Neural Comput & Applic (2018) 29:305–315
https://doi.org/10.1007/s00521-017-2965-0

EANN 2016

Constructive lower bounds on model complexity of shallow perceptron networks

Věra Kůrková[1]

Received: 10 November 2016 / Accepted: 23 March 2017 / Published online: 26 April 2017

Abstract Limitations of shallow (one-hidden-layer) perceptron networks are investigated with respect to computing multivariable functions on finite domains. Lower bounds are derived on growth of the number of network units or sizes of output weights in terms of variations of functions to be computed. A concrete construction is presented with a class of functions which cannot be computed by signum or Heaviside perceptron networks with considerably smaller numbers of units and smaller output weights than the sizes of the function's domains. A subclass of these functions is described whose elements can be computed by two-hidden-layer perceptron networks with the number of units depending on logarithm of the size of the domain linearly.

Keywords Shallow and deep networks · Model complexity and sparsity · Signum perceptron networks · Finite mappings · Variational norms · Hadamard matrices

1 Introduction

Originally, biologically inspired neural networks were introduced as multilayer computational models, but later, one-hidden-layer architectures became dominant in applications (see, e.g., [12, 20] and the references therein). Training of networks with several hidden layers had been inefficient until the advent of fast graphic processing units [32]. These networks were called *deep* [4, 14] to distinguish them from *shallow* networks with merely one hidden layer. Currently, deep networks with several convolutional and pooling layers are the state of the art in computer vision and speech recognition tasks (see the survey article [32] and the references therein).

However, reservations about overall superiority of deep networks over shallow ones have appeared. An empirical study demonstrated that shallow networks can learn some functions previously learned by deep ones using the same numbers of parameters as the original deep networks [1]. On the other hand, it was conjectured: "most functions that can be represented compactly by deep architectures cannot be represented by a compact shallow architecture" [6].

Theoretical analysis complementing the experimental evidence, obtained by some comparisons of deep and shallow networks solving the same tasks, is still in its early stages. Bianchini and Scarselli [7] bounded numbers of units in shallow and deep networks in terms of topological properties of input-output functions. Mhaskar et al. [36] suggested that due to their hierarchical structure, deep networks could outperform shallow networks in visual recognition of pictures with objects of different scales.

Characterization of functions, which can be computed by deep networks of smaller model complexities than shallow ones, can be derived by comparing lower bounds on numbers of units in shallow networks with upper bounds on numbers of units in deep ones. Generally, derivation of lower bounds is much more difficult than derivation of upper ones. For shallow networks, various upper bounds on numbers of units in dependence on types of units, input dimensions, and types of functions to be computed are known (see, e.g., [16] and the references therein), but only a few lower bounds are available. A method of deriving lower bounds exploiting continuous selection of best approximation [37] cannot be used for most types of common neural

✉ Věra Kůrková
vera@cs.cas.cz

[1] Institute of Computer Science, Czech Academy of Sciences, Pod Vodárenskou věží 2, 182 07 Prague, Czech Republic

networks [17, 18]. Other lower bounds hold merely for types of computational units that are not commonly used such as perceptrons with specially designed activation functions [34] or guarantee asymptotically existence of worst-case errors in Sobolev spaces [35].

In practical applications, feedforward networks compute functions on finite domains (such as pixels of photographs or digitized high-dimensional cubes). It is well-known that shallow networks with many common types of units can represent any finite mapping provided that the number of units is potentially as large as the size of the domain. For large domains, such networks might be too large for efficient implementations. Thus, it is desirable to investigate which functions allow sparse representations by shallow networks.

An example of a class of functions which cannot be sparsely represented by shallow networks with Gaussian SVM units was given in [5] where it was proved that a shallow network with less than 2^{d-1} Gaussian support vectors cannot classify correctly the elements of the set $\{0,1\}^d$ of points of a d-dimensional Boolean cube according to the parity of the number of 1's. It was suggested that a cause of large model complexities of shallow networks might be the "amount of variations" of functions to be computed.

Following up on this idea, we showed in [30] that the effect of "high variations" of a function depends on a type of computational units. We proposed to use a concept of variational norm tailored to the type of computational units as a measure of variations of a function influencing model complexity of networks with units of the given type. Using a probabilistic argument, we proved that almost any uniformly randomly chosen function on a sufficiently large domain is highly varying with respect to Heaviside or signum perceptrons, and thus, it cannot be represented by perceptron networks with a reasonably small number of units and sizes of output weights. However, our argument proving existence of large sets of functions whose implementations by shallow perceptron networks require large numbers of units or large sizes of output weights is not constructive [30] and is based on the probabilistic Chernoff bound related to the law of large numbers.

In this paper, we supplement the existential arguments presented in [30] with constructive ones. A concrete class of functions is described which cannot be sparsely represented by shallow signum and Heaviside perceptrons. As minimization of the number of nonzero output weights is a difficult nonconvex problem, we focus on a related concept of sparsity used in weight-decay regularization techniques minimizing l_1-norms of output weights [12].

Variational norms enable an estimate of the rate at which approximation accuracy increases as more network units are added. We show that for finite dictionaries of computational units, l_1-sparsity is related to variational norm. Geometrical properties of variational norm imply that functions which are nearly orthogonal to any function computable by a signum perceptron cannot be computed by perceptron networks with reasonably small numbers of units and sizes of output weights. It is shown that Hadamard matrices, due to their quasi-random distribution of $+1$'s and -1's entries, generate functions which are not correlated with any signum of Heaviside perceptron. We prove that functions on finite domains in $\mathbb{R}^d$ in the form of $n \times n$ rectangles generated by Hadamard matrices cannot be computed by shallow Heaviside or signum perceptron networks having both number of units and sizes of output weights smaller than $\frac{\sqrt{n}}{\lceil \log_2 n \rceil}$. In particular, for domains of sizes $2^k \times 2^k$, such functions cannot be computed by shallow perceptron networks with numbers or units or sizes of output weights depending on k polynomially.

Many concrete examples of functions to which our lower bounds apply can be obtained from various constructions of Hadamard matrices. Their listings are available at Neil Sloane's Library of Hadamard matrices [38]. The oldest known type of recursive construction of Hadamard matrices due to Sylvester is called Sylvester-Hadamard matrices. We show that functions induced by $2^k \times 2^k$ Sylvester-Hadamard matrices have a compositional structure, and thus, they can be represented by two-hidden-layer networks with k Heaviside perceptrons in each hidden layer. On the other hand, our results imply that they cannot be represented by shallow perceptron networks with both number of units and output weights smaller than $\frac{2^k}{k}$. A preliminary version of some results appeared in a conference proceedings [24] where main theorems of this paper with sketches of proofs were stated.

The paper is organized as follows. Section 2 contains basic concepts on shallow networks and dictionaries of computational units. Section 3 introduces variational norms as tools for investigation of network complexity. Section 4 presents existential probabilistic results on functions which cannot be computed by shallow signum perceptron networks with "small" numbers of units and sizes of output weights. In Section 5, construction of concrete classes of such functions is presented. In Section 6, there is described a class of functions which can be computed by two-hidden-layer networks with much smaller number of units than by networks with one hidden layer. Section 7 is a brief discussion. To make our exposition easier to follow, in the main sections, we explain basic ideas of our arguments intuitively, while detailed rigorous mathematical proofs are presented in the Appendix.

2 Preliminaries

The most widespread type of a feedforward neural network architecture is a *one-hidden-layer network with a single*

linear output. Networks of this type compute input-output functions belonging to sets of the form

$$\text{span}_n\ G := \left\{ \sum_{i=1}^{n} w_i g_i | w_i \in \mathbb{R},\ g_i \in G \right\},$$

where the coefficients w_i are called *output weights*, G is a set of functions computable by a given type of computational units called a *dictionary*, and n is the number of network units. Recently, one-hidden-layer networks became called *shallow networks* to distinguish them from *deep* ones with two or more hidden layers of computational units.

A common type of a computational unit is *perceptron*, which computes functions of the form

$$\sigma(v \cdot x + b) : X \to \mathbb{R},$$

where $v \in \mathbb{R}^d$ is called an *input weight*, $b \in \mathbb{R}$ a *bias*, and $\sigma : \mathbb{R} \to \mathbb{R}$ an *activation function*. It is called *sigmoid* when it is monotonic increasing and $\lim_{t\to-\infty} \sigma(t) = 0$ and $\lim_{t\to\infty} \sigma(t) = 1$. Important types of activation functions are the *Heaviside function* $\vartheta : \mathbb{R} \to \{0, 1\}$ defined as

$$\vartheta(t) := 0 \text{ for } t < 0 \quad \text{and} \quad \vartheta(t) := 1 \text{ for } t \geq 0$$

and the *signum function* $\text{sgn} : \mathbb{R} \to \{-1, 1\}$, defined as

$$\text{sgn}(t) := -1 \text{ for } t < 0 \quad \text{and} \quad \text{sgn}(t) := 1 \quad \text{for } t \geq 0.$$

We denote by $H_d(X)$ the dictionary of functions on $X \subset \mathbb{R}^d$ computable by *Heaviside perceptrons*, i.e.,

$$H_d(X) := \{\vartheta(v \cdot x + b) : X \to \{0, 1\} \,|\, v \in \mathbb{R}^d, b \in \mathbb{R}\},$$

and by $P_d(X)$ the dictionary of functions on X computable by *signum perceptrons*, i.e.,

$$P_d(X) := \{\text{sgn}(v \cdot x + b) : X \to \{-1, 1\} \,|\, v \in \mathbb{R}^d, b \in \mathbb{R}\}.$$

For a domain $X \subset \mathbb{R}^d$, we denote by

$$\mathcal{F}(X) := \{f \mid f : X \to \mathbb{R}\}$$

the *set of all real-valued functions on X* and by

$$\mathcal{B}(X) := \{f \mid f : X \to \{-1, 1\}\}$$

its *subset of functions on X with values in* $\{-1, 1\}$.

In practical applications, domains $X \subset \mathbb{R}^d$ of functions to be computed by neural networks are finite, but their sizes card X and/or input dimensions d can be quite large. It is easy to see that when card $X = m$ and $X = \{x_1, \ldots, x_m\}$ is a linear ordering of X, then the mapping $\iota : \mathcal{F}(X) \to \mathbb{R}^m$ defined as $\iota(f) := (f(x_1), \ldots, f(x_m))$ is an isomorphism. So on $\mathcal{F}(X)$, we have the Euclidean inner product defined as

$$\langle f, g \rangle := \sum_{u \in X} f(u) g(u)$$

and the Euclidean norm $\|f\| := \sqrt{\langle f, f \rangle}$. In contrast to the inner product $\langle ., . \rangle$ on $\mathcal{F}(X)$, we denote by $\cdot$ the inner product on $X \subset \mathbb{R}^d$, i.e., for $u, v \in X$,

$$u \cdot v := \sum_{i=1}^{d} u_i v_i.$$

3 Variational norms as measures of sparsity

Shallow networks with many types of computational units (including perceptrons and positive definite kernel units) can exactly compute any function on any finite domain. We call this capability the *universal representation property*. The following general condition proven by Ito [15] guarantees this property for a class of one-hidden-layer networks with units from a dictionary of general computational units in the form of a parameterized family

$$G_\phi(X) = \{\phi(x, y) : X \to \mathbb{R}\ \ y \in Y\},$$

where $\phi : X \times Y \to \mathbb{R}$ is a function of two variables: an input vector $x \in X$ and a parameter vector Y.

Proposition 1 *Let d, m be positive integers, $X = \{x_1, \ldots, x_m\} \subset \mathbb{R}^d$, and $G_\phi(X) = \{\phi(., y) : X \to \mathbb{R} \,|\, y \in Y\}$ be a dictionary of computational units. If there exist $y_1 \ldots, y_m \in Y$ such that the matrix M with entries $M_{i,j} = \phi(x_i, y_j)$ is non singular, then for every function $f : X \to \mathbb{R}$, there exist $w_1, \ldots, w_m$ such that $f(x) = \sum_{i=1}^{m} w_i \phi(x, y_i)$ for all $x \in X$.*

It is well known that many dictionaries of computational units, e.g., positive definite kernel units and sigmoidal perceptrons, satisfy the assumptions of Proposition 1 [15]. However, this proposition assumes that the number of hidden units is potentially equal to the size of the domain of functions to be computed. For large domains, such networks may be too large for efficient implementations. It is desirable that dictionaries of computational units are chosen in such a way that input-output functions representing optimal solutions (or reasonably suboptimal ones) of given tasks can be computed by sufficiently sparse networks.

Minimization of the number of nonzero output weights in a shallow network computing a given function f is a difficult non convex task. In some literature, the number of nonzero coefficients among w_i's in an input-output function

$$f = \sum_{i=1}^{n} w_i g_i \tag{1}$$

from $\text{span}_n G$ is called an "l_0-pseudo-norm" in quotation marks and denoted $\|w\|_0$. It can be defined as the Hamming distance of the vector $(\hat{w}_1, \ldots, \hat{w}_n)$ from zero, where $\hat{w}_i = 0$ when $w_i = 0$ and $\hat{w}_i = 1$ when $w_i \neq 0$. It is neither

a norm nor a pseudo-norm. The quantity $\|w\|_0$ is always an integer, and thus, $\|\cdot\|_0$ does not satisfy the homogeneity property of a norm ($\|\lambda x\| = |\lambda|\|x\|$ for all λ). Moreover, the "unit ball" $\{w \in \mathbb{R}^n \mid \|w\|_0 \leq 1\}$ is non convex and unbounded as it is equal to the union of all one-dimensional subspaces of span G.

Instead of l_0-minimization, minimizations of l_1 and l_2-norms of output weights vectors $w = (w_1, \ldots, w_n)$ have been used in weight-decay regularization techniques (see, e.g., [12, p. 220]). Note that l_1-norm minimization also plays an important role in compressed sensing [8, 9]. Thus, it is useful to consider an alternative concept of sparsity defined in terms of l_1-norm of output weight vector. Neither small "l_0-pseudo-norm" implies small l_1-norm nor small l_1-norm implies small "l_0-pseudo-norm" of an output weight vector. However, small l_1-norm guarantees that an input-output function of a network can be well approximated by input-output functions computable by networks with small "l_0-pseudo-norms". To formulate this statement rigorously, we first define a variational norm tailored to a dictionary of computational units.

The representation (1) of a function f as an input-output function of a shallow network with units from a dictionary G is unique when the dictionary is linearly independent. Many common dictionaries formed by functions defined on $\mathbb{R}^d$ or its infinite compact subsets are linearly independent (see, e.g., [25–27, 39]); however, this property may not be guaranteed for dictionaries on finite domains. For a general dictionary, we consider the minimum of l_1-norms of output weights over all representations of a given function f as input-output functions of shallow networks with units from the dictionary G,

$$\min \left\{ \sum_{i=1}^{k} |w_i| \,\middle|\, f = \sum_{i=1}^{k} w_i\, g_i\,,\ w_i \in \mathbb{R},\ g_i \in G \right\}. \tag{2}$$

The quantity (2) can be studied in terms of a concept of variational norm from nonlinear approximation theory. For a bounded subset G of a normed linear space $(\mathcal{X}, \|.\|)$, *G-variation (variation with respect to the dictionary G)*, denoted by $\|.\|_G$, is defined as

$$\|f\|_G := \inf \left\{ c \in \mathbb{R}_+ \,\middle|\, \frac{f}{c} \in \mathrm{cl}_{\mathcal{X}}\, \mathrm{conv}\,(G \cup -G) \right\},$$

where $-G := \{-g \mid g \in G\}$, $\mathrm{cl}_{\mathcal{X}}$ denotes the closure with respect to the topology induced by the norm $\|\cdot\|_{\mathcal{X}}$, and conv is the convex hull. Variation with respect to Heaviside perceptrons (called *variation with respect to half-spaces*) was introduced by Barron [2] and extended to general dictionaries by Kůrková [21]. For properties of variational norm and its role in estimates of rates of approximation, see, e.g., [13, 16, 19, 22, 23, 28].

The next proposition, which follows easily from the definition, shows the role of G-variation in estimates of l_1-sparsity.

Proposition 2 *Let G be a finite subset of $(\mathcal{X}, \|.\|)$ with card $G = k$. Then, for every $f \in \mathcal{X}$*

$$\|f\|_G = \min \left\{ \sum_{i=1}^{k} |w_i| \,\middle|\, f = \sum_{i=1}^{k} w_i\, g_i\,,\ w_i \in \mathbb{R},\ g_i \in G \right\}.$$

Thus, any representation of a function with "large" G-variation by a shallow network with units from a dictionary G must have "large" number of units and/or absolute values of some output weights must be "large".

By Proposition 2, G-variation of a function is equal to the smallest l_1-norm of an output weight vector in a shallow network representing the function. Moreover, G-variation is related to sparsity measured by the minimal number of nonzero coefficients in a representation of a functions as an input-output function of given type of shallow network.

The following theorem is a reformulation of Maurey-Jones-Barron's theorem [3] in terms of a variational norm from [21, 23]. As in this paper we focus on dictionaries on finite domains X, we state the theorem for finite dimensional Hilbert spaces $\mathcal{F}(X)$ which is isomorphic to the finite dimensional Euclidean space $\mathbb{R}^{\mathrm{card}X}$.

Theorem 1 *Let $X \subset \mathbb{R}^d$ be finite, G be a finite subset of $\mathcal{F}(X)$, $s_G = \max_{g \in G} \|g\|$, and $f \in \mathcal{F}(X)$. Then for every n, there exists a function $f_n \in \mathrm{span}_n\, G$ such that*

$$\|f - f_n\| \leq \frac{s_G\, \|f\|_G}{\sqrt{n}}.$$

Theorem 1 shows the relationship between sparsity expressed in terms of l_1-norm and "l_0-pseudo-norm". A function f can be approximated within an error at most $\frac{\|f\|_G}{\sqrt{n}}$ by a function computable by a shallow network with units from G with at most n nonzero output weights. Classes of d-variable functions with G-variations growing with d polynomially are of particular interest [16, 29].

In Hilbert spaces (in particular in the finite dimensional space $\mathcal{F}(X)$), $\|.\|_G$ can be investigated geometrically in terms of angles between the function f and functions from the dictionary G. The following theorem from [31] (see also [23] for a more general version) shows that lower bounds on G-variation of a function f can be obtained by estimating correlations of f with functions from the dictionary G. By $G^{\perp}$ is denoted the *orthogonal complement of G* in the Hilbert space $\mathcal{F}(X)$.

Theorem 2 *Let X be a finite subset of $\mathbb{R}^d$ and G be a bounded subset of $\mathcal{F}(X)$. Then for every $f \in \mathcal{F}(X) \setminus G^{\perp}$,*

$$\|f\|_G \geq \frac{\|f\|^2}{\sup_{g\in G}|\langle g, f\rangle|}.$$

Theorem 2 shows that functions which are almost orthogonal to all elements of a dictionary G have large variations with respect to G. We will use it as a tool for derivation of lower bounds on G-variation.

4 Shallow networks with signum perceptrons

It was shown by Ito [15], that the dictionary of perceptrons with any sigmoidal activation function satisfies the assumptions of Proposition 1. Thus, one-hidden-layer Heaviside as well as signum perceptron networks have the universal representation property.

From the point of view of the number of network units, there is only a minor difference between networks with signum and Heaviside perceptrons as

$$\mathrm{sgn}(t) = 2\vartheta(t) - 1 \quad \text{and} \quad \vartheta(t) = \frac{\mathrm{sgn}(t)+1}{2}. \tag{3}$$

An advantage of signum perceptrons is that they all have the same norms equal to $\sqrt{\mathrm{card} X}$, where X is the domain. Thus, in some arguments concerning computations of functions on finite domains, it is more convenient to consider signum perceptrons than Heaviside ones.

The dictionary of signum perceptrons

$$P_d(X) := \{\mathrm{sgn}(v \cdot x + b) : X \to \{-1, 1\} \mid v \in \mathbb{R}^d, b \in \mathbb{R}\}$$

occupies a relatively small subset of the set $\mathcal{B}(X)$ of all functions on X with values in $\{-1, 1\}$. The size of $P_d(X)$ grows with increasing size m of the domain X only polynomially (the degree of the polynomial is the dimension d of the space $\mathbb{R}^d$ where X is embedded), while the size 2^m of the set $\mathcal{B}(X)$ of all functions from X to $\{-1, 1\}$ grows with m exponentially. The following upper bound is a direct consequence of an upper bound on the number of linearly separable dichotomies of m points in $\mathbb{R}^d$ from [10] combined with a well-known estimate of partial sum of binomials (see [30]).

Theorem 3 *For every d and every $X \subset \mathbb{R}^d$ with card $X = m$,*

$$\mathrm{card}\ P_d(X) \leq 2\frac{m^d}{d!}.$$

In [30], combining the probabilistic Chernoff bound, the geometric lower bound on variational norm from Theorem 2, and the relatively small size of the dictionary $P_d(X)$, we proved the following theorem.

Theorem 4 *Let d be a positive integer, $X \subset \mathbb{R}^d$ with card $X = m$, f uniformly randomly chosen in $\mathcal{B}(X)$, and $b > 0$. Then,*

$$\Pr\left(\|f\|_{P_d(X)} \geq b\right) \geq 1 - 4\frac{m^d}{d!}e^{-\frac{m}{2b^2}}.$$

Thus, for large domains X, almost any uniformly randomly chosen function from X to $\{-1, 1\}$ has large variation with respect to signum perceptrons and so it cannot be l_1-sparsely represented by a shallow network with signum perceptrons. In particular, for card $X = 2^d$ and $b = 2^{\frac{d}{4}}$, Theorem 4 implies the following corollary.

Corollary 1 *Let d be a positive integer, $X \subset \mathbb{R}^d$ with card $X = m$, and f uniformly randomly chosen in $\mathcal{B}(X)$. Then,*

$$\Pr\left(\|f\|_{P_d(X)} \geq 2^{\frac{d}{4}}\right) \geq 1 - 4\frac{2^{d^2}}{d!}e^{-(2^{\frac{d}{2}}-1)}.$$

Corollary 1 shows that almost all uniformly randomly chosen functions on the d-dimensional Boolean cube $\{0, 1\}^d$ cannot be computed by shallow signum perceptron networks with the sum of absolute values of output weights depending on d polynomially.

Theorem 4 is existential. It proves that there exists a lot of functions which cannot be l_1-sparsely represented by shallow signum perceptron networks, but it does not suggest how to construct such functions.

5 Lower bounds on variational norms with respect to perceptrons

In this section, we complement the existential probabilistic Theorem 4 by a concrete construction of a class of binary-valued functions having relatively large variations with respect to signum perceptrons. We prove that for large domains such functions cannot be computed by shallow networks with small numbers of signum perceptrons and small sizes of output weights.

By Theorem 2, functions which are nearly orthogonal to all elements of a dictionary G have large G-variations. Thus, our aim is to construct functions which are nearly orthogonal to all signum perceptrons. It is quite difficult to estimate inner products of functions on large finite domains formed by points in $\mathbb{R}^d$ in general positions. To simplify our task, we focus on functions on square domains

$$X = \{x_1, \ldots, x_n\} \times \{y_1, \ldots, y_n\} \subset \mathbb{R}^d$$

formed by points in grid-like positions. For example, pixels of pictures in $\mathbb{R}^d$ as well as digitized high-dimensional cubes can form such square domains.

Functions on square domains can be represented by square matrices. For a function f on $X = \{x_1, \ldots, x_n\} \times \{y_1, \ldots, y_n\}$, we denote by $M(f)$ the $n \times n$ matrix defined as

$$M(f)_{i,j} = f(x_i, y_j).$$

On the other hand, an $n \times n$ matrix M induces a function f_M on X such that

$$f_M(x_i, y_j) = M_{i,j}.$$

The following lemma shows that the geometrical shape of the square domain guarantees that matrices $M(g)$ representing signum perceptrons $g \in P_d(X)$ can be reordered in such a way that each row and each column of the reordered matrix starts with a segment of -1's followed by a segment of $+1$'s (for proof see the Appendix).

Lemma 1 *Let $d = d_1 + d_2$, $\{x_i \,|\, i = 1, \ldots, n\} \subset \mathbb{R}^{d_1}$, $\{y_j \,|\, j = 1, \ldots, n\} \subset \mathbb{R}^{d_2}$, and $X = \{x_1, \ldots, x_n\} \times \{y_1, \ldots, y_n\} \subset \mathbb{R}^d$. Then for every $g \in P_d(X)$, there exists a reordering of rows and columns of the $n \times n$ matrix $M(g)$ such that in the reordered matrix, each row and each column start with a (possibly empty) initial segment of -1's followed by a (possibly empty) segment of $+1$'s.*

The inner product of two functions f and g on a square domain $X = \{x_1, \ldots, x_n\} \times \{y_1, \ldots, y_n\}$ is equal to the sum of entries of the matrices $M(f)$ and $M(g)$, i.e.,

$$\langle f, g \rangle = \sum_{i,j}^{n} M(f)_{i,j} M(g)_{i,j},$$

and thus, it is invariant under permutations of rows and columns performed jointly on both matrices $M(f)$ and $M(g)$. So to estimate inner products of functions represented by matrices, we can reorder rows and columns whenever it is convenient. In particular, we can use reorderings of matrices induced by signum perceptrons guaranteed by Lemma 1.

Signum perceptrons as functions with values in $\{-1, 1\}$ induce matrices with entries equal to -1 or $+1$. The reordering assembling -1's and $+1$'s in the matrix representing a signum perceptron (guaranteed by Lemma 1) reduces estimation of their inner products with functions $f : X \to \{-1, 1\}$ to estimation of differences of -1's and $+1$'s in submatrices of $M(f)$. To obtain small inner products, we need matrices whose submatrices have relatively small differences of -1's and $+1$'s.

One class of such matrices is formed by Hadamard matrices. A *Hadamard matrix* of order n is an $n \times n$ square matrix M with entries in $\{-1, 1\}$ such that any two distinct rows (or equivalently columns) of M are orthogonal. It follows directly from the definition that this property is invariant under permutations of rows and columns and sign flips of all elements in a row or a column. Note that Hadamard matrices were introduced as extremal ones among all $n \times n$ matrices with entries in $\{-1, 1\}$ as they have the largest determinants equal to $\sqrt{n}$. The well-known Lindsay Lemma bounds from above differences of $+1$'s and -1's in submatrices of Hadamard matrices (see, e.g., [11, p. 88]).

Lemma 2 (Lindsay) *Let n be a positive integer and M be an $n \times n$ Hadamard matrix. Then for any subset I of the set of indices of rows and any subset J of the set of indices of columns of M,*

$$\left| \sum_{i \in I} \sum_{j \in J} M_{i,j} \right| \leq \sqrt{n \operatorname{card} I \operatorname{card} J}.$$

Our main theorem shows that functions induced by Hadamard matrices of large orders have large variations with respect to signum perceptrons.

Theorem 5 *Let $d = d_1 + d_2$, $\{x_i \,|\, i = 1, \ldots, n\} \subset \mathbb{R}^{d_1}$, $\{y_j \,|\, j = 1, \ldots, n\} \subset \mathbb{R}^{d_2}$, $X = \{x_i \,|\, i = 1, \ldots, n\} \times \{y_j \,|\, j = 1, \ldots, n\} \subset \mathbb{R}^d$, and $f_M : X \to \{-1, 1\}$ be defined as $f_M(x_i, y_j) = M_{i,j}$, where M is an $n \times n$ Hadamard matrix. Then,*

$$\| f_M \|_{P_d(X)} \geq \frac{\sqrt{n}}{\lceil \log_2 n \rceil}.$$

A detailed proof is presented in the Appendix. An essential part of the proof is a construction of a partition of a matrix induced by a signum perceptron into submatrices which have all entries either equal to $+1$ or all entries equal to -1 and application of the Lindsay Lemma to a corresponding partition of a Hadamard matrix (see Fig 1).

Theorem 5 combined with Proposition 2 implies the following corollary.

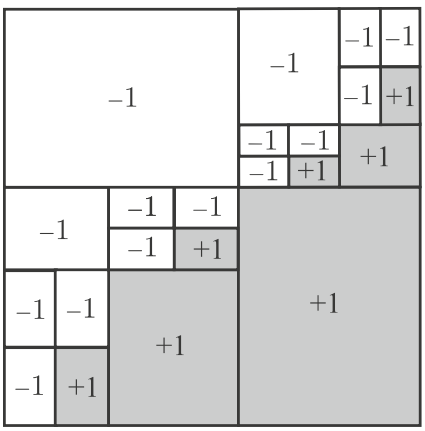

Fig. 1 Partition of the matrix $M(g)$

Corollary 2 *Let* $d = d_1 + d_2$, $\{x_i \mid i = 1, \ldots, n\} \subset \mathbb{R}^{d_1}$, $\{y_j \mid j = 1, \ldots, n\} \subset \mathbb{R}^{d_2}$, $X = \{x_i \mid i = 1, \ldots, n\} \times \{y_j \mid j = 1, \ldots, n\} \subset \mathbb{R}^d$, *and* $f_M : X \to \{-1, 1\}$ *be defined as* $f_M(x_i, y_j) = M_{i,j}$, *where* M *is an* $n \times n$ *Hadamard matrix. Then* f_M *cannot be computed by a shallow signum perceptron network having both the number of units and absolute values of all output weights depending on* $\log_2 n$ *polynomially.*

Corollary 2 shows that functions induced by Hadamard matrices cannot be computed by shallow networks with signum or Heaviside perceptrons with numbers of units and sizes of output weights considerably smaller than sizes of their domains. Numbers of units and sizes of output weights in these networks cannot be bounded by polynomials of $\log_2$ of the size of their domains.

Theorem 5 can be applied to domains containing sufficiently large squares, for example, domains representing pictures formed by two-dimensional squares with $2^k \times 2^k$ pixels or to digitized d-dimensional cubes.

Corollary 3 *Let* k *be a positive integer and* $f_M : \{0, 1\}^k \times \{0, 1\}^k \to \{-1, 1\}$ *be defined as* $f_M(x_i, y_j) = M_{i,j}$, *where* M *is a* $2^k \times 2^k$ *Hadamard matrix. Then,*

$$\|f_M\|_{P_d(\{0,1\}^{2k})} \geq \frac{2^{k/2}}{k}.$$

Functions generated by $2^k \times 2^k$ Hadamard matrices (such as the function generated by the matrix in the Fig. 2) cannot be computed by shallow signum perceptron networks with the sum of the absolute values of output weights bounded by a polynomial of k. This implies that the numbers of units and absolute values of all output weights in these networks cannot be bounded by any polynomial of k.

Similarly, functions defined on $2k$-dimensional discretized cubes of sizes $s^{2k} = s^k \times s^k$ cannot be computed by networks with numbers of signum perceptrons and output weights smaller than

$$\frac{s^{k/2}}{\lceil k \log_2 s \rceil}. \tag{4}$$

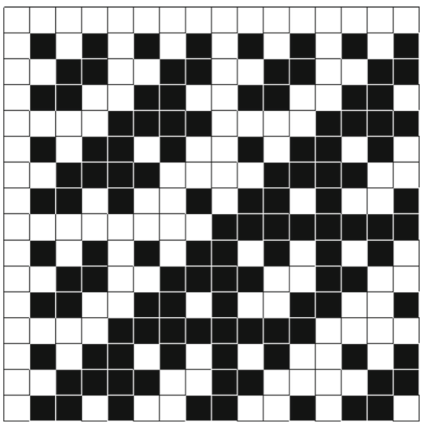

Fig. 2 $2^4 \times 2^4$ Sylvester-Hadamard matrix

6 Comparison of representations by one and two-hidden-layer networks

Applying Corollary 2 to a variety of types of Hadamard matrices, one obtains many examples of functions which cannot be computed by shallow perceptron networks with numbers of units and sizes of output weights bounded by

$$p(\log_2 \operatorname{card} X),$$

where p is a polynomial and X is the domain of the function.

Recall that if a Hadamard matrix of order $n > 2$ exists, then n is divisible by 4 (see, e.g., [33, p. 44]). It is conjectured that there exists a Hadamard matrix of every order divisible by 4. Various constructions of Hadamard matrices are known, such as Sylvester's recursive construction of $2^k \times 2^k$ matrices, Paley's construction based on quadratic residues, as well as constructions based on Latin squares, and on Steiner triples.

Two Hadamard matrices are called equivalent when one can be obtained from the second one by permutations of rows and columns and sign flips of all entries in a row or a column. Listings of known constructions of Hadamard matrices and enumeration of non-equivalent Hadamard matrices of some orders can be found in [38].

The oldest construction of a class of $2^k \times 2^k$ matrices with orthogonal rows and columns was discovered by Sylvester [40]. A $2^k \times 2^k$ matrix is called *Sylvester-Hadamard* and denoted $S(k)$ if it is constructed recursively starting from the matrix

$$S(2) = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

and iterating the Kronecker product

$$S(l+1) = S(2) \otimes S(l) = \begin{vmatrix} S(l) & S(l) \\ S(l) & -S(l) \end{vmatrix}$$

for $l - 1, \ldots, k - 1$ (see Fig. 2).

Corollary 3 implies that functions generated by $2^k \times 2^k$ Sylvester-Hadamard matrices cannot be represented by shallow signum perceptron networks with numbers of units and sizes of output weights smaller than $\frac{2^{k/2}}{k}$.

The following theorem shows that model complexities of signum or Heaviside perceptron networks computing functions generated by Sylvester-Hadamard matrices can be considerably decreased when two hidden layers are used instead of merely one hidden layer.

Theorem 6 *Let $S(k)$ be a $2^k \times 2^k$ Sylvester-Hadamard matrix, $h_k : \{0,1\}^k \times \{0,1\}^k \to \{-1,1\}$ be defined as $h_k(u,v) = S(k)_{u,v}$. Then, h_k can be represented by a network with one linear output and two hidden layers with k Heaviside perceptrons in each hidden layer.*

The proof of this theorem, given in the Appendix, utilizes the well-known (and easily verifiable by induction) equivalence of $2^k \times 2^k$ Sylvester-Hadamard matrix to the matrix with rows formed by generalized parities $p_u(v) : \{0,1\}^k \to \{-1,1\}$ defined as

$$p_u(v) = -1^{u \cdot v},$$

see, e.g., [33]. Thus, functions generated by Sylvester-Hadamard matrices are compositional functions. Inner products $u \cdot v$ can be computed by the first hidden layer and the classification of odd and even numbers by the second hidden layer.

Combining Theorems 6 and 5 with the relationship between signum and Heaviside perceptrons (3), we obtain an example of a class of functions on $\{0,1\}^{2k}$ which can be "l_0"-sparsely represented by two-hidden-layer Heaviside perceptron network but cannot be l_1-sparsely represented by a network with merely one hidden layer of Heaviside perceptrons.

Corollary 4 *Let $S(k)$ be a $2^k \times 2^k$ Sylvester-Hadamard matrix, $h_k : \{0,1\}^k \times \{0,1\}^k \to \{-1,1\}$ be defined as $h_k(u,v) = S(k)_{u,v}$. Then, h_k can be represented by a two-hidden-layer network with k Heaviside perceptrons in each hidden layer, but every representation of h_k by one-hidden-layer Heaviside perceptron network has at least $\frac{2^k}{k}$ units, or some of absolute values of output weights are greater or equal to $\frac{2^k}{k}$.*

Although generally, a small "l_0-pseudo-norm" does not imply a small l_1-norm, inspection of the proof of Theorem 6 (see the Appendix) shows that the function h_k induced by the $2^k \times 2^k$ Sylvester-Hadamard matrix can be expressed as an input-output functions of a two-hidden-layer Heaviside perceptron network having both the "l_0-pseudo-norm" and the l_1-norm of output weights equal to k. More precisely, all parameters of the network computing h_k are in the set $\{0,-1,1,b\}$, where $b \in (1,2)$. In particular, all output weights connecting the k units in the second hidden layer with the linear output unit are equal either to -1 or to $+1$. Thus, the sum of their absolute values is equal to k. On the other hand, by Theorem 5, the function induced by any $2^k \times 2^k$ Hadamard marix cannot be computed by a shallow signum perceptron network with the number of hidden units as well as the absolute values of all output weights bounded by any polynomial of k.

7 Discussion

In this paper, we investigated when deep networks are provably more efficient than shallow ones by comparing complexities of these two types of architectures. As estimation of minimal numbers of network units needed for a computation of a given function is a difficult nonconvex problem, we focused on sparsity defined in terms of the l_1-norms of their output weights. These norms have been used in weight-decay regularization techniques and are related to the concept of a variational norm tailored to a type of computational units which is a crucial factor in estimates of rates of approximation of functions by shallow networks with increasing numbers of units [16].

We derived constructive results which complement existential probabilistic results from [30] showing that almost any uniformly randomly chosen function on a large domain cannot be computed by a sparse shallow perceptron network.

Our arguments are based on estimation of correlations between functions to be computed by a shallow network and its computational units. This is quite difficult for finite domains formed by points in a general position. Thus, we focused on square domains which can represent, e.g., pixels of two-dimensional pictures or digitized cubes of even dimensions. Describing such functions as matrices, we proved that functions induced by Hadamard matrices are not correlated with any signum perceptron. We showed that these functions cannot be computed by shallow signum perceptron networks with both numbers of units and sizes of output weights bounded by polynomials of logarithm of the size of the domain. Thus, many concrete examples of functions which cannot be sparsely represented by shallow signum perceptron networks can be obtained from listings of Hadamard matrices (e.g., [38]).

In particular, our results imply that functions induced by Hadamard matrices on d-dimensional Boolean cubes cannot be computed by shallow perceptron networks with numbers of units and output weights depending on d polynomially. To compare one- and two-hidden-layer networks, we showed that functions induced by a subclass of Hadamard matrices formed by $2^k \times 2^k$ Sylvester-Hadamard matrices can be computed by two-hidden-layer networks with merely k perceptrons in each hidden layer. The representation of Sylvester-Hadamard matrices as two-hidden-layer perceptron network is due to their compositional structure and thus cannot easily be extended to other types of Hadamard matrices.

Although for large domains, almost any uniformly randomly chosen function has a large variation with respect to perceptrons [30], it is not easy to find examples of such functions. We constructed a class of such functions generated by matrices with rather extreme properties. However,

it is quite likely that many classification tasks of real data can be represented by functions which can be computed by reasonably small shallow networks. Deep networks seem to be more efficient than shallow ones in tasks which can be naturally described in terms of compositional functions. Such functions can be suitable for description of visual recognition tasks.

Acknowledgments This work was partially supported by the Czech Grant Agency grant GA15-18108S and institutional support of the Institute of Computer Science RVO 67985807.

Compliance with ethical standards

Conflict of interest The author declares that she has no conflict of interests.

Appendix

Proof of Lemma 1 Choose an expression of $g \in P_d(X)$ as $g(z) = \text{sign}(a \cdot z + b)$, where $z = (x, y) \in \mathbb{R}^{d_1} \times \mathbb{R}^{d_2}$, $a \in \mathbb{R}^d = \mathbb{R}^{d_1} \times \mathbb{R}^{d_2}$, and $b \in \mathbb{R}$. Let a_l and a_r denote the left and the right part, resp, of a, i.e., $a_{li} = a_i$ for $i = 1, \ldots, d_1$ and $a_{ri} = a_{d_1+i}$ for $i = 1, \ldots, d_2$. Then, $\text{sign}(a \cdot z + b) = \text{sign}(a_l \cdot x + a_r \cdot y + b)$. Let ρ and κ be permutations of the set $\{1, \ldots, n\}$ such that $a_l \cdot x_{\rho(1)} \leq a_l \cdot x_{\rho(2)} \leq \ldots \leq a_l \cdot x_{\rho(n)}$ and $a_r \cdot y_{\kappa(1)} \leq a_r \cdot y_{\kappa(2)} \leq \ldots \leq a_r \cdot y_{\kappa(n)}$.

Denote by $M(g)^*$ the matrix obtained from $M(g)$ by permuting its rows and columns by ρ and κ, resp. It follows from the definition of the permutations ρ and κ that each row and each column of $M(g)^*$ starts with a (possibly empty) initial segment of -1's followed by a (possibly empty) segment of $+1$'s. □

Proof of Theorem 5 By Theorem 2,

$$\|f_M\|_{P_d(X)} \geq \frac{\|f_M\|^2}{\sup_{g \in P_d(X)} |\langle f_M, g \rangle|} = \frac{n^2}{\sup_{g \in P_d(X)} |\langle f_M, g \rangle|}. \tag{5}$$

The inner product of f_M with g is equal to the sum of entries of the matrices M and $M(g)$, i.e., $\langle f_M, g \rangle = \sum_{i,j}^n M_{i,j} M(g)_{i,j}$, and so it is invariant under permutations of rows and columns performed simultaneously on both matrices M and $M(g)$.

Thus, without loss of generality, we can assume that each row and each column of $M(g)$ starts with a (possibly empty) initial segment of -1's followed by a (possibly empty) segment of $+1$'s. Otherwise, we reorder rows and columns in both matrices $M(g)$ and M applying permutations from Lemma 1.

To estimate $\langle f_M, g \rangle = \sum_{i,j=1}^n M_{i,j} M(g)_{i,j}$, we define a partition of the matrix $M(g)$ into a family of submatrices such that each submatrix from the partition of $M(g)$ has either all entries equal to -1 or all entries equal to $+1$ (see Fig. 1). We construct the partition of $M(g)$ as a union of sequence of families of submatrices (possibly some of them empty)

$$\mathcal{P}(g, k) = \{P(g, k, 1), \ldots, P(g, k, 2^k)\}, \quad k = 1, \ldots, \lceil \log_2 n \rceil,$$

defined recursively. To construct it, we also define an auxiliary sequence of families of submatrices

$$\mathcal{Q}(g, k) = \{Q(g, k, 1), \ldots, Q(g, k, 2^k)\}, \quad k = 1, \ldots, \lceil \log_2 n \rceil,$$

such that for each k,

$$\{P(g, k, 1), \ldots, P(g, k, 2^k), Q(g, k, 1), \ldots, Q(g, k, 2^k)\}$$

is a partition of the whole matrix $M(g)$.

First, we define $\mathcal{P}(g, 1) = \{P(g, 1, 1), P(g, 1, 2)\}$ and $\mathcal{Q}(g, 1) = \{Q(g, 1, 1), Q(g, 1, 2)\}$. Let $r_{1,1}$ and $c_{1,1}$ be such that the submatrix $P(g, 1, 1)$ of $M(g)$ formed by the entries from the first $r_{1,1}$ rows and the first $c_{1,1}$ columns of $M(g)$ has all entries equal to -1 and the submatrix $P(g, 1, 2)$ by the entries from the last $r_{1,2} = n - r_{1,1}$ rows and the last $c_{1,2} = n - c_{1,1}$ of $M(g)$ has all entries equal to $+1$. Let $Q(g, 1, 1)$ be the submatrix formed by the last $r_{1,2} = n - r_{1,1}$ rows and the first $c_{1,1}$ columns of $M(g)$ and $Q(g, 1, 2)$ be the the submatrix formed by the first $r_{1,2}$ rows and the last $c_{1,2} = n - c_{1,1}$ columns. So $\{P(g, 1, 1), P(g, 1, 2), Q(g, 1, 1), Q(g, 1, 2)\}$ is a partition of $M(g)$ into four submatrices (see Fig. 1).

Now, assume that the families $\mathcal{P}(g, k)$ and $\mathcal{Q}(g, k)$ are constructed. To define $\mathcal{P}(g, k + 1)$ and $\mathcal{Q}(g, k + 1)$, we divide each of 2^k submatrices $Q(g, k, j)$, $j = 1, \ldots, 2^k$ into four submatrices: $P(g, k + 1, 2j - 1)$, $P(g, k + 1, 2j)$, $Q(g, k+1, 2j-1)$, and $Q(g, k+1, 2j)$ such that each of the submatrices $P(g, k + 1, 2j - 1)$ has all entries equal to -1 and each of the submatrices $P(g, k + 1, 2j)$ has all entries equal to $+1$.

Iterating this construction at most $\lceil \log_2 n \rceil$ times, we obtain a partition of $M(g)$ formed by the union of families of submatrices $\mathcal{P}(g, k)$. It follows from the construction that for each k, the sum of the numbers of rows $\{r_{k,t} \mid t = 1, \ldots, 2^k\}$ and the sum of the numbers of columns $\{c_{k,t} \mid t = 1, \ldots, 2^k\}$ of these submatrices satisfy

$$\sum_{t=1}^{2^k} r_{k,t} = n \quad \text{and} \quad \sum_{l=1}^{2^k} c_{k,t} = n.$$

Let $\mathcal{P}(k) = \{P(k, 1), \ldots, P(k, 2^k)\}$ be the family of submatrices of M formed by the entries from the same rows and columns as corresponding submatrices from the family

$\mathcal{P}(g,k) = \{P(g,k,1),\ldots,P(g,k,2^k)\}$ of submatrices of $M(g)$.

To derive an upper bound on $|\langle f_M, g\rangle|$, we express it as

$$|\langle f_M, g\rangle| = \left|\sum_{i,j}^{n} M_{i,j} M(g)_{i,j}\right| = \left|\sum_{k=1}^{\lceil \log_2 n\rceil}\sum_{t=1}^{2^k} P(k,t)_{i,j}\, P(g,k,t)_{i,j}\right|. \quad (6)$$

As all the matrices $P(k,t)$ are submatrices of the Hadamard matrix M, by the Lindsay Lemma 2 for each submatrix $P(k,t)$,

$$\left|\sum_{i=1}^{r_{k,t}}\sum_{j=1}^{c_{k,t}} P(k,t)_{i,j}\right| \le \sqrt{n\, r_{k,t}\, c_{k,t}}.$$

All the matrices $P(g,k,t)$ have all entries either equal to 1 or all entries equal to -1. Thus,

$$\left|\sum_{i=1}^{r_{k,t}}\sum_{j=1}^{c_{k,t}} P(k,t)_{i,j}\, P(g,k,t)_{i,j}\right| \le \sqrt{n\, r_{k,t}\, c_{k,t}}.$$

As for all k, $\sum_{t=1}^{2^k} r_{k,t} = n$ and $\sum_{t=1}^{2^k} c_{k,t} = n$, we obtain by the Cauchy-Schwartz inequality

$$\sum_{t=1}^{2^k} \sqrt{r_{k,t}\, c_{k,t}} \le n.$$

Thus, for each k,

$$\sum_{t=1}^{2^k} |P(k,t)_{i,j}\, P(g,k,t)_{i,j}| \le \sum_{t=1}^{2^k} \sqrt{n\, r_{k,t}\, c_{k,t}} \le n\sqrt{n}.$$

Hence, by (6),

$$|\langle f_M, g\rangle| \le \sum_{k=1}^{\lceil \log_2 n\rceil} \left|\sum_{t=1}^{2^k} P(k,t)_{i,j}\, P(g,k,t)_{i,j}\right| \le n\sqrt{n}\,\lceil \log_2 n\rceil.$$

So by (5),

$$\|f_M\|_{P_d(X)} \ge \frac{n^2}{n\sqrt{n}\,\lceil \log_2 n\rceil} \ge \frac{\sqrt{n}}{\lceil \log_2 n\rceil}$$

□

Proof of Theorem 6 Any $2^k \times 2^k$ Sylvester-Hadamard matrix $S(k)$ is equivalent to the matrix $M(k)$ with rows and columns indexed by vectors $u, v \in \{0,1\}^k$ and entries

$$M(k)_{u,v} = -1^{u\cdot v}$$

(see, e.g., [33]). Thus, without loss of generality, we can assume that $S(k)_{u,v} = -1^{u\cdot v}$ (otherwise, we permute rows and columns).

To represent the function $h_k : \{0,1\}^k \times \{0,1\}^k \to \{-1,1\}$ by a two-hidden-layer network, we first define k Heaviside perceptrons from the first hidden layer. Choose any bias $b \in (1,2)$ and define input weights $c^i = (c^{i,l}, c^{i,r}) \in \mathbb{R}^k \times \mathbb{R}^k$, $i = 1,\ldots,k$, as $c^{il}_j = 1$ and $c^{ir}_j = 1$ when $j = i$, otherwise $c^{il}_j = 0$ and $c^{ir}_j = 0$. So for an input vector $x = (u,v) \in \{0,1\}^k \times \{0,1\}^k$, the output $y_i(x)$ of the i-th perceptron in the first hidden layer satisfies $y_i(x) = \vartheta(c^i \cdot x - b) = 1$ if and only if both $u_i = 1$ and $v_i = 1$; otherwise, $y_i(x)$ is equal to zero.

Let $w = (w_1,\ldots,w_k)$ be such that $w_j = 1$ for all $j = 1,\ldots,k$. In the second hidden layer, define k perceptrons by $z_j(y) := \vartheta(w \cdot y - j + 1/2)$. Finally, for all $j = 1,\ldots,k$, let the j-th unit from the second hidden layer be connected with one linear output unit with the weight $(-1)^j$.

The two-hidden-layer network obtained in this way computes the function $\sum_{j=1}^{k}(-1)^j \vartheta(w \cdot y(x) - j + 1/2)$, where $y_i(x) = \vartheta(c^i \cdot x - b)$, i.e., it computes the function $\sum_{j=1}^{k}(-1)^j \vartheta\left(\sum_{i=1}^{d/2}\vartheta(c^i \cdot x - b) - j + 1/2\right) = h_k(x) = h_k(u,v) = -1^{u\cdot v}$. □

References

1. Ba LJ, Caruana R (2014) Do deep networks really need to be deep? In: Ghahrani Z et al (eds) Advances in neural information processing systems, vol 27, pp 1–9
2. Barron AR (1992) Neural net approximation. In: Narendra K (ed) Proceedings 7th Yale workshop on adaptive and learning systems, pp 69–72. Yale University Press
3. Barron AR (1993) Universal approximation bounds for superpositions of a sigmoidal function. IEEE Trans Inf Theory 39:930–945
4. Bengio Y (2009) Learning deep architectures for AI. Foundations and Trends in Machine Learning 2:1–127
5. Bengio Y, Delalleau O, Roux NL (2006) The curse of highly variable functions for local kernel machines. In: Advances in neural information processing systems 18, pp 107–114. MIT Press
6. Bengio Y, LeCun Y (2007) Scaling learning algorithms towards AI. In: Bottou LO, Chapelle D, DeCoste, Weston J (eds) Large-Scale Kernel Machines. MIT Press
7. Bianchini M, Scarselli F (2014) On the complexity of neural network classifiers: a comparison between shallow and deep architectures. IEEE Trans Neural Netw Learning Syst 25(8):1553–1565
8. Candès EJ (2008) The restricted isometric property and its implications for compressed sensing. C R Acad Sci Paris I 346:589–592
9. Candès EJ, Tao T (2005) Decoding by linear programming. IEEE Trans Inf Process 51:4203–4215
10. Cover T (1965) Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. IEEE Trans Electron Comput 14:326–334
11. Erdös P, Spencer JH (1974) Probabilistic methods in combinatorics. Academic Press
12. Fine TL (1999) Feedforward neural network methodology. Springer, Berlin Heidelberg
13. Gnecco G, Sanguineti M (2011) On a variational norm tailored to variable-basis approximation schemes. IEEE Trans Inf Theory 57:549–558

14. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18:1527–1554
15. Ito Y (1992) Finite mapping by neural networks and truth functions. Mathematical Scientist 17:69–77
16. Kainen PC, Kůrková V, Sanguineti M (2012) Dependence of computational models on input dimension: tractability of approximation and optimization tasks. IEEE Trans Inf Theory 58:1203–1214
17. Kainen PC, Kůrková V, Vogt A (1999) Approximation by neural networks is not continuous. Neurocomputing 29:47–56
18. Kainen PC, Kůrková V, Vogt A (2000) Geometry and topology of continuous best and near best approximations. J Approx Theory 105:252–262
19. Kainen PC, Kůrková V, Vogt A (2007) A Sobolev-type upper bound for rates of approximation by linear combinations of Heaviside plane waves. J Approx Theory 147:1–10
20. Kecman V (2001) Learning and soft computing. MIT Press, Cambridge
21. Kůrková V (1997) Dimension-independent rates of approximation by neural networks. In: Warwick K, Kárný M (eds) Computer-intensive methods in control and signal processing. The curse of dimensionality, pp 261–270. Birkhäuser, Boston
22. Kůrková V (2008) Minimization of error functionals over perceptron networks. Neural Comput 20:250–270
23. Kůrková V (2012) Complexity estimates based on integral transforms induced by computational units. Neural Netw 33:160–167
24. Kůrková V (2016) Lower bounds on complexity of shallow perceptron networks. In: Jayne C, Iliadis L (eds) Engineering applications of neural networks. Communications in computer and information sciences, vol 629, pp 283–294. Springer
25. Kůrková V, Kainen PC (1994) Functionally equivalent feedforward neural networks. Neural Comput 6(3):543–558
26. Kůrková V, Kainen PC (1996) Singularities of finite scaling functions. Appl Math Lett 9(2):33–37
27. Kůrková V, Kainen PC (2014) Comparing fixed and variable-width Gaussian kernel networks. Neural Netw 57:23–28
28. Kůrková V, Sanguineti M (2002) Comparison of worst-case errors in linear and neural network approximation. IEEE Trans Inf Theory 48:264–275
29. Kůrková V, Sanguineti M (2008) Approximate minimization of the regularized expected error over kernel models. Math Oper Res 33:747–756
30. Kůrková V, Sanguineti M (2016) Model complexities of shallow networks representing highly varying functions. Neurocomputing 171:598–604
31. Kůrková V, Savický P, Hlaváčková K (1998) Representations and rates of approximation of real-valued Boolean functions by neural networks. Neural Netw 11:651–659
32. LeCunn Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444
33. MacWilliams F, Sloane NJA (1977) The theory of error-correcting codes. North-Holland, Amsterdam
34. Maiorov V, Pinkus A (1999) Lower bounds for approximation by MLP neural networks. Neurocomputing 25:81–91
35. Maiorov VE, Meir R (2000) On the near optimality of the stochastic approximation of smooth functions by neural networks. Adv Comput Math 13:79–103
36. Mhaskar HN, Liao Q, Poggio T (2016) Learning functions: when is deep better than shallow. Center for brains, minds & machines **CBMM Memo No. 045v3**, pp 1–12
37. Mhaskar HN, Liao Q, Poggio T (2016) Learning functions: when is deep better than shallow. Center for brains, minds & machines **CBMM Memo No. 045v4**, pp 1–12
38. Sloane NJA A library of Hadamard matrices. http://www.research.att.com/~njas/hadamard/
39. Sussman HJ (1992) Uniqueness of the weights for minimal feedforward nets with a given input-output map. Neural Netw 5(4):589–593
40. Sylvester J (1867) Thoughts on inverse orthogonal matrices, simultaneous sign successions, and tessellated pavements in two or more colours, with applications to Newton's rule, ornamental tile-work, and the theory of numbers. Phil Mag 34:461–475