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

Amit Daniely Roy Frostig Yoram Singer

Google Research

NIPS, 2017

Presenter: Chao Jiang

Outline

Outline

Overview

- They define an object termed a computation skeleton that describes a distilled structure of feed-forward networks.
- They show that the representation generated by random initialization is sufficiently rich to approximately express the functions in $\mathcal{H}$
- all functions in $\mathcal{H}$ can be approximated by tuning the weights of the last layer, which is a convex optimization task.

Outline

Part I: Function Basis

- In $\mathbb{R}^n$ space, we can use n independent vectors to represent any vector by linear combination. The n independent vectors can be viewed as a set of basis.
- if $x = (x_1, x_2, \cdots, x_n)$, $y = (y_1, y_2, \cdots, y_n)$, we can get

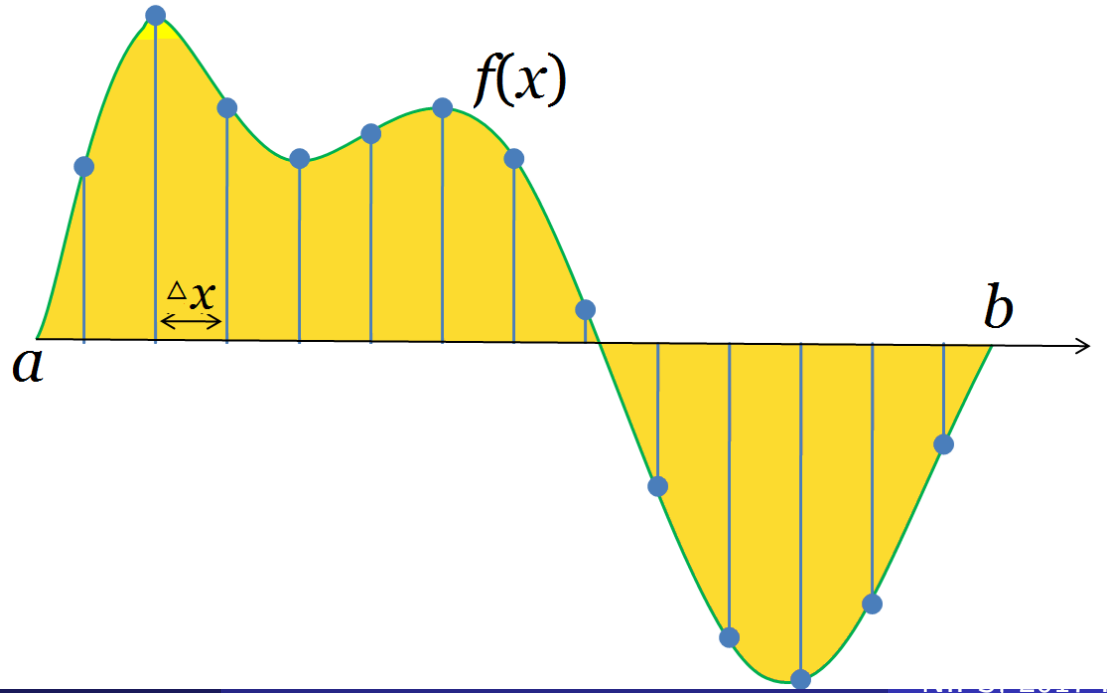
$$< x, y >= \sum_{i=1}^{n} x_i y_i$$

- Until now, this is the review of vector basis. These knowledge can be extended to functions and function space.

Function Basis

A function is an infinite vector.

- For a function defined on the interval $[a, b]$, we take samples by an interval Δx.
- If we sample the function $f(x)$ at points $a, x_1 \cdots, x_n, b$, then we can transform the function into a vector $(f(a), f(x_1), \cdot, f(x_n), f(b))^T$.
- When $\Delta x \to 0$, the vector should be more and more close to the function and at last, it becomes infinite.



Inner Product of Functions Similarly

- Since functions are so close to vectors, we can also define the inner product of functions similarly.
- For two functions f and g sampling by interval Δx, the inner product could be defined as:

$$< f, g >= \lim_{\Delta x \to 0} \sum_i f(x_i) g(x_i) \Delta x = \int f(x) g(x) dx$$

Inner Product of Functions Similarly

- The expression of function inner product is seen everywhere. It has various meanings in various context.
- For example, if X is a continuous random variable with probability density function $f(x)$, i.e., $f(x) > 0$ and $\int f(x)dx = 1$, then the expectation

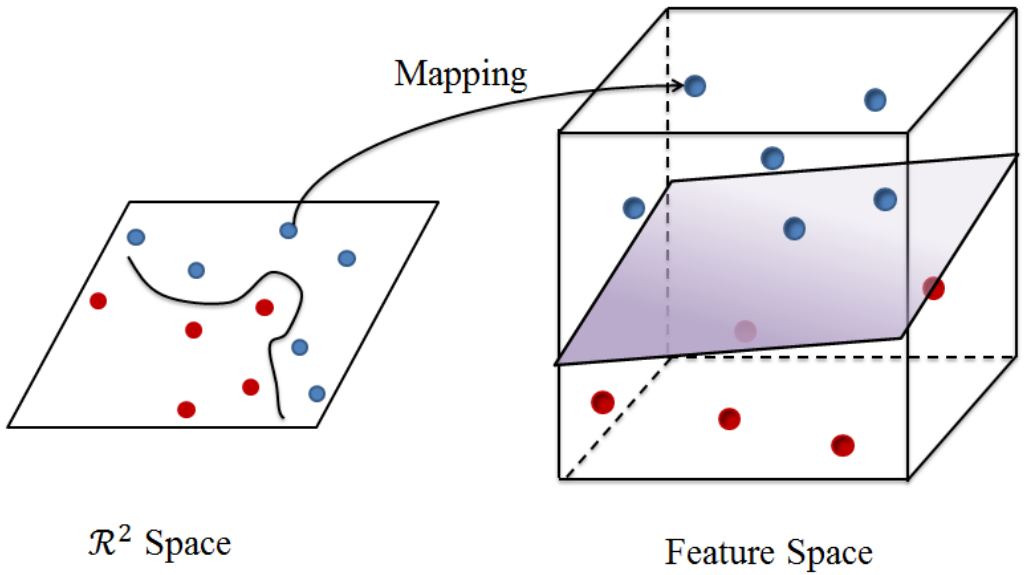
$$E[g(x)] = \int f(x)g(x)dx =< f, g >$$

Inner Product of Functions Similarly

- Similar to vector basis, we can use a set of functions to represent other functions.
- The difference is that in a vector space, we only need **finite** vectors to construct a complete basis set, but in function space, we may need **infinite** basis functions.
- Two functions can be regarded as orthogonal if their inner product is zero.
- In function space, we can also have a set of function basis that are mutually orthogonal.

Inner Product of Functions Similarly

- Kernel methods have been widely used in a variety of data analysis techniques.
- The motivation of kernel method arises in mapping a vector in $\mathbb{R}^n$ space as another vector in a feature space.



Eigen Decomposition

- For a real symmetric matrix A , there exists real number λ and vector q so that

$$Aq = \lambda q$$

- For $A \in \mathbb{R}^{n \times n}$, we can find n eigenvalues (λ_i) along with n orthogonal eigenvectors (q_i). As a result, A can be decomposited as

$$\begin{aligned}
\mathbf{A} = \mathbf{Q}\mathbf{D}\mathbf{Q}^T &= (\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_n) \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix} \begin{pmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_n^T \end{pmatrix} \\
&= (\lambda_1 \mathbf{q}_1, \lambda_2 \mathbf{q}_2, \cdots, \lambda_n \mathbf{q}_n) \begin{pmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_n^T \end{pmatrix} \\
&= \sum_{i=1}^{n} \lambda_i \mathbf{q}_i \mathbf{q}_i^T
\end{aligned}$$

- Here $\{\mathbf{q}_i\}_{i=1}^n$ is a set of orthogonal basis of $\mathbb{R}^n$.

Eigen Decomposition

- A matrix is a description of the transformation in a linear space.
- The transformation direction is eigenvector.
- The transformation scale is eigenvalue.

Kernel Function

- A function $f(x)$ can be viewed as an infinite vector, then for a function with two independent variables $K(x, y)$, we can view it as an infinite matrix.
- If $K(x, y) = K(y, x)$ and

$$\int \int f(x)K(x, y)f(y)dxdy \leq 0$$

for any function f, then $K(x, y)$ is symmetric and positive definite, then $K(x, y)$ is a kernel function.

- Similar to matrix eigenvalue and eigenvector, there exists eigenvalue λ and eigenfunction $\psi(x)$, so that

$$\int K(x, y)\psi(x)dx = \lambda\psi(y)$$

Kernel Function

$$K(x, y) = \sum_{i=0}^{\infty} \lambda_i \psi_i(x) \psi_i(y) \tag{1}$$

- Here, $< \psi_i, \psi_j >= 0 for i \neq j$
- Therefore, $\{\psi_i\}_{i=1}^{\infty}$ construct a set of orthogonal basis for a function space.

Reproducing Kernel Hilbert Space

- $\{\sqrt{\lambda_i}\psi_i\}_{i=1}^{\infty}$ is a set of orthogonal basis and construct a Hilbert space $\mathcal{H}$
- Any function or vector in the space can be represented as the linear combination of the basis. Suppose

$$f = \sum_{i=1}^{\infty} f_i \sqrt{\lambda_i}\psi_i$$

we can denote f as an infinite vector in $\mathcal{H}$:

$$f = (f_1, f_2, \cdots)_{\mathcal{H}}^T$$

for another function

$$g = (g_1, g_2, \cdots)_{\mathcal{H}}^T$$

we have

$$< f, g >_{\mathcal{H}} = \sum_{i=1}^{\infty} f_i g_i$$

Reproducing Kernel Hilbert Space

- We use $K(x, y)$ to denote the number of K at point x, y, use $K(\cdot, \cdot)$ to denote the function itself, and use $K(x, \cdot)$ to denote the x-th "row" of the matrix.

$$K(x, \cdot) = \sum_{i=0}^{\infty} \lambda_i \psi_i(x) \psi_i$$

In space $\mathcal{H}$, we can denote

$$K(x, \cdot) = (\sqrt{\lambda_1}\psi_1(x), \sqrt{\lambda_2}\psi_2(x), \cdots)^T_{\mathcal{H}}$$

Therefore,

$$< K(x, \cdot, K(y, \cdot)) >_{\mathcal{H}} = \sum_{i=0}^{\infty} \lambda_i \psi_i(x) \psi_i(y) = K(x, y)$$

- This is the reproducing property, thus $\mathcal{H}$ is called reproducing kernel Hilbert space (RKHS).

Go Back: How to Map a Point into A Feature Space

- we define a mapping:

$$\Phi(x) = K(x, \cdot) = (\sqrt{\lambda_1}\psi_1(x), \sqrt{\lambda_2}\psi_2(x), \cdots)^T_{\mathcal{H}}$$

- then we can map the point x to $\mathcal{H}$.

$$< \Phi(x), \Phi(y) >_{\mathcal{H}} = < K(x, \cdot), K(y, \cdot) >_{\mathcal{H}} = K(x, y)$$

- As a result, we do not need to actually know what is the mapping, where is the feature space, or what is the basis of the feature space.
- Kernel trick: for a symmetric positive-definite function K, there must exist at least one mapping Φ and one feature space $\mathcal{H}$, so that

$$< \Phi(x), \Phi(y) >_{\mathcal{H}} = K(x, y)$$

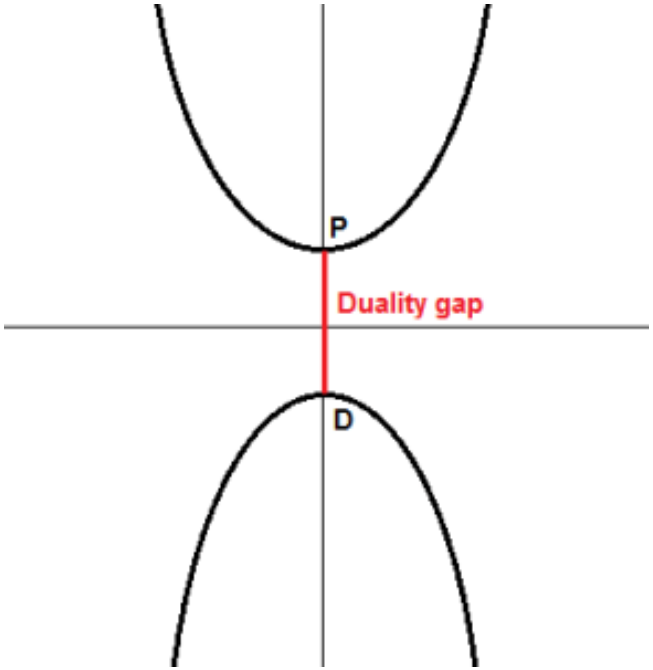
Outline

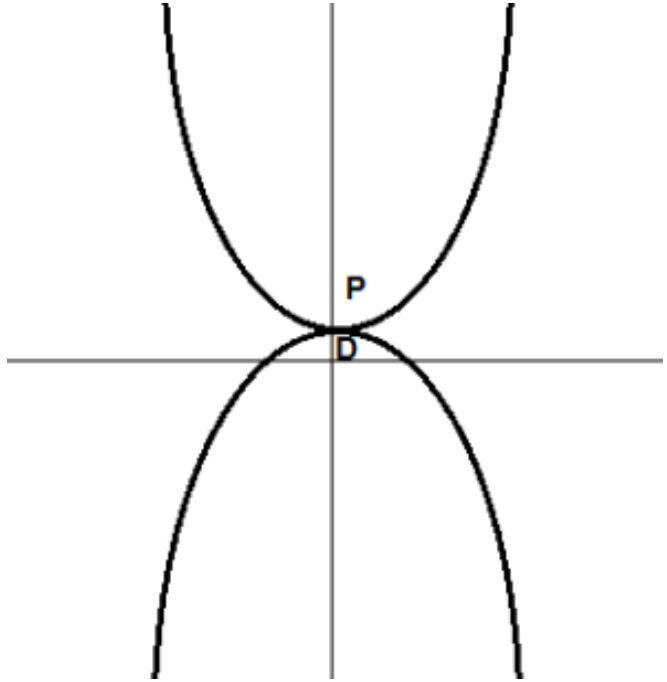
Duality

- In mathematical optimization theory, duality means that optimization problems may be viewed from either of two perspectives, the primal problem or the dual problem (the duality principle). The solution to the dual problem provides a lower bound to the solution of the primal (minimization) problem.
- If you have a minimization problem, you can also see it as a maximization problem. And when you find the maximum of this problem, it will be a lower bound to the solution of the minimization problem

Duality

We want to minimize the function at the top of the graph. Its minimum is P, D is the maximum for its dual problem.





- $P - D$ is duality gap. if $P - D > 0$, we say weak duality holds.
- $P - D = 0$, there is no duality gap, and we say that strong duality holds.

Notation

- G = (V, E) is a directed acyclic graph
- The set of neighbors incoming to a vertex v is denoted $in(v) := u \in V | uv \in E$
- The d - 1 dimensional sphere is denoted $\mathbb{S}^{d-1} = \{x \in \mathbb{R}^d | \|x\| = 1\}$
- The ball of radius B in Hilbert space $\mathcal{H}$: $\{x \in \mathcal{H}^d | \|x\|_{\mathcal{H}} \leq B\}$
- input: $x = (x^1, \cdots, x^n), where x^i \in S^{d-1}$
- A network $\mathcal{N}$ with a weight kernel vector $w = \{W_{uv} | uv \in E\}$ defines a predictor $h_{\mathcal{N},w} : \mathcal{X} \to \mathbb{R}^k$
- For an input node v, its output is $h_{v,\mathbf{w}}(x) = \theta_v(\sum_{u \in in(v)} w_{uv} h_{u,\mathbf{w}}(X))$
- The representation induced by the weight w is $\mathbb{R}_{\mathcal{N},w} = h_{rep(\mathcal{N}),\mathbf{w}}$

Computation Skeleton

Definition 1

Definition 1. A computation skeleton $\mathcal{S}$ is directed acyclic graph (DAG) whose non-input nodes are labeled by activations.

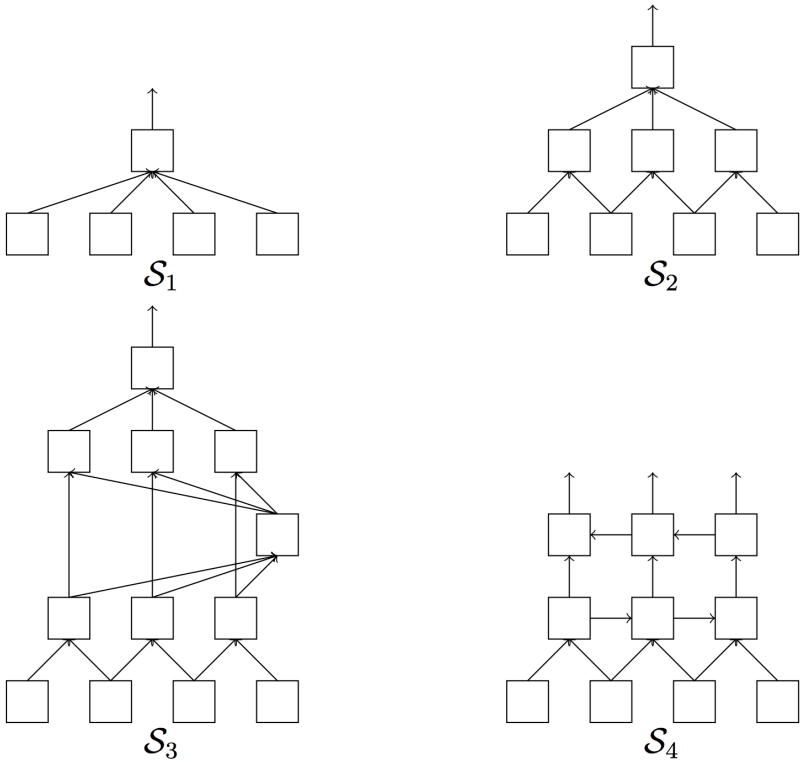


Figure 1: Examples of computation skeletons.

Fully connected layer of a skeleton

Terminology

An induced subgraph of a skeleton with $r+1$ nodes, $u_1, \cdots, u_r, v$ is called a fully connected layer if ite edges are $u_1 v, \cdots u_r v$

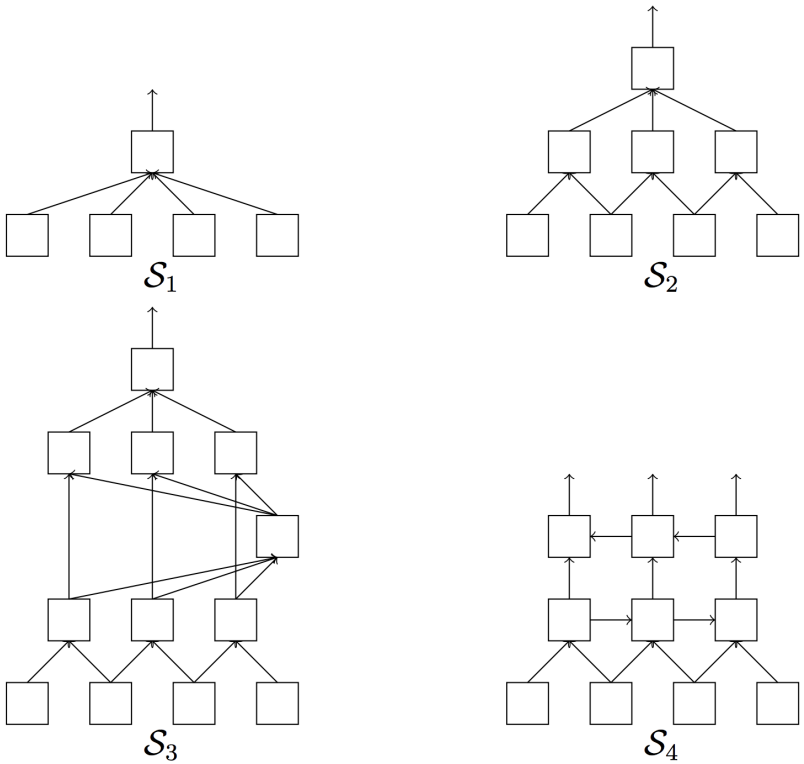


Figure 1: Examples of computation skeletons.

Convolution layer of a skeleton

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, $u_1, \ldots, u_n, v_1, \ldots, v_q$, and qw edges, $u_{s(i-1)+j}\, v_i$, for $1 \le i \le q, 1 \le j \le w$.*

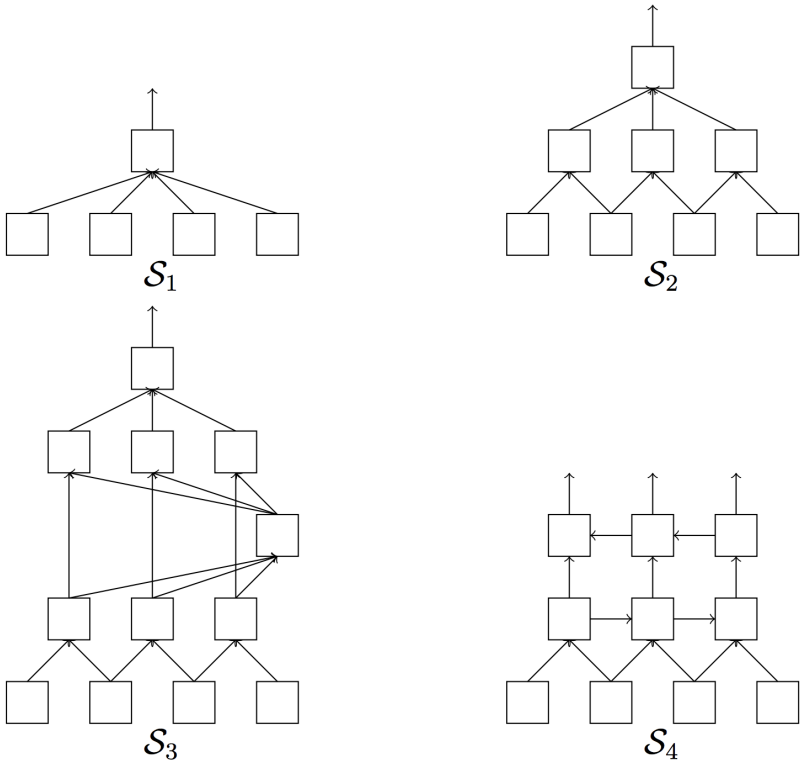


Figure 1: Examples of computation skeletons.

Realization of a skeleton

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 $v^1, \ldots, v^r$, each with an activation σ and weight $1/r$. In addition, $\mathcal{N}$ has k output neurons $o_1, \ldots, o_k$ 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 $o_1, \ldots, o_k$. We term $\mathcal{N}$ the (r, k)-*fold realization *of $\mathcal{S}$. We also define the* r-fold realization *of $\mathcal{S}$ as*[1] $\mathcal{N}(\mathcal{S}, r) = \mathrm{rep}\left(\mathcal{N}(\mathcal{S}, r, 1)\right)$.

Realization of a skeleton

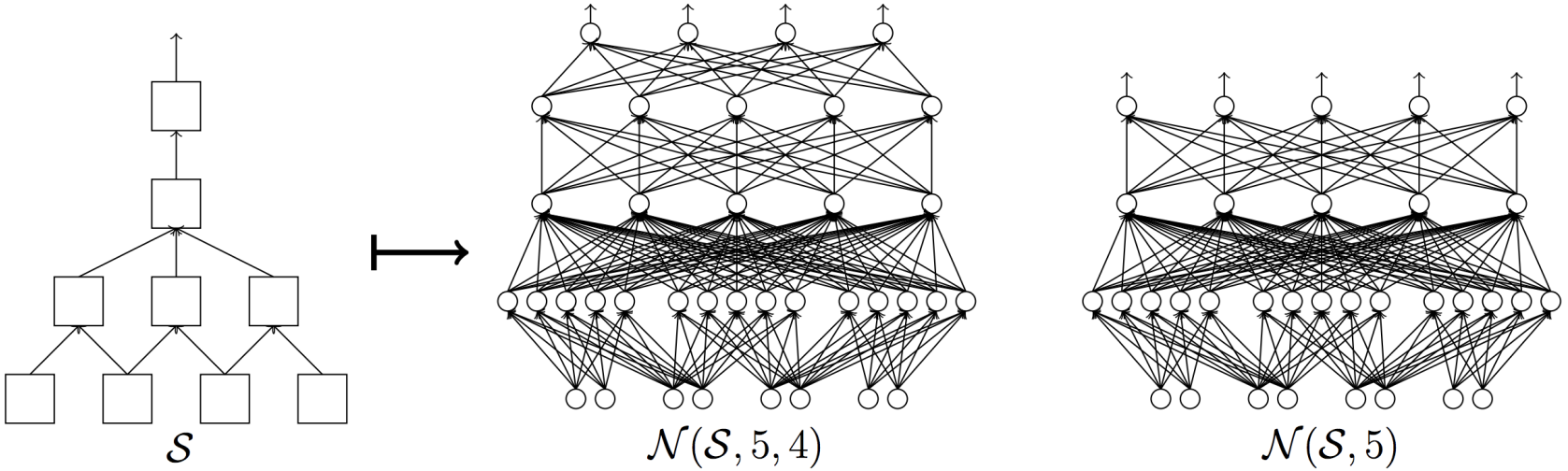


Figure 2: A $(5, 4)$-fold and 5-fold realizations of the computation skeleton $\mathcal{S}$ with $d = 2$.

- 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.

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 variance*[2] $d\delta(u)/\delta(\text{in}(v))$ *if* u *is an input neuron and* $\delta(u)/(\|\sigma_u\|^2 \, \delta(\text{in}(v)))$ *otherwise.*

Dual activation kernel

- A computation skeleton $\mathcal{S}$ also defines a normalized kernel $k_{\mathcal{S}} : \mathcal{X} \times \mathcal{X} \to [-1, 1]$ and a corresponding norm $\|\cdot\|_{\mathcal{S}}$ on functions $f : \mathcal{S} \to \mathcal{R}$
- This norm has the property that $\|f\|_{\mathcal{S}}$ is small iff 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 $\mathcal{R}^2$ with mean 0 and covariance matrix $\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$

Dual activation kernel

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

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

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

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

Dual activation kernel

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 $\kappa_v : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ as follows. For an input node v corresponding to the ith 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.

Dual activation kernel

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

Dual activation kernel

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*

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

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

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

Dual activation kernel

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*

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

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

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