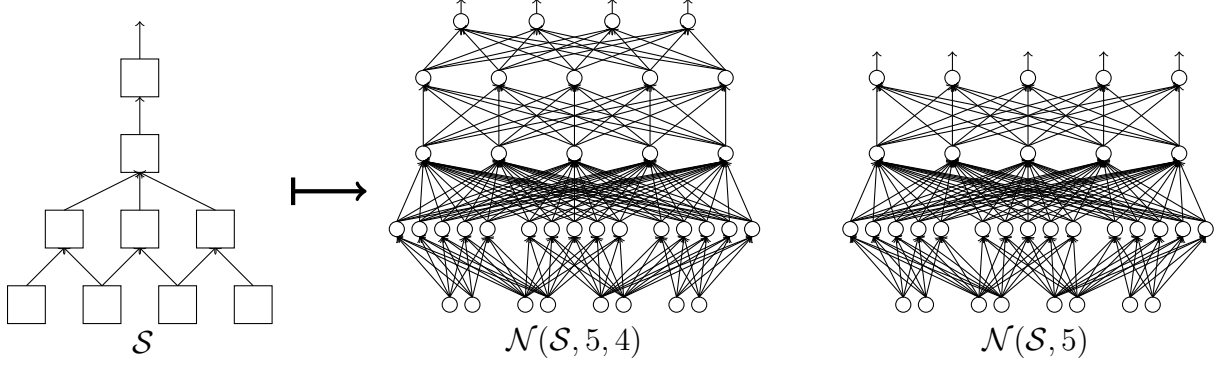


Figure 2: A $(5, 4)$ -fold and 5-fold realizations of the computation skeleton \mathcal{S} with $d = 2$.

We next define a scheme for random initialization of the weights of a neural network, that is similar to what is often done in practice. We employ the definition throughout the paper whenever we refer to random weights.

Definition 3 (Random weights). *A random initialization of a neural network \mathcal{N} is a multivariate Gaussian $\mathbf{w} = (w_{uv})_{uv \in E(\mathcal{N})}$ such that each weight w_{uv} is sampled independently from a normal distribution with mean 0 and variance² $d\delta(u)/\delta(\text{in}(v))$ if u is an input neuron and $\delta(u)/(\|\sigma_u\|^2 \delta(\text{in}(v)))$ otherwise.*

Architectures such as convolutional nets have weights that are shared across different edges. Again, it is straightforward to extend our results to these cases and for simplicity we assume no explicit weight sharing.

4.2 From computation skeletons to reproducing kernels

In addition to networks' architectures, a computation skeleton \mathcal{S} also defines a normalized kernel $\kappa_{\mathcal{S}} : \mathcal{X} \times \mathcal{X} \rightarrow [-1, 1]$ and a corresponding norm $\|\cdot\|_{\mathcal{S}}$ on functions $f : \mathcal{X} \rightarrow \mathbb{R}$. This norm has the property that $\|f\|_{\mathcal{S}}$ is small if and only if f can be obtained by certain simple compositions of functions according to the structure of \mathcal{S} . To define the kernel, we introduce a *dual activation* and *dual kernel*. For $\rho \in [-1, 1]$, we denote by N_{ρ} the multivariate Gaussian distribution on \mathbb{R}^2 with mean 0 and covariance matrix $\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$.

Definition 4 (Dual activation and kernel). *The dual activation of an activation σ is the function $\hat{\sigma} : [-1, 1] \rightarrow \mathbb{R}$ defined as*

$$\hat{\sigma}(\rho) = \mathbb{E}_{(X,Y) \sim N_{\rho}} \sigma(X)\sigma(Y).$$

The dual kernel w.r.t. to a Hilbert space \mathcal{H} is the kernel $\kappa_{\sigma} : \mathcal{H}^1 \times \mathcal{H}^1 \rightarrow \mathbb{R}$ defined as

$$\kappa_{\sigma}(\mathbf{x}, \mathbf{y}) = \hat{\sigma}(\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{H}}).$$

²For $U \subset V(\mathcal{N})$ we denote $\delta(U) = \sum_{u \in U} \delta(u)$.

Activation		Dual Activation	Kernel	Ref
Identity	x	ρ	linear	
2nd Hermite	$\frac{x^2-1}{\sqrt{2}}$	ρ^2	poly	
ReLU	$\sqrt{2} [x]_+$	$\frac{1}{\pi} + \frac{\rho}{2} + \frac{\rho^2}{2\pi} + \frac{\rho^4}{24\pi} + \dots = \frac{\sqrt{1-\rho^2+(\pi-\cos^{-1}(\rho))\rho}}{\pi}$	\arccos_1	[13]
Step	$\sqrt{2} \mathbf{1}[x \geq 0]$	$\frac{1}{2} + \frac{\rho}{\pi} + \frac{\rho^3}{6\pi} + \frac{3\rho^5}{40\pi} + \dots = \frac{\pi-\cos^{-1}(\rho)}{\pi}$	\arccos_0	[13]
Exponential	e^{x-2}	$\frac{1}{e} + \frac{\rho}{e} + \frac{\rho^2}{2e} + \frac{\rho^3}{6e} + \dots = e^{\rho-1}$	RBF	[36]

Table 1: Activation functions and their duals.

Section 7 shows that κ_σ is indeed a kernel for every activation σ that adheres with the square-integrability requirement. In fact, any continuous $\mu : [-1, 1] \rightarrow \mathbb{R}$, such that $(\mathbf{x}, \mathbf{y}) \mapsto \mu(\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{H}})$ is a kernel for all \mathcal{H} , is the dual of some activation. Note that κ_σ is normalized iff σ is normalized. We show in Section 8 that dual activations are closely related to Hermite polynomial expansions, and that these can be used to calculate the duals of activation functions analytically. Table 1 lists a few examples of normalized activations and their corresponding dual (corresponding derivations are in Section 8). The following definition gives the kernel corresponding to a skeleton having normalized activations.³

Definition 5 (Compositional kernels). *Let \mathcal{S} be a computation skeleton with normalized activations and (single) output node o . For every node v , inductively define a kernel $\kappa_v : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ as follows. For an input node v corresponding to the i th coordinate, define $\kappa_v(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}^i, \mathbf{y}^i \rangle$. For a non-input node v , define*

$$\kappa_v(\mathbf{x}, \mathbf{y}) = \hat{\sigma}_v \left(\frac{\sum_{u \in \text{in}(v)} \kappa_u(\mathbf{x}, \mathbf{y})}{|\text{in}(v)|} \right).$$

The final kernel $\kappa_{\mathcal{S}}$ is κ_o , the kernel associated with the output node o . The resulting Hilbert space and norm are denoted $\mathcal{H}_{\mathcal{S}}$ and $\|\cdot\|_{\mathcal{S}}$ respectively, and \mathcal{H}_v and $\|\cdot\|_v$ denote the space and norm when formed at node v .

As we show later, $\kappa_{\mathcal{S}}$ is indeed a (normalized) kernel for every skeleton \mathcal{S} . To understand the kernel in the context of learning, we need to examine which functions can be expressed as moderate norm functions in $\mathcal{H}_{\mathcal{S}}$. As we show in section 7, these are the functions obtained by certain simple compositions according to the feed-forward structure of \mathcal{S} . For intuition, the following example contrasts two commonly used skeletons.

Example 1 (Convolutional vs. fully connected skeletons). Consider a network whose activations are all ReLU, $\sigma(z) = [z]_+$, and an input space $\mathcal{X}_{n,1} = \{\pm 1\}^n$. Say that \mathcal{S}_1 is a skeleton comprising a single fully connected layer, and that \mathcal{S}_2 is one comprising a convolutional layer

³For a skeleton \mathcal{S} with unnormalized activations, the corresponding kernel is the kernel of the skeleton \mathcal{S}' obtained by normalizing the activations of \mathcal{S} .

of stride 1 and width $q = \log^{0.999}(n)$, followed by a single fully-connected layer. (The skeleton \mathcal{S}_2 from Figure 1 is a concrete example of the convolutional skeleton with $q = 2$ and $n = 4$.) The kernel $\kappa_{\mathcal{S}_1}$ takes the form $\kappa_{\mathcal{S}_1}(\mathbf{x}, \mathbf{y}) = \hat{\sigma}(\langle \mathbf{x}, \mathbf{y} \rangle / n)$. It is a symmetric kernel and therefore functions with small norm in $\mathcal{H}_{\mathcal{S}_1}$ are essentially low-degree polynomials. For instance, fix a bound $R = n^{1.001}$ on the norm of the functions. In this case, the space $\mathcal{H}_{\mathcal{S}_1}^R$ contains multiplication of one or two input coordinates. However, multiplication of 3 or more coordinates are no-longer in $\mathcal{H}_{\mathcal{S}_1}^R$. Moreover, this property holds true regardless of the choice of activation function. On the other hand, $\mathcal{H}_{\mathcal{S}_2}^R$ contains functions whose dependence on adjacent input coordinates is far more complex. It includes, for instance, any function $f : \mathcal{X} \rightarrow \{\pm 1\}$ that is symmetric (i.e. $f(x) = f(-x)$) and that depends on q adjacent coordinates $\mathbf{x}_i, \dots, \mathbf{x}_{i+q}$. Furthermore, any sum of n such functions is also in $\mathcal{H}_{\mathcal{S}_2}^R$.

5 Main results

We review our main results. Let us fix a compositional kernel \mathcal{S} . There are a few upshots to underscore upfront. First, our analysis implies that a representation generated by a random initialization of $\mathcal{N} = \mathcal{N}(\mathcal{S}, r, k)$ approximates the kernel $\kappa_{\mathcal{S}}$. The sense in which the result holds is twofold. First, with the proper rescaling we show that $\langle \mathcal{R}_{\mathcal{N}, \mathbf{w}}(\mathbf{x}), \mathcal{R}_{\mathcal{N}, \mathbf{w}}(\mathbf{x}') \rangle \approx \kappa_{\mathcal{S}}(\mathbf{x}, \mathbf{x}')$. Then, we also show that the functions obtained by composing bounded linear functions with $\mathcal{R}_{\mathcal{N}, \mathbf{w}}$ are approximately the bounded-norm functions in $\mathcal{H}_{\mathcal{S}}$. In other words, the functions expressed by \mathcal{N} under varying the weights of the last layer are approximately bounded-norm functions in $\mathcal{H}_{\mathcal{S}}$. For simplicity, we restrict the analysis to the case $k = 1$. We also confine the analysis to either bounded activations, with bounded first and second derivatives, or the ReLU activation. Extending the results to a broader family of activations is left for future work. Through this and remaining sections we use \gtrsim to hide universal constants.

Definition 6. *An activation $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ is C -bounded if it is twice continuously differentiable and $\|\sigma\|_{\infty}, \|\sigma'\|_{\infty}, \|\sigma''\|_{\infty} \leq \|\sigma\|C$.*

Note that many activations are C -bounded for some constant $C > 0$. In particular, most of the popular sigmoid-like functions such as $1/(1 + e^{-x})$, $\text{erf}(x)$, $x/\sqrt{1 + x^2}$, $\tanh(x)$, and $\tan^{-1}(x)$ satisfy the boundedness requirements. We next introduce terminology that parallels the representation layer of \mathcal{N} with a kernel space. Concretely, let \mathcal{N} be a network whose representation part has q output neurons. Given weights \mathbf{w} , the *normalized representation* $\Psi_{\mathbf{w}}$ is obtained from the representation $R_{\mathcal{N}, \mathbf{w}}$ by dividing each output neuron v by $\|\sigma_v\|/\sqrt{q}$. The *empirical kernel* corresponding to \mathbf{w} is defined as $\kappa_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') = \langle \Psi_{\mathbf{w}}(\mathbf{x}), \Psi_{\mathbf{w}}(\mathbf{x}') \rangle$. We also define the *empirical kernel space* corresponding to \mathbf{w} as $\mathcal{H}_{\mathbf{w}} = \mathcal{H}_{\kappa_{\mathbf{w}}}$. Concretely,

$$\mathcal{H}_{\mathbf{w}} = \{h_{\mathbf{v}}(\mathbf{x}) = \langle \mathbf{v}, \Psi_{\mathbf{w}}(\mathbf{x}) \rangle \mid \mathbf{v} \in \mathbb{R}^q\} ,$$

and the norm of $\mathcal{H}_{\mathbf{w}}$ is defined as $\|h\|_{\mathbf{w}} = \inf\{\|\mathbf{v}\| \mid h = h_{\mathbf{v}}\}$. Our first result shows that the empirical kernel approximates the kernel $\kappa_{\mathcal{S}}$.

Theorem 2. Let \mathcal{S} be a skeleton with C -bounded activations. Let \mathbf{w} be a random initialization of $\mathcal{N} = \mathcal{N}(\mathcal{S}, r)$ with

$$r \geq \frac{(4C^4)^{\text{depth}(\mathcal{S})+1} \log(8|\mathcal{S}|/\delta)}{\epsilon^2}.$$

Then, for all \mathbf{x}, \mathbf{x}' , with probability of at least $1 - \delta$,

$$|k_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') - k_{\mathcal{S}}(\mathbf{x}, \mathbf{x}')| \leq \epsilon.$$

We note that if we fix the activation and assume that the depth of \mathcal{S} is logarithmic, then the required bound on r is polynomial. For the ReLU activation we get a stronger bound with only quadratic dependence on the depth. However, it requires that $\epsilon \leq 1/\text{depth}(\mathcal{S})$.

Theorem 3. Let \mathcal{S} be a skeleton with ReLU activations. Let \mathbf{w} be a random initialization of $\mathcal{N}(\mathcal{S}, r)$ with

$$r \gtrsim \frac{\text{depth}^2(\mathcal{S}) \log(|\mathcal{S}|/\delta)}{\epsilon^2}.$$

Then, for all \mathbf{x}, \mathbf{x}' and $\epsilon \lesssim 1/\text{depth}(\mathcal{S})$, with probability of at least $1 - \delta$,

$$|\kappa_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') - \kappa_{\mathcal{S}}(\mathbf{x}, \mathbf{x}')| \leq \epsilon.$$

For the remaining theorems, we fix a L -Lipschitz loss $\ell : \mathbb{R} \times \mathcal{Y} \rightarrow [0, \infty)$. For a distribution \mathcal{D} on $\mathcal{X} \times \mathcal{Y}$ we denote by $\|\mathcal{D}\|_0$ the cardinality of the support of the distribution. We note that $\log(\|\mathcal{D}\|_0)$ is bounded by, for instance, the number of bits used to represent an element in $\mathcal{X} \times \mathcal{Y}$. We use the following notion of approximation.

Definition 7. Let \mathcal{D} be a distribution on $\mathcal{X} \times \mathcal{Y}$. A space $\mathcal{H}_1 \subset \mathbb{R}^{\mathcal{X}}$ ϵ -approximates the space $\mathcal{H}_2 \subset \mathbb{R}^{\mathcal{X}}$ w.r.t. \mathcal{D} if for every $h_2 \in \mathcal{H}_2$ there is $h_1 \in \mathcal{H}_1$ such that $\mathcal{L}_{\mathcal{D}}(h_1) \leq \mathcal{L}_{\mathcal{D}}(h_2) + \epsilon$.

Theorem 4. Let \mathcal{S} be a skeleton with C -bounded activations. Let \mathbf{w} be a random initialization of $\mathcal{N}(\mathcal{S}, r)$ with

$$r \gtrsim \frac{L^4 R^4 (4C^4)^{\text{depth}(\mathcal{S})+1} \log\left(\frac{LRC|\mathcal{S}|}{\epsilon\delta}\right)}{\epsilon^4}.$$

Then, with probability of at least $1 - \delta$ over the choices of \mathbf{w} we have that $\mathcal{H}_{\mathbf{w}}^{\sqrt{2}R}$ ϵ -approximates $\mathcal{H}_{\mathcal{S}}^R$ and $\mathcal{H}_{\mathcal{S}}^{\sqrt{2}R}$ ϵ -approximates $\mathcal{H}_{\mathbf{w}}^R$.

Theorem 5. Let \mathcal{S} be a skeleton with ReLU activations and $\epsilon \lesssim 1/\text{depth}(\mathcal{C})$. Let \mathbf{w} be a random initialization of $\mathcal{N}(\mathcal{S}, r)$ with

$$r \gtrsim \frac{L^4 R^4 \text{depth}^2(\mathcal{S}) \log\left(\frac{\|\mathcal{D}\|_0 |\mathcal{S}|}{\delta}\right)}{\epsilon^4}.$$

Then, with probability of at least $1 - \delta$ over the choices of \mathbf{w} we have that $\mathcal{H}_{\mathbf{w}}^{\sqrt{2}R}$ ϵ -approximates $\mathcal{H}_{\mathcal{S}}^R$ and $\mathcal{H}_{\mathcal{S}}^{\sqrt{2}R}$ ϵ -approximates $\mathcal{H}_{\mathbf{w}}^R$.

As in Theorems 2 and 3, for a fixed C -bounded activation and logarithmically deep \mathcal{S} , the required bounds on r are polynomial. Analogously, for the ReLU activation the bound is polynomial even without restricting the depth. However, the polynomial growth in Theorems 4 and 5 is rather large. Improving the bounds, or proving their optimality, is left to future work.

5.1 Incorporating bias terms

Our results can be extended to incorporate bias terms. Namely, in addition to the weights we can add a bias vector $\mathbf{b} = \{b_v \mid v \in V\}$ and let each neuron compute the function

$$h_{v,\mathbf{w},\mathbf{b}}(\mathbf{x}) = \sigma_v \left(\sum_{u \in \text{in}(v)} w_{uv} h_{u,\mathbf{w}}(\mathbf{x}) + b_v \right).$$

To do so, we extend the definition of random initialization and compositional kernel as follows:

Definition 8 (Random weights with bias terms). *Let $0 \leq \beta \leq 1$. A β -biased random initialization of a neural network \mathcal{N} is a multivariate Gaussian $(\mathbf{b}, \mathbf{w}) = ((w_{uv})_{uv \in E(\mathcal{N})}, (b_v)_{v \in V(\mathcal{N})})$ such that each weight w_{uv} is sampled independently from a normal distribution with mean 0 and variance $(1 - \beta)d\delta(u)/\delta(\text{in}(v))$ if u is an input neuron and $(1 - \beta)\delta(u)/(\|\sigma_u\|^2 \delta(\text{in}(v)))$ otherwise. Finally, each bias term b_v is sampled independently from a normal distribution with mean 0 and variance β .*

Definition 9 (Compositional kernels with bias terms). *Let \mathcal{S} be a computation skeleton with normalized activations and (a single) output node o , and let $0 \leq \beta \leq 1$. For every node v , inductively define a kernel $\kappa_v^\beta : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ as follows. For an input node v corresponding to the i th coordinate, define $\kappa_v^\beta(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}^i, \mathbf{y}^i \rangle$. For a non-input node v , define*

$$\kappa_v^\beta(\mathbf{x}, \mathbf{y}) = \hat{\sigma}_v \left((1 - \beta) \frac{\sum_{u \in \text{in}(v)} \kappa_u^\beta(\mathbf{x}, \mathbf{y})}{|\text{in}(v)|} + \beta \right).$$

The final kernel $\kappa_{\mathcal{S}}^\beta$ is κ_o^β , the kernel associated with the output node o .

Note that our original definitions correspond to $\beta = 0$. With the above definitions, Theorems 2, 3, 4 and 5 extend to the case when there exist bias terms. To simplify the notation, we focus on the case when there are no biases.

6 Mathematical background

Reproducing kernel Hilbert spaces (RKHS). The proofs of all the theorems we quote here are well-known and can be found in Chapter 2 of [48] and similar textbooks. Let \mathcal{H} be a Hilbert space of functions from \mathcal{X} to \mathbb{R} . We say that \mathcal{H} is a *reproducing kernel Hilbert space*, abbreviated RKHS or kernel space, if for every $\mathbf{x} \in \mathcal{X}$ the linear functional $f \mapsto f(\mathbf{x})$ is bounded. The following theorem provides a one-to-one correspondence between kernels and kernel spaces.

Theorem 6. (i) For every kernel κ there exists a unique kernel space \mathcal{H}_κ such that for every $\mathbf{x} \in \mathcal{X}$, $\kappa(\cdot, \mathbf{x}) \in \mathcal{H}_\kappa$ and for all $f \in \mathcal{H}_\kappa$, $f(\mathbf{x}) = \langle f(\cdot), \kappa(\cdot, \mathbf{x}) \rangle_{\mathcal{H}_\kappa}$. (ii) A Hilbert space $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{X}}$ is a kernel space if and only if there exists a kernel $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ such that $\mathcal{H} = \mathcal{H}_\kappa$.

The following theorem describes a tight connection between embeddings of \mathcal{X} into a Hilbert space and kernel spaces.

Theorem 7. A function $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a kernel if and only if there exists a mapping $\Phi : \mathcal{X} \rightarrow \mathcal{H}$ to some Hilbert space for which $\kappa(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{\mathcal{H}}$. In addition, the following two properties hold,

- $\mathcal{H}_\kappa = \{f_{\mathbf{v}} : \mathbf{v} \in \mathcal{H}\}$, where $f_{\mathbf{v}}(\mathbf{x}) = \langle \mathbf{v}, \Phi(\mathbf{x}) \rangle_{\mathcal{H}}$.
- For every $f \in \mathcal{H}_\kappa$, $\|f\|_{\mathcal{H}_\kappa} = \inf\{\|\mathbf{v}\|_{\mathcal{H}} \mid f = f_{\mathbf{v}}\}$.

Positive definite functions. A function $\mu : [-1, 1] \rightarrow \mathbb{R}$ is *positive definite* (PSD) if there are non-negative numbers b_0, b_1, \dots such that

$$\sum_{i=0}^{\infty} b_i < \infty \quad \text{and} \quad \forall x \in [-1, 1], \quad \mu(x) = \sum_{i=0}^{\infty} b_i x^i.$$

The *norm* of μ is defined as $\|\mu\| := \sqrt{\sum_i b_i} = \sqrt{\mu(1)}$. We say that μ is *normalized* if $\|\mu\| = 1$.

Theorem 8 (Schoenberg, [50]). A continuous function $\mu : [-1, 1] \rightarrow \mathbb{R}$ is PSD if and only if for all $d = 1, 2, \dots, \infty$, the function $\kappa : \mathbb{S}^{d-1} \times \mathbb{S}^{d-1} \rightarrow \mathbb{R}$ defined by $\kappa(\mathbf{x}, \mathbf{x}') = \mu(\langle \mathbf{x}, \mathbf{x}' \rangle)$ is a kernel.

The restriction to the unit sphere of many of the kernels used in machine learning applications corresponds to positive definite functions. An example is the Gaussian kernel,

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right).$$

Indeed, note that for unit vectors \mathbf{x}, \mathbf{x}' we have

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}\|^2 + \|\mathbf{x}'\|^2 - 2\langle \mathbf{x}, \mathbf{x}' \rangle}{2\sigma^2}\right) = \exp\left(-\frac{1 - \langle \mathbf{x}, \mathbf{x}' \rangle}{\sigma^2}\right).$$

Another example is the Polynomial kernel $\kappa(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle^d$.

Hermite polynomials. The normalized *Hermite polynomials* is the sequence h_0, h_1, \dots of orthonormal polynomials obtained by applying the Gram-Schmidt process to the sequence $1, x, x^2, \dots$ w.r.t. the inner-product $\langle f, g \rangle = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x)g(x)e^{-\frac{x^2}{2}} dx$. Recall that we define activations as square integrable functions w.r.t. the Gaussian measure. Thus, Hermite polynomials form an orthonormal basis to the space of activations. In particular, each activation σ can be uniquely described in the basis of Hermite polynomials,

$$\sigma(x) = a_0 h_0(x) + a_1 h_1(x) + a_2 h_2(x) + \dots, \quad (2)$$

where the convergence holds in ℓ^2 w.r.t. the Gaussian measure. This decomposition is called the *Hermite expansion*. Finally, we use the following facts (see Chapter 11 in [41] and the relevant [entry](#) in Wikipedia):

$$\forall n \geq 1, h_{n+1}(x) = \frac{x}{\sqrt{n+1}} h_n(x) - \sqrt{\frac{n}{n+1}} h_{n-1}(x), \quad (3)$$

$$\forall n \geq 1, h'_n(x) = \sqrt{n} h_{n-1}(x) \quad (4)$$

$$\mathbb{E}_{(X,Y) \sim N_\rho} h_m(X) h_n(Y) = \begin{cases} \rho^n & n = m \\ 0 & n \neq m \end{cases} \quad \text{where } n, m \geq 0, \rho \in [-1, 1], \quad (5)$$

$$h_n(0) = \begin{cases} 0, & \text{if } n \text{ is odd} \\ \frac{1}{\sqrt{n!}} (-1)^{\frac{n}{2}} (n-1)!! & \text{if } n \text{ is even} \end{cases}, \quad (6)$$

where

$$n!! = \begin{cases} 1 & n \leq 0 \\ n \cdot (n-2) \cdots 5 \cdot 3 \cdot 1 & n > 0 \text{ odd} \\ n \cdot (n-2) \cdots 6 \cdot 4 \cdot 2 & n > 0 \text{ even} \end{cases}.$$

7 Compositional kernel spaces

We now describe the details of compositional kernel spaces. Let \mathcal{S} be a skeleton with normalized activations and n input nodes associated with the input's coordinates. Throughout the rest of the section we study the functions in $\mathcal{H}_{\mathcal{S}}$ and their norm. In particular, we show that $\kappa_{\mathcal{S}}$ is indeed a normalized kernel. Recall that $\kappa_{\mathcal{S}}$ is defined inductively by the equation,

$$\kappa_v(\mathbf{x}, \mathbf{x}') = \hat{\sigma}_v \left(\frac{\sum_{u \in \text{in}(v)} \kappa_u(\mathbf{x}, \mathbf{x}')}{|\text{in}(v)|} \right). \quad (7)$$

The recursion (7) describes a means for generating a kernel from another kernel. Since kernels correspond to kernel spaces, it also prescribes an operator that produces a kernel space from other kernel spaces. If \mathcal{H}_v is the space corresponding to v , we denote this operator by

$$\mathcal{H}_v = \hat{\sigma}_v \left(\frac{\oplus_{u \in \text{in}(v)} \mathcal{H}_u}{|\text{in}(v)|} \right). \quad (8)$$

The reason for using the above notation becomes clear in the sequel. The space $\mathcal{H}_{\mathcal{S}}$ is obtained by starting with the spaces \mathcal{H}_v corresponding to the input nodes and propagating them according to the structure of \mathcal{S} , where at each node v the operation (8) is applied. Hence, to understand $\mathcal{H}_{\mathcal{S}}$ we need to understand this operation as well as the spaces corresponding to input nodes. The latter spaces are rather simple: for an input node v corresponding to the variable \mathbf{x}^i , we have that $\mathcal{H}_v = \{f_{\mathbf{w}} \mid \forall \mathbf{x}, f_{\mathbf{w}}(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x}^i \rangle\}$ and $\|f_{\mathbf{w}}\|_{\mathcal{H}_v} = \|\mathbf{w}\|$. To understand (8), it is convenient to decompose it into two operations. The first operation, termed the *direct average*, is defined through the equation $\tilde{\kappa}_v(\mathbf{x}, \mathbf{x}') = \frac{\sum_{u \in \text{in}(v)} \kappa_u(\mathbf{x}, \mathbf{x}')}{|\text{in}(v)|}$, and the resulting kernel space is denoted $\mathcal{H}_{\tilde{v}} = \frac{\oplus_{u \in \text{in}(v)} \mathcal{H}_u}{|\text{in}(v)|}$. The second operation, called the *extension* according to $\hat{\sigma}_v$, is defined through $\kappa_v(\mathbf{x}, \mathbf{x}') = \hat{\sigma}_v(\tilde{\kappa}_v(\mathbf{x}, \mathbf{x}'))$. The resulting kernel space is denoted $\mathcal{H}_v = \hat{\sigma}_v(\mathcal{H}_{\tilde{v}})$. We next analyze these two operations.

The direct average of kernel spaces. Let $\mathcal{H}_1, \dots, \mathcal{H}_n$ be kernel spaces with kernels $\kappa_1, \dots, \kappa_n : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$. Their *direct average*, denoted $\mathcal{H} = \frac{\mathcal{H}_1 \oplus \dots \oplus \mathcal{H}_n}{n}$, is the kernel space corresponding to the kernel $\kappa(\mathbf{x}, \mathbf{x}') = \frac{1}{n} \sum_{i=1}^n \kappa_i(\mathbf{x}, \mathbf{x}')$.

Lemma 9. *The function κ is indeed a kernel. Furthermore, the following properties hold.*

1. *If $\mathcal{H}_1, \dots, \mathcal{H}_n$ are normalized then so is \mathcal{H} .*
2. $\mathcal{H} = \left\{ \frac{f_1 + \dots + f_n}{n} \mid f_i \in \mathcal{H}_i \right\}$
3. $\|f\|_{\mathcal{H}}^2 = \inf \left\{ \frac{\|f_1\|_{\mathcal{H}_1}^2 + \dots + \|f_n\|_{\mathcal{H}_n}^2}{n} \text{ s.t. } f = \frac{f_1 + \dots + f_n}{n}, f_i \in \mathcal{H}_i \right\}$

Proof. (outline) The fact that κ is a kernel follows directly from the definition of a kernel and the fact that an average of PSD matrices is PSD. Also, it is straight forward to verify item 1. We now proceed to items 2 and 3. By Theorem 7 there are Hilbert spaces $\mathcal{G}_1, \dots, \mathcal{G}_n$ and mappings $\Phi_i : \mathcal{X} \rightarrow \mathcal{G}_i$ such that $\kappa_i(\mathbf{x}, \mathbf{x}') = \langle \Phi_i(\mathbf{x}), \Phi_i(\mathbf{x}') \rangle_{\mathcal{G}_i}$. Consider now the mapping

$$\Psi(\mathbf{x}) = \left(\frac{\Phi_1(\mathbf{x})}{\sqrt{n}}, \dots, \frac{\Phi_n(\mathbf{x})}{\sqrt{n}} \right).$$

It holds that $\kappa(\mathbf{x}, \mathbf{x}') = \langle \Psi(\mathbf{x}), \Psi(\mathbf{x}') \rangle$. Properties 2 and 3 now follow directly from Thm. 7 applied to Ψ . \square

The extension of a kernel space. Let \mathcal{H} be a normalized kernel space with a kernel κ . Let $\mu(x) = \sum_i b_i x^i$ be a PSD function. As we will see shortly, a function is PSD if and only if it is a dual of an activation function. The *extension* of \mathcal{H} w.r.t. μ , denoted $\mu(\mathcal{H})$, is the kernel space corresponding to the kernel $\kappa'(\mathbf{x}, \mathbf{x}') = \mu(\kappa(\mathbf{x}, \mathbf{x}'))$.

Lemma 10. *The function κ' is indeed a kernel. Furthermore, the following properties hold.*

1. $\mu(\mathcal{H})$ is normalized if and only if μ is.

$$2. \mu(\mathcal{H}) = \overline{\text{span}} \left\{ \prod_{g \in A} g \mid A \subset \mathcal{H}, b_{|A|} > 0 \right\} \text{ where } \overline{\text{span}}(\mathcal{A}) \text{ is the closure of the span of } \mathcal{A}.$$

$$3. \|f\|_{\mu(\mathcal{H})} \leq \inf \left\{ \sum_A \frac{\prod_{g \in A} \|g\|_{\mathcal{H}}}{\sqrt{b_{|A|}}} \text{ s.t. } f = \sum_A \prod_{g \in A} g, A \subset \mathcal{H} \right\}$$

Proof. (outline) Let $\Phi : \mathcal{X} \rightarrow \mathcal{G}$ be a mapping from \mathcal{X} to the unit ball of a Hilbert space \mathcal{G} such that $\kappa(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$. Define

$$\Psi(\mathbf{x}) = \left(\sqrt{b_0}, \sqrt{b_1}\Phi(\mathbf{x}), \sqrt{b_2}\Phi(\mathbf{x}) \otimes \Phi(\mathbf{x}), \sqrt{b_3}\Phi(\mathbf{x}) \otimes \Phi(\mathbf{x}) \otimes \Phi(\mathbf{x}), \dots \right)$$

It is not difficult to verify that $\langle \Psi(\mathbf{x}), \Psi(\mathbf{x}') \rangle = \mu(\kappa(\mathbf{x}, \mathbf{x}'))$. Hence, by Thm. 7, κ' is indeed a kernel. Verifying property 1 is a straightforward task. Properties 2 and 3 follow by applying Thm. 7 on the mapping Ψ . \square

8 The dual activation function

The following lemma describes a few basic properties of the dual activation. These properties follow easily from the definition of the dual activation and equations (2), (4), and (5).

Lemma 11. *The following properties of the mapping $\sigma \mapsto \hat{\sigma}$ hold:*

- (a) *If $\sigma = \sum_i a_i h_i$ is the Hermite expansion of σ , then $\hat{\sigma}(\rho) = \sum_i a_i^2 \rho^i$.*
- (b) *For every σ , $\hat{\sigma}$ is positive definite.*
- (c) *Every positive definite function is a dual of some activation.*
- (d) *The mapping $\sigma \mapsto \hat{\sigma}$ preserves norms.*
- (e) *The mapping $\sigma \mapsto \hat{\sigma}$ commutes with differentiation.*
- (f) *For $a \in \mathbb{R}$, $\widehat{a\sigma} = a^2 \hat{\sigma}$.*
- (g) *For every σ , $\hat{\sigma}$ is continuous in $[-1, 1]$ and smooth in $(-1, 1)$.*
- (h) *For every σ , $\hat{\sigma}$ is non-decreasing and convex in $[0, 1]$.*
- (i) *For every σ , the range of $\hat{\sigma}$ is $[-\|\sigma\|^2, \|\sigma\|^2]$.*
- (j) *For every σ , $\hat{\sigma}(0) = (\mathbb{E}_{X \sim N(0,1)} \sigma(X))^2$ and $\hat{\sigma}(1) = \|\sigma\|^2$.*

We next discuss a few examples for activations and calculate their dual activation and kernel. Note that the dual of the exponential activation was calculated in [36] and the duals of the step and the ReLU activations were calculated in [13]. Here, our derivations are different and may prove useful for future calculations of duals for other activations.

The exponential activation. Consider the activation function $\sigma(x) = Ce^{ax}$ where $C > 0$ is a normalization constant such that $\|\sigma\| = 1$. The actual value of C is e^{-2a^2} but it will not be needed for the derivation below. From properties (e) and (f) of Lemma 11 we have that,

$$(\hat{\sigma})' = \widehat{\sigma'} = \widehat{a\sigma} = a^2 \hat{\sigma}.$$

The the solution of ordinary differential equation $(\hat{\sigma})' = a^2 \hat{\sigma}$ is of the form $\hat{\sigma}(\rho) = b \exp(a^2 \rho)$. Since $\hat{\sigma}(1) = 1$ we have $b = e^{-a^2}$. We therefore obtain that the dual activation function is

$$\hat{\sigma}(\rho) = e^{a^2 \rho - a^2} = e^{a^2(\rho-1)}.$$

Note that the kernel induced by σ is the RBF kernel, restricted to the d -dimensional sphere,

$$\kappa_\sigma(\mathbf{x}, \mathbf{x}') = e^{a^2(\langle \mathbf{x}, \mathbf{x}' \rangle - 1)} = e^{-\frac{a^2 \|\mathbf{x} - \mathbf{x}'\|^2}{2}}.$$

The Sine activation and the Sinh kernel. Consider the activation $\sigma(x) = \sin(ax)$. We can write $\sin(ax) = \frac{e^{iax} - e^{-iax}}{2i}$. We have

$$\begin{aligned} \hat{\sigma}(\rho) &= \mathbb{E}_{(X,Y) \sim N_\rho} \left(\frac{e^{iaX} - e^{-iaX}}{2i} \right) \left(\frac{e^{iaY} - e^{-iaY}}{2i} \right) \\ &= -\frac{1}{4} \mathbb{E}_{(X,Y) \sim N_\rho} (e^{iaX} - e^{-iaX}) (e^{iaY} - e^{-iaY}) \\ &= -\frac{1}{4} \mathbb{E}_{(X,Y) \sim N_\rho} [e^{ia(X+Y)} - e^{ia(X-Y)} - e^{ia(-X+Y)} + e^{ia(-X-Y)}]. \end{aligned}$$

Recall that the characteristic function, $\mathbb{E}[e^{itX}]$, when X is distributed $N(0, 1)$ is $e^{-\frac{1}{2}t^2}$. Since $X+Y$ and $-X-Y$ are normal variables with expectation 0 and variance of $2+2\rho$, it follows that,

$$\mathbb{E}_{(X,Y) \sim N_\rho} e^{ia(X+Y)} = \mathbb{E}_{(X,Y) \sim N_\rho} e^{-ia(X+Y)} = e^{-\frac{a^2(2+2\rho)}{2}}.$$

Similarly, since the variance of $X-Y$ and $Y-X$ is $2-2\rho$, we get

$$\mathbb{E}_{(X,Y) \sim N_\rho} e^{ia(X-Y)} = \mathbb{E}_{(X,Y) \sim N_\rho} e^{ia(-X+Y)} = e^{-\frac{a^2(2-2\rho)}{2}}.$$

We therefore obtain that

$$\hat{\sigma}(\rho) = \frac{e^{-a^2(1-\rho)} - e^{-a^2(1+\rho)}}{2} = e^{-a^2} \sinh(a^2 \rho).$$

Hermite activations and polynomial kernels. From Lemma 11 it follows that the dual activation of the Hermite polynomial h_n is $\hat{h}_n(\rho) = \rho^n$. Hence, the corresponding kernel is the polynomial kernel.

The normalized step activation. Consider the activation

$$\sigma(x) = \begin{cases} \sqrt{2} & x > 0 \\ 0 & x \leq 0 \end{cases}.$$

To calculate $\hat{\sigma}$ we compute the Hermite expansion of σ . For $n \geq 0$ we let

$$a_n = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \sigma(x) h_n(x) e^{-\frac{x^2}{2}} dx = \frac{1}{\sqrt{\pi}} \int_0^{\infty} h_n(x) e^{-\frac{x^2}{2}} dx.$$

Since $h_0(x) = 1$, $h_1(x) = x$, and $h_2(x) = \frac{x^2-1}{\sqrt{2}}$, we get the corresponding coefficients,

$$\begin{aligned} a_0 &= \mathbb{E}_{X \sim N(0,1)} [\sigma(X)] = \frac{1}{\sqrt{2}} \\ a_1 &= \mathbb{E}_{X \sim N(0,1)} [\sigma(X)X] = \frac{1}{\sqrt{2}} \mathbb{E}_{X \sim N(0,1)} [|X|] = \frac{1}{\sqrt{\pi}} \\ a_2 &= \frac{1}{\sqrt{2}} \mathbb{E}_{X \sim N(0,1)} [\sigma(X)(X^2 - 1)] = \frac{1}{2} \mathbb{E}_{X \sim N(0,1)} [X^2 - 1] = 0. \end{aligned}$$

For $n \geq 3$ we write $g_n(x) = h_n(x) e^{-\frac{x^2}{2}}$ and note that

$$\begin{aligned} g'_n(x) &= [h'_n(x) - x h_n(x)] e^{-\frac{x^2}{2}} \\ &= [\sqrt{n} h_{n-1}(x) - x h_n(x)] e^{-\frac{x^2}{2}} \\ &= -\sqrt{n+1} h_{n+1}(x) e^{-\frac{x^2}{2}} \\ &= -\sqrt{n+1} g_{n+1}(x). \end{aligned}$$

Here, the second equality follows from (4) and the third from (3). We therefore get

$$\begin{aligned} a_n &= \frac{1}{\sqrt{\pi}} \int_0^{\infty} g_n(x) dx \\ &= -\frac{1}{\sqrt{n\pi}} \int_0^{\infty} g'_{n-1}(x) dx \\ &= \frac{1}{\sqrt{n\pi}} \left(g_{n-1}(0) - \overbrace{g_{n-1}(\infty)}^{=0} \right) \\ &= \frac{1}{\sqrt{n\pi}} h_{n-1}(0) \\ &= \begin{cases} \frac{(-1)^{\frac{n-1}{2}} (n-2)!!}{\sqrt{n\pi} \sqrt{(n-1)!}} = \frac{(-1)^{\frac{n-1}{2}} (n-2)!!}{\sqrt{\pi n!}} & \text{if } n \text{ is odd} \\ 0 & \text{if } n \text{ is even} \end{cases}. \end{aligned}$$

The second equality follows from (3) and the last equality follows from (6). Finally, from Lemma 11 we have that $\hat{\sigma}(\rho) = \sum_{n=0}^{\infty} b_n \rho^n$ where

$$b_n = \begin{cases} \frac{((n-2)!!)^2}{\pi n!} & \text{if } n \text{ is odd} \\ \frac{1}{2} & \text{if } n = 0 \\ 0 & \text{if } n \text{ is even } \geq 2 \end{cases}.$$

In particular, $(b_0, b_1, b_2, b_3, b_4, b_5, b_6) = (\frac{1}{2}, \frac{1}{\pi}, 0, \frac{1}{6\pi}, 0, \frac{3}{40\pi}, 0)$. Note that from the Taylor expansion of \cos^{-1} it follows that $\hat{\sigma}(\rho) = 1 - \frac{\cos^{-1}(\rho)}{\pi}$.

The normalized ReLU activation. Consider the activation $\sigma(x) = \sqrt{2} \max(0, x)$. We now write $\hat{\sigma}(\rho) = \sum_i b_i \rho^i$. The first coefficient is

$$b_0 = \left(\mathbb{E}_{X \sim N(0,1)} \sigma(X) \right)^2 = \frac{1}{2} \left(\mathbb{E}_{X \sim N(0,1)} |X| \right)^2 = \frac{1}{\pi}.$$

To calculate the remaining coefficients we simply note that the derivative of the ReLU activation is the step activation and the mapping $\sigma \mapsto \hat{\sigma}$ commutes with differentiation. Hence, from the calculation of the step activation we get,

$$b_n = \begin{cases} \frac{((n-3)!!)^2}{\pi n!} & \text{if } n \text{ is even} \\ \frac{1}{2} & \text{if } n = 1 \\ 0 & \text{if } n \text{ is odd } \geq 3 \end{cases}.$$

In particular, $(b_0, b_1, b_2, b_3, b_4, b_5, b_6) = (\frac{1}{\pi}, \frac{1}{2}, \frac{1}{2\pi}, 0, \frac{1}{24\pi}, 0, \frac{1}{80\pi})$. We see that the coefficients corresponding to the degrees 0, 1, and 2 sum to 0.9774. The sums up to degrees 4 or 6 are 0.9907 and 0.9947 respectively. That is, we get an excellent approximation of less than 1% error with a dual activation of degree 4.

The collapsing tower of fully connected layers. To conclude this section we discuss the case of very deep networks. The setting is taken for illustrative purposes but it might surface when building networks with numerous fully connected layers. Indeed, most deep architectures that we are aware of do not employ more than five *consecutive* fully connected layers.

Consider a skeleton \mathcal{S}_m consisting of m fully connected layers, each layer associated with the same (normalized) activation σ . We would like to examine the form of the compositional kernel as the number of layers becomes very large. Due to the repeated structure and activation we have

$$\kappa_{\mathcal{S}_m}(\mathbf{x}, \mathbf{y}) = \alpha_m \left(\frac{\langle \mathbf{x}, \mathbf{y} \rangle}{n} \right) \quad \text{where} \quad \alpha_m = \hat{\sigma}^m = \overbrace{\hat{\sigma} \circ \dots \circ \hat{\sigma}}^{m \text{ times}}.$$

Hence, the limiting properties of $\kappa_{\mathcal{S}_m}$ can be understood from the limit of α_m . In the case that $\sigma(x) = x$ or $\sigma(x) = -x$, $\hat{\sigma}$ is the identity function. Therefore $\alpha_m(\rho) = \hat{\sigma}(\rho) = \rho$ for all m and $\kappa_{\mathcal{S}_m}$ is simply the linear kernel. Assume now that σ is neither the identity nor its negation. The following claim shows that α_m has a point-wise limit corresponding to a degenerate kernel.

Claim 1. *There exists a constant $0 \leq \alpha_\sigma \leq 1$ such that for all $-1 < \rho < 1$,*

$$\lim_{m \rightarrow \infty} \alpha_m(\rho) = \alpha_\sigma$$

Before proving the claim, we note that for $\rho = 1$, $\alpha_m(1) = 1$ for all m , and therefore $\lim_{m \rightarrow \infty} \alpha_m(1) = 1$. For $\rho = -1$, if σ is anti-symmetric then $\alpha_m(-1) = -1$ for all m , and in particular $\lim_{m \rightarrow \infty} \alpha_m(-1) = -1$. In any other case, our argument can show that $\lim_{m \rightarrow \infty} \alpha_m(-1) = \alpha_\sigma$.

Proof. Recall that $\hat{\sigma}(\rho) = \sum_{i=0}^{\infty} b_i \rho^i$ where the b_i 's are non-negative numbers that sum to 1. By the assumption that σ is not the identity or its negation, $b_1 < 1$. We first claim that there is a unique $\alpha_\sigma \in [0, 1]$ such that

$$\forall x \in (-1, \alpha_\sigma), \quad \hat{\sigma}(\rho) > \rho \quad \text{and} \quad \forall x \in (\alpha_\sigma, 1), \quad \alpha_\sigma < \hat{\sigma}(\rho) < \rho \quad (9)$$

To prove (9) it suffices to prove the following properties.

- (a) $\hat{\sigma}(\rho) > \rho$ for $\rho \in (-1, 0)$
- (b) $\hat{\sigma}$ is non-decreasing and convex in $[0, 1]$
- (c) $\hat{\sigma}(1) = 1$
- (d) the graph of $\hat{\sigma}$ has at most a single intersection in $[0, 1)$ with the graph of $f(\rho) = \rho$

If the above properties hold we can take α_σ to be the intersection point or 1 if such a point does not exist. We first show (a). For $\rho \in (-1, 0)$ we have that

$$\hat{\sigma}(\rho) = b_0 + \sum_{i=1}^{\infty} b_i \rho^i \geq b_0 - \sum_{i=1}^{\infty} b_i |\rho|^i > - \sum_{i=1}^{\infty} b_i |\rho| \geq -|\rho| = \rho.$$

Here, the third inequality follows from the fact that $b_0 \geq 0$ and for all i , $-b_i |\rho|^i \geq -b_i |\rho|$. Moreover since $b_1 < 1$, one of these inequalities must be strict. Properties (b) and (c) follows from Lemma 11. Finally, to show (d), we note that the second derivative of $\hat{\sigma}(\rho) - \rho$ is $\sum_{i \geq 2} i(i-1)b_i \rho^{i-2}$ which is non-negative in $[0, 1)$. Hence, $\hat{\sigma}(\rho) - \rho$ is convex in $[0, 1]$ and in particular intersects with the x -axis at either 0, 1, 2 or infinitely many times in $[0, 1]$. As we assume that $\hat{\sigma}$ is not the identity, we can rule out the option of infinitely many intersections. Also, since $\hat{\sigma}(1) = 1$, we know that there is at least one intersection in $[0, 1]$. Hence, there are 1 or 2 intersections in $[0, 1]$ and because one of them is in $\rho = 1$, we conclude that there is at most one intersection in $[0, 1)$.

Lastly, we derive the conclusion of the claim from equation (9). Fix $\rho \in (-1, 1)$. Assume first that $\rho \geq \alpha_\sigma$. By (9), $\alpha_m(\rho)$ is a monotonically non-increasing sequence that is lower bounded by α_σ . Hence, it has a limit $\alpha_\sigma \leq \tau \leq \rho < 1$. Now, by the continuity of $\hat{\sigma}$ we have

$$\hat{\sigma}(\tau) = \hat{\sigma}\left(\lim_{m \rightarrow \infty} \alpha_m(\rho)\right) = \lim_{m \rightarrow \infty} \hat{\sigma}(\alpha_m(\rho)) = \lim_{m \rightarrow \infty} \alpha_{m+1}(\rho) = \tau.$$

Since the only solution to $\hat{\sigma}(\rho) = \rho$ in $(-1, 1)$ is α_σ , we must have $\tau = \alpha_\sigma$. We next deal with the case that $-1 < \rho < \alpha_\sigma$. If for some m , $\alpha_m(\rho) \in [\alpha_\sigma, 1)$, the argument for $\alpha_\sigma \leq \rho$ shows that $\alpha_\sigma = \lim_{m \rightarrow \infty} \alpha_m(\rho)$. If this is not the case, we have that for all m , $\alpha_m(\rho) \leq \alpha_{m+1}(\rho) \leq \alpha_\sigma$. As in the case of $\rho \geq \alpha_\sigma$, this can be used to show that $\alpha_m(\rho)$ converges to α_σ . \square

9 Proofs

9.1 Well-behaved activations

The proof of our main results applies to activations that are decent, i.e. well-behaved, in a sense defined in the sequel. We then show that C -bounded activations as well as the ReLU activation are decent. We first need to extend the definition of the dual activation and kernel to apply to vectors in \mathbb{R}^d , rather than just \mathbb{S}^d . We denote by \mathcal{M}_+ the collection of 2×2 positive semi-definite matrices and by \mathcal{M}_{++} the collection of positive definite matrices.

Definition 10. Let σ be an activation. Define the following,

$$\bar{\sigma} : \mathcal{M}_+^2 \rightarrow \mathbb{R} \quad , \quad \bar{\sigma}(\Sigma) = \mathbb{E}_{(X,Y) \sim N(0,\Sigma)} \sigma(X)\sigma(Y) \quad , \quad k_\sigma(\mathbf{x}, \mathbf{y}) = \bar{\sigma} \begin{pmatrix} \|\mathbf{x}\|^2 & \langle \mathbf{x}, \mathbf{y} \rangle \\ \langle \mathbf{x}, \mathbf{y} \rangle & \|\mathbf{y}\|^2 \end{pmatrix}.$$

We underscore the following properties of the extension of a dual activation.

- (a) The following equality holds,

$$\hat{\sigma}(\rho) = \bar{\sigma} \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

- (b) The restriction of the extended k_σ to the sphere agrees with the restricted definition.
- (c) The extended dual activation and kernel are defined for every activation σ such that for all $a \geq 0$, $x \mapsto \sigma(ax)$ is square integrable with respect to the Gaussian measure.
- (d) For $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, if $\mathbf{w} \in \mathbb{R}^d$ is a multivariate normal distribution with zero mean vector and identity covariance matrix, then

$$k_\sigma(\mathbf{x}, \mathbf{y}) = \mathbb{E}_{\mathbf{w}} \sigma(\langle \mathbf{w}, \mathbf{x} \rangle) \sigma(\langle \mathbf{w}, \mathbf{y} \rangle).$$

Denote

$$\mathcal{M}_+^\gamma := \left\{ \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12} & \Sigma_{22} \end{pmatrix} \in \mathcal{M}_+ \mid 1 - \gamma \leq \Sigma_{11}, \Sigma_{22} \leq 1 + \gamma \right\}.$$

Definition 11. A normalized activation σ is (α, β, γ) -decent for $\alpha, \beta, \gamma \geq 0$ if the following conditions hold.

- (i) The dual activation $\bar{\sigma}$ is β -Lipschitz in \mathcal{M}_+^γ with respect to the ∞ -norm.
- (ii) If $(X_1, Y_1), \dots, (X_r, Y_r)$ are independent samples from $N(0, \Sigma)$ for $\Sigma \in \mathcal{M}_+^\gamma$ then

$$\Pr \left(\left| \frac{\sum_{i=1}^r \sigma(X_i) \sigma(Y_i)}{r} - \bar{\sigma}(\Sigma) \right| \geq \epsilon \right) \leq 2 \exp \left(-\frac{r\epsilon^2}{2\alpha^2} \right).$$

Lemma 12 (Bounded activations are decent). *Let $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ be a C -bounded normalized activation. Then, σ is $(C^2, 2C^2, \gamma)$ -decent for all $\gamma \geq 0$.*

Proof. It is enough to show that the following properties hold.

1. The (extended) dual activation $\bar{\sigma}$ is $2C^2$ -Lipschitz in \mathcal{M}_{++} w.r.t. the ∞ -norm.
2. If $(X_1, Y_1), \dots, (X_r, Y_r)$ are independent samples from $N(0, \Sigma)$ then

$$\Pr \left(\left| \frac{\sum_{i=1}^r \sigma(X_i) \sigma(Y_i)}{r} - \bar{\sigma}(\Sigma) \right| \geq \epsilon \right) \leq 2 \exp \left(-\frac{r\epsilon^2}{2C^4} \right)$$

From the boundedness of σ it holds that $|\sigma(X)\sigma(Y)| \leq C^2$. Hence, the second property follows directly from Hoeffding's bound. We next prove the first part. Let $\mathbf{z} = (x, y)$ and $\phi(\mathbf{z}) = \sigma(x)\sigma(y)$. Note that for $\Sigma \in \mathcal{M}_{++}$ we have

$$\bar{\sigma}(\Sigma) = \frac{1}{2\pi\sqrt{\det(\Sigma)}} \int_{\mathbb{R}^2} \phi(\mathbf{z}) e^{-\frac{\mathbf{z}^\top \Sigma^{-1} \mathbf{z}}{2}} d\mathbf{z}.$$

Thus we get that,

$$\begin{aligned} \frac{\partial \bar{\sigma}}{\partial \Sigma} &= \frac{1}{2\pi} \int_{\mathbb{R}^2} \phi(\mathbf{z}) \left[\frac{\frac{1}{2}\sqrt{\det(\Sigma)}\Sigma^{-1} - \frac{1}{2}\sqrt{\det(\Sigma)}(\Sigma^{-1}\mathbf{z}\mathbf{z}^\top\Sigma^{-1})}{\det(\Sigma)} \right] e^{-\frac{\mathbf{z}^\top \Sigma^{-1} \mathbf{z}}{2}} d\mathbf{z} \\ &= \frac{1}{2\pi\sqrt{\det(\Sigma)}} \int_{\mathbb{R}^2} \phi(\mathbf{z}) \frac{1}{2} [\Sigma^{-1} - \Sigma^{-1}\mathbf{z}\mathbf{z}^\top\Sigma^{-1}] e^{-\frac{\mathbf{z}^\top \Sigma^{-1} \mathbf{z}}{2}} d\mathbf{z} \end{aligned}$$

Let $g(\mathbf{z}) = e^{-\frac{\mathbf{z}^\top \Sigma^{-1} \mathbf{z}}{2}}$. Then, the first and second order partial derivatives of g are

$$\begin{aligned} \frac{\partial g}{\partial \mathbf{z}} &= -\Sigma^{-1}\mathbf{z} e^{-\frac{\mathbf{z}^\top \Sigma^{-1} \mathbf{z}}{2}} \\ \frac{\partial^2 g}{\partial^2 \mathbf{z}} &= [-\Sigma^{-1} + \Sigma^{-1}\mathbf{z}\mathbf{z}^\top\Sigma^{-1}] e^{-\frac{\mathbf{z}^\top \Sigma^{-1} \mathbf{z}}{2}}. \end{aligned}$$

We therefore obtain that,

$$\frac{\partial \bar{\sigma}}{\partial \Sigma} = -\frac{1}{4\pi\sqrt{\det(\Sigma)}} \int_{\mathbb{R}^2} \phi \frac{\partial^2 g}{\partial^2 \mathbf{z}} d\mathbf{z}.$$

By the product rule we have

$$\frac{\partial \bar{\sigma}}{\partial \Sigma} = -\frac{1}{2\pi\sqrt{\det(\Sigma)}} \frac{1}{2} \int_{\mathbb{R}^2} \frac{\partial^2 \phi}{\partial^2 \mathbf{z}} g d\mathbf{z} = -\frac{1}{2} \mathbb{E}_{(X,Y) \sim \mathcal{N}(0,\Sigma)} \left[\frac{\partial^2 \phi}{\partial^2 \mathbf{z}}(X, Y) \right]$$

We conclude that $\bar{\sigma}$ is differentiable in \mathcal{M}_{++} with partial derivatives that are point-wise bounded by $\frac{C^2}{2}$. Thus, $\bar{\sigma}$ is $2C^2$ -Lipschitz in \mathcal{M}_+ w.r.t. the ∞ -norm. \square

We next show that the ReLU activation is decent.

Lemma 13 (ReLU is decent). *There exists a constant $\alpha_{\text{ReLU}} \geq 1$ such that for $0 \leq \gamma \leq 1$, the normalized ReLU activation $\sigma(x) = \sqrt{2} \max(0, x)$ is $(\alpha_{\text{ReLU}}, 1 + o(\gamma), \gamma)$ -decent.*

Proof. The measure concentration property follows from standard concentration bounds for sub-exponential random variables (e.g. [53]). It remains to show that $\bar{\sigma}$ is $(1 + o(\gamma))$ -Lipschitz in \mathcal{M}_+^γ . We first calculate an exact expression for $\bar{\sigma}$. The expression was already calculated in [13], yet we give here a derivation for completeness.

Claim 2. *The following equality holds for all $\Sigma \in \mathcal{M}_+^2$,*

$$\bar{\sigma}(\Sigma) = \sqrt{\Sigma_{11}\Sigma_{22}} \hat{\sigma}\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right).$$

Proof. Let us denote

$$\tilde{\Sigma} = \begin{pmatrix} 1 & \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}} \\ \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}} & 1 \end{pmatrix}.$$

By the positive homogeneity of the ReLU activation we have

$$\begin{aligned} \bar{\sigma}(\Sigma) &= \mathbb{E}_{(X,Y) \sim \mathcal{N}(0,\Sigma)} \sigma(X)\sigma(Y) \\ &= \sqrt{\Sigma_{11}\Sigma_{22}} \mathbb{E}_{(X,Y) \sim \mathcal{N}(0,\Sigma)} \sigma\left(\frac{X}{\sqrt{\Sigma_{11}}}\right) \sigma\left(\frac{Y}{\sqrt{\Sigma_{22}}}\right) \\ &= \sqrt{\Sigma_{11}\Sigma_{22}} \mathbb{E}_{(\tilde{X}, \tilde{Y}) \sim \mathcal{N}(0, \tilde{\Sigma})} \sigma(\tilde{X}) \sigma(\tilde{Y}) \\ &= \sqrt{\Sigma_{11}\Sigma_{22}} \hat{\sigma}\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right). \end{aligned}$$

which concludes the proof. \square

For brevity, we henceforth drop the argument from $\bar{\sigma}(\Sigma)$ and use the abbreviation $\bar{\sigma}$. In order to show that $\bar{\sigma}$ is $(1 + o(\gamma))$ -Lipschitz w.r.t. the ∞ -norm it is enough to show that for every $\Sigma \in \mathcal{M}_+^\gamma$ we have,

$$\|\nabla \bar{\sigma}\|_1 = \left| \frac{\partial \bar{\sigma}}{\partial \Sigma_{12}} \right| + \left| \frac{\partial \bar{\sigma}}{\partial \Sigma_{11}} \right| + \left| \frac{\partial \bar{\sigma}}{\partial \Sigma_{22}} \right| \leq 1 + o(\gamma). \quad (10)$$

First, Note that $\partial\bar{\sigma}/\partial\Sigma_{11}$ and $\partial\bar{\sigma}/\partial\Sigma_{22}$ have the same sign, hence,

$$\|\nabla\bar{\sigma}\|_1 = \left| \frac{\partial\bar{\sigma}}{\partial\Sigma_{12}} \right| + \left| \frac{\partial\bar{\sigma}}{\partial\Sigma_{11}} + \frac{\partial\bar{\sigma}}{\partial\Sigma_{22}} \right|.$$

Next we get that,

$$\begin{aligned} \frac{\partial\bar{\sigma}}{\partial\Sigma_{11}} &= \frac{1}{2} \sqrt{\frac{\Sigma_{22}}{\Sigma_{11}}} \hat{\sigma}\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \frac{1}{2} \sqrt{\frac{\Sigma_{22}}{\Sigma_{11}}} \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}} \hat{\sigma}'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) \\ \frac{\partial\bar{\sigma}}{\partial\Sigma_{22}} &= \frac{1}{2} \sqrt{\frac{\Sigma_{11}}{\Sigma_{22}}} \hat{\sigma}\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \frac{1}{2} \sqrt{\frac{\Sigma_{11}}{\Sigma_{22}}} \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}} \hat{\sigma}'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) \\ \frac{\partial\bar{\sigma}}{\partial\Sigma_{12}} &= \hat{\sigma}'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right). \end{aligned}$$

We therefore get that the 1-norm of $\nabla\bar{\sigma}$ is,

$$\|\nabla\bar{\sigma}\|_1 = \frac{1}{2} \frac{\Sigma_{11} + \Sigma_{22}}{\sqrt{\Sigma_{11}\Sigma_{22}}} \left| \hat{\sigma}\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}} \hat{\sigma}'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) \right| + \hat{\sigma}'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right).$$

The gradient of $\frac{1}{2} \frac{\Sigma_{11} + \Sigma_{22}}{\sqrt{\Sigma_{11}\Sigma_{22}}}$ at $(\Sigma_{11}, \Sigma_{22}) = (1, 1)$ is $(0, 0)$. Therefore, from the mean value theorem we get, $\frac{1}{2} \frac{\Sigma_{11} + \Sigma_{22}}{\sqrt{\Sigma_{11}\Sigma_{22}}} = 1 + o(\gamma)$. Furthermore, $\hat{\sigma}$, $\hat{\sigma}'$ and $\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}$ are bounded by 1 in absolute value. Hence, we can write,

$$\|\nabla\bar{\sigma}\|_1 = \left| \hat{\sigma}\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) - \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}} \hat{\sigma}'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) \right| + \hat{\sigma}'\left(\frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}\right) + o(\gamma).$$

Finally, if we let $t = \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}$, we can further simplify the expression for $\nabla\bar{\sigma}$,

$$\begin{aligned} \|\nabla\bar{\sigma}(\Sigma)\|_1 &= |\hat{\sigma}(t) - t\hat{\sigma}'(t)| + |\hat{\sigma}'(t)| + o(\gamma) \\ &= \frac{\sqrt{1-t^2}}{\pi} + 1 - \frac{\cos^{-1}(t)}{\pi} + o(\gamma). \end{aligned}$$

Finally, the proof is obtained from the fact that the function $f(t) = \frac{\sqrt{1-t^2}}{\pi} + 1 - \frac{\cos^{-1}(t)}{\pi}$ satisfies $0 \leq f(t) \leq 1$ for every $t \in [-1, 1]$. Indeed, it is simple to verify that $f(-1) = 0$ and $f(1) = 1$. Hence, it suffices to show that f' is non-negative in $[-1, 1]$ which is indeed the case since,

$$f'(t) = \frac{1}{\pi} \frac{1-t}{\sqrt{1-t^2}} = \frac{1}{\pi} \sqrt{\frac{1-t}{1+t}} \geq 0. \quad \square$$

9.2 Proofs of Thms. 2 and 3

We start by an additional theorem which serves as a simple stepping stone for proving the aforementioned main theorems.

Theorem 14. Let \mathcal{S} be a skeleton with (α, β, γ) -decent activations, $0 < \epsilon \leq \gamma$, and $B_d = \sum_{i=0}^{d-1} \beta^i$. Let \mathbf{w} be a random initialization of the network $\mathcal{N} = \mathcal{N}(\mathcal{S}, r)$ with

$$r \geq \frac{2\alpha^2 B_{\text{depth}(\mathcal{S})}^2 \log\left(\frac{8|\mathcal{S}|}{\delta}\right)}{\epsilon^2}.$$

Then, for every \mathbf{x}, \mathbf{y} with probability of at least $1 - \delta$, it holds that

$$|\kappa_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) - \kappa_{\mathcal{S}}(\mathbf{x}, \mathbf{y})| \leq \epsilon.$$

Before proving the theorem we show that together with Lemmas 12 and 13, Theorems 2 and 3 follow from Theorem 14. We restate them as corollaries, prove them, and then proceed to the proof of Theorem 14.

Corollary 15. Let \mathcal{S} be a skeleton with C -bounded activations. Let \mathbf{w} be a random initialization of $\mathcal{N} = \mathcal{N}(\mathcal{S}, r)$ with

$$r \geq \frac{(4C^4)^{\text{depth}(\mathcal{S})+1} \log\left(\frac{8|\mathcal{S}|}{\delta}\right)}{\epsilon^2}.$$

Then, for every \mathbf{x}, \mathbf{y} , w.p. $\geq 1 - \delta$,

$$|\kappa_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) - \kappa_{\mathcal{S}}(\mathbf{x}, \mathbf{y})| \leq \epsilon.$$

Proof. From Lemma 12, for all $\gamma > 0$, each activation is $(C^2, 2C^2, \gamma)$ -decent. By Theorem 14, it suffices to show that

$$2(C^2)^2 \left(\sum_{i=0}^{\text{depth}(\mathcal{S})-1} (2C^2)^i \right)^2 \leq (4C^4)^{\text{depth}(\mathcal{S})+1}.$$

The sum of can be bounded above by,

$$\sum_{i=0}^{\text{depth}(\mathcal{S})-1} (2C^2)^i = \frac{(2C^2)^{\text{depth}(\mathcal{S})} - 1}{2C^2 - 1} \leq \frac{(2C^2)^{\text{depth}(\mathcal{S})}}{C^2}.$$

Therefore, we get that,

$$2(C^2)^2 \left(\sum_{i=0}^{\text{depth}(\mathcal{S})-1} (2C^2)^i \right)^2 \leq \frac{2C^4 (4C^4)^{\text{depth}(\mathcal{S})}}{C^4} \leq (4C^4)^{\text{depth}(\mathcal{S})+1},$$

which concludes the proof. \square

Corollary 16. Let \mathcal{S} be a skeleton with ReLU activations, and \mathbf{w} a random initialization of $\mathcal{N}(\mathcal{S}, r)$ with $r \geq c_1 \frac{\text{depth}^2(\mathcal{S}) \log(\frac{8|\mathcal{S}|}{\delta})}{\epsilon^2}$. For all \mathbf{x}, \mathbf{y} and $\epsilon \leq \min(c_2, \frac{1}{\text{depth}(\mathcal{S})})$, w.p. $\geq 1 - \delta$,

$$|\kappa_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) - \kappa_{\mathcal{S}}(\mathbf{x}, \mathbf{y})| \leq \epsilon$$

Here, $c_1, c_2 > 0$ are universal constants.

Proof. From Lemma 13, each activation is $(\alpha_{\text{ReLU}}, 1 + o(\epsilon), \epsilon)$ -decent. By Theorem 14, it is enough to show that

$$\sum_{i=0}^{\text{depth}(\mathcal{S})-1} (1 + o(\epsilon))^i = O(\text{depth}(\mathcal{S})).$$

This claim follows from the fact that $(1 + o(\epsilon))^i \leq e^{o(\epsilon)\text{depth}(\mathcal{S})}$ as long as $i \leq \text{depth}(\mathcal{S})$. Since we assume that $\epsilon \leq 1/\text{depth}(\mathcal{S})$, the expression is bounded by e for sufficiently small ϵ . \square

We next prove Theorem 14.

Proof. (Theorem 14) For a node $u \in \mathcal{S}$ we denote by $\Psi_{u, \mathbf{w}} : \mathcal{X} \rightarrow \mathbb{R}^r$ the normalized representation of \mathcal{S} 's sub-skeleton rooted at u . Analogously, $\kappa_{u, \mathbf{w}}$ denotes the empirical kernel of that network. When u is the output node of \mathcal{S} we still use $\Psi_{\mathbf{w}}$ and $\kappa_{\mathbf{w}}$ for $\Psi_{u, \mathbf{w}}$ and $\kappa_{u, \mathbf{w}}$. Given two fixed $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ and a node $u \in \mathcal{S}$, we denote

$$\mathcal{K}_{\mathbf{w}}^u = \begin{pmatrix} \kappa_{u, \mathbf{w}}(\mathbf{x}, \mathbf{x}) & \kappa_{u, \mathbf{w}}(\mathbf{x}, \mathbf{y}) \\ \kappa_{u, \mathbf{w}}(\mathbf{x}, \mathbf{y}) & \kappa_{u, \mathbf{w}}(\mathbf{y}, \mathbf{y}) \end{pmatrix}, \quad \mathcal{K}^u = \begin{pmatrix} \kappa_u(\mathbf{x}, \mathbf{x}) & \kappa_u(\mathbf{x}, \mathbf{y}) \\ \kappa_u(\mathbf{x}, \mathbf{y}) & \kappa_u(\mathbf{y}, \mathbf{y}) \end{pmatrix}$$

$$\mathcal{K}_{\mathbf{w}}^{\leftarrow u} = \frac{\sum_{v \in \text{in}(u)} \mathcal{K}_{\mathbf{w}}^v}{|\text{in}(u)|}, \quad \mathcal{K}^{\leftarrow u} = \frac{\sum_{v \in \text{in}(u)} \mathcal{K}^v}{|\text{in}(u)|}.$$

For a matrix $\mathcal{K} \in \mathcal{M}_+$ and a function $f : \mathcal{M}_+ \rightarrow \mathbb{R}$, we denote

$$f^p(\mathcal{K}) = \begin{pmatrix} f\left(\begin{pmatrix} \mathcal{K}_{11} & \mathcal{K}_{11} \\ \mathcal{K}_{11} & \mathcal{K}_{11} \end{pmatrix}\right) & f(\mathcal{K}) \\ f(\mathcal{K}) & f\left(\begin{pmatrix} \mathcal{K}_{22} & \mathcal{K}_{22} \\ \mathcal{K}_{22} & \mathcal{K}_{22} \end{pmatrix}\right) \end{pmatrix}$$

Note that $\mathcal{K}^u = \bar{\sigma}_u^p(\mathcal{K}^{\leftarrow u})$. We say that a node $u \in \mathcal{S}$, is *well-initialized* if

$$\|\mathcal{K}_{\mathbf{w}}^u - \mathcal{K}^u\|_{\infty} \leq \epsilon \frac{B_{\text{depth}(u)}}{B_{\text{depth}(\mathcal{S})}}. \quad (11)$$

Here, we use the convention that $B_0 = 0$. It is enough to show that with probability of at least $\geq 1 - \delta$ all nodes are well-initialized. We first note that input nodes are well-initialized by construction since $\mathcal{K}_{\mathbf{w}}^u = \mathcal{K}^u$. Next, we show that given that all incoming nodes for a certain node are well-initialized, then w.h.p. the node is well-initialized as well.

Claim 3. Assume that all the nodes in $\text{in}(u)$ are well-initialized. Then, the node u is well-initialized with probability of at least $1 - \frac{\delta}{|\mathcal{S}|}$.

Proof. It is easy to verify that $\mathcal{K}_{\mathbf{w}}^u$ is the empirical covariance matrix of r independent variables distributed according to $(\sigma(X), \sigma(Y))$ where $(X, Y) \sim \mathcal{N}(0, \mathcal{K}_{\mathbf{w}}^{\leftarrow u})$. Given the assumption that all nodes incoming to u are well-initialized, we have,

$$\begin{aligned} \|\mathcal{K}_{\mathbf{w}}^{\leftarrow u} - \mathcal{K}^{\leftarrow u}\|_{\infty} &= \left\| \frac{\sum_{v \in \text{in}(v)} \mathcal{K}_{\mathbf{w}}^v}{|\text{in}(v)|} - \frac{\sum_{v \in \text{in}(v)} \mathcal{K}^v}{|\text{in}(v)|} \right\|_{\infty} \\ &\leq \frac{1}{|\text{in}(v)|} \sum_{v \in \text{in}(v)} \|\mathcal{K}_{\mathbf{w}}^v - \mathcal{K}^v\|_{\infty} \\ &\leq \epsilon \frac{B_{\text{depth}(u)-1}}{B_{\text{depth}(\mathcal{S})}}. \end{aligned} \quad (12)$$

Further, since $\epsilon \leq \gamma$ then $\mathcal{K}_{\mathbf{w}}^{\leftarrow u} \in \mathcal{M}_+^{\gamma}$. Using the fact that σ_u is (α, β, γ) -decent and that $r \geq \frac{2\alpha^2 B_{\text{depth}(\mathcal{S})}^2 \log(\frac{8|\mathcal{S}|}{\delta})}{\epsilon^2}$, we get that w.p. of at least $1 - \frac{\delta}{|\mathcal{S}|}$,

$$\|\mathcal{K}_{\mathbf{w}}^u - \bar{\sigma}_u^p(\mathcal{K}_{\mathbf{w}}^{\leftarrow u})\|_{\infty} \leq \frac{\epsilon}{B_{\text{depth}(\mathcal{S})}}. \quad (13)$$

Finally, using (12) and (13) along with the fact that $\bar{\sigma}$ is β -Lipschitz, we have

$$\begin{aligned} \|\mathcal{K}_{\mathbf{w}}^u - \mathcal{K}^u\|_{\infty} &= \|\mathcal{K}_{\mathbf{w}}^u - \bar{\sigma}_u^p(\mathcal{K}_{\mathbf{w}}^{\leftarrow u})\|_{\infty} \\ &\leq \|\mathcal{K}_{\mathbf{w}}^u - \bar{\sigma}_u^p(\mathcal{K}_{\mathbf{w}}^{\leftarrow u})\|_{\infty} + \|\bar{\sigma}_u^p(\mathcal{K}_{\mathbf{w}}^{\leftarrow u}) - \bar{\sigma}_u^p(\mathcal{K}^{\leftarrow u})\|_{\infty} \\ &\leq \frac{\epsilon}{B_{\text{depth}(\mathcal{S})}} + \beta \|\mathcal{K}_{\mathbf{w}}^{\leftarrow u} - \mathcal{K}^{\leftarrow u}\|_{\infty} \\ &\leq \frac{\epsilon}{B_{\text{depth}(\mathcal{S})}} + \beta \epsilon \frac{B_{\text{depth}(u)-1}}{B_{\text{depth}(\mathcal{S})}} = \epsilon \frac{B_{\text{depth}(u)}}{B_{\text{depth}(\mathcal{S})}}. \quad \square \end{aligned}$$

We are now ready to conclude the proof. Let $u_1, \dots, u_{|\mathcal{S}|}$ be an ordered list of the nodes in \mathcal{S} in accordance to their depth, starting with the shallowest nodes, and ending with the output node. Denote by A_q the event that u_1, \dots, u_q are well-initialized. We need to show that $\Pr(A_{|\mathcal{S}|}) \geq 1 - \delta$. We do so using an induction on q for the inequality $\Pr(A_q) \geq 1 - \frac{q\delta}{|\mathcal{S}|}$. Indeed, for $q = 1, \dots, n$, u_q is an input node and $\Pr(A_q) = 1$. Thus, the base of the induction hypothesis holds. Assume that $q > n$. By Claim (3) we have that $\Pr(A_q | A_{q-1}) \geq 1 - \frac{\delta}{|\mathcal{S}|}$. Finally, from the induction hypothesis we have,

$$\Pr(A_q) \geq \Pr(A_q | A_{q-1}) \Pr(A_{q-1}) \geq \left(1 - \frac{\delta}{|\mathcal{S}|}\right) \left(1 - \frac{(q-1)\delta}{|\mathcal{S}|}\right) \geq 1 - \frac{q\delta}{|\mathcal{S}|}. \quad \square$$

9.3 Proofs of Thms. 4 and 5

Theorems 4 and 5 follow from using the following lemma combined with Theorems 2 and 3. When we apply the lemma, we always focus on the special case where one of the kernels is constant w.p. 1.

Lemma 17. Let \mathcal{D} be a distribution on $\mathcal{X} \times \mathcal{Y}$, $\ell : \mathbb{R} \times \mathcal{Y} \rightarrow \mathbb{R}$ be an L -Lipschitz loss, $\delta > 0$, and $\kappa_1, \kappa_2 : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be two independent random kernels sample from arbitrary distributions. Assume that the following properties hold.

- For some $C > 0$, $\forall \mathbf{x} \in \mathcal{X}$, $\kappa_1(\mathbf{x}, \mathbf{x}), \kappa_2(\mathbf{x}, \mathbf{x}) \leq C$.
- $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$, $\Pr_{\kappa_1, \kappa_2} (|\kappa_1(\mathbf{x}, \mathbf{y}) - \kappa_2(\mathbf{x}, \mathbf{y})| \geq \epsilon) \leq \tilde{\delta}$ for $\tilde{\delta} < c_2 \frac{\epsilon^2 \delta}{C^2 \log^2(\frac{1}{\delta})}$ where $c_2 > 0$ is a universal constant.

Then, w.p. $\geq 1 - \delta$ over the choices of κ_1, κ_2 , for every $f_1 \in \mathcal{H}_{\kappa_1}^M$ there is $f_2 \in \mathcal{H}_{\kappa_2}^{\sqrt{2}M}$ such that $\mathcal{L}_{\mathcal{D}}(f_2) \leq \mathcal{L}_{\mathcal{D}}(f_1) + \sqrt{\epsilon} 4LM$.

To prove the above lemma, we state another lemma below followed by a basic measure concentration result.

Lemma 18. Let $\mathbf{x}_1, \dots, \mathbf{x}_m \in \mathbb{R}^d$, $\mathbf{w}^* \in \mathbb{R}^d$ and $\epsilon > 0$. There are weights $\alpha_1, \dots, \alpha_m$ such that for $\mathbf{w} := \sum_{i=1}^m \alpha_i \mathbf{x}_i$ we have,

- $\mathcal{L}(\mathbf{w}) := \frac{1}{m} \sum_{i=1}^m |\langle \mathbf{w}, \mathbf{x}_i \rangle - \langle \mathbf{w}^*, \mathbf{x}_i \rangle| \leq \epsilon$
- $\sum_i |\alpha_i| \leq \frac{\|\mathbf{w}^*\|^2}{\epsilon}$
- $\|\mathbf{w}\| \leq \|\mathbf{w}^*\|$

Proof. Denote $M = \|\mathbf{w}^*\|$, $C = \max_i \|\mathbf{x}_i\|$, and $y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle$. Suppose that we run stochastic gradient decent on the sample $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$ w.r.t. the loss $\mathcal{L}(\mathbf{w})$, with learning rate $\eta = \frac{\epsilon}{C^2}$, and with projections onto the ball of radius M . Namely, we start with $\mathbf{w}_0 = 0$ and at each iteration $t \geq 1$, we choose at random $i_t \in [m]$ and perform the update,

$$\tilde{\mathbf{w}}_t = \begin{cases} \mathbf{w}_{t-1} - \eta \mathbf{x}_{i_t} & \langle \mathbf{w}_{t-1}, \mathbf{x}_{i_t} \rangle \geq y_{i_t} \\ \mathbf{w}_{t-1} + \eta \mathbf{x}_{i_t} & \langle \mathbf{w}_{t-1}, \mathbf{x}_{i_t} \rangle < y_{i_t} \end{cases}$$

$$\mathbf{w}_t = \begin{cases} \tilde{\mathbf{w}}_t & \|\tilde{\mathbf{w}}_t\| \leq M \\ \frac{M \tilde{\mathbf{w}}_t}{\|\tilde{\mathbf{w}}_t\|} & \|\tilde{\mathbf{w}}_t\| > M \end{cases}$$

After $T = \frac{M^2 C^2}{\epsilon^2}$ iterations the loss in expectation would be at most ϵ (see for instance Chapter 14 in [53]). In particular, there exists a sequence of at most $\frac{M^2 C^2}{\epsilon^2}$ gradient steps that attains a solution \mathbf{w} with $\mathcal{L}(\mathbf{w}) \leq \epsilon$. Each update adds or subtracts $\frac{\epsilon}{C^2} \mathbf{x}_i$ from the current solution. Hence \mathbf{w} can be written as a weighted sum of \mathbf{x}_i 's where the sum of each coefficient is at most $T \frac{\epsilon}{C^2} = \frac{M^2}{\epsilon}$. \square

Theorem 19 (Bartlett and Mendelson [8]). Let \mathcal{D} be a distribution over $\mathcal{X} \times \mathcal{Y}$, $\ell : \mathbb{R} \times \mathcal{Y} \rightarrow \mathbb{R}$ a 1-Lipschitz loss, $\kappa : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ a kernel, and $\epsilon, \delta > 0$. Let $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$ be i.i.d. samples from \mathcal{D} such that $m \geq c \frac{M^2 \max_{\mathbf{x} \in \mathcal{X}} \kappa(\mathbf{x}, \mathbf{x}) + \log(\frac{1}{\delta})}{\epsilon^2}$ where c is a constant. Then, with probability of at least $1 - \delta$ we have,

$$\forall f \in \mathcal{H}_{\kappa}^M, |\mathcal{L}_{\mathcal{D}}(f) - \mathcal{L}_S(f)| \leq \epsilon.$$

Proof. (of Lemma 17) By rescaling ℓ , we can assume w.l.o.g that $L = 1$. Let $\epsilon_1 = \sqrt{\epsilon}M$ and $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\} \sim \mathcal{D}$ be i.i.d. samples which are independent of the choice of κ_1, κ_2 . By Theorem 19, for a large enough constant c , if $m = c \frac{CM^2 \log(\frac{1}{\delta})}{\epsilon_1^2} = c \frac{C \log(\frac{1}{\delta})}{\epsilon}$, then w.p. $\geq 1 - \frac{\delta}{2}$ over the choice of the samples we have,

$$\forall f \in \mathcal{H}_{\kappa_1}^M \cup \mathcal{H}_{\kappa_2}^{\sqrt{2}M}, |\mathcal{L}_{\mathcal{D}}(f) - \mathcal{L}_S(f)| \leq \epsilon_1 \quad (14)$$

Now, if we choose $c_2 = \frac{1}{2c^2}$ then w.p. $\geq 1 - m^2\tilde{\delta} \geq 1 - \frac{\delta}{2}$ (over the choice of the examples and the kernel), we have that

$$\forall i, j \in [m], |\kappa_1(\mathbf{x}_i, \mathbf{x}_j) - \kappa_2(\mathbf{x}_i, \mathbf{x}_j)| < \epsilon. \quad (15)$$

In particular, w.p. $\geq 1 - \delta$ (14) and (15) hold and therefore it suffices to prove the conclusion of the theorem under these conditions. Indeed, let $\Psi_1, \Psi_2 : \mathcal{X} \rightarrow \mathcal{H}$ be two mapping from \mathcal{X} to a Hilbert space \mathcal{H} so that $\kappa_i(\mathbf{x}, \mathbf{y}) = \langle \Psi_i(\mathbf{x}), \Psi_i(\mathbf{y}) \rangle$. Let $f_1 \in \mathcal{H}_{\kappa_1}^M$. By lemma 18 there are $\alpha_1, \dots, \alpha_m$ so that for the vector $\mathbf{w} = \sum_{i=1}^m \alpha_i \Psi_1(\mathbf{x}_i)$ we have

$$\frac{1}{m} \sum_{i=1}^m |\langle \mathbf{w}, \Psi_1(\mathbf{x}_i) \rangle - f_1(\mathbf{x}_i)| \leq \epsilon_1, \quad \|\mathbf{w}\| \leq M, \quad (16)$$

and

$$\sum_{i=1}^m |\alpha_i| \leq \frac{M^2}{\epsilon_1}. \quad (17)$$

Consider the function $f_2 \in \mathcal{H}_2$ defined by $f_2(\mathbf{x}) = \sum_{i=1}^m \alpha_i \langle \Psi_2(\mathbf{x}_i), \Psi_2(\mathbf{x}) \rangle$. We note that

$$\begin{aligned} \|f_2\|_{\mathcal{H}_{\kappa_2}}^2 &\leq \left\| \sum_{i=1}^m \alpha_i \Psi_2(\mathbf{x}_i) \right\|^2 \\ &= \sum_{i,j=1}^m \alpha_i \alpha_j \kappa_2(\mathbf{x}_i, \mathbf{x}_j) \\ &\leq \sum_{i,j=1}^m \alpha_i \alpha_j \kappa_1(\mathbf{x}_i, \mathbf{x}_j) + \epsilon \sum_{i,j=1}^m |\alpha_i \alpha_j| \\ &= \|\mathbf{w}\|^2 + \epsilon \left(\sum_{i=1}^m |\alpha_i| \right)^2 \\ &\leq M^2 + \epsilon \frac{M^4}{\epsilon_1^2} = 2M^2. \end{aligned}$$

Denote by $\tilde{f}_1(\mathbf{x}) = \langle \mathbf{w}, \Psi_1(\mathbf{x}) \rangle$ and note that for every $i \in [m]$ we have,

$$\begin{aligned} |\tilde{f}_1(\mathbf{x}_i) - f_2(\mathbf{x}_i)| &= \left| \sum_{j=1}^m \alpha_j (\kappa_1(\mathbf{x}_i, \mathbf{x}_j) - \kappa_2(\mathbf{x}_i, \mathbf{x}_j)) \right| \\ &\leq \epsilon \sum_{i=1}^m |\alpha_i| \leq \epsilon \frac{M^2}{\epsilon_1} = \epsilon_1. \end{aligned}$$

Finally, we get that,

$$\begin{aligned}
\mathcal{L}_{\mathcal{D}}(f_2) &\leq \mathcal{L}_{\mathcal{S}}(f_2) + \epsilon_1 \\
&= \frac{1}{m} \sum_{i=1}^m \ell(f_2(\mathbf{x}_i), y_i) + \epsilon_1 \\
&\leq \frac{1}{m} \sum_{i=1}^m \ell(\tilde{f}_1(\mathbf{x}_i), y_i) + \epsilon_1 + \epsilon_1 \\
&\leq \frac{1}{m} \sum_{i=1}^m \ell(f_1(\mathbf{x}_i), y_i) + |\tilde{f}_1(\mathbf{x}_i) - f_1(\mathbf{x}_i)| + 2\epsilon_1 \\
&\leq \frac{1}{m} \sum_{i=1}^m \ell(f_1(\mathbf{x}_i), y_i) + 3\epsilon_1 \\
&\leq \mathcal{L}_{\mathcal{S}}(f_1) + 3\epsilon_1 \leq \mathcal{L}_{\mathcal{D}}(f_1) + 4\epsilon_1,
\end{aligned}$$

which concludes the proof. \square

10 Discussion

Role of initialization and training. Our results surface the question of the extent to which random initialization accounts for the success of neural networks. While we mostly leave this question for future research, we would like to point to empirical evidence supporting the important role of initialization. First, numerous researchers and practitioners demonstrated that random initialization, similar to the scheme we analyze, is crucial to the success of neural network learning (see for instance [20]). This suggests that starting from arbitrary weights is unlikely to lead to a good solution. Second, several studies show that the contribution of optimizing the representation layers is relatively small [49, 26, 44, 43, 15]. For example, competitive accuracy on CIFAR-10, STL-10, MNIST and MONO datasets can be achieved by optimizing merely the last layer [36, 49]. Furthermore, Saxe et al. [49] show that the performance of training the last layer is quite correlated with training the entire network. The effectiveness of optimizing solely the last layer is also manifested by the popularity of the random features paradigm [46]. Finally, other studies show that the metrics induced by the initial and fully trained representations are not substantially different. Indeed, Giryes et al. [19] demonstrated that for the MNIST and CIFAR-10 datasets the distances' histogram of different examples barely changes when moving from the initial to the trained representation. For the ImageNet dataset the difference is more pronounced yet still moderate.

The role of architecture. By using skeletons and compositional kernel spaces, we can reason about functions that the network can actually learn rather than merely express. This may explain in retrospect past architectural choices and potentially guide future choices. Let us consider for example the task of object recognition. It appears intuitive, and is supported by visual processing mechanisms in mammals, that in order to perform object recognition,

the first processing stages are confined to local receptive fields. Then, the result of the local computations are applied to detect more complex shapes which are further combined towards a prediction. This processing scheme is naturally expressed by convolutional skeletons. A two dimensional version of Example 1 demonstrates the usefulness of convolutional networks for vision and speech applications.

The rationale we described above was pioneered by LeCun and colleagues [32]. Alas, the mere fact that a network can express desired functions does not guarantee that it can actually learn them. Using for example Barron’s theorem [7], one may claim that vision-related functions are expressed by fully connected two layer networks, but such networks are inferior to convolutional networks in machine vision applications. Our result mitigates this gap. First, it enables use of the original intuition behind convolutional networks in order to design function spaces that are provably learnable. Second, as detailed in Example 1, it also explains why convolutional networks perform better than fully connected networks.

The role of other architectural choices. In addition to the general topology of the network, our theory can be useful for understanding and guiding other architectural choices. We give two examples. First, suppose that a skeleton \mathcal{S} has a fully connected layer with the dual activation $\hat{\sigma}_1$, followed by an additional fully connected layer with dual activation $\hat{\sigma}_2$. It is straightforward to verify that if these two layers are replaced by a single layer with dual activation $\hat{\sigma}_2 \circ \hat{\sigma}_1$, the corresponding compositional kernel space remains the same. This simple observation can be useful in potentially saving a whole layer in the corresponding networks.

The second example is concerned with the ReLU activation, which is one of the most common activations used in practice. Our theory suggests a somewhat surprising explanation for its usefulness. First, the dual kernel of the ReLU activation enables expression of non-linear functions. However, this property holds true for many activations. Second, Theorem 3 shows that even for quite deep networks with ReLU activations, random initialization approximates the corresponding kernel. While we lack a proof at the time of writing, we conjecture that this property holds true for many other activations. What is then so special about the ReLU? Well, an additional property of the ReLU is being *positive homogeneous*, i.e. satisfying $\sigma(ax) = a\sigma(x)$ for all $a \geq 0$. This fact makes the ReLU activation robust to small perturbations in the distribution used for initialization. Concretely, if we multiply the variance of the random weights by a constant, the distribution of the generated representation and the space $\mathcal{H}_{\mathbf{w}}$ remain the same up to a scaling. Note moreover that training algorithms are sensitive to the initialization. Our initialization is very similar to approaches used in practice, but encompasses a small “correction”, in the form of a multiplication by a small constant which depends on the activation. For most activations, ignoring this correction, especially in deep networks, results in a large change in the generated representation. The ReLU activation is more robust to such changes. We note that similar reasoning applies to the max-pooling operation.

Future work. Though our formalism is fairly general, we mostly analyzed fully connected and convolutional layers. Intriguing questions remain, such as the analysis of max-pooling and recursive neural network components from the dual perspective. On the algorithmic side, it is yet to be seen whether our framework can help in understanding procedures such as dropout [54] and batch-normalization [25]. Beside studying existing elements of neural network learning, it would be interesting to devise new architectural components inspired by duality. More concrete questions are concerned with quantitative improvements of the main results. In particular, it remains open whether the dependence on $2^{O(\text{depth}(\mathcal{S}))}$ can be made polynomial and the quartic dependence on $1/\epsilon$, R , and L can be improved. In addition to being interesting in their own right, improving the bounds may further underscore the effectiveness of random initialization as a way of generating low dimensional embeddings of compositional kernel spaces. Randomly generating such embeddings can be also considered on its own, and we are currently working on design and analysis of random features a la Rahimi and Recht [45].

Acknowledgments

We would like to thank Yossi Arjevani, Elad Eban, Moritz Hardt, Elad Hazan, Percy Liang, Nati Linial, Ben Recht, and Shai Shalev-Shwartz for fruitful discussions, comments, and suggestions.

References

- [1] A. Andoni, R. Panigrahy, G. Valiant, and L. Zhang. Learning polynomials with neural networks. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1908–1916, 2014.
- [2] F. Anselmi, L. Rosasco, C. Tan, and T. Poggio. Deep convolutional networks are hierarchical kernel machines. *arXiv:1508.01084*, 2015.
- [3] M. Anthony and P. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.
- [4] S. Arora, A. Bhaskara, R. Ge, and T. Ma. Provable bounds for learning some deep representations. In *Proceedings of The 31st International Conference on Machine Learning*, pages 584–592, 2014.
- [5] F. Bach. Breaking the curse of dimensionality with convex neural networks. *arXiv:1412.8690*, 2014.
- [6] F. Bach. On the equivalence between kernel quadrature rules and random feature expansions. 2015.
- [7] A.R. Barron. Universal approximation bounds for superposition of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

- [8] P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.
- [9] P.L. Bartlett. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE Transactions on Information Theory*, 44(2):525–536, March 1998.
- [10] E.B. Baum and D. Haussler. What size net gives valid generalization? *Neural Computation*, 1(1):151–160, 1989.
- [11] L. Bo, K. Lai, X. Ren, and D. Fox. Object recognition with hierarchical kernel descriptors. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1729–1736. IEEE, 2011.
- [12] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, 2013.
- [13] Y. Cho and L.K. Saul. Kernel methods for deep learning. In *Advances in neural information processing systems*, pages 342–350, 2009.
- [14] A. Choromanska, M. Henaff, M. Mathieu, G. Ben Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *AISTATS*, pages 192–204, 2015.
- [15] D. Cox and N. Pinto. Beyond simple features: A large-scale feature search approach to unconstrained face recognition. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 8–15. IEEE, 2011.
- [16] A. Daniely. Complexity theoretic limitations on learning halfspaces. In *STOC*, 2016.
- [17] A. Daniely and S. Shalev-Shwartz. Complexity theoretic limitations on learning DNFs. In *COLT*, 2016.
- [18] A. Daniely, N. Linial, and S. Shalev-Shwartz. From average case complexity to improper learning complexity. In *STOC*, 2014.
- [19] R. Giryes, G. Sapiro, and A.M. Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy? *arXiv preprint arXiv:1504.08291*, 2015.
- [20] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256, 2010.
- [21] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Tenth IEEE International Conference on Computer Vision*, volume 2, pages 1458–1465, 2005.
- [22] M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. *arXiv:1509.01240*, 2015.

- [23] Z.S. Harris. Distributional structure. *Word*, 1954.
- [24] T. Hazan and T. Jaakkola. Steps toward deep kernel methods from infinite neural networks. *arXiv:1508.05133*, 2015.
- [25] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167*, 2015.
- [26] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2146–2153. IEEE, 2009.
- [27] P. Kar and H. Karnick. Random feature maps for dot product kernels. *arXiv:1201.6530*, 2012.
- [28] R.M. Karp and R.J. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the twelfth annual ACM symposium on Theory of computing*, pages 302–309. ACM, 1980.
- [29] M. Kearns and L.G. Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. In *STOC*, pages 433–444, May 1989.
- [30] A.R. Klivans and A.A. Sherstov. Cryptographic hardness for learning intersections of halfspaces. In *FOCS*, 2006.
- [31] A. Krizhevsky, I. Sutskever, and G.E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [33] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- [34] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, pages 2177–2185, 2014.
- [35] R. Livni, S. Shalev-Shwartz, and O. Shamir. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, pages 855–863, 2014.
- [36] J. Mairal, P. Koniusz, Z. Harchaoui, and Cordelia Schmid. Convolutional kernel networks. In *Advances in Neural Information Processing Systems*, pages 2627–2635, 2014.
- [37] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

- [38] R.M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.
- [39] B. Neyshabur, R. R Salakhutdinov, and N. Srebro. Path-SGD: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2413–2421, 2015.
- [40] B. Neyshabur, N. Srebro, and R. Tomioka. Norm-based capacity control in neural networks. In *COLT*, 2015.
- [41] R. O’Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.
- [42] J. Pennington, F. Yu, and S. Kumar. Spherical random features for polynomial kernels. In *Advances in Neural Information Processing Systems*, pages 1837–1845, 2015.
- [43] N. Pinto and D. Cox. An evaluation of the invariance properties of a biologically-inspired system for unconstrained face recognition. In *Bio-Inspired Models of Network, Information, and Computing Systems*, pages 505–518. Springer, 2012.
- [44] N. Pinto, D. Doukhan, J.J. DiCarlo, and D.D. Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS Computational Biology*, 5(11):e1000579, 2009.
- [45] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, pages 1177–1184, 2007.
- [46] A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, pages 1313–1320, 2009.
- [47] I. Safran and O. Shamir. On the quality of the initial basin in overspecified neural networks. *arxiv:1511.04210*, 2015.
- [48] S. Saitoh. *Theory of reproducing kernels and its applications*. Longman Scientific & Technical England, 1988.
- [49] A. Saxe, P.W. Koh, Z. Chen, M. Bhand, B. Suresh, and A.Y. Ng. On random weights and unsupervised feature learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1089–1096, 2011.
- [50] I.J. Schoenberg et al. Positive definite functions on spheres. *Duke Mathematical Journal*, 9(1):96–108, 1942.
- [51] B. Schölkopf, P. Simard, A. Smola, and V. Vapnik. Prior knowledge in support vector kernels. In *Advances in Neural Information Processing Systems 10*, pages 640–646. MIT Press, 1998.

- [52] H. Sedghi and A. Anandkumar. Provable methods for training neural networks with sparse connectivity. *arXiv:1412.2693*, 2014.
- [53] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.
- [54] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
- [55] I. Sutskever, O. Vinyals, and Q.V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.
- [56] C.K.I. Williams. Computation with infinite neural networks. pages 295–301, 1997.