UNDERSTANDING AND TRAINING DEEP DIAGONAL CIRCULANT NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we study deep diagonal circulant neural networks, that is deep neural networks in which weight matrices are the product of diagonal and circulant ones. Besides making a theoretical analysis of their expressivity, we introduced principled techniques for training these models: we devise an initialization scheme and proposed a smart use of non-linearity functions in order to train deep diagonal circulant networks. Furthermore, we show that these networks outperform recently introduced deep networks with other types of structured layers. We conduct a thorough experimental study to compare the performance of deep diagonal circulant networks with state of the art models based on structured matrices and with dense models. We show that our models achieve better accuracy than other structured approaches while required 2x fewer weights as the next best approach. Finally we train deep diagonal circulant networks to build a compact and accurate models on a real world video classification dataset with over 3.8 million training examples.

1 INTRODUCTION

The deep learning revolution has yielded models of increasingly large size. In recent years, designing compact and accurate neural networks with a small number of trainable parameters has been an active research topic, motivated by practical applications in embedded systems (to reduce memory footprint (Sainath & Parada, 2015)), federated and distributed learning (to reduce communication (Konečný et al., 2016)), derivative-free optimization in reinforcement learning (to simplify the computation of the approximated gradient (Choromanski et al., 2018)). Besides a number of practical applications, it is also an important research question whether or not models really need to be this big or if smaller results can achieve similar accuracy (Ba & Caruana, 2014) .

Structured matrices are at the very core of most of the work on compact networks. In these models, dense weight matrices are replaced by matrices with a prescribed structure (e.g. low rank matrices, Toeplitz matrices, circulant matrices, LDR, etc.). Despite substantial efforts (e.g. Cheng et al. (2015); Moczulski et al. (2015)), the performance of compact models is still far from achieving an acceptable accuracy motivating their use in real-world scenarios. This raises several questions about the effectiveness of such models and about our ability to train them. In particular two main questions call for investigation:

Q1 *How to efficiently train deep neural networks with a large number of structured layers?*

Q2 *What is the expressive power of structured layers compared to dense layers?*

In this paper, we provide principled answers to these questions for the particular case of deep neural networks based on diagonal and circulant matrices (a.k.a. Diagonal-circulant networks or DCNNs).

The idea of using diagonal and circulant matrices together comes from a series of results in linear algebra by Müller-Quade et al. (1998) and Huhtanen & Perämäki (2015). The most recent result from Huhtanen & Perämäki demonstrates that any matrix A in $\mathbb{C}^{n\times n}$ can be decomposed into the product of $2n-1$ alternating diagonal and circulant matrices. The diagonal-circulant decomposition inspired Moczulski et al. (2015) to design the *AFDF* structured layer, which is the building block of DCNNs. However, Moczulski et al. (2015) were not able to train deep neural networks based on AFDF.

To answer **Q1**, we first describe a theoretically sound initialization procedure for DCNN which allows the signal to propagate through the network without vanishing or exploding. Furthermore, we provide a number of empirical insights to explain the behaviour of DCNNs, and show the impact of the number of the non-linearities in the network on the convergence rate and the accuracy of the network. By combining all these insights, we are able (for the first time) to train large and deep DCNNs. We demonstrate the good performance of DCNNs on a large scale application (the *YouTube-8M* video classification problem) and obtain very competitive accuracy.

To answer **Q2**, we propose an analysis of the expressivity of DCNNs by extending the results by Huhtanen & Perämäki (2015). We introduce a new bound on the number of diagonal-circulant required to approximate a matrix that depends on its rank. Building on this result, we demonstrate that a DCNN with bounded width and small depth can approximate any dense networks with ReLU activations.

Outline of the paper: We present in Section 2 the related work on structured neural networks and several compression techniques. Section 3 introduces circulant matrices, our new result extending the one from Huhtanen & Perämäki (2015). Section 4 proposes an theoretical analysis on the expressivity on DCNNs. Section 5 describes two efficient techniques for training deep diagonal circulant neural networks. Finally, Section 6 presents extensive experiments to compare the performance of deep diagonal circulant neural networks in different settings w.r.t. other state of the art approaches. Section 7 provides a discussion and concluding remarks.

2 Related Work

Structured matrices exhibit a number of good properties which have been exploited by deep learning practitioners, mainly to compress large neural networks architectures into smaller ones. For example Hinrichs & Vybíral (2011) have demonstrated that a single circulant matrix can be used to approximate the Johson-Lindenstrauss transform, often used in machine learning to perform dimensionality reduction. Building upon this result, Cheng et al. (2015) proposed to replace the weight matrix of a fully connected layer by a circulant matrix effectively replacing the complex transform modeled by the fully connected layer by a simple dimensionality reduction. Despite the reduction of expressivity, the resulting network demonstrated good accuracy using only a fraction of its original size (90% reduction).

Comparison with ACDC. Moczulski et al. (2015) have introduced two *Structured Efficient Linear Layers* (SELL) called AFDF and ACDC. The AFDF structured layer benefits from the theoretical results introduced by Huhtanen & Perämäki and can be seen the building block of DCNNs. However, Moczulski et al. (2015) only experiment using ACDC, a different type of layer that does not involve circulant matrices. As far as we can tell, the theoretical guarantees available for the AFDF layer do not apply on the ACDC layer since the cosine transform does not diagonalize circulant matrices (Sanchez et al., 1995). Another possible limit of the ACDC paper is that they only train large neural networks involving ACDC layers combined with many other expressive layers. Although the resulting network demonstrates good accuracy, it is difficult the characterize the true contribution of the ACDC layers in this setting.

Comparison with Low displacement rank structures. More recently, Thomas et al. (2018) have generalized these works by proposing neural networks with low-displacement rank matrices (LDR), that are structured matrices encompassing a large family of structured matrices, including Toeplitz-like, Vandermonde-like, Cauchy-like and more notably DCNNs. To obtain this result, LDR represents a structured matrix using two displacement operators and a low-rank residual. Despite being elegant and general, we found that the LDR framework suffers from several limits which are inherent to its generality, and makes it difficult to use in the context of large and deep neural networks. First, the training procedure for learning LDR matrices is highly involved and implies many complex mathematical objects such as Krylov matrices. Then, as acknowledged by the authors, the number of parameters required to represent a given structured matrix (e.g. a Toeplitz matrix) in practice is unnecessarily high (higher than required in theory).

Other compression techniques. Besides structured matrices, a variety of techniques have been proposed to build more compact deep learning models. These include *model distillation* (Hinton et al., 2015), Tensor Train (Novikov et al., 2015), Low-rank decomposition (Denil et al., 2013), to mention a few. However, Circulant networks show good performances in several contexts (the interested reader can refer to the results reported by Moczulski et al. (2015) and Thomas et al. (2018)).

3 A PRIMER ON CIRCULANT MATRICES AND A NEW RESULT

An n-by-n circulant matrix C is a matrix where each row is a cyclic right shift of the previous one as illustrated below.

$$C = circ(c) = \begin{bmatrix} c_0 & c_{n-1} & c_{n-2} & \dots & c_1 \\ c_1 & c_0 & c_{n-1} & & c_2 \\ c_2 & c_1 & c_0 & & c_3 \\ \vdots & & & \ddots & \vdots \\ c_{n-1} & c_{n-2} & c_{n-3} & & c_0 \end{bmatrix}$$

Circulant matrices exhibit several interesting properties from the perspective of numerical computations. Most importantly, any n-by-n circulant matrix C can be represented using only n coefficients instead of the n^2 coefficients required to represent classical unstructured matrices. In addition, the matrix-vector product is simplified from $O(n^2)$ to $O(n\ log(n))$ using the convolution theorem.

As we will show in this paper, circulant matrices also have a strong expressive power. So far, we know that a single circulant matrix can be used to represent a variety of important linear transforms such as random projections (Hinrichs & Vybíral, 2011). When they are combined with diagonal matrices, they can also be used as building blocks to represent any linear transform (Schmid et al., 2000; Huhtanen & Perämäki, 2015) with an arbitrary precision. Huhtanen & Perämäki were able to bound the number of factors that is required to approximate any matrix A with arbitrary precision.

Relation between diagonal circulant matrices and low rank matrices We recall this result in Theorem 1 as it is the starting point of our theoretical analysis (note that in the rest of the paper, $\|\cdot\|$ denotes the ℓ_2 norm when applied to vectors, and the operator norm when applied to matrices).

Theorem 1. *(Reformulation Huhtanen & Perämäki (2015)) For every matrix $A \in \mathbb{C}^{n \times n}$, for any $\epsilon > 0$, there exists a sequence of matrices $B_1 \dots B_{2n-1}$ where B_i is a circulant matrix if i is odd, and a diagonal matrix otherwise, such that $\|B_1 B_2 \dots B_{2n-1} - A\| < \epsilon$.*

Unfortunately, this theorem is of little use to understand the expressive power of diagonal-circulant matrices when they are used in deep neural networks. This is because: 1) the bound only depends on the dimension of the matrix A, not on the matrix itself, 2) the theorem does not provide any insights regarding the expressive power of m diagonal-circulant factors when m is much lower than $2n - 1$ as it is the case in most practical scenarios we consider in this paper.

In the following theorem, we enhance the result by Huhtanen & Perämäki by expressing the number of factors required to approximate A, *as a function of the rank of A*. This is useful when one deals with low-rank matrices, which is common in machine learning problems.

Theorem 2. *(Rank-based circulant decomposition) Let $A \in \mathbb{C}^{n \times n}$ be a matrix of rank at most k. Assume that n can be divided by k. For any $\epsilon > 0$, there exists a sequence of $4k + 1$ matrices $B_1, \dots, B_{4k+1}$, where B_i is a circulant matrix if i is odd, and a diagonal matrix otherwise, such that $\|B_1 B_2 \dots B_{4k+1} - A\| < \epsilon$*

A direct consequence of Theorem 2, is that if the number of diagonal-circulant factors is set to a value K, we can represent all linear transform A whose rank is $\frac{K-1}{4}$.

Compared to Huhtanen & Perämäki (2015), this result shows that structured matrices with fewer than $2n$ diagonal-circulant matrices (as it is the case in practice) can still represent a large class of matrices. As we will show in the following section, this result will be useful to analyze the expressivity of neural networks based on diagonal and circulant matrices.

4 ANALYSIS OF DIAGONAL CIRCULANT NEURAL NETWORKS (DCNNS)

Zhao et al. (2017) have shown that circulant networks with 2 layers and unbounded width are universal approximators. However, results on unbounded networks offer weak guarantees and two important questions have remained open until now: 1) *Can we approximate any function with a bounded-width circulant networks?* 2) *What function can we approximate with a circulant network that has a bounded width and a small depth?* We answer these two questions in this section.

First, we introduce some necessary definitions regarding neural networks and we provide a theoretical analysis of their approximation capabilities.

Definition 1 (Deep ReLU network)**.** *Given L weight matrices $W = (W_1, \ldots, W_L)$ with $W_i \in \mathbb{C}^{n \times n}$ and L bias vectors $b = (b_1, \ldots, b_L)$ with $b_i \in \mathbb{C}^n$, a* deep ReLU network *is a function $f_{W_L,b_L} : \mathbb{C}^n \to \mathbb{C}^n$ such that $f_{W,b}(x) = (f_{W_L,b_L} \circ \ldots \circ f_{W_1,b_1})(x)$ where $f_{W_i,b_i}(x) = \phi(W_i x + b_i)$ and $\phi(.)$ is a ReLU non-linearity* [1] *In the rest of this paper, we call L and n respectively the depth and the width of the network. Moreover, we call* total rank *k, the sum of the ranks of the matrices $W_1 \ldots W_L$. i.e. $k = \sum_{i=1}^{L} rank(W_i)$.*

We also need to introduce DCNNs, similarly to Moczulski et al. (2015).

Definition 2 (Diagonal Circulant Neural Networks)**.** *Given L diagonal matrices $D = (D_1, \ldots, D_L)$ with $D_i \in \mathbb{C}^{n \times n}$, L circulant matrices $C = (C_1, \ldots, C_L)$ with $C_i \in \mathbb{C}^{n \times n}$ and L bias vectors $b = (b_1, \ldots, b_L)$ with $b_i \in \mathbb{C}^n$, a* Diagonal Circulant Neural Networks *(DCNN) is a function $f_{W_L,b_L} : \mathbb{C}^n \to \mathbb{C}^n$ such that $f_{D,C,b}(x) = (f_{D_L,C_L,b_L} \circ \ldots \circ f_{D_1,C_1,b_1})(x)$ where $f_{D_i,C_i,b_i}(x) = \phi_i(D_i C_i x + b_i)$ and where $\phi_i(.)$ is a ReLU non-linearity or the identity function.*

We can now show that bounded-width DCNNs can approximate any Deep ReLU Network, and as a corollary, that they are universal approximators.

Lemma 1. *Let $\mathcal{N}$ be a deep ReLU network of width n and depth L, and let $\mathcal{X} \subset \mathbb{C}^n$ be a bounded set. For any $\epsilon > 0$, there exists a DCNN $\mathcal{N}'$ of width n and of depth $(2n-1)L$ such that $\|\mathcal{N}(x) - \mathcal{N}'(x)\| < \epsilon$ for all $x \in \mathcal{X}$.*

The proof is in the supplemental material. We can now state the universal approximation corollary:

Corollary 1. *Bounded width DCNNs are universal approximators in the following sense: for any continuous function $f : [0,1]^n \to \mathbb{R}_+$ of bounded supremum norm, for any $\epsilon > 0$, there exists a DCNN $\mathcal{N}_\epsilon$ of width $n+3$ such that $\forall x \in [0,1]^{n+3}$, $|f(x_1 \ldots x_n) - (\mathcal{N}_\epsilon(x))_1| < \epsilon$, where $(\cdot)_i$ represents the i^{th} component of a vector.*

This is a first result, however $(2n+5)L$ is not a small depth (in our experiments, n can be over 300 000), and a number of work provided empirical evidences that DCNN with small depth can offer good performances (e.g. Araujo et al. (2018); Cheng et al. (2015)). To improve our result, we introduce our main theorem which studies the approximation properties of these small depth networks.

Theorem 3. *(Rank-based expressive power of DCNNs) Let $\mathcal{N}$ be a deep ReLU network of width n, depth L and a total rank k and assume n is a power of 2. Let $\mathcal{X} \subset \mathbb{C}^n$ be a bounded set. Then, for any $\epsilon > 0$, there exists a DCNN with ReLU activation $\mathcal{N}'$ of width n such that $\|\mathcal{N}(x) - \mathcal{N}'(x)\| < \epsilon$ for all $x \in \mathcal{X}$ and the depth of $\mathcal{N}'$ is bounded by $9k$.*

Remark that in the theorem, we require that n is a power of 2. We conjecture that the result still holds even without this condition.

This result refines Lemma 1, and answer our second question: a DCNN of bounded width and small depth can approximate a Deep ReLU network of low total rank. Note that the converse is not true: because n-by-n circulant matrix can be of rank n, approximating a DCNN of depth 1 can require a deep ReLU network of total rank equals to n.

Expressivity of DCNNs For the sake of clarity, we highlight the significance of these results with the two following properties.

Properties. Given an arbitrary fixed integer n, let $\mathcal{R}_k$ be the set of all functions $f : \mathbb{R}^n \to \mathbb{R}^n$ representable by a deep ReLU network of total rank at most k and let $\mathcal{C}_l$ the set of all functions $f : \mathbb{R}^n \to \mathbb{R}^n$ representable by deep diagonal-circulant networks of depth at most l, then:

$$\forall k, \exists l \quad \mathcal{R}_k \subsetneq \mathcal{C}_l \tag{1}$$

$$\forall l, \nexists k \quad \mathcal{C}_l \subseteq \mathcal{R}_k \tag{2}$$

[1] Because our networks deal with complex numbers, we use an extension of the ReLU function to the complex domain. The most straightforward extension defined in Trabelsi et al. (2018) is as follows: $\mathrm{ReLU}(z) = \mathrm{ReLU}\left(\Re(z)\right) + i\mathrm{ReLU}\left(\Im(z)\right)$, where $\Re$ and $\Im$ refer to the real and imaginary parts of z.

We illustrate the meaning of this properties using Figure 1. As we can see, the set $\mathcal{R}_k$ of all the functions representable by a deep ReLU network of total rank k is strictly included in the set $\mathcal{C}_{9k}$ of all DCNN of depth $9k$ (as by Theorem 3).

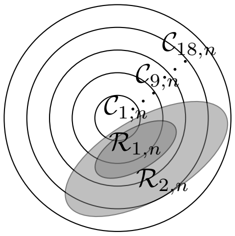
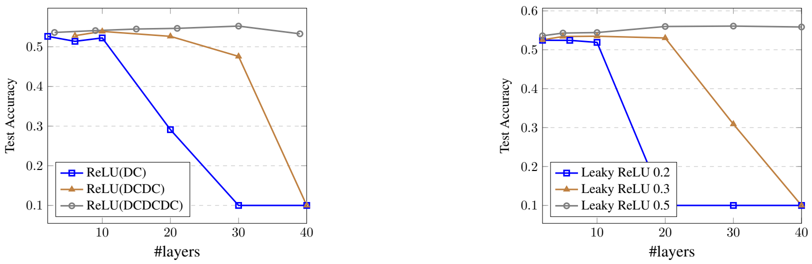


Figure 1: Illustration of Properties (1) and (2).

These properties are interesting for many reasons. First, Property (2) shows that diagonal-circulant networks are *strictly more expressive* than networks with low total rank. Second and most importantly, in standard deep neural networks, it is known that the most of the singular values are close to zero (see e.g. Sedghi et al. (2018); Arora et al. (2019)). Property (1) shows that these networks can efficiently be approximated by diagonal-circulant networks. Finally, several publications have shown that neural networks can be trained explicitly to have low-rank weight matrices (Li & Shi, 2018; Goyal et al., 2019). This opens the possibility of learning compact and accurate diagonal-circulant networks.

5 How to train very deep DCNNs



(a)

(b)

Figure 2: Experiments on training DCNNs and other structured neural networks on CIFAR-10. Figure 2(a): impact of increasing the number of ReLU activations in a DCNN. Deep DCNNs with fewer ReLUs are easier to train. Figure 2(b): impact of increasing the slope of a Leaky-ReLU in DCNNs. Deep DCNNs with a larger slope are easier to train.

Training DCNNs has revealed to be a challenging problem. We devise two techniques to facilitate the training of deep DCNNs. First, we propose an initialization procedure which guarantee the signal is propagated across the network without vanishing nor exploding. Secondly, we study the behavior of DCNNs with different non-linearity functions and determine the best parameters for different settings.

Initialization scheme The following initialization procedure which is a variant of Xavier initialization. First, for each circulant matrix $C = circ(c_1 \dots c_n)$, each c_i is randomly drawn from $\mathcal{N}\left(0, \sigma^2\right)$, with $\sigma = \sqrt{\frac{2}{n}}$. Next, for each diagonal matrix $D = diag(d_1 \dots d_n)$, each d_i is drawn randomly and uniformly from $\{-1, 1\}$ for all i. Finally, all biases in the network are randomly drawn from $\mathcal{N}\left(0, \sigma'^2\right)$, for some small value of σ'. The following proposition states that the covariance matrix at the output of any layer in a DCNN, independent of the depth, is constant.

Proposition 4. *Let $\mathcal{N}$ be a DCNN of depth L initialized according to our procedure, with $\sigma' = 0$. Assume that all layers 1 to $L - 1$ have ReLU activation functions, and that the last layer has the* *identity activation function. Then, for any $x \in \mathbb{R}^n$, the covariance matrix of $\mathcal{N}(x)$ is $\frac{2.Id}{n} \left\lVert x \right\rVert_2^2$. Moreover, note that this covariance does not depend on the depth of the network.*

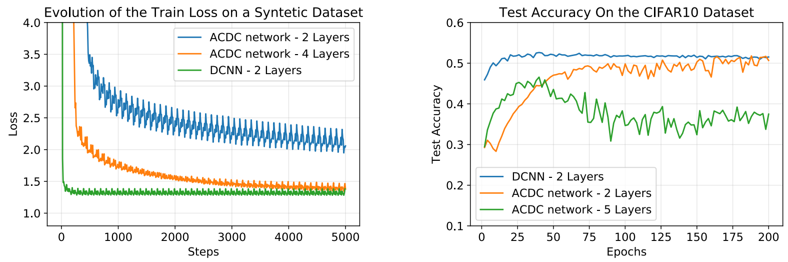
Non-linearity function We empirically found that reducing the number of non-linearities in the networks simplifies the training of deep neural networks. To support this claim, we conduct a series of experiments on various DCNNs with a varying number of ReLU activations (to reduce the number of non-linearities, we replace some ReLU activations with the identity function). In a second experiment, we replace the ReLU activations with Leaky-ReLU activations and vary the slope of the Leaky ReLU (a higher slope means an activation function that is closer to a linear function). The results of this experiment are presented in Figure 2(a) and 2(b). In 2(a), "ReLU(DC)" means that we interleave on ReLU activation functions between every diagonal-circulant matrix, whereas ReLU(DCDC) means we interleave a ReLU activation every other block etc. In both Figure 2(a) and Figure 2(b), we observe that reducing the non-linearity of the networks can be used to train deeper networks. This is an interesting result, since we can use this technique to adjust the number of parameters in the network, without facing training difficulties. We obtain a maximum accuracy of 0.56 with one ReLU every three layers and leaky-ReLUs with a slope of 0.5. We hence rely on this setting in the experimental section.

6 Empirical Evaluation

This experimental section aims at answering the following questions:

> **Q6.1** – How do DCNNs compare to other approaches such as ACDC, LDR or other structured approaches?
>
> **Q6.2** – How do DCNNs compare to other compression based techniques?
>
> **Q6.3** – How do DCNNs perform in the context of large scale real-world machine learning applications?

6.1 Comparison with other structured approaches (Q6.1)



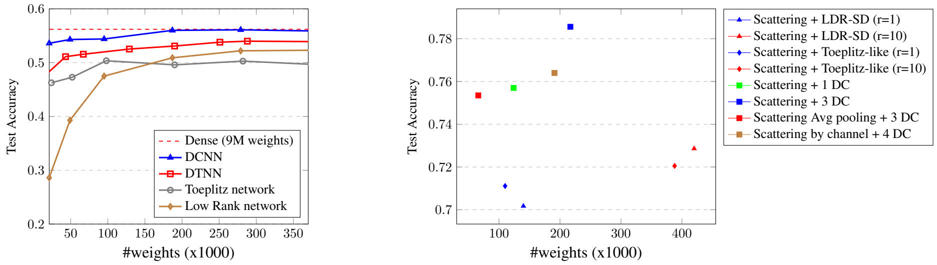
(a) (b)

Figure 3: Comparison of DCNNs and ACDC networks on two different tasks. Figure 3(a) shows the evolution of the training loss on a regression task with synthetic data. Figure 3(b) shows the test accuracy on the CIFAR-10 dataset.

Comparison with ACDC Moczulski et al. (2015). In Section 2, we have discussed the differences between the ACDC framework and our approach from a theoretical perspective. In this section, we conduct experiments to compare the performance of DCNNs with neural networks based on ACDC layers. We first reproduce the experimental setting from Moczulski et al. (2015), and compare both approaches using only linear networks (i.e. networks without any ReLU activations). The results are presented in Figure 3(a). On this simple setting, both architectures demonstrate good performance, however, DCNNs offer better convergence rate. In Figure 3(b), we compare neural networks with ReLU activations on CIFAR-10. The synthetic dataset has been created in order to reproduce the experiment on the regression linear problem proposed by Moczulski et al. (2015). We draw X, Y and W from a uniform distribution between [-1, +1] and ϵ from a normal distribution with mean 0 and variance 0.01. The relationship between X and Y is define by $Y = XW + \epsilon$.

We found that networks which are based only on ACDC layers are difficult to train and offer poor accuracy on CIFAR. (We have tried different initialization schemes including the one from the original paper, and the one we propose in this paper.) Moczulski et al. (2015) manage to train a large VGG network however these networks are generally highly redundant, the contribution of the structured layer is difficult to quantify. We also observe that adding a single dense layer improves the convergence rate of ACDC in the linear case networks, which explain the good results of Moczulski et al. (2015). However, it is difficult to characterize the true contribution of the ACDC layers when the network involved a large number of other expressive layers.

In contrast, deep DCNNs can be trained and offer good performance without additional dense layers (these results are in line with our experiments on the *YouTube-8M* dataset). We can conclude that DCNNs are able to model complex relations at a low cost.



(a)

(b)

Figure 4: Figure 4(a): network size vs. accuracy compared on Dense networks, DCNNs (our approach), DTNNs (our approach), neural networks based on Toeplitz matrices and neural networks based on Low Rank-based matrices. DCNNs outperforms alternatives structured approaches. Figure 4(b) shows the accuracy of different structured architecture given the number of trainable parameters.

Comparison with Dense networks, Toeplitz networks and Low Rank networks. We now compare DCNNs with other state-of-the-art structured networks by measuring the accuracy on a flattened version of the CIFAR-10 dataset. Our baseline is a dense feed-forward network with a fixed number of weights (9 million weights). We compare with DCNNs and with DTNNs (see below), Toeplitz networks, and Low-Rank networks Yu et al. (2017). We first consider Toeplitz networks which are stacked Toeplitz matrices interleaved with ReLU activations since Toeplitz matrices are closely related to circulant matrices. Since Toeplitz networks have a different structure (they do not include diagonal matrices), we also experiment using DTNNs, a variant of DCNNs where all the circulant matrices have been replaced by Toeplitz matrices. Finally we conduct experiments using networks based on low-rank matrices as they are also closely related to our work. For each approach, we report the accuracy of several networks with a varying depth ranging from 1 to 40 (DCNNs, Toeplitz networks) and from 1 to 30 (from DTNNs). For low-rank networks, we used a fixed depth network and increased the rank of each matrix from 7 to 40. We also tried to increase the depth of low rank matrices, but we found that deep low-rank networks are difficult to train so we do not report the results here. We compare all the networks based on the number of weights from 21K (0.2% of the dense network) to 370K weights (4% of the dense network) and we report the results in Figure 4(a). First we can see that the size of the networks correlates positively with their accuracy which demonstrate successful training in all cases. We can also see that the DCNNs achieves the maximum accuracy of 56% with 20 layers ($\sim$ 200K weights) which as as good as the dense networks with only 2% of the number of weights. Other approaches also offer good performance but they are not able to reach the accuracy of a dense network.

Comparison with LDR networks Thomas et al. (2018). We now compare DCNNs with the LDR framework using the network configuration experimented in the original paper: a single LDR structured layer followed by a dense layer. In the LDR framework, we can change the size of a network by adjusting the rank of the residual matrix, effectively capturing matrices with a structure that is close to a known structure but not exactly (e.g. in the LDR framework, Toeplitz matrices

[2] Remark: the numbers may differ from the original experiments by Thomas et al. because we use the original dataset instead of a monochrome version)

Table 1: LDR networks compared with DCNNs on a flattend version of CIFAR-10. DCNNs outperform all LDR configurations with fewer weights.[2]

Architectures	#Params	Acc.
Dense	*9.4M*	*0.562*
DCNN ($5\ layers$)	**49K**	**0.543**
DCNN ($2\ layers$)	**21K**	**0.536**
LDR–TD ($r = 2$)	64K	0.511
LDR–TD ($r = 3$)	70K	0.473
Toeplitz-like ($r = 2$)	46K	0.483
Toeplitz-like ($r = 3$)	52K	0.496

Table 2: Two depths scattering on CIFAR-10 followed by LDR or DC layer. Networks with DC layers outperform all LDR configurations with fewer weights.

Architectures	#Params	Acc.
DC ($1\ layers$)	**124K**	**0.757**
DC ($3\ layers$)	**217K**	**0.785**
Ensemble x5 DC ($3\ layers$)	**1.08M**	**0.811**
LDR-SD ($r = 1$)	140K	0.701
LDR-SD ($r = 10$)	420K	0.728
Toeplitz-like ($r = 1$)	110K	0.711
Toeplitz-like ($r = 10$)	388K	0.720

can be encoded with a residual matrix with rank=2, so a matrix that can be encoded with a residual of rank=3 can be seen as Toeplitz-like.). The results are presented in Table 1 and demonstrate that DCNNs outperforms all LDR networks both in terms in size and accuracy.

Exploiting image features. Dense layers and DCNNs are not designed to capture task-specific features such as the translation invariance inherently useful in image classification. We can further improve the accuracy of such general purpose architectures on image classification without dramatically increasing the number of trained parameters by stacking them on top of fixed (i.e. non-trained) transforms such as the scattering transform (Mallat, 2010). In this section we compare the accuracy of various structured networks, enhanced with the scattering transform, on an image classification task, and run comparative experiments on CIFAR-10.

Our test architecture consists of 2 depth scattering on the RGB images followed by a batch norm and LDR or DC layer. To vary the number of parameters of Scattering+LDR architecture, we increase the rank of the matrix (stacking several LDR matrices quickly exhausted the memory). The Figure 4(b) and 2 shows the accuracy of these architectures given the number of trainable parameters.

First, we can see that the DCNN architecture very much benefits from the scattering transform and is able to reach a competitive accuracy over 78%. We can also see that scattering followed by a DC layer systematically outperforms scattering + LDR or scattering + Toeplitz-like with less parameters.

6.2 Comparison with other compression based approaches (Q6.2)

Table 3: Comparison with compression based approaches

Architecture	Settings	#Params	Error (%)
LeNet Lecun et al. (1998)	-	*4 257 674*	*0.61*
DCNN	**8 DC layers**	**25 620**	**1.74**
	10 DC layers	**31 764**	**1.60**
Fast Food (FF) Yang et al. (2015)	Conv + FF 1024 Softmax layer	38 821	0.71
	Conv + FF 2048 Softmax Layer	52 124	0.71
HashNet Chen et al. (2015)	3 layers, 1/64 compress. factor	46 875	2.79
	5 layers, 1/64 compress. factor	78 125	1.99
Dark Knowledge Hinton et al. (2015)	3 layers, 1/64 compress.factor	46 875	6.32
	5 layers, 1/64 compress. factor	78 125	2.16

We provide a comparison with other compression based approaches such as HashNet Chen et al. (2015), Dark Knowledge Hinton et al. (2015) and Fast Food Transform (FF) Yang et al. (2015). Table 3 shows the test error of DCNN against other know compression techniques on the MNIST datasets. We can observe that DCNN outperform easily HashNet Chen et al. (2015) and Dark Knowledge Hinton et al. (2015) with fewer number of parameters. The architecture with Fast Food (FF) Yang et al. (2015) achieves better performance but with convolutional layers and only 1 Fast Food Layer as the last Softmax layer.

Table 4: This table shows the GAP score for the *YouTube-8M* dataset with DCNNs. We can see a large increase in the score with deeper networks.

Architecture	#Weights	GAP@20
original	*5.7M*	*0.773*
4 DC	25 410 (**0.44**)	0.599
32 DC	122 178 *(2.11)*	0.685
4 DC + 1 FC	4.46M *(77)*	**0.747**

Table 5: This table shows the GAP score for the *YouTube-8M* dataset with different layer represented with our DC decomposition.

Architecture	#Weights	GAP@20
original	*45M*	*0.846*
DBoF with DC	36M (*80*)	0.838
FC with DC	41M (*91*)	**0.845**
MoE with DC	12M (**26**)	0.805

6.3 DCNNs for large-scale video classification on the *YouTube-8M* dataset (Q6.3)

To understand the performance of deep DCNNs on large scale applications, we conducted experiments on the *YouTube-8M* video classification with 3.8 training examples introduced by Abu-El-Haija et al. (2016b). Notice that we favour this experiment over ImageNet applications because modern image classification architectures involve a large number of convolutional layers, and compressing convolutional layers is out of our scope. Also, as mentioned earlier, testing the performance of DCNN architectures mixed with a large number of expressive layers makes little sense.

The *YouTube-8M* includes two datasets describing 8 million labeled videos. Both datasets contain audio and video features for each video. In the first dataset (*aggregated*) all audio and video features have been aggregated every 300 frames. The second dataset (*full*) contains the descriptors for all the frames. To compare the models we use the GAP metric (Global Average Precision) proposed by Abu-El-Haija et al. (2016b). On the simpler *aggregated* dataset we compared off-the-shelf DCNNs with a dense baseline with 5.7M weights. On the full dataset, we designed three new compact architectures based on the state-of-the-art architecture introduced by Abu-El-Haija et al. (2016b).

Experiments on the *aggregated* dataset with DCNNs: We compared DCNNs with a dense baseline with 5.7 millions weights. The goal of this experiment is to discover a good trade-off between depth and model accuracy. To compare the models we use the GAP metric (Global Average Precision) following the experimental protocol in Abu-El-Haija et al. (2016b), to compare our experiments.

Table 4 shows the results of our experiments on the *aggrgated YouTube-8M* dataset in terms of number of weights, compression rate and GAP. We can see that the compression ratio offered by the circulant architectures is high. This comes at the cost of a little decrease of GAP measure. The 32 layers DCNN is 46 times smaller than the original model in terms of number of parameters while having a close performance.

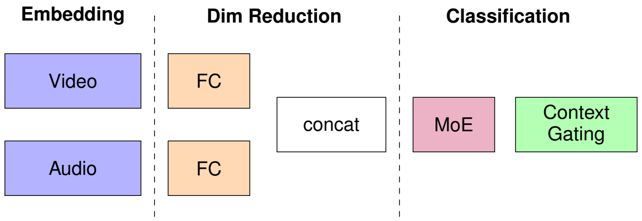


Figure 5: This figure shows the state-of-the-art neural network architecture, initially proposed by Abu-El-Haija et al. (2016b) and later improved by Miech et al. (2017), used in our experiment.

Experiments with DCNNs Deep Bag-of-Frames Architecture: The Deep Bag-of-Frames architecture can be decomposed into three blocks of layers, as illustrated in Figure 5. The first block of layers, composed of the Deep Bag-of-Frames embedding (DBoF), is meant to model an embedding of these frames in order to make a simple representation of each video. A second block of fully connected layers (FC) reduces the dimensionality of the output of the embedding and merges the resulting output with a concatenation operation. Finally, the classification block uses a combination of Mixtures-of-Experts (MoE) Jordan & Jacobs (1993); Abu-El-Haija et al. (2016a) and Context Gating Miech et al. (2017) to calculate the final class probabilities.

Table 5 shows the results in terms of number of weights, size of the model (MB) and GAP on the full dataset, replacing the DBoF block reduces the size of the network without impacting the accuracy. We obtain the best compression ratio by replacing the MoE block with DCNNs (26%) of the size of the original dataset with a GAP score of 0.805 (95% of the score obtained with the original architecture). We conclude that DCNN are both theoretically sound and of practical interest in real, large scale applications.

7 Conclusion

This paper deals with the training of diagonal circulant neural networks. To the best of our knowledge, training such networks with a large number of layers had not been done before. We also endowed this kind of models with theoretical guarantees, hence enriching and refining previous theoretical work from the literature. More importantly, we showed that DCNNs outperform their competing structured alternatives, including the very recent general approach based on LDR networks. Our results suggest that stacking diagonal circulant layers with non linearities improves the convergence rate and the final accuracy of the network. Formally proving these statements constitutes the future directions of this work. As future work, we would like to generalize the good results of DCNNs to convolutions neural networks. We also believe that circulant matrices deserve a particular attention in deep learning because of their strong ties with convolutions: a circulant matrix operator is equivalent to the convolution operator with circular paddings (as shown in [5]). This fact makes any contribution to the area of circulant matrices particularly relevant to the field of deep learning with impacts beyond the problem of designing compact models. As future work, we would like to generalize our results to deep convolutional neural networks.

References

Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Apostol (Paul) Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. In *arXiv:1609.08675*, 2016a.

Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016b.

Alexandre Araujo, Benjamin Negrevergne, Yann Chevaleyre, and Jamal Atif. Training compact deep learning models for video classification using circulant matrices. In *The 2nd Workshop on YouTube-8M Large-Scale Video Understanding at ECCV 2018*, 2018.

Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In *Neurips*, 05 2019.

Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pp. 2654–2662, 2014.

Wenlin Chen, James T. Wilson, Stephen Tyree, Kilian Q. Weinberger, and Yixin Chen. Compressing neural networks with the hashing trick. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 2285–2294. JMLR.org, 2015.

Y. Cheng, F. X. Yu, R. S. Feris, S. Kumar, A. Choudhary, and S. F. Chang. An exploration of parameter redundancy in deep networks with circulant projections. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2857–2865, Dec 2015.

Krzysztof Choromanski, Mark Rowland, Vikas Sindhwani, Richard E. Turner, and Adrian Weller. Structured evolution with compact architectures for scalable policy optimization. In *ICML*, 2018. URL https://arxiv.org/pdf/1804.02395.pdf.

Misha Denil, Babak Shakibi, Laurent Dinh, Marc' Aurelio Ranzato, and Nando de Freitas. Predicting parameters in deep learning. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26*, pp. 2148–2156. Curran Associates, Inc., 2013.

S. Goyal, A. Roy Choudhury, and V. Sharma. Compression of deep neural networks by combining pruning and low rank decomposition. In *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 952–958, 2019. doi: 10.1109/IPDPSW.2019.00162.

Aicke Hinrichs and Jan Vybíral. Johnson-lindenstrauss lemma for circulant matrices. *Random Structures & Algorithms*, 39(3):391–398, 2011.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

Marko Huhtanen and Allan Perämäki. Factoring matrices into the product of circulant and diagonal matrices. *Journal of Fourier Analysis and Applications*, 21(5):1018–1033, Oct 2015. ISSN 1531-5851. doi: 10.1007/s00041-015-9395-0.

M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the em algorithm. In *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*, volume 2, pp. 1339–1344 vol.2, Oct 1993. doi: 10.1109/IJCNN.1993.716791.

Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtarik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. In *NIPS Workshop on Private Multi-Party Machine Learning*, 2016. URL https://arxiv.org/abs/1610.05492.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Chong Li and C. J. Richard Shi. Constrained optimization based low-rank approximation of deep neural networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Computer Vision – ECCV 2018*, pp. 746–761, Cham, 2018. Springer International Publishing.

Stéphane Mallat. Recursive interferometric representation. In *Proc. of EUSICO conference, Danemark*, 2010.

Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with context gating for video classification. *CoRR*, abs/1706.06905, 2017.

Marcin Moczulski, Misha Denil, Jeremy Appleyard, and Nando de Freitas. Acdc: A structured efficient linear layer. *arXiv preprint arXiv:1511.05946*, 2015.

Jörn Müller-Quade, Harald Aagedal, Th Beth, and Michael Schmid. Algorithmic design of diffractive optical systems for information processing. *Physica D: Nonlinear Phenomena*, 120(1-2):196–205, 1998.

Alexander Novikov, Dmitrii Podoprikhin, Anton Osokin, and Dmitry P Vetrov. Tensorizing neural networks. In *Advances in Neural Information Processing Systems*, pp. 442–450, 2015.

Tara Sainath and Carolina Parada. Convolutional neural networks for small-footprint keyword spotting. In *Interspeech*, 2015.

Victoria Sanchez, Pedro Garcia, Antonio M Peinado, José C Segura, and Antonio J Rubio. Diagonalizing properties of the discrete cosine transforms. *IEEE transactions on Signal Processing*, 43(11): 2631–2641, 1995.

Michael Schmid, Rainer Steinwandt, Jörn Müller-Quade, Martin Rötteler, and Thomas Beth. Decomposing a matrix into circulant and diagonal factors. *Linear Algebra and its Applications*, 306(1-3): 131–143, 2000.

Hanie Sedghi, Vineet Gupta, and Philip Long. The singular values of convolutional layers. In *ICLR*, 2018.

Anna Thomas, Albert Gu, Tri Dao, Atri Rudra, and Christopher Ré. Learning compressed transforms with low displacement rank. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 9066–9078. Curran Associates, Inc., 2018.

Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J. Pal. Deep complex networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

Z. Yang, M. Moczulski, M. Denil, N. d. Freitas, A. Smola, L. Song, and Z. Wang. Deep fried convnets. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1476–1483, Dec 2015. doi: 10.1109/ICCV.2015.173.

X. Yu, T. Liu, X. Wang, and D. Tao. On compressing deep models by low rank and sparse decomposition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 67–76, July 2017. doi: 10.1109/CVPR.2017.15.

Liang Zhao, Siyu Liao, Yanzhi Wang, Zhe Li, Jian Tang, and Bo Yuan. Theoretical properties for neural networks with weight matrices of low displacement rank. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 4082–4090, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

SUPPLEMENTAL MATERIAL – UNDERSTANDING AND TRAINING DEEP DIAGONAL CIRCULANT NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

1 NOTATIONS & DEFINITION

We note $\Re(z)$ and $\Im(z)$ the real and imaginary parts the complex number z. We note $(\cdot)_t$ is the t^{th} component of a vector. Let $\mathbf{i}$ be the imaginary number defined by $\mathbf{i}^2 = -1$. Define $\mathbf{1}_n$ as the n-vector of ones. Also, we note $[n] = \{1, \dots, n\}$. The rectified linear unit on the complex domain is defined by $ReLU(z) = \max(0, \Re(z)) + \mathbf{i}\max(0, \Im(z))$. The notation $|\cdot|$ refers to the complex modulus. Finally, define the *cyclic shift* matrix $S \in \mathbb{R}^{n\times n}$ as follows:

$$S = \begin{bmatrix} 0 & & & & 1 \\ 1 & 0 & & & \\ & 1 & \ddots & & \\ & & \ddots & 0 & \\ & & & 1 & 0 \end{bmatrix}$$

We introduce some necessary definitions regarding neural networks.

Definition 1 (Deep ReLU network)**.** *Given L weight matrices $W = (W_1, \dots, W_L)$ with $W_i \in \mathbb{C}^{n\times n}$ and L bias vectors $b = (b_1, \dots, b_L)$ with $b_i \in \mathbb{C}^n$, a* deep ReLU network *is a function $f_{W_L,b_L} : \mathbb{C}^n \to \mathbb{C}^n$ such that $f_{W,b}(x) = (f_{W_L,b_L} \circ \dots \circ f_{W_1,b_1})(x)$ where $f_{W_i,b_i}(x) = \phi(W_i x + b_i)$ and $\phi(.)$ is a ReLU non-linearity [1] In the rest of this paper, we call L and n respectively the depth and the width of the network. Moreover, we call* total rank *k, the sum of the ranks of the matrices $W_1 \dots W_L$. i.e. $k = \sum_{i=1}^{L} rank(W_i)$.*

In the rest of this paper, we call L and n respectively the depth and the width of the network. Moreover, we call *total rank* k, the sum of the ranks of the matrices $W_1 \dots W_L$. i.e. $k = \sum_{i=1}^{L} rank(W_i)$.

2 PROOFS OF SECTION 3

Theorem 1. *(Reformulation Huhtanen & Perämäki (2015)) For any given matrix $A \in \mathbb{C}^{n\times n}$, for any $\epsilon > 0$, there exists a sequence of matrices $B_1 \dots B_{2n-1}$ where B_i is a circulant matrix if i is odd, and a diagonal matrix otherwise, such that $\|B_1 B_2 \dots B_{2n-1} - A\| < \epsilon$. Moreover, if A can be decomposed as $A = \sum_{i=1}^{k} D_i S^{i-1}$ where S is the cyclic-shift matrix and $D_1 \dots D_k$ are diagonal matrices, then A can be written as a product $B_1 B_2 \dots B_{2k-1}$ where B_i is a circulant matrix if i is odd, and a diagonal matrix otherwise.*

Theorem 2. *(Rank-based circulant decomposition) Let $A \in \mathbb{C}^{n\times n}$ be a matrix of rank at most k. Assume that n can be divided by k. For any $\epsilon > 0$, there exists a sequence of $4k + 1$ matrices $B_1, \dots, B_{4k+1}$, where B_i is a circulant matrix if i is odd, and a diagonal matrix otherwise, such that $\|B_1 B_2 \dots B_{4k+1} - A\| < \epsilon$*

Proof. *(Theorem 2)* Let $U\Sigma V^T$ be the SVD decomposition of M where U, V and Σ are $n \times n$ matrices. Because M is of rank k, the last $n - k$ columns of U and V are null. In the following, we

[1] Because our networks deal with complex numbers, we use an extension of the ReLU function to the complex domain. The most straightforward extension defined in Trabelsi et al. (2018) is as follows: $\mathrm{ReLU}(z) = \mathrm{ReLU}(\Re(z)) + i\mathrm{ReLU}(\Im(z))$, where $\Re$ and $\Im$ refer to the real and imaginary parts of z.

will first decompose U into a product of matrices WRO, where R and O are respectively circulant and diagonal matrices, and W is a matrix which will be further decomposed into a product of diagonal and circulant matrices. Then, we will apply the same decomposition technique to V. Ultimately, we will get a product of $4k + 2$ matrices alternatively diagonal and circulant.

Let $R = circ(r_1 \dots r_n)$. Let O be a $n \times n$ diagonal matrix where $O_{i,i} = 1$ if $i \leq k$ and 0 otherwise. The k first columns of the product RO will be equal to that of R, and the $n - k$ last colomns of RO will be zeros. For example, if $k = 2$, we have:

$$RO = \begin{pmatrix} r_1 & r_n & 0 & \cdots & 0 \\ r_2 & r_1 & & & \\ r_3 & r_2 & \vdots & & \vdots \\ \vdots & \vdots & & & \\ r_n & r_{n-1} & 0 & \cdots & 0 \end{pmatrix}$$

Let us define k diagonal matrices $D_i = diag(d_{i1} \dots d_{in})$ for $i \in [k]$. For now, the values of d_{ij} are unknown, but we will show how to compute them. Let $W = \sum_{i=1}^{k} D_i S^{i-1}$. Note that the $n - k$ last columns of the product WRO will be zeros. For example, with $k = 2$, we have:

$$W = \begin{bmatrix} d_{1,1} & & & & d_{2,1} \\ d_{2,2} & d_{1,2} & & & \\ & d_{2,3} & \ddots & & \\ & & \ddots & & \\ & & & d_{2,n} & d_{1,n} \end{bmatrix}$$

$$WRO = \begin{pmatrix} r_1 d_{11} + r_n d_{21} & r_n d_{11} + r_{n-1} d_{21} & 0 & \cdots & 0 \\ r_2 d_{12} + r_1 d_{22} & r_1 d_{12} + r_n d_{22} & & & \\ & & \vdots & & \vdots \\ \vdots & \vdots & & & \\ r_n d_{1n} + r_{n-1} d_{2n} & r_{n-1} d_{1n} + r_{n-2} d_{2n} & 0 & \cdots & 0 \end{pmatrix}$$

We want to find the values of d_{ij} such that $WRO = U$. We can formulate this as linear equation system. In case $k = 2$, we get:

$$\begin{pmatrix} r_n & r_1 & & & & & & \\ r_{n-1} & r_n & & & & & & \\ & & r_1 & r_2 & & & & \\ & & r_n & r_1 & & & & \\ & & & & r_2 & r_3 & & \\ & & & & r_1 & r_2 & & \\ & & & & & & \ddots & \\ & & & & & & & \ddots \end{pmatrix} \times \begin{pmatrix} d_{2,1} \\ d_{1,1} \\ d_{2,2} \\ d_{1,2} \\ d_{2,3} \\ d_{1,3} \\ \vdots \\ \vdots \end{pmatrix} = \begin{pmatrix} U_{1,1} \\ U_{1,2} \\ U_{2,1} \\ U_{2,2} \\ \\ \\ \vdots \\ \\ \end{pmatrix}$$

The i^{th} bloc of the bloc-diagonal matrix is a Toeplitz matrix induced by a subsequence of length k of $(r_1, \dots r_n, r_1 \dots r_n)$. Set $r_j = 1$ for all $j \in \{k, 2k, 3k, \dots n\}$ and set $r_j = 0$ for all other values of j. Then it is easy to see that each bloc is a permutation of the identity matrix. Thus, all blocs are invertible. This entails that the block diagonal matrix above is also invertible. So by solving this set of linear equations, we find $d_{1,1} \dots d_{k,n}$ such that $WRO = U$. We can apply the same idea to factorize $V = W'.R.O$ for some matrix W'. Finally, we get

$$A = U \Sigma V^T = WRO \Sigma O^T R^T W'^T$$

Thanks to Theorem 1, W and W' can both be factorized in a product of $2k - 1$ circulant and diagonal matrices. Note that $O \Sigma O^T$ is diagonal, because all three are diagonal. Overall, A can be represented with a product of $4k + 2$ matrices, alternatively diagonal and circulant. □

3 Proofs of Section 4

Lemma 1. *Let $W_L, \ldots W_1 \in \mathbb{C}^{n\times n}$, $b \in \mathbb{C}^n$ and let $\mathcal{X} \subset \mathbb{C}^n$ be a bounded set. There exists $\beta_L \ldots \beta_1 \in \mathbb{C}^n$ such that for all $x \in \mathcal{X}$ we have $f_{W_L,\beta_L} \circ \ldots \circ f_{W_1,\beta_1}(x) = ReLU\left(W_L W_{L-1} \ldots W_1 x + b\right)$.*

Proof. (Lemma 1) Define $S = \left\{ \left(\left(\prod_{k=1}^{j} W_k \right) x \right)_t : x \in \mathcal{X}, t \in [n], j \in [L] \right\}$. Let $\Omega = \max\left\{\Re(v) : v \in S\right\} + \mathbf{i} \max\left\{\Im(v) : v \in S\right\}$. Intuitively, the real and imaginary parts of Ω are the largest any activation in the network can have. Define $h_j(x) = W_j x + \beta_j$. Let $\beta_1 = \Omega \mathbf{1}_n$. Clearly, for all $x \in \mathcal{X}$ we have $h_1(x) \geq 0$, so $ReLU \circ h_1(x) = h_1(x)$. More generally, for all $j < n-1$ define $\beta_{j+1} = \mathbf{1}_n \Omega - W_{j+1}\beta_j$. It is easy to see that for all $j < n$ we have $h_j \circ \ldots \circ h_1(x) = W_j W_{j-1} \ldots W_1 x + \mathbf{1}_n \Omega$. This guarantees that for all $j < n$, $h_j \circ \ldots \circ h_1(x) = ReLU \circ h_j \circ \ldots \circ ReLU \circ h_1(x)$. Finally, define $\beta_L = b - A_L \beta_{L-1}$. We have, $ReLU \circ h_L \circ \ldots \circ ReLU \circ h_1(x) = ReLU\left(W_j \ldots W_1 x + b\right)$. □

Lemma 2. *Let $\mathcal{N}$ be a deep ReLU network of width n and depth L, and let $\mathcal{X} \subset \mathbb{C}^n$ be a bounded set. For any $\epsilon > 0$, there exists a DCNN $\mathcal{N}'$ of width n and of depth $(2n-1)L$ such that $\|\mathcal{N}(x) - \mathcal{N}'(x)\| < \epsilon$ for all $x \in \mathcal{X}$.*

Proof. (Lemma 2) Assume $\mathcal{N} = f_{W_L,b_L} \circ \ldots \circ f_{W_1,b_1}$. By theorem 1, for any $\epsilon' > 0$, any matrix W_i, there exists a sequence of $2n-1$ matrices $C_{i,n}D_{i,n-1}C_{i,n-1}\ldots D_{i,1}C_{i,1}$ such that $\left\| \prod_{j=0}^{n-1} D_{i,n-j} C_{i,n-j} - W_i \right\| < \epsilon'$, where $D_{i,1}$ is the identity matrix. By lemma 1, we know that there exists $\{\beta_{ij}\}_{i\in[L], j\in[n]}$ such that for all $i \in [L]$, $f_{D_{in}C_{in},\beta_{in}} \circ \ldots \circ f_{D_{i1}C_{i1},\beta_{i1}}(x) = ReLU\left(D_{in}C_{in} \ldots C_{i1}x + b_i\right)$.

Now if ϵ' tends to zero, $\| f_{D_{in}C_{in},\beta_{in}} \circ \ldots \circ f_{D_{i1}C_{i1},\beta_{i1}} - ReLU\left(W_i x + b_i\right) \|$ will also tend to zero for any $x \in \mathcal{X}$, because the ReLU function is continuous and $\mathcal{X}$ is bounded. Let $\mathcal{N}' = f_{D_{1n}C_{1n},\beta_{1n}} \circ \ldots \circ f_{D_{i1}C_{i1},\beta_{i1}}$. Again, because all functions are continuous, for all $x \in \mathcal{X}$, $\|\mathcal{N}(x) - \mathcal{N}'(x)\|$ tends to zero as ϵ' tends to zero. □

Corollary 1. *Bounded width DCNNs are universal approximators in the following sense: for any continuous function $f : [0,1]^n \to \mathbb{R}_+$ of bounded supremum norm, for any $\epsilon > 0$, there exists a DCNN $\mathcal{N}_\epsilon$ of width $n+3$ such that $\forall x \in [0,1]^{n+3}$, $\left|f(x_1 \ldots x_n) - \left(\mathcal{N}_\epsilon(x)\right)_1\right| < \epsilon$, where $(\cdot)_i$ represents the i^{th} component of a vector.*

Proof. (Corollary 1) It has been shown recently in Hanin (2017) that for any continuous function $f : [0,1]^n \to \mathbb{R}_+$ of bounded supremum norm, for any $\epsilon > 0$, there exists a dense neural network $\mathcal{N}$ with an input layer of width n, an output layer of width 1, hidden layers of width $n+3$ and ReLU activations such that $\forall x \in [0,1]^n$, $|f(x) - \mathcal{N}(x)| < \epsilon$. From $\mathcal{N}$, we can easily build a deep ReLU network $\mathcal{N}'$ of width exactly $n+3$, such that $\forall x \in [0,1]^{n+3}$, $\left|f(x_1 \ldots x_n) - \left(\mathcal{N}'(x)\right)_1\right| < \epsilon$. Thanks to lemma 2, this last network can be approximated arbitrarily well by a DCNN of width $n+3$. □

Theorem 3. *(Rank-based expressive power of diagonal circulant neural networks)*
Let $\mathcal{N} : f_{W_L,b_L} \circ \ldots \circ f_{W_1,b_1}$ be a deep ReLU network of width n, depth L and a total rank k. Assume n is a power of 2. Let $\mathcal{X} \subset \mathbb{C}^n$ be a bounded set. For any $\epsilon > 0$, there exists a DCNN $\mathcal{N}'$ of width n such that $\|\mathcal{N}(x) - \mathcal{N}'(x)\| < \epsilon$ for all $x \in \mathcal{X}$. In addition, the depth of $\mathcal{N}'$ is bounded by $9k$. Moreover, if the rank of each matrix A_i divides n, then the depth of $\mathcal{N}'$ is bounded by $L + 4k$.

Proof. (Theorem 3) Let $k_1 \ldots k_L$ be the ranks of matrices $W_1 \ldots W_L$, which are n-by-n matrices. For all i, there exists $k'_i \in \{k_i \ldots 2k_i\}$ such that k'_i is a power of 2. Due to the fact that n is also a power of 2, k'_i divides n. By theorem 2, for all i each matrix W_i can be decomposed as an alternating product of diagonal-circulant matrices $B_{i,1} \ldots B_{i,4k'_i+1}$ such that $\left\| W_i - B_{i,1} \times \ldots \times B_{i,4k'_i+1} \right\| < \epsilon$. Using the exact same technique as in lemma 2, we can build a DCNN $\mathcal{N}'$ using matrices $B_{1,1} \ldots B_{L,4k'_L+1}$, such that $\|\mathcal{N}(x) - \mathcal{N}'(x)\| < \epsilon$ for all $x \in \mathcal{X}$. The total number of layers is $\sum_i (4k'_i + 1) \leq L + 8\sum_i k_i \leq L + 8$.total rank ≤ 9.total rank.

□

Finally, what if we choose to use small depth networks to approximate deep ReLU networks where matrices are not of low rank? To answer this question, we first need to show the negative impact of replacing matrices by their low rank approximators in neural networks:

Proposition 4. *Let $\mathcal{N} = f_{W_L,b_L} \circ \ldots \circ f_{W_1,b_1}$ be a Deep ReLU network, where $W_i \in \mathbb{C}^{n\times n}, b_i \in \mathbb{C}^n$ for all $i \in [L]$. Let $\tilde{W}_i$ be the matrix obtained by an SVD approximation of rank k of matrix W_i. Let $\sigma_{i,j}$ be the j^{th} singular value of W_i. Define $\tilde{\mathcal{N}} = f_{\tilde{W}_L,b_L} \circ \ldots \circ f_{\tilde{W}_1,b_1}$. Then, for any $x \in \mathbb{C}^n$, we have:*

$$\left\|\mathcal{N}(x) - \tilde{\mathcal{N}}(x)\right\| \leq \frac{\left(\sigma_{max,1}^L - 1\right) R\sigma_{max,k}}{\sigma_{max,1} - 1}$$

where R is an upper bound on norm of the output of any layer in $\mathcal{N}$, and $\sigma_{max,j} = \max_i \sigma_{i,j}$.

Proof. (Proposition 4) Let $x_0 \in \mathbb{C}^n$ and $\tilde{x}_0 = x_0$. For all $i \in [L]$, define $x_i = ReLU\left(W_i x_{i-1} + b\right)$ and $\tilde{x}_i = ReLU\left(\tilde{W}_i \tilde{x}_{i-1} + b\right)$. By lemma 3, we have

$$\left\|x_i - \tilde{x}_i\right\| \leq \sigma_{i,k+1}\left\|x_{i-1}\right\| + \sigma_{i,1}\left\|x_{i-1} - \tilde{x}_{i-1}\right\|$$

Observe that for any sequence $a_0, a_1 \ldots$ defined recurrently by $a_0 = 0$ and $a_i = ra_{i-1} + s$, the recurrence relation can be unfold as follows: $a_i = \frac{s\left(r^i - 1\right)}{r-1}$. We can apply this formula to bound our error as follows:

$$\left\|x_l - \tilde{x}_l\right\| \leq \frac{\left(\sigma_{max,1}^l - 1\right)\sigma_{max,k}\max_i\left\|x_i\right\|}{\sigma_{max,1} - 1}$$

□

Lemma 3. *Let $W \in \mathbb{C}^{n\times n}$ with singular values $\sigma_1 \ldots \sigma_n$, and let $x, \tilde{x} \in \mathbb{C}^n$. Let $\tilde{W}$ be the matrix obtained by a SVD approximation of rank k of matrix W. Then we have:*

$$\left\|ReLU\left(Wx + b\right) - ReLU\left(\tilde{W}\tilde{x} + b\right)\right\| \leq \sigma_{k+1}\left\|x\right\| + \sigma_1\left\|\tilde{x} - x\right\|$$

Proof. (Lemma 3) Recall that $\left\|W\right\|_2 = \sup_z \frac{\|Wz\|_2}{\|z\|_2} = \sigma_1 = \left\|\tilde{W}\right\|_2$, because σ_1 is the greatest singular value of both W and $\tilde{W}$. Also, note that $\left\|W - \tilde{W}\right\|_2 = \sigma_{k+1}$. Let us bound the formula without ReLUs:

$$\begin{aligned}\left\|\left(Wx + b\right) - \left(\tilde{W}\tilde{x} + b\right)\right\| &= \left\|\left(Wx + b\right) - \left(\tilde{W}\tilde{x} + b\right)\right\| \\ &= \left\|Wx - \tilde{W}x - \tilde{W}\left(\tilde{x} - x\right)\right\| \\ &\leq \left\|\left(W - \tilde{W}\right)x\right\| + \left\|\tilde{W}\right\|_2\left\|\tilde{x} - x\right\| \\ &\leq \left\|x\right\|\sigma_{k+1} + \sigma_1\left\|\tilde{x} - x\right\|\end{aligned}$$

Finally, it is easy to see that for any pair of vectors $a, b \in \mathbb{C}^n$, we have $\|ReLU(a) - ReLU(b)\| \leq \|a - b\|$. This concludes the proof. □

Corollary 2. *Consider any deep ReLU network $\mathcal{N} = f_{W_L,b_L} \circ \ldots \circ f_{W_1,b_1}$ of depth L and width n. Let $\sigma_{max,j} = \max_i \sigma_{i,j}$ where $\sigma_{i,j}$ is the j^{th} singular value of W_i. Let $\mathcal{X} \subset \mathbb{C}^n$ be a bounded set. Let k be an integer dividing n. There exists a DCNN $\mathcal{N}' = f_{D_m C_m, b'_m} \circ \ldots \circ f_{D_1 C_1, b'_1}$ of width n and of depth $m = L(4k + 1)$, such that for any $x \in \mathcal{X}$:*

$$\left\|\mathcal{N}(x) - \mathcal{N}'(x)\right\| < \frac{\left(\sigma_{max,1}^L - 1\right) R\sigma_{max,k}}{\sigma_{max,1} - 1}$$

where R is an upper bound on the norm of the outputs of each layer in $\mathcal{N}$.

Proof. (Corollary 2) Let $\tilde{\mathcal{N}} = f_{\tilde{W}_L, b_L} \circ \ldots \circ f_{\tilde{W}_1, b_1}$, where each $\tilde{W}_i$ is the matrix obtained by an SVD approximation of rank k of matrix W_i. With Proposition 4, we have an error bound on $\|\mathcal{N}(x) - \tilde{\mathcal{N}}(x)\|$. Now each matrix $\tilde{W}_i$ can be replaced by a product of k diagonal-circulant matrices. By theorem 3, this product yields a DCNN of depth $m = L(4k+1)$, strictly equivalent to $\tilde{\mathcal{N}}$ on $\mathcal{X}$. The result follows. □

4 Proof of Section 5

Proposition 5. *Let $\mathcal{N}$ be a DCNN of depth L initialized according to our procedure, with $\sigma' = 0$. Assume that all layers 1 to $L-1$ have ReLU activation functions, and that the last layer has the identity activation function. Then, for any $x \in \mathbb{R}^n$, the covariance matrix of $\mathcal{N}(x)$ is $\frac{2.Id}{n} \|x\|_2^2$. Moreover, note that this covariance does not depend on the depth of the network.*

Proof. (Proposition 5) Let $\mathcal{N} = f_{D_L, C_L} \circ \ldots \circ f_{D_1, C_1}$ be a L layer DCNN. All matrices are initialized as described in the statement of the proposition. Let $y = D_1 C_1 x$. Lemma 4 shows that $cov(y_i, y_{i'}) = 0$ for $i \neq i'$ and $var(y_i) = \frac{2}{n} \|x\|_2^2$. For any $j \leq L$, define $z^j = f_{D_j, C_j} \circ \ldots \circ f_{D_1, C_1}(x)$. By a recursive application of lemma 4, we get that then $cov(z_i^j, z_{i'}^j) = 0$ and $var(z_i^j) = \frac{2}{n} \|x\|_2^2$. □

Lemma 4. *Let $c_1 \ldots c_n, d_1 \ldots d_n, b_1 \ldots b_n$ be random variables in $\mathbb{R}$ such that $c_i \sim \mathcal{N}(0, \sigma^2)$, $b_i \sim \mathcal{N}(0, \sigma'^2)$ and $d_i \sim \{-1, 1\}$ uniformly. Define $C = circ(c_1 \ldots c_n)$ and $D = diag(d_1 \ldots d_n)$. Define $y = DCu$ and $z = CDu$ for some vector u in $\mathbb{R}^n$. Also define $\bar{y} = y + b$ and $\bar{z} = z + b$. Then, for all i, the p.d.f. of y_i, $\bar{y}_i$, z_i and $\bar{z}_i$ are symmetric. Also:*

- *Assume $u_1 \ldots u_n$ is fixed. Then, we have for $i \neq i'$:*

$$cov(y_i, y_{i'}) = cov(z_i, z_{i'}) = cov(\bar{y}_i, \bar{y}_{i'}) = cov(\bar{z}_i, \bar{z}_{i'}) = 0$$
$$var(y_i) = var(z_i) = \sum_j u_j^2 \sigma^2$$
$$var(\bar{y}_i) = var(\bar{z}_i) = \sigma'^2 + \sum_j u_j^2 \sigma^2$$

- *Let $x_1 \ldots x_n$ be random variables in $\mathbb{R}$ such that the p.d.f. of x_i is symmetric for all i, and let $u_i = ReLU(x_i)$. We have for $i \neq i'$:*

$$cov(y_i, y_{i'}) = cov(z_i, z_{i'}) = cov(\bar{y}_i, \bar{y}_{i'}) = cov(\bar{z}_i, \bar{z}_{i'}) = 0$$
$$var(y_i) = var(z_i) = \frac{1}{2} \sum_j var(x_i).\sigma^2$$
$$var(\bar{y}_i) = var(\bar{z}_i) = \sigma'^2 + \frac{1}{2} \sum_j var(x_i).\sigma^2$$

Proof. (Lemma 4) By an abuse of notation, we write $c_0 = c_n, c_{-1} = c_{n-1}$ and so on. First, note that: $y_i = \sum_{j=1}^n c_{j-i} u_j d_j$ and $z_i = \sum_{j=1}^n c_{j-i} u_j d_i$. Observe that each term $c_{j-i} u_j d_j$ and $c_{j-i} u_j d_i$ have symmetric p.d.f. because of d_i and d_j. Thus, y_i and z_i have symmetric p.d.f. Now let us compute the covariance.

$$\begin{aligned} cov(y_i, y_{i'}) &= \sum_{j,j'=1}^n cov\left(c_{j-i} u_j d_j, c_{j'-i'} u_{j'} d_{j'}\right) \\ &= \sum_{j,j'=1}^n \mathbb{E}\left[c_{j-i} u_j d_j c_{j'-i'} u_{j'} d_{j'}\right] - \mathbb{E}\left[c_{j-i} u_j d_j\right] \mathbb{E}\left[c_{j'-i'} u_{j'} d_{j'}\right] \end{aligned}$$

Observe that $\mathbb{E}\left[c_{j-i} u_j d_j\right] = \mathbb{E}\left[c_{j-i} u_j\right] \mathbb{E}\left[d_j\right] = 0$ because d_j is independent from $c_{j-i} u_j$. Also, observe that if $j \neq j'$ then $\mathbb{E}\left[d_j d_{j'}\right] = 0$ and thus $\mathbb{E}\left[c_{j-i} u_j d_j c_{j'-i'} u_{j'} d_{j'}\right] =$

$\mathbb{E}[d_j d_{j'}]\,\mathbb{E}[c_{j-i}u_j c_{j'-i'}u_{j'}] = 0$. Thus, the only non null terms are those for which $j = j'$. We get:

$$
\begin{aligned}
cov(y_i, y_{i'}) &= \sum_{j=1}^{n} \mathbb{E}[c_{j-i}u_j d_j c_{j-i'}u_j d_j] \\
&= \sum_{j=1}^{n} \mathbb{E}\left[c_{j-i}c_{j-i'}u_j^2\right]
\end{aligned}
$$

Assume u is a fixed vector. Then, $var(y_i) = \sum_{j=1}^{n} u_j^2 \sigma^2$ and $cov(y_i, y_{i'}) = 0$ for $i \neq i'$ because c_{j-i} is independent from $c_{j-i'}$.

Now assume that $u_j = ReLU(x_j)$ where x_j is a r.v. Clearly, u_j^2 is independent from c_{j-i} and $c_{j-i'}$. Thus:

$$
cov(y_i, y_{i'}) = \sum_{j=1}^{n} \mathbb{E}[c_{j-i}c_{j-i'}]\,\mathbb{E}\left[u_j^2\right]
$$

For $i \neq i'$, then c_{j-i} and $c_{j-i'}$ are independent, and thus $\mathbb{E}[c_{j-i}c_{j-i'}] = \mathbb{E}[c_{j-i}]\,\mathbb{E}[c_{j-i'}] = 0$. Therefore, $cov(y_i, y_{i'}) = 0$ if $i \neq i'$. Let us compute the variance. We get $var(y_i) = \sum_{j=1}^{n} var(c_{j-i}).\mathbb{E}\left[u_j^2\right]$. Because the p.d.f. of x_j is symmetric, $\mathbb{E}\left[x_j^2\right] = 2\mathbb{E}\left[u_j^2\right]$ and $\mathbb{E}[x_j] = 0$. Thus, $var(y_i) = \frac{1}{2}\sum_{j=1}^{n} var(c_{j-i}).\mathbb{E}\left[x_j^2\right] = \frac{1}{2}\sum_{j=1}^{n} var(c_{j-i}).var(x_j)$.

Finally, note that $cov(\bar{y}_i, \bar{y}_{i'}) = cov(y_i, y_{i'}) + cov(b_i, b_{i'})$. This yields the covariances of $\bar{y}$.

To derive $cov(z_i, z_{i'})$ and $cov(\bar{z}_i, \bar{z}_{i'})$, the required calculus is nearly identical. We let the reader check by himself/herself. □

5 Additional Information on the Empirical Evaluation

Architectures & Hyper-Parameters: For the first set of our experiments (e.g. experiments on CIFAR-10), we train all networks for 200 epochs, a batch size of 200, Leaky ReLU activation with a different slope. We minimize the Cross Entropy Loss with Adam optimizer and use a piecewise constant learning rate of $5 \times 10e-5$, $2.5 \times 10e-5$, $5 \times 10e-6$ and $1 \times 10e-6$ after respectively 40000, 60000 and 80000 steps.

For the *YouTube-8M* dataset experiments, we built a neural network based on the state-of-the-art architecture initially proposed by Abu-El-Haija et al. (2016) and later improved by Miech et al. (2017). Remark that no convolution layer is involved in this application since the input vectors are embeddings of video frames processed using state-of-the-art convolutional neural networks trained on ImageNet.

We trained our models with the CrossEntropy loss and used Adam optimizer with a 0.0002 learning rate and a 0.8 exponential decay every 4 million examples. All fully connected layers are composed of 512 units. DBoF, NetVLAD and NetFV are respectively 8192, 64 and 64 of cluster size for video frames and 4096, 32, 32 for audio frames. We used 4 mixtures for the MoE Layer. We used all the available 300 frames for the DBoF embedding. In order to stabilize and accelerate the training, we used batch normalization before each non linear activation and gradient clipping.

References

Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

B. Hanin. Universal Function Approximation by Deep Neural Nets with Bounded Width and ReLU Activations. *ArXiv e-prints*, August 2017.

Marko Huhtanen and Allan Perämäki. Factoring matrices into the product of circulant and diagonal matrices. *Journal of Fourier Analysis and Applications*, 21(5):1018–1033, Oct 2015. ISSN 1531-5851. doi: 10.1007/s00041-015-9395-0.

Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with context gating for video classification. *CoRR*, abs/1706.06905, 2017.

Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J. Pal. Deep complex networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.