Representation Costs of Linear Neural Networks: Analysis and Design

Anonymous Author(s)
Affiliation
Address
email

Abstract

For different parameterizations (mappings from parameters to predictors), we study the regularization cost in predictor space induced by l_2 regularization on the parameters (weights). We focus on linear neural networks as parameterizations of linear predictors. We identify the representation cost of certain sparse linear ConvNets and residual networks. In order to get a better understanding of how the architecture and parameterization affect the representation cost, we also study the reverse problem, identifying which regularizers on linear predictors (e.g., l_p norms, group norms, the k-support-norm, elastic net) can be the representation cost induced by simple l_2 regularization, and designing the parameterizations that do so.

1 Introduction

In a class of parameterized models, penalizing the l_2 norm of the parameters induces regularization on function space which can be interpreted as a complexity measure on the class of learned functions. In this paper, we study how different parameterizations induce different complexity measures.

We consider parameterized mappings $f : \mathcal{X} \times \mathbb{R}^p \longrightarrow \mathbb{R}^m$, from input $x \in \mathcal{X}$ and parameters $w \in \mathbb{R}^p$ to predictions $f(x; w)$. We denote the predictor implemented with parameters w by $F(w) : \mathcal{X} \longrightarrow \mathbb{R}^m$ defined as $F(w)(x) := f(x; w)$. Then $\mathrm{image}(F)$ is the set of functions from $\mathcal{X}$ to $\mathbb{R}$ which can be obtained from this class of parameterized models.

The l_2 regularization on parameters, either explicitly or implicitly, is a common phenomenon. As an example, in deep learning, explicit l_2 regularization on parameters (a.k.a. weight decay) improves generalization ([13, 26]). Implicit regularization of l_2 norm of parameters appears when we use gradient descent (GD) to train the model ([1, 25, 24, 12, 16, 23, 8, 18, 6, 11]). In particular, GD on homogeneous neural networks with logistic loss implicitly regularizes l_2 norm on weights [16, 23].

The representation cost ([9, 22, 20]) of a function g in $\mathrm{image}(F)$ under the parametrization F is

$$R_F(g) = \min\{\|w\|_2^2 : F(w) = g\}. \tag{1}$$

Consider learning a predictor $F(w)$ with some loss function $L(\cdot)$ while controlling the l_2 norm of parameters w by minimizing

$$\min_{w \in \mathbb{R}^p} L(F(w)) + \lambda \|w\|_2^2 . \tag{2}$$

This is clearly equivalent to learning a function g in $\mathrm{image}(F)$ by controlling $R_F(g)$ defined in Eq. (1):

$$\min_{g \in \mathrm{image}(F)} L(g) + \lambda R_F(g). \tag{3}$$

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Architectures (of depth d)	Induced Complexity Measures
Fully Connected Network with multiple outputs	Schatten $2/d$ norm
Diagonal Network with multiple outputs	Matrix $l_{2/d,2}$ norm
Convolutional Network with multiple outputs	Matrix $l_{2/d,2}$ norm on Fourier domain
Residual Network	An interpolation between two norms

Table 1: **From Architecture to Induced Complexity Measure**

Induced Complexity Measure	Conditions	Architectures
l_p norms	if and only if $2/p \in \mathbb{N}$	Diagonal networks
k-support norms	if and only if $k \in [n]$	k-balanced networks (Fig. 1a)
$l_{p,q}$ group norms	if $2/p, 2/q \in \mathbb{N}$ and $2/p \geq 2/q - 1$	Group networks (Fig. 1)
Elastic nets	None	None
$l_{p,q}$ with overlapping groups	None	None

Table 2: **From Induced Complexity Measure to Architecture**

In other words, the representation cost under the parameterization F, $R_F(\cdot)$, captures the regularization on function space $\mathrm{image}(F)$ induced by l_2 regularization on parameter space. In this paper, we are interested in understanding how different parameterizations, regularize the function space differently.

One way to motivate this question is by considering the popularity of the overparameterized models, in which the number of parameters is greater than the number of samples. Surprisingly, it has been observed that in the overparameterized regime, interpolative predictor generalizes well [5, 28, 10, 2]. One way to explain this is that although there are many predictors which perfectly fit the training data, gradient based algorithms choose the one with the smallest representation cost ([9, 16]). In these cases, representation cost operates as a regularization in the function space which enables good generalization. Thus, understanding representation cost helps us understand the generalization of the model. In particular, representation cost of predictors induces an ordering on the space of predictors. Since representation cost is determined by the specific parametrization, each parameterization induces an ordering on the function space. This can be interpreted as an *induced complexity measure* of the predictor space, where penalizing the cost is the same as minimizing the complexity.

Definition 1.1 (**Induced complexity measure**). Let $\mathcal{F}$ be a set of functions from $\mathcal{X}$ to $\mathbb{R}^m$ and $\mathcal{F}^*$ be the set of all functions from $\mathcal{F}$ to $\mathbb{R}$. We define the *equivalence relation* in $\mathcal{F}^*$ such that h_1 and $h_2 \in \mathcal{F}^*$ are equivalent ($h_1 \sim h_2$) if there exists a strictly increasing function $\psi : \mathbb{R} \longrightarrow \mathbb{R}$ such that $h_1 = \psi \circ h_2$. Let $\mathcal{F}^*/\sim$ be the set of equivalence classes of $\mathcal{F}^*$ under $\sim$.

For parameterization $F : \mathbb{R}^p \to \mathcal{F}$, let $\mathcal{F}$ be $\mathrm{image}(F)$ and the representation cost R_F be as in Eq. (1). We say that equivalence class $\bar{h} \in \mathcal{F}^*/\sim$ is the *induced complexity measure* of the parameterization F if for any representative h in class $\bar{h}$, $R_F \sim h$ i.e., R_F is a strictly increasing function of h.

We study the dependence of induced complexity measures on parameterizations from two perspectives: First, given a parameterization F, we analyze the its representation cost. Second, given a regularizer on function space, we study when and how it can be the induced complexity measure of some parametrizations. In this paper, we start answering these questions by focusing on linear predictors parameterized by linear neural networks.

In the first part of the paper (Section 3), we identify the representation costs of various architectures. Specifically, we look into fully connected networks and convolutional networks with multiple outputs. In addition, we show how the representation cost of convolutional networks with finite filter width changes as their filter width changes. Then, we show that the representation cost of residual networks interpolates between the representation costs of two of their component networks. Finally, we give

(a) (b) (c) (d)

Figure 1: **k-balanced and Group networks: architectures for $l_{p,q}$ norms.** Figure 1b induces $l_{2,1}$ norm. Figure 1c induces $l_{2/d_2,2/d_1}$ norm. Figure 1d induces $l_{2/d_1,2/(d_1+1)}$ norm. In all plots, nodes in same color are in same group.

two characterizations of the representation costs of depth-two neural networks. The results of this part are summarized in Table 1.

In the second part (Section 4), given a few regularizers, we study when and how they can be the induced complexity measure of some architectures. Specifically, we show that l_p norm can be the induced complexity measure induced by l_2 regularization on some linear neural networks if and only if $2/p$ is an integer. Moreover, when $2/p$ is an integer, we characterize all the architectures whose induced complexity measures are l_p norms. In addition, we design architectures whose induced complexity measures are k-support-norm and $l_{p,q}$ group norms. Then, we show that elastic nets and $l_{p,q}$ norm with overlapping groups cannot be the induced complexity measure of any linear neural network. On the contrary, we show that there exist homogeneous parametrizations whose induced induced complexity measures are elastic nets and $l_{p,q}$ norm with overlapping groups. The results of this part are summarized in Table 2. Finally, in the conclusion, we discuss some interesting future directions.

Further related works: Traditional works tend to focus on the expressive power $\mathrm{image}(F)$ of the model [15, 21, 27, 14]. However, as discussed in [19, 17], some other capacity control, different from network size, plays a role in deep learning. This motivates the study of representation cost and its relation to parametrizations.

Representation cost has been studied before under various models. [9] showed that the representation cost of a linear convolutional neural network of depth d is strictly increasing in the Schatten-$2/d$ norm on the Fourier domain, whereas the representation cost of a linear fully connected network of depth d is strictly increasing in the l_2 norm. [22] and [20] studied depth-two fully connected network with infinite width and ReLU activation. They show that the representation cost of any continuous function depends on the Laplacian of that function.

2 Setup

The *homogeneous* model f of degree L satisfies $f(x;\lambda w) = \lambda^L f(x;w)$, for all $\lambda > 0$. Note that $L = d$ when f is a homogeneous neural networks of depth d. When $\mathcal{X} = \mathbb{R}^n$ and g is a linear function, we identify g with the matrix $B \in \mathbb{R}^{m\times n}$ such that $g(x) = Bx$. In particular, in the case of one-dimensional output space, we identify g with the vector $b \in \mathbb{R}^n$ such that $g(x) = b^T x$.

In this paper, we mainly focus on *linear neural networks* which are linear predictors $y = (\prod_{i=1}^{d} C_{d-i+1})x$, where C_j is the matrix in the jth hidden layer of the neural network which depends on the architecture and the parameters of the network. Let $\mathcal{N}$ be a depth d homogeneous feedforward linear neural network with single output. For $i = 0, 1 \ldots, d$, let N_i represent the set of nodes in the i-th layer, where $N_0 = [n]$ denote the input layer and $N_d = \{O\}$ denote the output layer. We associate $\mathcal{N}$ with a directed graph $G = (V, E)$, where $V = \cup_{i=0}^{d} N_i$ is the set of nodes and $(u, v) \in E$ for nodes $u \in N_i$ and $v \in N_{i+1}$ if and only if the corresponding nodes are connected by an edge in $\mathcal{N}$. For each $v \in V$, let

$$S_v = \{i \in N_0 : \quad \text{there exists a directed path from } \ i \ \text{ to } \ v\}. \tag{4}$$

Without loss of generality, we assume that for all $v \in V$, $|S_v| \geq 1$ and there exists a directed path from v to O. Otherwise, it does not affect the output of the network.

Let w be the weights on $\mathcal{N}$ and with slight abuse of notation, let $F_{\mathcal{N}}(w) \in \mathbb{R}^n$ be the vector corresponding to the linear predictor generated by w on $\mathcal{N}$. Let $R_{\mathcal{N}} := R_{F_{\mathcal{N}}}$ denote the representation cost (Eq. 1) under $F_{\mathcal{N}}$. We say that h is the *induced complexity measure* of $\mathcal{N}$ if it is the induced complexity measure of $F_{\mathcal{N}}$ as defined in Definition 1.1.

Notation: We will use $b \in \mathbb{R}^n$ to denote a column vector, and b_i or $b[i]$ to denote the i-th component of b. We will use $\hat{b}$ to denote the discrete Fourier transform of b. For groups $G_1, G_2 \ldots, G_k \subseteq [n]$, we use the definition of the $l_{p,q}$ group norm, $\|b\|_{p,q} = \left(\sum_{j=1}^{k} \left(\sum_{i \in G_j} |b_i|^q\right)^{p/q}\right)^{1/p}$. Unless stated otherwise, $G_1, G_2 \ldots,$ and G_k form a partition of $[n] := \{1, 2, \ldots, n\}$.

3 Representation cost analysis

To understand the dependence of induced complexity measure on architectures, we analyze the representation costs of some commonly used architectures. The authors in [9] studied single output fully connected network, diagonal network, and convolutional neural network (CNN) with full filter width. In this section, we first generalize their results to multiple output case and then, we study CNN's with finite filter width, which is what used in practice. Next, we look into the non-homogeneous residual neural network and observe that its representation cost interpolates between the representation costs of two of its component networks. Finally, we characterize the representation costs of depth-two neural network in two ways.

3.1 Multiple output networks

3.1.1 Fully connected network

In a *fully connected neural network*, $F_{FC}(w) = \prod_{i=1}^{d} C_{d+1-i}$ where $w = (C_1, C_2, \cdots, C_d)$ is the weights of the network. For $i \in [d]$, the matrix C_i is in $\mathbb{R}^{n_{i+1} \times n_i}$ where $n_i \geq \min(m, n)$, $n_1 = n$ and $n_{d+1} = m$. Let $R_{FC} := R_{F_{FC}}$ be the representation cost under F_{FC} defined in Eq. (1).

Proposition 1. *For any $B \in \mathbb{R}^{m \times n}$, let $\sigma_1, \sigma_2, \cdots, \sigma_r$ be the positive singular values of B. Then, $R_{FC}(B) = d \sum_{i=1}^{r} \sigma_i^{2/d}$.*

Trace norm is used as regularization in multitask learning [3].

The above proposition shows that linear fully connected network induces regularization that improves generalization in multitask learning.

3.1.2 Diagonal network

In a *diagonal network*, $F_{DNN}(w) = C_d \prod_{i=2}^{d} \operatorname{diag}(c_{d+1-i})$, where $w = (c_1, c_2, \cdots c_{d-1}, C_d)$ is the parameters of a diagonal neural network. For $i \in [d-1]$, $c_i \in \mathbb{R}^n$, and $C_d \in \mathbb{R}^{m \times n}$. So a diagonal network consists of some *diagonal layers* followed by a *fully connected layer*. Let $R_{DNN} := R_{F_{DNN}}$ be the representation cost under F_{DNN} defined in Eq. (1).

Proposition 2. *For any $B = (b^{(1)}, b^{(2)}, \cdots, b^{(n)}) \in \mathbb{R}^{m \times n}$, $R_{DNN}(B) = d \sum_{i=1}^{n} \left\|b^{(i)}\right\|_2^{2/d}$.*

Matrix $l_{1,2}$ norm is used as regularization in multitask learning [7].

The above proposition shows that linear diagonal network induces regularization that improves generalization in multitask learning.

3.1.3 Convolutional neural network (CNN)

In a *Convolutional neural network* (CNN), $F_{CNN}(w) = \prod_{i=1}^{d} C_{d+1-i}$ where $w = (c_1, c_2, \cdots c_{d-1}, C_d)$ is the parameters of CNN and for each $i \in [d-1]$,

$$C_i = \frac{1}{\sqrt{n}} \begin{pmatrix} c_i[1] & c_i[2] & \cdots & c_i[n] \\ c_i[n] & c_i[1] & \cdots & c_i[n-1] \\ \vdots & \vdots & & \vdots \\ c_i[2] & c_i[3] & \cdots & c_i[1] \end{pmatrix}. \tag{5}$$

In a *finite width CNN*, let q be the filter width and $w_q = (c_1, c_2, \cdots c_{d-1}, C_d)$ be the weights, such that $c_i \in \mathbb{R}^q \times \{0\}^{n-q}$ and $C_d \in \mathbb{R}^{m \times n}$. Let $R_{CNN}(B; n, q) := R_{F_{CNN}}(B; n, q)$ be the representation cost under F_{CNN} defined in Eq. (1), with inputs in $\mathbb{R}^n$ and filter width q.

Let $\mathsf{F} \in \mathbb{C}^{n\times n}$ be the discrete Fourier transform matrix defined by $\mathsf{F}[j,k] = \frac{1}{\sqrt{n}}\omega_n^{(j-1)(k-1)}$, where $\omega_n = e^{2\pi i/n}$. For $B \in \mathbb{R}^{m\times n}$, let $\hat{B} := B\mathsf{F}$.

Proposition 3. *For any $B \in \mathbb{R}^{m\times n}$, let $\hat{B} := B\mathsf{F}$ and $\hat{b}^{(i)}$ be the i-th column of $\hat{B}$. Then,* $R_{CNN}(B;n,n) = d\sum_{i=1}^{n}\left\|\hat{b}^{(i)}\right\|_2^{2/d}$.

3.2 Linear convolutional neural networks with finite filter width

In this section, we consider single output, depth-two CNN with finite filter width q such that $q|n$. Let $\hat{b} = \mathsf{F}b$ be the discrete Fourier transform of b. The following results can be easily generalized to the multiple outputs and general depth case.

Theorem 4. *For single output, depth-two CNN with filter width q such that $q|n$,*

$$R_{CNN}(b;n,q) \geq 2\sqrt{\frac{n}{q}}\sqrt{\sum_{t=1}^{n/q}(\sum_{j\in\mathcal{S}_t}|\hat{b}_j|)^2}, \tag{6}$$

where $\mathcal{S}_t = \{t + \frac{n}{q}v : v \in \{0,1,\ldots,q-1\}\}$.

Next corollary is a direct consequence of Theorem 4.

Corollary 5. *For $b \in \mathbb{R}^n$ and $q|n$, $\frac{1}{\sqrt{q}}R_{CNN}(b;n,1) \leq R_{CNN}(b;n,q) \leq R_{CNN}(b;n,1)$. In particular,*

$$2\sqrt{\frac{n}{q}}\sqrt{\sum_{t=1}^{n/q}(\sum_{j\in\mathcal{S}_t}|\hat{b}_j|)^2} \leq R_{CNN}(b;n,q) \leq 2\sqrt{n}\sqrt{\sum_{t=1}^{n/q}(\sum_{j\in\mathcal{S}_t}|\hat{b}_j|)^2}. \tag{7}$$

Note that the above corollary implies that our bound in Theorem 4 is tight up to a multiplicative factor of $\sqrt{q}$.

Now, we compute the representation costs of some special vectors. We say that $b \in \mathbb{R}^n$ is q-periodic if $b_{i+q} = b_i$ for all i, and q-antiperiodic if $b_{i+q} = -b_i$ for all i.

Theorem 6. *If either b is q-periodic or n is even, q is odd, and b is is q-antiperiodic, then* $R_{CNN}(b;n,q) = 2\sqrt{\frac{n}{q}}\left\|\hat{b}\right\|_1$.

Next, we look into the representation costs of two predictors $b^{(1)}$ and $b^{(2)}$ to see how the induced complexity measure of CNN changes with q.

Example Assume $q_1|q_2$, $q_2|n$. Let $e_j \in \mathbb{R}^n$ be the jth standard basis vector and $\epsilon > 0$ be a constant such that $(1-\epsilon)\sqrt{q_2} > \sqrt{q_1}$. Assume $b^{(1)} = (1-\epsilon)\sum_{j=0}^{n/q_1-1} e_{1+jq_1}$ and $b^{(2)} = \sum_{j=0}^{n/q_2-1} e_{1+jq_2}$. Then $R_{CNN}(b^{(1)};n,q_1) > R_{CNN}(b^{(2)};n,q_1)$ but $R_{CNN}(b^{(1)};n,q_2) < R_{CNN}(b^{(2)};n,q_2)$. Proof of this claim is in the Appendix.

In the Appendix, we will use the above example to construct a data set and look into the minimum representation cost interpolation of the data. We design our data set so that this interpolation changes with filter width.

We will also study some variations of CNN such as CNN with sum pooling and CNN with multiple channels in the Appendix.

3.3 Linear non-homogeneous residual neural networks

Let $\mathcal{N}_1,\ldots,\mathcal{N}_k$ be k homogeneous feedforward linear neural networks. Let w be the weights. Let $F_{\mathcal{N}_j}(w) \in \mathbb{R}^n$ be the vector corresponding to the linear predictor generated by w in $\mathcal{N}_j$. We consider the ResNet $\mathcal{N}_{ResNet}$, which is the sum of the networks $\mathcal{N}_1,\ldots,\mathcal{N}_k$, i.e., $F_{\mathcal{N}_{ResNet}}(w) = \sum_{j=1}^{k} F_{\mathcal{N}_j}(w)$. Let $R_{ResNet} := R_{F_{\mathcal{N}_{ResNet}}}$ be the representation cost under $F_{\mathcal{N}_{ResNet}}$, and $R_j := R_{F_{\mathcal{N}_j}}$ be the representation cost under $F_{\mathcal{N}_j}$ for each $j \in [k]$.

Theorem 7. *Let $\mathcal{N}_{ResNet}$ be the sum of k homogeneous feedforward neural networks $\mathcal{N}_1, \ldots, \mathcal{N}_k$ such that $\mathcal{N}_j$ is of depth d_j. Suppose that $d_1 < d_2 < \cdots < d_k$. Then, $R_{ResNet}(\lambda b)/R_1(\lambda b) \longrightarrow 1$ as $\lambda \to 0$. In addition, if $\mathcal{N}_k$ is the only architecture (among $\mathcal{N}_1, \ldots, \mathcal{N}_k$) that contains the first hidden layer in $\mathcal{N}_{ResNet}$, then $R_{ResNet}(\lambda b)/R_k(\lambda b) \longrightarrow 1$ as $\lambda \to \infty$.*

The Appendix includes some examples.

3.4 Depth two neural networks

In this section, we characterize the representation costs of depth two homogeneous feedforward neural networks in two ways. We will use these two characterizations to find architectures that induce k-support norms [4] and $l_{2,1}$ norms as induced complexity measures.

Let $d = 2$. Note that the definition given in Eq. (4) becomes

$$S_h = \{i \in N_0 : (i, h) \in E\}, \quad \text{for each} \quad h \in N_1. \tag{8}$$

Lemma 8. $R_{\mathcal{N}}(b) = 2 \min \left\{ \sum_{h \in N_H} \|v_h\|_2 : \text{supp}(v_h) \subseteq S_h, \sum_{h \in N_H} v_h = b \right\}.$

In the above lemma, each vector v_h corresponds to the linear predictor generated by the part of the network which includes the hidden node h and its neighbors. Lemma 8 implies that $R_{\mathcal{N}}(\cdot)$ is a norm. Now, we give a characterization of its dual norm.

Lemma 9. $R^*_{\mathcal{N}}(b) = \frac{1}{2} \max \left\{ \left(\sum_{i \in S_h} b_i^2 \right)^{1/2} : h \in N_H \right\}.$

By the Lemma 9, if there exists $h_1, h_2 \in N_H$ such that $h_1 \neq h_2$ and $S_{h_1} \subseteq S_{h_2}$, then removing h_1 from $\mathcal{N}$ would not change the representation cost since $\sum_{i \in S_{h_1}} b_i^2 \leq \sum_{i \in S_{h_2}} b_i^2$. Results in the next two sections are two of the applications of Lemma 8 and Lemma 9.

4 Parameterization design

In order to further understand the dependence of induced complexity measure on architectures, we study when and how regularizers can be induced as the induced complexity measure by some architectures. In this section, we study a few regularizers such as l_p norms, $l_{p,q}$ group norms with and without overlapping between groups, $k-$support norms, and elastic nets.

4.1 Basic properties

Restricting a network to a subset of input nodes is a useful tool in this section:

Definition 4.1 (Subnetwork). For $S \subseteq [n]$, the restriction of $\mathcal{N}$ to S is called $\mathcal{N}_S$. The subnetwork $\mathcal{N}_S$ is obtained from $\mathcal{N}$ by first removing all input nodes in $\mathcal{N}$ except for those in $S \subseteq [n]$ and then removing all hidden nodes that are isolated from the remaining input nodes.

If a network $\mathcal{N}$ induces $h(\cdot)$ as the induced complexity measure, then the subnetwork $\mathcal{N}_S$ induces $h_S(\cdot)$ as the *restricted induced complexity measure*. An important observation is that for any $b' \in \mathbb{R}^{|S|}$, $h_S(b')$ can be obtained from h as

$$h_S(b') = h(b), \tag{9}$$

where b is the lifting of b' defined as: $b_i = b'_i$ for $i \in S$ and $b_i = 0$ otherwise.

In Section 4.3, we will use this observation to design architectures that induce certain desired complexity measures. At a high level, given h, the above property informs how the subnetwork $\mathcal{N}_S$ looks like and that indicate certain properties of $\mathcal{N}$.

4.2 l_p norms

We show that the architectures that induce l_p norms have the same *mixing depths* for all $S \subseteq [n]$, where mixing depth is defined as follows.

Definition 4.2 (**Mixing Depths**). For any $S \subseteq [n]$, the mixing depth of S with respect to $\mathcal{N}$ is defined as:

$$M_{\mathcal{N}}(S) := \min\{i \in \mathbb{N} : \quad \text{there exists} \quad v \in N_i \quad \text{such that} \quad S \subseteq S_v\}, \tag{10}$$

where N_i is the set of nodes in the i-th hidden layer of $\mathcal{N}$, and S_v is defined in Eq. (4).

The notion of mixing depths capture how fast information from the a subset of nodes in the input layers is mixed together, as defined below. Note that $M_{\mathcal{N}}(\{s\}) = 1$ for all $s \in [n]$. Thus, we only consider mixing depths of sets of size at least two. The following theorem identifies architectures that induce l_p norms as architectures with *uniform* mixing depths.

Theorem 10. *There exists a linear homogeneous feedforward neural network $\mathcal{N}$ that induces l_p norm if and only if $2/p \in \mathbb{N}$. Furthermore, a linear homogeneous feedforward neural network $\mathcal{N}$ induces l_p norm if and only if $M_{\mathcal{N}}(S) = 2/p$, for all $S \subseteq [n], |S| \geq 2$.*

Roughly speaking, architectures with small mixing depths usually have small representation costs, and vice versa. The following proposition is a motivating example of this intuition.

Proposition 11. *For all linear homogeneous feedforward neural networks $\mathcal{N}$ of depth d,*

$$R_{FC}(b) = d\|b\|_2^{2/d} \leq R_{\mathcal{N}}(b) \leq d\|b\|_{2/d}^{2/d} = R_{DNN}(b). \tag{11}$$

Furthermore, the lower bound is achieved for all $b \in \mathbb{R}^n$ if and only if the mixing depths $M_{\mathcal{N}}(S) = 1$ for all $S \subseteq [n]$, and the upper bound is achieved for all $b \in \mathbb{R}^n$ if and only if the mixing depths $M_{\mathcal{N}}(S) = d$ for all $S \subseteq [n]$ such that $|S| \geq 2$.

4.3 $l_{p,q}$ group norms

Similar to the previous sections, we want to know if and when $l_{p,q}$ group norm is the induced complexity measure of $\mathcal{N}$. Remember the definition of of $l_{p,q}$ norm, $\|b\|_{p,q} = \left(\sum_{j=1}^{k}\left(\sum_{i\in G_j}|b_i|^q\right)^{p/q}\right)^{1/p}$, where $G_1, G_2 \ldots G_k$ form a partition of $[n]$. In particular, we will show that $l_{p,q}$ norm is the induced complexity measure of $\mathcal{N}$, only when $2/p, 2/q \in \mathbb{N}$ and identify some architectures which induce $l_{p,q}$ norm for p, q such that $2/p, 2/q \in \mathbb{N}$ and $2/p \geq 2/q - 1$.

The main observation is that *if $\mathcal{N}$ induces $l_{p,q}$ norm as the complexity measure, then the subnetwork $\mathcal{N}_S$ with $S = \{i, j\}$ (Def 4.1) induces l_q or l_p norm depending on whether i and j are in the same group or not.* The reasoning for this observation is two-fold: Restricting a network to two nodes gives a subnetwork with "restricted induced complexity measure" as in Eq. (9); and restricting $l_{p,q}$ group norm to 2 sparse vectors gives l_p or l_q norms. Theorem 10 identifies the architectures which induce l_p norm as the ones with *uniform* mixing depths $2/p$. Hence, the mixing depths of any architectures that induces $l_{p,q}$ norm satisfy

$$M_{\mathcal{N}}(\{i,j\}) = \begin{cases} 2/q & \text{if} \quad \text{i and j are in the same group;} \\ 2/p & \text{if} \quad \text{i and j are in different groups.} \end{cases} \tag{12}$$

Based on this, it is natural to consider architectures that consists of some diagonal layers followed by a *grouping layer* and then followed by a diagonal network (Section 3.1.2). We call these architectures *group networks*. Two examples of such networks are in Figures 1c and 1d. The *grouping layer* is the first layer that mixes information from different input nodes. Depending on whether $p < q$ or $p > q$ [1], we define two types of grouping layers:

Definition 4.3 (**Type I and II Grouping Layers**). For each $i \in [d]$, N_i is a *type I grouping layer* if N_j is diagonal for all $j < i$, $|N_i| = k$, where k is the number of groups, and for each $j \in [k]$, there exists $u \in N_i$ such that $S_u = G_j$.

For each $i \in [d]$, N_i is a *type II grouping layer* if N_j is diagonal for all $j < i$, $|N_i| = \prod_{j=1}^{k}|G_j|$, where k is the number of groups, and for each $h \in \prod_{j=1}^{k} G_j$, there exists $u \in N_i$ such that $S_u = h$.

Next, we compare the representation cost of networks with these two types of grouping layers.

[1] When $p = q$, $l_{p,q}$ norm becomes l_p norm which we already studied.

4.3.1 Architectures with type I grouping layer

For $d_1, d_2 \in \mathbb{N}$, let $\mathcal{N}^{1;d_1,d_2}$ be the architecture with $d_1 - 1$ diagonal layers, followed by a type I grouping layer (Def 4.3), and then followed by a diagonal network of depth $d_2 - d_1$ (Section 3.1.2). See Figure 1c for an example of this kind of *group network*.

Theorem 12. *Let b_{G_j} be the projection of b on G_j. Then, $R_{\mathcal{N}^{1;d_1,d_2}}(b) = d_2 \sum_{j=1}^{k} \left\| b_{G_j} \right\|_{2/d_1}^{2/d_2} = d_2 \|b\|_{2/d_2,2/d_1}^{2/d_2}$.*

4.3.2 Architectures with type II grouping layer

Let $\mathcal{N}^{2;d_1,d_2}$ denote the architecture consisting of $d_1 - 1$ diagonal layers, followed by a type II grouping layer (Def 4.3), and then followed by a diagonal network of depth $d_2 - d_1$ (Section 3.1.2). Figure 1d is an example of $\mathcal{N}^{2;d_1,d_2}$.

In particular, when $d_1 = 1$ and $d_2 = 2$ (as in Figure 1b), $\mathcal{N}^{2;1,2}$ induces $l_{2,1}$ norm. This can be proved by the dual characterization of representation cost of depth-two networks in Lemma 9 and the fact that $\|b\|_{2,1} = \|b\|_{2,\infty}^{*}$.

Theorem 13. *When $d_2 = d_1 + 1$, $R_{\mathcal{N}^{2;d_1,d_2}}(b) = d_2 \|b\|_{2/d_1,2/d_2}^{2/d_2}$.*

This theorem implies that $\mathcal{N}^{2;d_1,d_2}$ induces $l_{2/d_1,2/d_2}$ norm when $d_2 = d_1 + 1$. Surprisingly, $\mathcal{N}^{2;d_1,d_2}$ does not induce $l_{2/d_1,2/d_2}$ norm when $d_2 > d_1 + 1$. The details are in the Appendix.

4.4 The k-support norms

In [4], the k-support norm is defined as

$$\|b\|_k^{sp} = \min \left\{ \sum_{I \in \mathcal{G}_k} \|v_I\|_2 : \operatorname{supp}(v_I) \subseteq I, \sum_{I \in \mathcal{G}_k} v_I = b \right\}, \quad \text{for} \quad k \in [n], \tag{13}$$

where $\mathcal{G}_k$ is the set of subsets of $[n]$ of size at most k.

To design an architecture which induces k-support-norm, we introduce the *k-balanced networks*. A two layer neural network is a k-balanced network, if it contains $\binom{n}{k}$ nodes in the hidden layer such that for each subset $I \subseteq [n]$ of size k, there is a node in the hidden layer which connects to input nodes in I. See Figure 1a for an example with $n = 3$ and $k = 2$.

Theorem 14. *For any $k \in [n]$, there exists a homogeneous feedforward depth two linear neural network that induces k-support norm as induced complexity measure. Furthermore, a homogeneous feedforward linear depth two neural network $\mathcal{N}$ induces k-support norm if and only if the mixing depths of subsets S satisfy*

$$M_{\mathcal{N}}(S) = \begin{cases} 1 & \text{if} \quad |S| \leq k; \\ 2 & \text{if} \quad |S| > k. \end{cases} \tag{14}$$

The proof of the above theorem is an application of Lemma 8 which characterizes the representation cost of depth-two networks.

By Theorem 14, the k-balanced networks induce k-support norm.

4.5 Beyond homogeneous neural networks

4.5.1 Negative results on homogeneous neural networks

Theorems 12, 13, and 14 give architectures that induce $l_{p,q}$ norms and k-support norms. Contrary to these results, elastic nets defined as $f(b) = \|b\|_1 + \alpha \|b\|_2{}^2$, and $l_{p,q}$ norms with overlapping between groups are not induced complexity measure of any architecture $\mathcal{N}$.

To show the impossibility of designing architectures with these induced complexities, note that any network with two input nodes only induces l_p norms (Details are in appendix). As a result, if

[2] Elastic nets and k-support norms are both interpolations between l_1 and l_2 norms.

regularizer h is the induced complexity measure of some homogeneous feedforward network $\mathcal{N}$, then h behaves like l_p norms when restricted to 2-sparse vectors due to the characterizations of induced complexity measure of subnetworks (See discussions after Def 4.1). Since neither elastic nets nor $l_{p,q}$ norms with overlapping between groups satisfy this property, they are not induced complexity measure of any architecture $\mathcal{N}$.

Given these negative results, it is natural to look beyond homogeneous networks. Two possible directions are non-homogeneous residual networks and general homogeneous parameterizations.

4.5.2 Linear non-homogeneous residual neural networks

We use the same definition of residual networks as in Section 3.3. Theorem 7 characterizes the asymptotic behavior of the representation costs of residual networks. The proof of the following proposition uses Theorem 7.

Proposition 15. *Let $\mathcal{N}_{ResNet}$ be the sum of k homogeneous feedforward neural networks $\mathcal{N}_1, \dots, \mathcal{N}_k$ such that $\mathcal{N}_j$ is of depth d_j. Suppose that $d_1 < d_2 < \cdots < d_k$. Let $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ be a homogeneous function. If $\mathcal{N}_{ResNet}$ induces f as induced complexity measure, then $\mathcal{N}_1$ also induces f as induced complexity measure.*

This proposition implies that the negative results in Section 4.5.1 and Theorem 10 on elastic nets, $l_{p,q}$ norm with overlapping groups, and l_p norms when $2/p \notin \mathbb{N}$ still hold even in the case of non-homogeneous residual networks.

4.5.3 Homogeneous parameterizations

As a next step looking beyond homogeneous networks, we can look into general form of homogeneous parameterizations which might not be associated with any networks. Contrary to neural networks, homogeneous parameterizations can indeed induce elastic nets and $l_{p,q}$ norms with overlapping groups for all $p, q > 0$, as induced complexity measure. The key idea is that we can use the $\min$ operator to design the parametrization with the desired induced complexity measure. For instance, the parametrization

$$f_{EN}(x; c_1, c_2, C_3, c_4) = \operatorname{sign}(c_2^T \operatorname{diag}(c_1)) \min(2|c_2^T \operatorname{diag}(c_1)|, 2\alpha^{-1}|c_4^T C_3|)\, x \tag{15}$$

induces elastic net $\|b\|_1 + \alpha\|b\|_2$, where $c_1, c_2, c_4 \in \mathbb{R}^n$ and $C_3 \in \mathbb{R}^{n\times n}$ are the parameters, and $\min(\cdot)$, $\operatorname{sign}(\cdot)$, and $|\cdot|$ are applied component-wise. The proof conveys the intuition behind the use of min operator.

5 Conclusion

This paper is the first step in the direction of focusing on the representation cost of parameterized models. In this paper, we study the dependency of induced complexity measure on the choice of parametrization. We study this problem by analyzing the induced complexity measures of some well-known architectures and designing architectures that induce some common regularizers on linear predictors. These directions are important for two reasons. First, it helps us understand why certain architectures generalize. Second, if we have a desired regularizer in mind, this helps us design an architecture which induces this regularizer as induced complexity measure.

Answering these two questions of design and analysis in a broad sense is an important step in understanding generalization and improving our current models. As a first step in this direction, this paper focuses on linear predictors and certain architectures and regularizers on them.

To move beyond, there are still many unanswered questions for the linear models: what is the induced complexity measure of CNN with finite filter width? Second, for $p, q \in \mathbb{N}$ such that $2/p < 2/q - 1$, does there exist an architecture that induce $l_{p,q}$ norm?

Next step would be looking beyond linear predictors. For example, the question of analyzing representation costs for neural networks with nonlinear activation functions such as ReLU is an open problem for most architectures. The other possible direction is studying the same questions (analysis and design) for functions with multiple outputs.

References

[1] Alnur Ali, J Zico Kolter, and Ryan J Tibshirani. A continuous-time view of early stopping for least squares regression. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1370–1378. PMLR, 2019.

[2] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018.

[3] Yonatan Amit, Michael Fink, Nathan Srebro, and Shimon Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the 24th international conference on Machine learning*, pages 17–24, 2007.

[4] Andreas Argyriou, Rina Foygel, and Nathan Srebro. Sparse prediction with the k-support norm. *arXiv preprint arXiv:1204.5043*, 2012.

[5] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[6] Lenaic Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on Learning Theory*, pages 1305–1338. PMLR, 2020.

[7] An Evgeniou and Massimiliano Pontil. Multi-task feature learning. *Advances in neural information processing systems*, 19:41, 2007.

[8] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. pages 1832–1841, 2018.

[9] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Implicit bias of gradient descent on linear convolutional networks. *arXiv preprint arXiv:1806.00468*, 2018.

[10] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *arXiv preprint arXiv:1705.08741*, 2017.

[11] Meena Jagadeesan, Ilya Razenshteyn, and Suriya Gunasekar. Inductive bias of multi-channel linear convolutional networks with bounded weight norm. *arXiv preprint arXiv:2102.12238*, 2021.

[12] Ziwei Ji and Matus Telgarsky. Risk and parameter convergence of logistic regression. *arXiv preprint arXiv:1803.07300*, 2018.

[13] Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

[14] Hongzhou Lin and Stefanie Jegelka. Resnet with one-neuron hidden layers is a universal approximator. *arXiv preprint arXiv:1806.10909*, 2018.

[15] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. *arXiv preprint arXiv:1709.02540*, 2017.

[16] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *arXiv preprint arXiv:1906.05890*, 2019.

[17] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. *arXiv preprint arXiv:1706.08947*, 2017.

[18] Behnam Neyshabur, Ryota Tomioka, Ruslan Salakhutdinov, and Nathan Srebro. Geometry of optimization and implicit regularization in deep learning. *arXiv preprint arXiv:1705.03071*, 2017.

[19] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*, 2014.

[20] Greg Ongie, Rebecca Willett, Daniel Soudry, and Nathan Srebro. A function space view of bounded norm infinite width relu nets: The multivariate case. *arXiv preprint arXiv:1910.01635*, 2019.

[21] David Rolnick and Max Tegmark. The power of deeper networks for expressing natural functions. *arXiv preprint arXiv:1705.05502*, 2017.

[22] Pedro Savarese, Itay Evron, Daniel Soudry, and Nathan Srebro. How do infinite width bounded norm networks look in function space? pages 2667–2690, 2019.

[23] Mor Shpigel Nacson, Suriya Gunasekar, Jason D Lee, Nathan Srebro, and Daniel Soudry. Lexicographic and depth-sensitive margins in homogeneous and non-homogeneous deep models. *arXiv e-prints*, pages arXiv–1905, 2019.

[24] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[25] Arun Suggala, Adarsh Prasad, and Pradeep K Ravikumar. Connecting optimization and regularization paths. *Advances in Neural Information Processing Systems*, 31:10608–10619, 2018.

[26] Colin Wei, Jason Lee, Qiang Liu, and Tengyu Ma. Regularization matters: Generalization and optimization of neural nets vs their induced kernel. 2019.

[27] Dmitry Yarotsky. Optimal approximation of continuous functions by very deep relu networks. In *Conference on Learning Theory*, pages 639–649. PMLR, 2018.

[28] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] In the conclusion
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [No] They will be in appendix
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [N/A]
(b) Did you mention the license of the assets? [N/A]
(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]