Theoretical Properties for Neural Networks with Weight Matrices of Low Displacement Rank

Liang Zhao [1] Siyu Liao [1] Yanzhi Wang [2] Zhe Li [2] Jian Tang [2] Bo Yuan [1]

Abstract

Recently low displacement rank (LDR) matrices, or so-called structured matrices, have been proposed to compress large-scale neural networks. Empirical results have shown that neural networks with weight matrices of LDR matrices, referred as LDR neural networks, can achieve significant reduction in space and computational complexity while retaining high accuracy. We formally study LDR matrices in deep learning. First, we prove the universal approximation property of LDR neural networks with a mild condition on the displacement operators. We then show that the error bounds of LDR neural networks are as efficient as general neural networks with both single-layer and multiple-layer structure. Finally, we propose back-propagation based training algorithm for general LDR neural networks.

1. Introduction

Neural networks, especially large-scale deep neural networks, have made remarkable success in various applications such as computer vision, natural language processing, etc. (Krizhevsky et al., 2012)(Sutskever et al., 2014). However, large-scale neural networks are both memory-intensive and computation-intensive, thereby posing severe challenges when deploying those large-scale neural network models on memory-constrained and energy-constrained embedded devices. To overcome these limitations, many studies and approaches, such as connection pruning (Han et al., 2015)(Gong et al., 2014), low rank approximation (Denton et al., 2014)(Jaderberg et al., 2014), sparsity regularization (Wen et al., 2016)(Liu et al., 2015) etc., have been proposed to reduce the model size of large-scale (deep) neural networks.

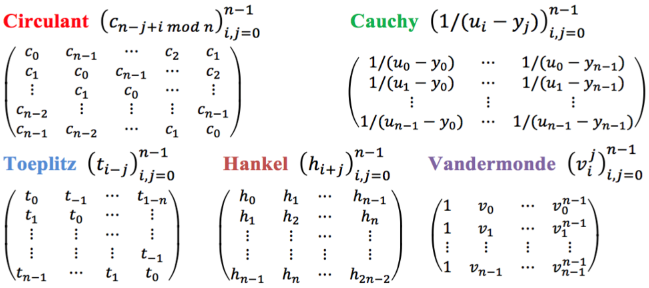
Circulant $(c_{n-j+i \bmod n})_{i,j=0}^{n-1}$
Cauchy $(1/(u_i - y_j))_{i,j=0}^{n-1}$
Toeplitz $(t_{i-j})_{i,j=0}^{n-1}$
Hankel $(h_{i+j})_{i,j=0}^{n-1}$
Vandermonde $(v_i^j)_{i,j=0}^{n-1}$

Figure 1. Examples of commonly used LDR (structured) matrices, i.e., circulant, Cauchy, Toeplitz, Hankel, and Vandermonde matrices.

LDR Construction and LDR Neural Networks: Among those efforts, *low displacement rank (LDR) construction* is a type of structure-imposing technique for network model reduction and computational complexity reduction. By regularizing the weight matrices of neural networks using the format of LDR matrices (when weight matrices are square) or the composition of multiple LDR matrices (when weight matrices are non-square), a *strong structure* is naturally imposed to the construction of neural networks. Since an LDR matrix typically requires $O(n)$ independent parameters and exhibits fast matrix operation algorithms (Pan, 2001), an immense space for network model and computational complexity reduction can be enabled. Pioneering work in this direction (Cheng et al., 2015)(Sindhwani et al., 2015) applied special types of LDR matrices (structured matrices), such as circulant matrices and Toeplitz matrices, for weight representation. Other types of LDR matrices exist such as Cauchy matrices, Vandermonde matrices, etc., as shown in Figure 1.

Benefits of LDR Neural Networks: Compared with other types of network compression approaches, the LDR construction shows several unique advantages. First, unlike heuristic weight-pruning methods (Han et al., 2015)(Gong et al., 2014) that produce irregular pruned networks, the LDR construction approach always guarantees the strong structure of the trained network, thereby avoiding the stor-

Co-first authors: Liang Zhao and Siyu Liao. [1]The City University of New York, New York, New York, USA [2]Syracuse University, Syracuse, New York, USA. Correspondence to: Bo Yuan <byuan@ccny.cuny.edu>.

Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

age space and computation time overhead incurred by the complicated indexing process. Second, as a "train from scratch" technique, LDR construction does not need extra re-training, and hence eliminating the additional complexity to the training process. Third, the reduction in space complexity and computational complexity by using the structured weight matrices are significant. Different from other network compression approaches that can only provide a heuristic compression factor, the LDR construction can enable the model reduction and computational complexity reduction in Big-O complexity: The storage requirement is reduced from $O(n^2)$ to $O(n)$, and the computational complexity can be reduced from $O(n^2)$ to $O(n \log n)$ or $O(n \log^2 n)$ because of the existence of fast matrix-vector multiplication algorithm (Pan, 2001)(Bini et al., 1996) for LDR matrices. For example, when applying structured matrices to the fully-connected layers of AlexNet using ImageNet dataset (Deng et al., 2009), the storage requirement can be reduced by more than 4,000X while incurring negligible degradation in overall accuracy (Cheng et al., 2015).

Motivation of This Work: Because of its inherent structure-imposing characteristic, convenient re-training-free training process and unique capability of simultaneous Big-O complexity reduction in storage and computation, LDR construction is a promising approach to achieve high compression ratio and high speedup for a broad category of network models. However, since imposing the structure to weight matrices results in substantial reduction of weight storage from $O(n^2)$ to $O(n)$, cautious researchers need to know whether the neural networks with LDR construction, referred to as LDR neural networks, will consistently yield the similar accuracy as compared with the uncompressed networks. Although (Cheng et al., 2015)(Sindhwani et al., 2015) have already shown that using LDR construction still results the same accuracy or minor degradation on various datasets, such as ImageNet (Deng et al., 2009), CIFAR (Krizhevsky & Hinton, 2009) etc., the *theoretical analysis*, which can provide the mathematically solid proofs that the LDR neural networks can converge to the same "effectiveness" as the uncompressed neural networks, is still very necessary in order to promote the wide application of LDR neural networks for emerging and larger-scale applications.

Technical Preview and Contributions: To address the above necessity, in this paper we study and provide a solid theoretical foundation of LDR neural networks on the ability to approximate an arbitrary continuous function, the error bound for function approximation, applications on shallow and deep neural networks, etc. More specifically, the main contributions of this paper include:

- We prove the *universal approximation property* for LDR neural networks, which states that the LDR neural networks could approximate an arbitrary continuous function with arbitrary accuracy given enough parameters/neurons. In other words, the LDR neural network will have the same "effectiveness" of classical neural networks without compression. This property serves as the theoretical foundation of the potential broad applications of LDR neural networks.
- We show that, for LDR matrices defined by $O(n)$ parameters, the corresponding LDR neural networks are still capable of achieving integrated squared error of order $O(1/n)$, which is identical to the error bound of unstructured weight matrices-based neural networks, thereby indicating that there is essentially no loss for restricting to the weight matrices to LDR matrices.
- We develop a universal training process for LDR neural networks with computational complexity reduction compared with backward propagation process for classical neural networks. The proposed algorithm is the generalization of the training process in (Cheng et al., 2015)(Sindhwani et al., 2015) that restricts the structure of weight matrices to circulant matrices or Toeplitz matrices.

Outline: The paper is outlined as follows. In Section 2 we review the related work on this topic. Section 3 presents necessary definitions and properties of matrix displacement and LDR neural networks. The problem statement is also presented in this section. In Section 4 we prove the universal approximation property for a broad family of LDR neural networks. Section 5 addresses the approximation potential (error bounds) with a limited amount of neurons on shallow LDR neural networks and deep LDR neural networks, respectively. The proposed detailed procedure for training general LDR neural networks are derived in Section 6. Section 7 concludes the article.

2. Related Work

Universal Approximation & Error Bound Analysis: For feedforward neural networks with one hidden layer, (Cybenko, 1989) and (Hornik et al., 1989) proved separately the universal approximation property, which guarantees that for any given continuous function or decision function and any error bound $\epsilon > 0$, there always exists a single-hidden layer neural network that approximates the function within ϵ integrated error. However, this property does not specify the number of neurons needed to construct such a neural network. In practice, there must be a limit on the maximum amount of neurons due to the computational limit. Moreover, the magnitude of the coefficients can be neither too large nor too small. To address these issues for general neural networks, (Hornik et al., 1989) proved that it is sufficient to approximate functions with weights and biases whose absolute values are bounded by a constant (depending on the activation function). (Hornik, 1991) further extended this result to an arbitrarily small bound. (Barron, 1993) showed that feedforward networks with one layer of sigmoidal nonlinearities achieve an integrated squared error with order of $O(1/n)$, where n is the number of neurons.

More recently, several interesting results were published on the approximation capabilities of deep neural networks. (Delalleau & Bengio, 2011) have shown that there exist certain functions that can be approximated by three-layer neural networks with a polynomial amount of neurons, while two-layer neural networks require exponentially larger amount to achieve the same error. (Montufar et al., 2014) and (Telgarsky, 2016) have shown the exponential increase of linear regions as neural networks grow deeper. (Liang & Srikant, 2016) proved that with $\log(1/\epsilon)$ layers, the neural network can achieve the error bound ϵ for any continuous function with $O(polylog(\epsilon))$ parameters in each layer.

LDR Matrices in Neural Networks: (Cheng et al., 2015) have analyzed the effectiveness of replacing conventional weight matrices in fully-connected layers with circulant matrices, which can reduce the time complexity from $O(n^2)$ to $O(n \log n)$, and the space complexity from $O(n^2)$ to $O(n)$, respectively. (Sindhwani et al., 2015) have demonstrated significant benefits of using Toeplitz-like matrices to tackle the issue of large space and computation requirement for neural networks training and inference. Experiments show that the use of matrices with low displacement rank offers superior tradeoffs between accuracy and time/space complexity.

3. Preliminaries on LDR Matrices and Neural Networks

3.1. Matrix Displacement

An $n \times n$ matrix $\mathbf{M}$ is called a *structured matrix* when it has a low displacement rank γ (Pan, 2001). More precisely, with the proper choice of operator matrices $\mathbf{A}$ and $\mathbf{B}$, if the Sylvester displacement

$$\nabla_{\mathbf{A},\mathbf{B}}(\mathbf{M}) := \mathbf{AM} - \mathbf{MB} \tag{1}$$

and the Stein displacement

$$\Delta_{\mathbf{A},\mathbf{B}}(\mathbf{M}) := \mathbf{M} - \mathbf{AMB} \tag{2}$$

of matrix $\mathbf{M}$ have a rank γ bounded by a value that is independent of the size of $\mathbf{M}$, then matrix $\mathbf{M}$ is referred to as a *matrix with a low displacement rank* (Pan, 2001). In this paper we will call these matrices as *LDR matrices*. Even a full-rank matrix may have small displacement rank with appropriate choice of displacement operators $(\mathbf{A}, \mathbf{B})$.

Operator Matrix		**Structured**	**Rank of**
A	**B**	**Matrix M**	$\mathbf{\Delta_{A,B}}(\mathrm{M})$
$\mathbf{Z_1}$	$\mathbf{Z_0}$	Circulant	≤ 2
$\mathbf{Z_1}$	$\mathbf{Z_0}$	Toeplitz	≤ 2
$\mathbf{Z_0}$	$\mathbf{Z_1}$	Henkel	≤ 2
$\mathbf{diag(t)}$	$\mathbf{Z_0}$	Vandermonde	≤ 1
$\mathbf{diag(s)}$	$\mathbf{diag(t)}$	Cauchy	≤ 1

Table 1. Pairs of Displacement Operators and Associated Structured Matrices. $\mathbf{Z_0}$ and $\mathbf{Z_1}$ represent the 0-unit-circulant matrix and the 1-unit-circulant matrix respectively, and vector $\mathbf{s}$ and $\mathbf{t}$ denote vectors defining Vandermonde and Cauchy matrices (cf. (Sindhwani et al., 2015)).

Figure 1 illustrates a series of commonly used structured matrices, including a circulant matrix, a Cauchy matrix, a Toeplitz matrix, a Hankel matrix, and a Vandermonde matrix, and Table 1 summarizes their displacement ranks and corresponding displacement operators.

The general procedure of handling LDR matrices generally takes three steps: *Compression, Computation with Displacements, Decompression*. Here compression means to obtain a low-rank displacement of the matrices, and decompression means to converting the results from displacement computations to the answer to the original computational problem. In particular, if one of the displacement operator has the property that its power equals the identity matrix, then one can use the following method to decompress directly:

Lemma 3.1. *If* $\mathbf{A}$ *is an* a*-potent matrix (i.e.,* $\mathbf{A}^q = a\mathbf{I}$ *for some positive integer* $q \leq n$*), then*

$$\mathbf{M} = \Big[\sum_{k=0}^{q-1} \mathbf{A}^k \Delta_{\mathbf{A},\mathbf{B}}(\mathbf{M})\mathbf{B}^k\Big](\mathbf{I} - a\mathbf{B}^q)^{-1}. \tag{3}$$

Proof. See Corollary 4.3.7 in (Pan, 2001). □

One of the most important characteristics of structured matrices is their low number of independent variables. The number of independent parameters is $O(n)$ for an n-by-n structured matrix instead of the order of n^2, which indicates that the storage complexity can be potentially reduced to $O(n)$. Besides, the computational complexity for many matrix operations, such as matrix-vector multiplication, matrix inversion, etc., can be significantly reduced when operating on the structured ones. The definition and analysis of structured matrices have been generalized to the case of n-by-m matrices where $m \neq n$, e.g., the block-circulant matrices (Pan et al., 2015). Our application of LDR matrices to neural networks would be the general n-by-m weight matrices. For certain lemmas and theorems such as Lemma 3.1, only the form on $n \times n$ square matrices is needed for the derivation procedure in this paper.

So we omit the generalized form of such statements unless necessary.

3.2. LDR Neural Networks

In this paper we study the viability of applying LDR matrices in neural networks. Without loss of generality, we focus on a feed-forward neural network with one fully-connected (hidden) layer, which is similar network setup as (Cybenko, 1989). Here the input layer (with n neurons) and the hidden layer (with kn neurons)[1] are assumed to be fully connected with a weight matrix $\mathbf{W} \in \mathbb{R}^{n\times kn}$ of displacement rank at most r corresponding to displacement operators $(\mathbf{A}, \mathbf{B})$, where $r \ll n$. The domain for the input vector $\mathbf{x}$ is the n-dimensional hypercube $I^n := [0,1]^n$, and the output layer only contains one neuron. The neural network can be expressed as:

$$y = G_{\mathbf{W},\boldsymbol{\theta}}(\mathbf{x}) = \sum_{j=1}^{kn} \alpha_j \sigma({\mathbf{w}_j}^T \mathbf{x} + \theta_j). \tag{4}$$

Here $\sigma(\cdot)$ is the activation function, $\mathbf{w}_j \in \mathbb{R}^n$ denotes the j-th column of the weight matrix $\mathbf{W}$, and $\alpha_j, \theta_j \in \mathbb{R}$ for $j = 1, ..., kn$. When the weight matrix $\mathbf{W} = [\mathbf{w}_1|\mathbf{w}_2|\cdots|\mathbf{w}_{kn}]$ has a low-rank displacement, we call it an LDR neural network. Matrix displacement techniques ensure that LDR neural network has much lower space requirement and higher computational speed comparing to classical neural networks of the similar size.

3.3. Problem Statement

In this paper, we aim at providing theoretical support on the accuracy of function approximation using LDR neural networks, which represents the "effectiveness" of LDR neural networks compared with the original neural networks. Given a continuous function $f(\mathbf{x})$ defined on $[0,1]^n$, we study the following tasks:

- For any $\epsilon > 0$, find an LDR weight matrix $\mathbf{W}$ so that the function defined by equation (4) satisfies

$$\max_{\mathbf{x}\in[0,1]^n} |f(\mathbf{x}) - G_{\mathbf{W},\boldsymbol{\theta}}(\mathbf{x})| < \epsilon. \tag{5}$$

- Fix a positive integer n, find an upper bound ϵ so that for any continuous function $f(\mathbf{x})$ there exists a bias vector $\boldsymbol{\theta}$ and an LDR matrix with at most n rows satisfying equation (5).
- Find a multi-layer LDR neural network that achieves error bound (5) but with fewer parameters.

The first task is handled in Section 4, which is the *universal approximation property* of LDR neural networks. It states that the LDR neural networks could approximate an arbitrary continuous function arbitrarily well and is the underpinning of the widespread applications. The error bounds for shallow and deep neural networks are derived in Section 5. In addition, we derived explicit back-propagation expressions for LDR neural networks in Section 6.

4. The Universal Approximation Property of LDR Neural Networks

We call a family of matrices S to have *representation property* if for any vector $\mathbf{v} \in \mathbb{R}^n$, there exists a matrix $\mathbf{M} \in S_{\mathbf{A},\mathbf{B}}$ such that $\boldsymbol{v}$ is a column of $\boldsymbol{M}$. Note that all five types of LDR matrices shown in Fig. 1 have this representation property because of their explicit pattern. In this section we will prove that this property also holds for many other LDR families. Based on this result, we are able to prove the universal approximation property of neural networks utilizing only LDR matrices.

Theorem 4.1. *Let* $\mathbf{A}$, $\mathbf{B}$ *be two* $n \times n$ *non-singular diagonalizable matrices. Define* $S^r_{\boldsymbol{A},\boldsymbol{B}}$ *as the set of matrices* $\mathbf{M}$ *such that* $\Delta_{\mathbf{A},\mathbf{B}}(\mathbf{M})$ *has rank at most* r*. Then the representation property holds for* $S^r_{\boldsymbol{A},\boldsymbol{B}}$ *if* $\boldsymbol{A}$ *and* $\boldsymbol{B}$ *satisfy*

i) $\mathbf{A}^q = a\mathbf{I}$ *for some positive integer* $q \le n$ *and a scalar* $a \neq 0$*; ii)* $(\mathbf{I} - a\mathbf{B}^q)$ *is nonsingular; iii) the eigenvalues of* $\mathbf{B}$ *have distinguishable absolute values.*

Proof. It suffices to prove for the case $r = 1$, as increasing r only provides more candidate matrices to choose from. By the property of Stein displacement, any matrix $\mathbf{M} \in S$ can be expressed in terms of $\mathbf{A}$, $\mathbf{B}$, and its displacement as follows:

$$\mathbf{M} = \sum_{k=0}^{q-1} \mathbf{A}^k \Delta_{\mathbf{A},\mathbf{B}}(\mathbf{M}) \mathbf{B}^k (\mathbf{I} - a\mathbf{B}^q)^{-1}. \tag{6}$$

Next we express $\Delta_{\mathbf{A},\mathbf{B}}(\mathbf{M})$ as a product of two vectors $\mathbf{g} \cdot \mathbf{h}^T$ since it has rank 1. Also write $\mathbf{A} = \mathbf{Q}^{-1}\mathbf{\Lambda}\mathbf{Q}$, where $\mathbf{\Lambda} = \mathbf{diag}(\lambda_1, ..., \lambda_n)$ is a diagonal matrix generated by the eigenvalues of $\mathbf{A}$. Now define $\mathbf{e}_j$ to be the j-th unit column vector for $j = 1, ..., n$. Write

$$\begin{aligned}\mathbf{QMe}_j =& \mathbf{Q}\sum_{k=0}^{q-1} \mathbf{A}^k \Delta_{\mathbf{A},\mathbf{B}}(\mathbf{M}) \mathbf{B}^k (\mathbf{I} - a\mathbf{B}^q)^{-1}\mathbf{e}_j \\ =& \mathbf{Q}\sum_{k=0}^{q-1} (\mathbf{Q}^{-1}\mathbf{\Lambda}\mathbf{Q})^k \mathbf{g}\mathbf{h}^T \mathbf{B}^k (\mathbf{I} - a\mathbf{B}^q)^{-1}\mathbf{e}_j \\ =& \Big(\sum_{k=0}^{q-1} s_{\mathbf{h},j} \mathbf{\Lambda}^k\Big)\mathbf{Qg}.\end{aligned} \tag{7}$$

[1] Please note that this assumption does not sacrifice any generality because the n-by-m case can be transformed to n-by-kn format with the nearest k using zero padding (Cheng et al., 2015).

Here we use $s_{\mathbf{h},j}$ to denote the resulting scalar from matrix product $\mathbf{h}^T\mathbf{B}^k(\mathbf{I}-a\mathbf{B}^q)^{-1}\mathbf{e}_j$ for $k=1,...,n$. Define $\mathbf{T} := (\mathbf{I}-a\mathbf{B}^q)^{-1}$. In order to prove the theorem, we need to show that there exists a vector $\mathbf{h}$ and an index k such that the matrix $\sum_{k=0}^{q-1} s_{\mathbf{h},j}\mathbf{\Lambda}^k$ is nonsingular. In order to distinguish scalar multiplication from matrix multiplication, we use notation $a\circ\mathbf{M}$ to denote the multiplication of a scalar value and a matrices whenever necessary. Rewrite the expression as

$$\begin{aligned}&\sum_{k=0}^{q-1} s_{\mathbf{h},j}\mathbf{\Lambda}^k\\ =&\sum_{k=0}^{q-1}\mathbf{h}^T\cdot\big(\mathbf{B}^k\mathbf{T}\mathbf{e}_j\circ\mathbf{diag}(\lambda_1^k,...,\lambda_n^k)\big)\\ =&\sum_{k=0}^{q-1}\mathbf{diag}(\mathbf{h}^T\cdot\mathbf{B}^k\cdot\mathbf{T}\cdot[\lambda_1^k\mathbf{e}_j|\cdots|\lambda_n^k\mathbf{e}_j])\\ =&\mathbf{diag}\Big(\mathbf{h}^T\cdot\Big(\sum_{k=0}^{q-1}\mathbf{B}^k\mathbf{T}\lambda_1^k\mathbf{e}_j\Big),...,\mathbf{h}^T\cdot\Big(\sum_{k=0}^{q-1}\mathbf{B}^k\mathbf{T}\lambda_n^k\mathbf{e}_j\Big)\Big).\end{aligned}$$

The diagonal matrix $\sum_{k=0}^{q-1}s_{\mathbf{h},j}\Lambda^k$ is nonsingular if and only if all of its diagonal entries are nonzero. Let $\mathbf{b}_{ij}$ denote the column vector $\sum_{k=0}^{q-1}\mathbf{B}\mathbf{T}^k\lambda_i^k\mathbf{e}_j$. Unless for every j there is an index i_j such that $\mathbf{b}_{i_jj}=\mathbf{0}$, we can always choose an appropriate vector $\mathbf{h}$ so that the resulting diagonal matrix is nonsingular. Next we will show that the former case is not possible using proof by contradiction. Assume that there is a column $\mathbf{b}_{i_jj}=\mathbf{0}$ for every $j=1,2,\cdots,n$, we must have:

$$\begin{aligned}\mathbf{0}=&[\mathbf{b}_{i_11}|\mathbf{b}_{i_22}|\cdots|\mathbf{b}_{i_nn}]\\ =&\Big[\sum_{k=0}^{q-1}\mathbf{B}^k\mathbf{T}\lambda_{i_1}^k\mathbf{e}_1|\cdots|\sum_{k=0}^{q-1}\mathbf{B}^k\mathbf{T}\lambda_{i_n}^k\mathbf{e}_n\Big]\\ =&\sum_{k=0}^{q-1}\mathbf{B}^k\mathbf{T}\cdot\mathbf{diag}(\lambda_{i_1}^k,...,\lambda_{i_n}^k).\end{aligned}$$

Since $\mathbf{B}$ is diagonalizable, we write $\mathbf{B}=\mathbf{P}^{-1}\mathbf{\Pi}\mathbf{P}$, where $\mathbf{\Pi}=\mathbf{diag}(\eta_1,...,\eta_n)$. Also we have $\mathbf{T}=(\mathbf{I}-a\mathbf{B}^q)^{-1}=\mathbf{P}^{-1}(\mathbf{I}-a\mathbf{\Pi}^q)^{-1}\mathbf{P}$. Then

$$\begin{aligned}\mathbf{0}=&\sum_{k=0}^{q-1}\mathbf{B}\mathbf{T}^k\mathbf{diag}(\lambda_{i_1}^k,...,\lambda_{i_n}^k)\\ =&\mathbf{P}^{-1}\Big[\sum_{k=0}^{q-1}\mathbf{\Pi}^k(\mathbf{I}-a\mathbf{\Pi}^q)^{-1}\mathbf{diag}(\lambda_{i_1}^k,...,\lambda_{i_n}^k)\Big]\mathbf{P}\\ =&\mathbf{P}^{-1}\sum_{k=0}^{q-1}\mathbf{diag}\Big((\lambda_{i_1}\eta_1)^k,...,(\lambda_{i_n}\eta_n)^k\Big)(\mathbf{I}-a\mathbf{\Pi}^q)^{-1}\mathbf{P}\\ =&\mathbf{P}^{-1}\mathbf{diag}\Big(\sum_{k=0}^{q-1}(\lambda_{i_1}\eta_1)^k,...,\sum_{k=0}^{q-1}(\lambda_{i_n}\eta_n)^k\Big)(\mathbf{I}-a\mathbf{\Pi}^q)^{-1}\mathbf{P}.\end{aligned}$$

This implies that $\lambda_{i_1}\eta_1,...,\lambda_{i_n}\eta_n$ are solutions to the equation

$$1+x+x^2+\cdots+x^{q-1}=0. \qquad (8)$$

By assumption of matrix $\mathbf{B}$, $\eta_1,...,\eta_k$ have different absolute values, and so are $\lambda_{i_1}\eta_1,...,\lambda_{i_1}\eta_1$, since all λ_k have the same absolute value because $\mathbf{A}^q=a\mathbf{I}$. This fact suggests that there are q distinguished solutions of equation (8), which contradicts the fundamental theorem of algebra. Thus it is incorrect to assume that matrix $\sum_{k=0}^{q-1}s_{\mathbf{h},j}\mathbf{\Lambda}^k$ is singular for all $\mathbf{h}\in\mathbb{R}^n$. With this property proven, given any vector $\mathbf{v}\in\mathbb{R}^n$, one can take the following procedure to find a matrix $\mathbf{M}\in S$ and a index j such that the j-th column of $\mathbf{M}$ equals $\mathbf{v}$:

i) Find a vector $\mathbf{h}$ and a index j such that matrix $\sum_{k=0}^{q-1}s_{\mathbf{h},j}\Lambda^k$ is non-singular;

ii) By equation (7), find

$$\mathbf{g}:=\mathbf{Q}^{-1}\Big(\sum_{k=0}^{q-1}s_{\mathbf{h},j}\mathbf{\Lambda}^k\Big)^{-1}\mathbf{Q}\mathbf{T}\mathbf{v};$$

iii) Construct $\mathbf{M}\in S$ with $\mathbf{g}$ and $\mathbf{h}$ by equation (6). Then its j-th column will equal to $\mathbf{v}$.

With the above construction, we have shown that for any vector $\mathbf{v}\in\mathbb{R}^n$ one can find a matrix $\mathbf{M}\in S$ and a index j such that the j-th column of $\mathbf{M}$ equals $\mathbf{v}$, thus the theorem is proved. □

Our main goal of this section is to show that neural networks with many types of LDR matrices (LDR neural networks) can approximate continuous functions arbitrarily well. In particular, we are going to show that Toeplitz matrices and circulant matrices, as specific cases of LDR matrices, have the same property. In order to do so, we need to introduce the following definition of a *discriminatory* function and state one of its key property as Lemma 4.1.

Definition 4.1. *A function $\sigma(u):\mathbb{R}\to\mathbb{R}$ is called as discriminatory if the zero measure is the only measure μ that satisfies the following property:*

$$\int_{I^n}\sigma(\mathbf{w^Tx}+\theta)d\mu(\mathbf{x})=0,\forall\mathbf{w}\in\mathbb{R}^n,\theta\in\mathbb{R}. \qquad (9)$$

Lemma 4.1. *[cf. (Cybenko, 1989)] Any bounded, measurable sigmoidal function is discriminatory.*

Now we are ready to present the universal approximation theorem of LDR neural networks with n-by-kn weight matrix $\mathbf{W}$:

Theorem 4.2 (Universal Approximation Theorem for LDR Neural Networks)**.** *Let σ be any continuous discriminatory function and $S^r_{\mathbf{A},\mathbf{B}}$ be a family of LDR matrices having representation property. Then for any continuous function*

$f(\mathbf{x})$ defined on I^n and any $\epsilon > 0$, there exists a function $G(\mathbf{x})$ in the form of equation (4) so that its weight matrix consists of k submatrices from $S^r_{\boldsymbol{A},\boldsymbol{B}}$ and

$$\max_{\mathbf{x}\in I^n} |G(\mathbf{x}) - f(\mathbf{x})| < \epsilon. \tag{10}$$

Proof. Denote the i-th $n \times n$ submatrix of $\mathbf{W}$ as $\mathbf{W}_i$. Then $\mathbf{W}$ can be written as

$$\mathbf{W} = \big[\mathbf{W}_1 | \mathbf{W}_2 | ... | \mathbf{W}_k\big]. \tag{11}$$

Let S_{I^n} denote the set of all continuous functions defined on I^n. Let U_{I^n} be the linear subspace of S_{I^n} that can be expressed in form of equation (4) where $\mathbf{W}$ consists of k sub-matrices with displacement rank at most r. We want to show that U_{I^n} is dense in the set of all continuous functions S_{I^n}.

Suppose not, by Hahn-Banach Theorem, there exists a bounded linear functional $L \neq 0$ such that $L(\bar{U}(I^n)) = 0$. Moreover, By Riesz Representation Theorem, L can be written as

$$L(h) = \int_{I^n} h(\mathbf{x}) d\mu(\mathbf{x}), \forall h \in S(I^n),$$

for some measure μ.

Next we show that for any $\mathbf{y} \in \mathbb{R}^n$ and $\theta \in \mathbb{R}$, the function $\sigma(\mathbf{y}^{\mathbf{T}}\mathbf{x} + \theta)$ belongs to the set U_{I^n}, and thus we must have

$$\int_{I^n} \sigma(\mathbf{y}^{\mathbf{T}}\mathbf{x} + \theta) d\mu(\mathbf{x}) = 0. \tag{12}$$

For any vector $\mathbf{y} \in \mathbb{R}^n$, Theorem 4.1 guarantees that there exists an $n \times n$ LDR matrix $\mathbf{M} = [\mathbf{b}_1 | \cdots | \mathbf{b}_n]$ and an index j such that $\mathbf{b}_j = \mathbf{y}$. Now define a vector $(\alpha_1, ..., \alpha_n)$ such that $\alpha_j = 1$ and $\alpha_1 = \cdots = \alpha_n = 0$. Also let the value of all bias be θ. Then the LDR neural network function becomes

$$\begin{aligned} G(\mathbf{x}) =& \sum_{i=1}^{n} \alpha_i \sigma(\mathbf{b}_i^{\mathbf{T}}\mathbf{x} + \theta) \\ =& \alpha_j \sigma(\mathbf{b}_j^{\mathbf{T}}\mathbf{x} + \theta) = \sigma(\mathbf{y}^{\mathbf{T}}\mathbf{x} + \theta). \end{aligned} \tag{13}$$

From the fact that $L(G(\mathbf{x})) = 0$, we derive that

$$\begin{aligned} 0 =& L(G(\mathbf{x})) \\ =& \int_{I^n} \sum_{i=1}^{n} \alpha_i \sigma(\mathbf{b}_i^{\mathbf{T}}\mathbf{x} + \theta) = \int_{I_n} \sigma(\mathbf{y}^{\mathbf{T}}\mathbf{x} + \theta) d\mu(\mathbf{x}). \end{aligned}$$

Since $\sigma(t)$ is a discriminatory function by Lemma 4.1. We can conclude that μ is the zero measure. As a result, the function defined as an integral with measure μ must be zero for any input function $h \in S(I^n)$. The last statement contradicts the property that $L \neq 0$ from the Hahn-Banach Theorem, which is obtained based on the assumption that the set U_{I^n} of LDR neural network functions are not dense in S_{I^n}. As this assumption is not true, we have the universal approximation property of LDR neural networks. □

Reference work (Cheng et al., 2015), (Sindhwani et al., 2015) have utilized a circulant matrix or a Toeplitz matrix for weight representation in deep neural networks. Please note that for the general case of n-by-m weight matrices, either the more general Block-circulant matrices should be utilized or padding extra columns or rows of zeroes are needed (Cheng et al., 2015). Circulant matrices and Topelitz matrices are both special form of LDR matrices, and thus we could apply the above universal approximation property of LDR neural networks and provide theoretical support for the use of circulant and Toeplitz matrices in (Cheng et al., 2015), (Sindhwani et al., 2015). Moreover, it is possible to consolidate the choice of parameters so that a block-Toeplitz matrix also shows Toeplitz structure globally. Therefore we arrive at the following corollary.

Corollary 4.1. *Any continuous function can be arbitrarily approximated by neural networks constructed with Toeplitz matrices or circulant matrices (with padding or using Block-circulant matrices).*

5. Error Bounds on LDR Neural Networks

With the universal approximation property proved, naturally we seek ways to provide error bound estimates for LDR neural networks. We are able to prove that for LDR matrices defined by $O(n)$ parameters (n represents the number of rows and has the same order as the number of columns), the corresponding structured neural network is capable of achieving integrated squared error of order $O(1/n)$, where n is the number of parameters. This result is asymptotically equivalent to Barron's aforementioned result on general neural networks, indicating that there is essentially no loss for restricting to LDR matrices.

The functions we would like to approximate are those who are defined on a n-dimensional ball $B_r = \{\mathbf{x} \in \mathbb{R}^n : |\mathbf{x}| \leq r\}$ such that $\int_{B_r} |\mathbf{x}||f(\mathbf{x})|\mu(d\mathbf{x}) \leq C$, where μ is an arbitrary measure normalized so that $\mu(B_r) = 1$. Let's call this set Γ_{C,B_r}. (Barron, 1993) considered the following set of bounded multiples of a sigmoidal function composed with linear functions:

$$G_\sigma = \{\alpha\sigma(\mathbf{y}^{\mathbf{T}}\mathbf{x} + \theta) : |\alpha| \leq 2C, \mathbf{y} \in \mathbb{R}^n, \theta \in \mathbb{R}\}. \tag{14}$$

He proved the following theorem:

Theorem 5.1 ((Barron, 1993)). *For every function in Γ_{C,B_r}, every sigmoidal function σ, every probability measure, and every $k \geq 1$, there exists a linear combination of* *sigmoidal functions $f_k(\mathbf{x})$ of the form*

$$f_k(\mathbf{x}) = \sum_{j=1}^{k} \alpha_j \sigma(\mathbf{y}_j^{\mathbf{T}} \mathbf{x} + \theta_j), \tag{15}$$

such that

$$\int_{B_r} (f(\mathbf{x}) - f_k(\mathbf{x}))^2 \mu(d\mathbf{x}) \leq \frac{4r^2 C}{k}. \tag{16}$$

Here $\mathbf{y}_j \in \mathbb{R}^n$ and $\theta_j \in \mathbb{R}$ for every $j = 1, 2, ..., N$, Moreover, the coefficients of the linear combination may be restricted to satisfy $\sum_{j=1}^{k} |c_j| \leq 2rC$.

Now we will show how to obtain a similar result for LDR matrices. Fix operator $(\mathbf{A}, \mathbf{B})$ and define

$$\begin{aligned} S_\sigma^{kn} = \Big\{ & \sum_{j=1}^{kn} \alpha_j \sigma(\mathbf{y}_j^{\mathbf{T}} \mathbf{x} + \theta_j) : |\alpha_j| \leq 2C, \mathbf{y}_j \in \mathbb{R}^n, \\ & \theta_j \in \mathbb{R}, j = 1, 2, ..., N, \\ & \textit{and } [\mathbf{y}_{(i-1)n+1} | \mathbf{y}_{(i-1)n+2} | \cdots | \mathbf{y}_{in}] \\ & \textit{is an LDR matrix, } \forall i = 1, ..., k \Big\}. \end{aligned} \tag{17}$$

Moreover, let G_σ^k be the set of function that can be expressed as a sum of no more than k terms from G_σ. Define the metric $||f - g||_\mu = \sqrt{\int_{B_r} (f(\mathbf{x}) - g(\mathbf{x}))^2 \mu(d\mathbf{x})}$. Theorem 5.1 essentially states that the minimal distance between a function $f \in \Gamma_{C,B}$ and G_σ^m is asymptotically $O(1/n)$. The following lemma proves that G_σ^k is in fact contained in S_σ^{kn}.

Lemma 5.1. *For any $k \geq 1$, $G_\sigma^k \subset S_\sigma^{kn}$.*

Proof. Any function $f_k(\mathbf{x}) \in G_\sigma^k$ can be written in the form

$$f_k(\mathbf{x}) = \sum_{j=1}^{k} \alpha_j \sigma(\mathbf{y}_j^{\mathbf{T}} \mathbf{x} + \theta_j). \tag{18}$$

For each $j = 1, ..., k$, define a $n \times n$ LDR matrix $\mathbf{W}_j$ such that one of its column is $\mathbf{y}_j$. Let $\mathbf{t}_{ij}$ be the i-th column of $\mathbf{W}_j$. Let i_j correspond to the column index such that $\mathbf{t}_{ij} = \mathbf{y}_j$ for all j. Now consider the following function

$$G(\mathbf{x}) := \sum_{j=1}^{k} \sum_{i=1}^{n} \beta_{ij} \sigma(\mathbf{t}_{ij}^{\mathbf{T}} \mathbf{x} + \theta_j), \tag{19}$$

where $\beta_{i_j j}$ equals α_j, and $\beta_{ij} = 0$ if $i \neq i_j$. Notice that we have the following equality

$$\begin{aligned} G(\mathbf{x}) &:= \sum_{j=1}^{k} \sum_{i=1}^{n} \beta_{ij} \sigma(\mathbf{t}_{ij}^{\mathbf{T}} \mathbf{x} + \theta_j) \\ &= \sum_{j=1}^{k} \beta_{i_j j} \sigma(\mathbf{t}_{ij}^{\mathbf{T}} \mathbf{x} + \theta_j) \\ &= \sum_{j=1}^{k} \alpha_j \sigma(\mathbf{y}_j^{\mathbf{T}} \mathbf{x} + \theta_j) = f_k(\mathbf{x}). \end{aligned}$$

Notice that the matrix $\mathbf{W} = [\mathbf{W}_1 | \mathbf{W}_2 | \cdots | \mathbf{W}_k]$ consists k LDR submatrices. Thus $f_k(\mathbf{x})$ belongs to the set S_σ^{kn}. □

By Lemma 5.1, we can replace G_σ^k with S_σ^{kn} in Theorem 5.1 and obtain the following error bound estimates on LDR neural networks:

Theorem 5.2. *For every disk $B_r \subset \mathbb{R}^n$, every function in Γ_{C,B_r}, every sigmoidal function σ, every normalized measure μ, and every $k \geq 1$, there exists neural network defined by a weight matrix consists of k LDR submatrices such that*

$$\int_{B_r} (f(\mathbf{x}) - f_{kn}(\mathbf{x}))^2 \mu(d\mathbf{x}) \leq \frac{4r^2 C}{k}. \tag{20}$$

Moreover, the coefficients of the linear combination may be restricted to satisfy $\sum_{k=1}^{N} |c_k| \leq 2rC$.

Theorem 5.2 is the first theoretical result that gives a general error bound on LDR neural networks. Empirically, (Cheng et al., 2015) reported that circulant neural networks are capable of achieving the same level of accuracy as AlexNet with more than 4,000X space saving on fully-connected layers. (Sindhwani et al., 2015) applied Toeplitz-type LDR matrices to several benchmark image classification datasets, retaining the performance of state-of-the-art models with very high compression ratio.

The next theorem naturally extended the result from (Liang & Srikant, 2016) to LDR neural networks, indicating that LDR neural networks can also benefit a parameter reduction if one uses more than one layers. More precisely, we have the following statement:

Theorem 5.3. *Let f be a continuous function on $[0, 1]$ and is $2n + 1$ times differentiable in $(0, 1)$ for $n = \lceil \log \frac{1}{\epsilon} + 1 \rceil$. If $|f^{(k)}(x)| \leq k!$ holds for all $x \in (0, 1)$ and $k \in \left[2n + 1\right]$, then for any $n \times n$ matrices $\mathbf{A}$ and $\mathbf{B}$ satisfying the conditions of Theorem 4.1, there exists a LDR neural network $G_{\mathbf{A},\mathbf{B}}(x)$ with $O(\log \frac{1}{\epsilon})$ layers, $O(\log^2 \frac{1}{\epsilon})$ binary step units, $O(\log^3 \frac{1}{\epsilon})$ rectifier linear units such that*

$$\max_{x \in [0,1]} |f(x) - G_{\mathbf{A},\mathbf{B}}(x)| < \epsilon.$$

Proof. The theorem with better bounds and without assumption of being LDR neural network is proved in (Liang

& Srikant, 2016) as Theorem 4. For each binary step unit or rectifier linear unit in the construction of the general neural network, attach $(n-1)$ dummy units, and expand the weights associated to this unit from a vector to an LDR matrix based on Theorem 4.1. By doing so we need to expand the number units by a factor of order $\log\frac{1}{\epsilon}$, and the asymptotic bounds are relaxed accordingly. □

6. Training LDR Neural Networks

In this section, we reformulate the gradient computation of LDR neural networks. The computation for propagating through a fully-connected layer can be written as

$$\mathbf{y} = \sigma(\mathbf{W}^T\mathbf{x} + \boldsymbol{\theta}), \tag{21}$$

where $\sigma(\cdot)$ is the activation function, $\mathbf{W} \in \mathbb{R}^{n\times kn}$ is the weight matrix, $\mathbf{x} \in \mathbb{R}^n$ is input vector and $\boldsymbol{\theta} \in \mathbb{R}^{kn}$ is bias vector. According to Equation (7), if $\mathbf{W}_i$ is an LDR matrix with operators $(\mathbf{A}_i, \mathbf{B}_i)$ satisfying conditions of Theorem 4.1, then it is essentially determined by two matrices $\mathbf{G}_i \in \mathbb{R}^{n\times r}, \mathbf{H}_i \in \mathbb{R}^{n\times r}$ as

$$\mathbf{W}_i = \Big[\sum_{k=0}^{q-1}\mathbf{A}_i^k\mathbf{G}_i\mathbf{H}_i^T\mathbf{B}_i^k\Big](\mathbf{I} - a\mathbf{B}_i^q)^{-1}. \tag{22}$$

To fit the back-propagation algorithm, our goal is to compute derivatives $\frac{\partial O}{\partial \mathbf{G}_i}$, $\frac{\partial O}{\partial \mathbf{H}_i}$ and $\frac{\partial O}{\partial \mathbf{x}}$ for any objective function $O = O(\mathbf{W}_1, \ldots, \mathbf{W}_k)$.

In general, given that $\mathbf{a} := \mathbf{W^T x} + \boldsymbol{\theta}$, we can have:

$$\frac{\partial O}{\partial \mathbf{W}} = \mathbf{x}(\frac{\partial O}{\partial \mathbf{a}})^T, \frac{\partial O}{\partial \mathbf{x}} = \mathbf{W}\frac{\partial O}{\partial \mathbf{a}}, \frac{\partial O}{\partial \boldsymbol{\theta}} = \frac{\partial O}{\partial \mathbf{a}}\mathbf{1}. \tag{23}$$

where $\mathbf{1}$ is a column vector full of ones. Let $\hat{\mathbf{G}}_{ik} := \mathbf{A}_i^k\mathbf{G}_i$, $\hat{\mathbf{H}}_{ik} := \mathbf{H}_i^T\mathbf{B}_i^k(\mathbf{I} - a\mathbf{B}_i^q)^{-1}$, and $\mathbf{W}_{ik} := \hat{\mathbf{G}}_{ik}\hat{\mathbf{H}}_{ik}$. The derivatives of $\frac{\partial O}{\partial \mathbf{W}_{ik}}$ can be computed as following:

$$\frac{\partial O}{\partial \mathbf{W}_{ik}} = \frac{\partial O}{\partial \mathbf{W}_i}. \tag{24}$$

According to Equation (23), if we let $\mathbf{a} = \mathbf{W}_{ik}$, $\mathbf{W} = \hat{\mathbf{G}}_{ik}^T$ and $\mathbf{x} = \hat{\mathbf{H}}_{ik}$, then $\frac{\partial O}{\partial \hat{\mathbf{G}}_{ik}}$ and $\frac{\partial O}{\partial \hat{\mathbf{H}}_{ik}}$ can be derived as:

$$\frac{\partial O}{\partial \hat{\mathbf{G}}_{ik}} = \big[\frac{\partial O}{\partial \hat{\mathbf{G}}_{ik}^T}\big]^T = \big[\hat{\mathbf{H}}_{ik}\frac{\partial O}{\partial \mathbf{W}_{ik}}\big]^T = (\frac{\partial O}{\partial \mathbf{W}_{ik}})^T\hat{\mathbf{H}}_{ik}^T, \tag{25}$$

$$\frac{\partial O}{\partial \hat{\mathbf{H}}_{ik}} = \hat{\mathbf{G}}_{ik}^T\frac{\partial O}{\partial \mathbf{W}_{ik}}. \tag{26}$$

Similarly, let $\mathbf{a} = \hat{\mathbf{G}}_{ik}$, $\mathbf{W} = (\mathbf{A}_i^k)^T$ and $\mathbf{x} = \mathbf{G}_i$, then $\frac{\partial O}{\partial \mathbf{G}_i}$ can be derived as:

$$\begin{aligned}\frac{\partial O}{\partial \mathbf{G}_i} &= \sum_{k=0}^{q-1}(\mathbf{A}_i^k)^T(\frac{\partial O}{\partial \hat{\mathbf{G}}_{ik}})\\ &= \sum_{k=0}^{q-1}(\mathbf{A}_i^k)^T(\frac{\partial O}{\partial \mathbf{W}_{ik}})^T\hat{\mathbf{H}}_{ik}^T.\end{aligned} \tag{27}$$

Substituting with $\mathbf{a} = \hat{\mathbf{H}}_{ik}$, $\mathbf{W} = \mathbf{H}_i^T$ and $\mathbf{x} = \mathbf{B}_i^k(\mathbf{I} - a\mathbf{B}_i^q)^{-1}$, we have $\frac{\partial O}{\partial \mathbf{H}_i}$ derived as:

$$\begin{aligned}\frac{\partial O}{\partial \mathbf{H}_i} &= \sum_{k=0}^{q-1}\mathbf{B}_i^k(\mathbf{I} - a\mathbf{B}_i^q)^{-1}(\frac{\partial O}{\partial \hat{\mathbf{H}}_{ik}})^T\\ &= \sum_{k=0}^{q-1}\mathbf{B}_i^k(\mathbf{I} - a\mathbf{B}_i^q)^{-1}(\frac{\partial O}{\partial \mathbf{W}_{ik}})^T\hat{\mathbf{G}}_{ik}.\end{aligned} \tag{28}$$

In this way, derivatives $\frac{\partial O}{\partial \mathbf{G}_i}$ and $\frac{\partial O}{\partial \mathbf{H}_i}$ can be computed given $\frac{\partial O}{\partial \mathbf{W}_{ik}}$ which is equal to $\frac{\partial O}{\partial \mathbf{W}_i}$. The essence of back-propagation algorithm is to propagate gradients backward from the layer with objective function to the input layer. $\frac{\partial O}{\partial \mathbf{W}_i}$ can be calculated from previous layer and $\frac{\partial O}{\partial \mathbf{x}}$ will be propagated to the next layer if necessary.

For practical use one may want to choose matrices $\mathbf{A}_i$ and $\mathbf{B}_i$ with fast multiplication method such as diagonal matrices, permutation matrices, banded matrices, etc. Then the space complexity (the number of parameters for storage) of $\mathbf{W}_i$ can be $O(2n + 2nr)$ rather than $O(n^2)$ of traditional dense matrix. The $2n$ is for $\mathbf{A}_i$ and $\mathbf{B}_i$ and $2nr$ is for $\mathbf{G}_i$ and $\mathbf{H}_i$. The time complexity of $\mathbf{W}_i^T\mathbf{x}$ will be $O(q(3n + 2nr))$ compared with $O(n^2)$ of dense matrix. Particularly, when $\mathbf{W}_i$ is a structured matrix like the Toeplitz matrix, the space complexity will be $O(2n)$. This is because the Toeplitz matrix is defined by $2n$ parameters. Moreover, its matrix-vector multiplication can be accelerated by using Fast Fourier Transform (for Toeplitz and circulant matrices), resulting in time complexity $O(n\log n)$. In this way the back-propagation computation for the layer can be done with near-linear time.

7. Conclusion

In this paper, we have proven that the universal approximation property of LDR neural networks. In addition, we also theoretically show that the error bounds of LDR neural networks are at least as efficient as general unstructured neural network. Besides, we also develop the back-propagation based training algorithm for universal LDR neural networks. Our study provides the theoretical foundation of the empirical success of LDR neural networks.

References

Barron, Andrew R. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945, 1993.

Bini, Dario, Pan, Victor, and Eberly, Wayne. Polynomial and matrix computations volume 1: Fundamental algorithms. *SIAM Review*, 38(1):161–164, 1996.

Cheng, Yu, Yu, Felix X, Feris, Rogerio S, Kumar, Sanjiv, Choudhary, Alok, and Chang, Shi-Fu. An exploration of parameter redundancy in deep networks with circulant projections. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2857–2865, 2015.

Cybenko, George. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, 1989.

Delalleau, Olivier and Bengio, Yoshua. Shallow vs. deep sum-product networks. In *Advances in Neural Information Processing Systems*, pp. 666–674, 2011.

Deng, Jia, Dong, Wei, Socher, Richard, Li, Li-Jia, Li, Kai, and Fei-Fei, Li. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255. IEEE, 2009.

Denton, Emily L, Zaremba, Wojciech, Bruna, Joan, LeCun, Yann, and Fergus, Rob. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in Neural Information Processing Systems*, pp. 1269–1277, 2014.

Gong, Yunchao, Liu, Liu, Yang, Ming, and Bourdev, Lubomir. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*, 2014.

Han, Song, Mao, Huizi, and Dally, William J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

Hornik, Kurt. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

Hornik, Kurt, Stinchcombe, Maxwell, and White, Halbert. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Jaderberg, Max, Vedaldi, Andrea, and Zisserman, Andrew. Speeding up convolutional neural networks with low rank expansions. *arXiv preprint arXiv:1405.3866*, 2014.

Krizhevsky, Alex and Hinton, Geoffrey. Learning multiple layers of features from tiny images. 2009.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Liang, Shiyu and Srikant, R. Why deep neural networks? *arXiv preprint arXiv:1610.04161*, 2016.

Liu, Baoyuan, Wang, Min, Foroosh, Hassan, Tappen, Marshall, and Pensky, Marianna. Sparse convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 806–814, 2015.

Montufar, Guido F, Pascanu, Razvan, Cho, Kyunghyun, and Bengio, Yoshua. On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pp. 2924–2932, 2014.

Pan, Victor. *Structured matrices and polynomials: unified superfast algorithms*. Springer Science & Business Media, 2001.

Pan, Victor Y, Svadlenka, John, and Zhao, Liang. Estimating the norms of random circulant and toeplitz matrices and their inverses. *Linear algebra and its applications*, 468:197–210, 2015.

Sindhwani, Vikas, Sainath, Tara, and Kumar, Sanjiv. Structured transforms for small-footprint deep learning. In *Advances in Neural Information Processing Systems*, pp. 3088–3096, 2015.

Sutskever, Ilya, Vinyals, Oriol, and Le, Quoc V. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Telgarsky, Matus. Benefits of depth in neural networks. *arXiv preprint arXiv:1602.04485*, 2016.

Wen, Wei, Wu, Chunpeng, Wang, Yandan, Chen, Yiran, and Li, Hai. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 2074–2082, 2016.