Norm-based Generalization Bounds for Compositionally Sparse Neural Networks

Tomer Galanti [1] Mengjia Xu [1 2] Liane Galanti [3] Tomaso Poggio [1]

Abstract

In this paper, we investigate the Rademacher complexity of deep sparse neural networks, where each neuron receives a small number of inputs. We prove generalization bounds for multilayered sparse ReLU neural networks, including convolutional neural networks. These bounds differ from previous ones, as they consider the norms of the convolutional filters instead of the norms of the associated Toeplitz matrices, independently of weight sharing between neurons.

As we show theoretically, these bounds may be orders of magnitude better than standard norm-based generalization bounds and empirically, they are almost non-vacuous in estimating generalization in various simple classification problems. Taken together, these results suggest that compositional sparsity of the underlying target function is critical to the success of deep neural networks.

1. Introduction

Over the last decade, deep learning with large neural networks has greatly advanced the solution of a wide range of tasks including image classification (He et al., 2016; Dosovitskiy et al., 2021; Zhai et al., 2021), language processing (Vaswani et al., 2017; Devlin et al., 2019; Brown et al., 2020), interacting with open-ended environments (Silver et al., 2016; Arulkumaran et al., 2019), and code synthesis (Chen et al., 2021). Despite traditional theories (Vapnik, 1998), recent findings (Zhang et al., 2017; Belkin, 2021) show that deep neural networks can generalize well even when their size far exceeds the number of training samples.

To address this question, recent efforts in deep learning theory study the generalization performance of deep networks by analyzing the complexity of the learned function.

[*]Equal contribution [1]Massachusetts Institute of Technology [2]Brown University [3]Tel-Aviv University. Correspondence to: Tomer Galanti <galanti@mit.edu>, Tomaso Poggio <tp@csail.mit.edu>.

Preprint

Recent work has suggested generalization guarantees for deep neural networks based on various norms of their weight matrices (Neyshabur et al., 2015; Golowich et al., 2017; Bartlett & Mendelson, 2001; Harvey et al., 2017; Bartlett et al., 2017; Neyshabur et al., 2018; Cao & Gu, 2019; Daniely & Granot, 2019; Wei & Ma, 2019; Allen-Zhu et al., 2019; Li et al., 2018). Many efforts have been made to improve the applicability of these bounds to realistic scales. Some studies have focused on developing norm-based generalization bounds for complex network architectures, such as residual networks (He et al., 2019). Other studies investigated ways to reduce the dependence of the bounds on the product of spectral norms (Wei & Ma, 2019; Nagarajan & Kolter, 2019), or to use compression bounds based on PAC-Bayes theory (Zhou et al., 2019; Lotfi et al., 2022), or on the optimization procedure used to train the networks (Cao & Gu, 2019; Arora et al., 2019; Richards & Kuzborskij, 2021). However, most of these studies have focused on fully-connected networks which empirically have lower performance compared to other architectures. In particular, these studies cannot directly explain the success of current successful architectures (LeCun et al., 1998; Vaswani et al., 2017; Dosovitskiy et al., 2020).

To fully understand the success of deep learning, it is necessary to analyze a wider scope of architectures beyond fully-connected networks. An interesting recent direction (Ledent et al., 2021; Long & Sedghi, 2020) suggests better generalization bounds for neural networks with shared parameters, such as convolutional neural networks. In fact, (Ledent et al., 2021) was the first to show that convolutional layers contribute to generalization bounds with a norm component smaller than the norm of the associated linear transformation. However, many questions remain unanswered, including **(a)** *Why certain compositionally sparse architectures, such as convolutional networks, perform better than fully-connected architectures?* **(b)** *Is weight sharing necessary for the success of convolutional neural networks?* In this paper, we contribute to an understanding of both of these questions.

1.1. Related Work

Approximation guarantees for multilayer sparse networks. While fully-connected networks, including shallow networks, are universal approximators (Cybenko, 1989; Hornik, 1991) of continuous functions, they are largely limited in theory and in practice. Classic results (Mhaskar, 1996; Maiorov & Pinkus, 1999; Maiorov et al., 1999; Maiorov, 1999; Hanin & Sellke, 2017) show that, in the worst-case, approximating r-continuously differentiable target functions (with bounded derivatives) using fully-connected networks requires $\Theta(\epsilon^{-d/r})$ parameters, where d is the input dimension and ϵ is the approximation rate. The exponential dependence on d is also known as the "curse of dimensionality".

A recent line of work (Mhaskar et al., 2017; Poggio et al., 2020; Poggio, 2022) shows that the curse of dimensionality can be avoided by deep, sparse networks, when the target function is itself compositionally sparse. Furthermore, it has been conjectured that efficiently computable functions, that is functions that are computable by a Turing machine in polynomial time, are compositionally sparse. This suggests, in turns, that, for practical functions, deep and sparse networks can avoid the curse of dimensionality.

These results, however, lack any implication about generalization; in particular, they do not show that overparametrized sparse networks have good generalization properties.

Norm-based generalization bounds. A recent thread in the literature (Neyshabur et al., 2015; Golowich et al., 2017; Bartlett & Mendelson, 2001; Harvey et al., 2017; Bartlett et al., 2017; Neyshabur et al., 2018; Cao & Gu, 2019; Daniely & Granot, 2019; Wei & Ma, 2019) has introduced norm-based generalization bounds for neural networks. In particular, let $S = \{(x_i, y_i)\}_{i=1}^m$ be a training dataset of m independently drawn samples from a probability measure P defined on the sample space $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} = \{\pm 1\}$. A fully-connected network is defined as

$$f_w(x) = W^L \sigma(W^{L-1}\sigma(\dots\sigma(W^2\sigma(W^1 x))\dots)), \quad (1)$$

where L is the depth of the network, $W^l \in \mathbb{R}^{d_{l+1}\times d_l}$ and $\sigma(x)$ is the element-wise ReLU activation function $\max(0, x)$. A common approach for estimating the gap between the train and test errors of a neural network is to use the Rademacher complexity of the network. For example, in (Neyshabur et al., 2015), an upper bound on the Rademacher complexity is introduced based on the norms of the weight matrices of the network of order $\mathcal{O}(\frac{2^L}{\sqrt{m}}\prod_{l=1}^L \|W^l\|_F)$. Later, (Golowich et al., 2017) showed that the exponential dependence on the depth can be avoided by using the contraction lemma and obtained a bound that scales with $\mathcal{O}(\sqrt{L})$. In (Bartlett et al., 2017), a Rademacher complexity bound based on covering numbers was introduced, which scales as $\tilde{\mathcal{O}}\left(\frac{\prod_{l=1}^L \|W^l\|_2}{\sqrt{m}} \cdot \left(\sum_{l=1}^L \frac{\|(W^l - M^l)^\top\|_{2,1}^{2/3}}{\|W^l\|_2^{2/3}}\right)^{3/2}\right)$, where $M^l \in \mathbb{R}^{d_{l+1}\times d_l}$ are fixed reference matrices and $\|\cdot\|_2$ is the spectral norm.

While these results provide solid upper bounds on the test error of deep neural networks, they only take into account very limited information about the architectural choices of the network. In particular, when applied to convolutional networks, the matrices W^l represent the linear operation performed by a convolutional layer whose filters are w^l. However, since W^l applies w^l to several patches (d_l patches), we have $\|W^l\|_F = \sqrt{d_l}\|w^l\|_F$. As a result, the bound scales with $\mathcal{O}(\sqrt{\prod_{l=1}^L d_l})$, that grows exponentially with L. This means that the bound is not suitable for convolutional networks with many layers as it would be very loose in practice. In this work, we establish generalization bounds that are customized for convolutional networks and scale with $\prod_{l=1}^L \|w^l\|_F$ instead of $\prod_{l=1}^L \|W^l\|_F$.

In (Jiang et al., 2020) they conducted a large-scale experiment evaluating multiple norm-based generalization bounds, including those of (Bartlett et al., 2017; Golowich et al., 2017). They argued that these bounds are highly non-vacuous and negatively correlated with the test error. However, in all of these experiments, they trained the neural networks with the cross-entropy loss which implicitly maximizes the network's weight norms once the network perfectly fits the training data. This can explain the observed negative correlation between the bounds and the error.

In this work, we empirically show that our bounds provide relatively tight estimations of the generalization gap for convolutional networks trained with weight normalization and weight decay using the MSE loss.

Generalization bounds for convolutional networks. Several recent papers have introduced generalization bounds for convolutional networks that take into account the unique structure of these networks. In (Li et al., 2018), a generalization bound for neural networks with weight sharing was introduced. However, this bound only holds under the assumption that the weight matrices are orthonormal, which is not realistic in practice. Other papers introduce generalization bounds based on parameter counting for convolutional networks that improve classic guarantees for fully-connected networks but are typically still vacuous by several orders of magnitude. In (Long & Sedghi, 2020), norm-based generalization bounds for convolutional networks were introduced by addressing their weight-sharing. However, this bound scales roughly as the square root of the number of parameters. In (Du et al., 2018), size-free bounds for convolutional networks in terms of the number of trainable parameters for two-layer networks were proved. In (Ledent et al., 2021), the generalization bounds in (Bartlett et al., 2017) were extended for convolutional networks where the linear transformation W^l at each layer is replaced with the trainable parameters. While this paper provides generalization bounds in which each convolutional filter contributes only once to the bound, it does not hold when different filters are used for different patches, even if their norms are the same. In short, their analysis treats different patches as "datapoints" in an augmented problem where only one linear function is applied at each layer. If several choices of linear functions (different weights for different patches) are allowed, the capacity of the function class would increase.

While all of these papers provide generalization bounds for convolutional networks, they all rely on the number of trainable parameters or depend on weight sharing. None of these works, in particular, address the question of whether weight sharing is necessary for convolutional neural networks to generalize well.

1.2. Contributions

In this work, we study the generalization performance of multilayered sparse neural networks (Mhaskar et al., 2017), such as convolutional neural networks. Sparse, deep neural networks are networks of neurons represented as a Directed Acyclic Graph (DAG), where each neuron is a function of a small set of other neurons. We derive norm-based generalization bounds for these networks. Unlike previous bounds (Long & Sedghi, 2020; Ledent et al., 2021), our bounds do not rely on weight sharing and provide favorable guarantees for sparse neural networks that do not use weight sharing. These results suggest that it is possible to obtain good generalization performance with sparse neural networks without relying on weight sharing.

Finally, we conduct multiple experiments to evaluate our bounds for convolutional neural networks trained on simple classification problems. We show that our bound is relatively tight, even in the overparameterized regime.

2. Problem Setup

We consider the problem of training a model for a standard classification problem. Formally, the target task is defined by a distribution P over samples $(x, y) \in \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subset \mathbb{R}^{c_0 d_0}$ is the instance space (e.g., images), and $\mathcal{Y} \subset \mathbb{R}^k$ is a label space containing the k-dimensional one-hot encodings of the integers $1, \dots, k$.

We consider a hypothesis class $\mathcal{F} \subset \{f' : \mathcal{X} \to \mathbb{R}^k\}$ (e.g., a neural network architecture), where each function $f_w \in \mathcal{F}$ is specified by a vector of parameters $w \in \mathbb{R}^N$ (i.e., trainable parameters). A function $f_w \in \mathcal{F}$ assigns a prediction to an input point $x \in \mathcal{X}$, and its performance on the distribution P is measured by the *expected error*

$$\text{err}_P(f_w) \;:=\; \mathbb{E}_{(x,y)\sim P}\left[\mathbb{I}[\text{sign}(f_w(x)) \neq y]\right], \quad (2)$$

where $\mathbb{I} : \{\text{True}, \text{False}\} \to \{0, 1\}$ be the indicator function (i.e., $\mathbb{I}[\text{True}] = 1$ and vice versa).

Since we do not have direct access to the full population distribution P, the goal is to learn a predictor, f_w, from some training dataset $S = \{(x_i, y_i)\}_{i=1}^m$ of independent and identically distributed (i.i.d.) samples drawn from P along with regularization to control the complexity of the learned model. In addition, since $\mathbb{I}$ is a non-continuous function, we typically use a surrogate loss function $\ell : \mathbb{R}^k \times \mathcal{Y} \to [0, \infty)$ is a non-negative, differentiable, loss function (e.g., MSE or cross-entropy losses).

2.1. Rademacher Complexities

In this paper, we examine the generalization abilities of overparameterized neural networks by investigating their Rademacher complexity. This quantity can be used to upper bound the worst-case generalization gap (i.e., the distance between train and test errors) of functions from a certain class. It is defined as the expected performance of the class when averaged over all possible labelings of the data, where the labels are chosen independently and uniformly at random from the set $\{\pm 1\}$. In other words, it is the average performance of the function class on random data. For more information, see (Mohri et al., 2018a; Shalev-Shwartz & Ben-David, 2014; Bartlett & Mendelson, 2002).

Definition 2.1 (Rademacher Complexity)**.** Let $\mathcal{F}$ be a set of real-valued functions $f_w : \mathcal{X} \to \mathbb{R}$ defined over a set $\mathcal{X}$. Given a fixed sample $X \in \mathcal{X}^m$, the empirical Rademacher complexity of $\mathcal{H}$ is defined as follows:

$$\mathcal{R}_X(\mathcal{H}) \;:=\; \frac{1}{m}\mathbb{E}_\xi\left[\sup_{f_w \in \mathcal{F}} \left|\sum_{i=1}^m \xi_i f_w(x_i)\right|\right].$$

The expectation is taken over $\xi = (\xi_i, \dots, \xi_m)$, where, $\xi_i \in \{\pm 1\}$ are i.i.d. and uniformly distributed samples.

In contrast to the Vapnik–Chervonenkis (VC) dimension, the Rademacher complexity has the added advantage that it is data-dependent and can be measured from finite samples.

The Rademacher complexity can be used to upper bound the generalization gap of a certain class of functions (Mohri et al., 2018a). In particular, we can easily upper bound the test classification error $\text{err}_P(f_w)$ using the Rademacher complexity for models f_w that perfectly fit the training samples, i.e., $f_w(x_i) = y_i = \pm 1$.

Lemma 2.2. *Let P be a distribution over $\mathbb{R}^{c_0 d_0} \times \{\pm 1\}$ and $\mathcal{F} \subset \{f' : \mathcal{X} \to \{\pm 1\}\}$. Let $S = \{(x_i, y_i)\}_{i=1}^m$ be a dataset of i.i.d. samples selected from P and $X = \{x_i\}_{i=1}^m$. Then, with probability at least $1 - \delta$ over the selection of S, for any $f_w \in \mathcal{F}$ that perfectly fits the data (i.e., $f_w(x_i) = y_i$), we have*

$$\text{err}_P(f_w) \;\leq\; 2\mathcal{R}_X(\mathcal{F}) + 3\sqrt{\frac{\log(2/\delta)}{2m}}. \quad (3)$$

Proof. We apply a standard Rademacher complexity-based generalization bound with the ramp loss function. The ramp loss function is defined as follows:

$$\ell_{ramp}(y,y') := \begin{cases} 1, & \text{if} \quad yy' \leq 0 \\ 1-yy', & \text{if} \quad 0 \leq yy' \leq 1 \\ 0, & \text{if} \quad yy' \geq 1 \end{cases}.$$

By Theorem 3.3 in (Mohri et al., 2018b), with probability at least $1-\delta$, for any function $f_w \in \mathcal{F}$, $\mathbb{E}_{(x,y)\sim P}[\ell_{ramp}(f_w(x),y)]$ is bounded by

$$\frac{1}{m}\sum_{i=1}^{m}\ell_{ramp}(f_w(x_i),y_i) + 2\mathcal{R}_X(\mathcal{F}) + 3\sqrt{\frac{\log(2/\delta)}{2m}}.$$

We note that for any function f_w for which $f_w(x_i) = y_i = \pm 1$, we have $\ell_{ramp}(f_w(x_i),y_i) = 0$. In addition, for any function f_w and pair (x,y), we have $\ell_{ramp}(f_w(x),y) \geq \mathbb{I}[\text{sign}(f_w(x)) \neq y]$, hence, $\mathbb{E}_{(x,y)\sim P}[\ell_{ramp}(f_w(x),y)] \geq \text{err}_P(f_w)$. Therefore, we conclude that with probability at least $1-\delta$, for any function $f_w \in \mathcal{F}$ that perfectly fits the training data, we have the desired inequality. □

The above lemma provides an upper bound on the test error of a trained model f_w that perfectly fits the training data. The bound is decomposed into two parts; one is the Rademacher complexity and the second scales as $\mathcal{O}(1/\sqrt{m})$ which is small when m is large. In section 3 we derive norm-based bounds on the Rademacher complexity of compositionally sparse neural networks.

2.2. Architectures

A neural network architecture can be formally defined using a Directed Acyclic Graph (DAG) $G = (V,E)$. The class of neural networks associated with this architecture is denoted as $\mathcal{F}_G$. The set of neurons in the network is given by $V = \bigcup_{l=0}^{L}\{z_1^l,\dots,z_{d_l}^l\}$, which is organized into L layers. An edge $(z_i^l, z_j^{l-1}) \in E$ indicates a connection between a neuron in layer $l-1$ and a neuron in layer l. The full set of neurons at the layer lth is denoted by $v^l := (z_j^l)_{j=1}^{d_l}$.

A neural network function $f_w : \mathbb{R}^{c_0 d_0} \to \mathbb{R}^k$ takes "flattened" images x as input, where c_0 is the number of input channels and d_0 is the image dimension represented as a vector. Each neuron $z_i^l : \mathbb{R}^{c_0 d_0} \to \mathbb{R}^{c_l}$ computes a vector of size c_l (the number of channels in layer l). The set of predecessor neurons of z_i^l, denoted by $\text{pred}(l,i)$, is the set of $j \in [d_{l-1}]$ such that $(z_i^l, z_j^{l-1}) \in E$, and $v_i^l := (z_j^l)_{j\in\text{pred}(l,i)}$ denotes the set of predecessor neurons of z_i^l. The neural network $z_{j_0}^L(x) := z_1^L(x) := f_w(x)$ is recursively defined as follows:

$$z_i^l(x) := w_i^l \sigma(v_i^{l-1}(x)),$$

where $w_i^l \in \mathbb{R}^{c_l \times (c_{l-1}\cdot|\text{pred}(l-1,i)|)}$ is a weight matrix, $x = (z_i^0(x))_{i=1}^{d_0}$ and each $z_i^0(x)$ is a vector of dimension c_0 representing a "pixel" in the image x.

The degree of sparsity of a neural network can be measured using the degree, which is defined as the maximum number of predecessors for each neuron.

$$\text{deg}(G) := \max_{l\in[L], j\in[d_l]} |\text{pred}(l,j)|.$$

A compositionally sparse neural network is a neural network architecture G for which the degree $\text{deg}(G) = \mathcal{O}(1)$ (with respect to $\max_{i=0,\dots,L}(d_i)$ and L). These considerations extend easily to networks that contain sparse layers as well as fully-connected layers.

Convolutional neural networks. A special type of compositionally sparse neural networks is convolutional neural networks. In such networks, each neuron acts upon a set of nearby neurons from the previous layer, using a kernel shared across the neurons of the same layer.

To formally analyze convolutional networks, we consider a broader set of neural network architectures that includes sparse networks with shared weights. Specifically, for an architecture G with $|\text{pred}(l,j)| = k_l$ for all $j \in [d_l]$, we define the set of neural networks $\mathcal{F}_G^{\text{sh}}$ to consist of all neural networks $f_w \in \mathcal{F}_G^{\text{sh}}$ that satisfy the weight sharing property $w^l := w_{j_1}^l = w_{j_2}^l$ for all $j_1, j_2 \in [d_l]$ and $l \in [L]$. Convolutional neural networks are essentially sparse neural networks with shared weights and locality (each neuron is a function of a set of nearby neurons of its preceding layer).

Norms of neural networks. As mentioned earlier, previous papers (Golowich et al., 2017; Neyshabur et al., 2018; Arora et al., 2018; Neyshabur et al., 2017; Bartlett et al., 2017) have proposed different generalization bounds based on different norms measuring the complexity of fully-connected networks. One approach that was suggested by (Golowich et al., 2017) is to use the product of the norms of the weight matrices given by $\tilde{\rho}(w) := \prod_{l=1}^{L}\|W^l\|_F$.

In this work, we derive generalization bounds based on the product of the maximal norms of the kernel matrices across layers, defined as:

$$\rho(w) := \|w_1^L\|_2 \cdot \prod_{l=1}^{L-1}\max_{j\in[d_l]}\|w_j^l\|_F, \tag{4}$$

where $\|\cdot\|_F$ and $\|\cdot\|_2$ are the Frobenius and the spectral norms. Specifically, for a convolutional neural network, we have a simplified form of $\rho(w) = \|w^L\|_2 \cdot \prod_{l=1}^{L-1}\|w^l\|_F$, due to the weight sharing property.

We observe that this quantity is significantly smaller than the quantity $\tilde{\rho}(w) = \prod_{l=1}^{L}\sqrt{\sum_{j=1}^{d_l}\|w_j^l\|_F^2}$ used by (Golowich et al., 2017). For instance, when weight sharing is applied, we can see that $\tilde{\rho}(w) = \rho(w)\cdot\sqrt{\prod_{l=1}^{L} d_l}$.

Classes of interest. In the next section, we study the Rademacher complexity of classes of compositionally sparse neural networks that are bounded in norm. We focus on two classes: $\mathcal{F}_{G,\rho} := \{f_w \in \mathcal{F}_G \mid \rho(w) \leq \rho\}$ and $\mathcal{F}^{\text{sh}}_{G,\rho} := \{f_w \in \mathcal{F}^{\text{sh}}_G \mid \rho(w) \leq \rho\}$, where G is a compositionally sparse neural network architecture and ρ is a bound on the norm of the network parameters.

3. Theoretical Results

In this section, we introduce our main theoretical results. The following theorem provides an upper bound on the Rademacher complexity of the class $\mathcal{F}_{G,\rho}$ of neural networks of architecture G of norm $\leq \rho$.

Proposition 3.1. *Let G be a neural network architecture of depth L and let $\rho > 0$. Let $X = \{x_i\}_{i=1}^m$ be a set of samples. Then,*

$$\mathcal{R}_X(\mathcal{F}_{G,\rho}) \leq \frac{\rho}{m} \cdot \left(1 + \sqrt{2L\log(2\deg(G))}\right) \cdot \sqrt{\max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \cdot \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2},$$

where the maximum is taken over $j_1, \dots, j_L$, such that, $j_{L-l+1} \in \text{pred}(l, j_{L-l})$ for all $l \in [L]$.

The proof for this theorem is provided in Appendix B and builds upon the proof of Theorem 1 in (Golowich et al., 2017). A summary of the proof is presented in section 3.1. As we show next, by combining Lemma 2.2 and Proposition 3.1 we can obtain an upper bound on the test error of compositionally sparse neural networks f_w that perfectly fit the training data (i.e., for all $i \in [m]: f_w(x_i) = y_i$).

Theorem 3.2. *Let P be a distribution over $\mathbb{R}^{c_0 d_0} \times \{\pm 1\}$. Let $S = \{(x_i, y_i)\}_{i=1}^m$ be a dataset of i.i.d. samples selected from P. Then, with probability at least $1 - \delta$ over the selection of S, for any $f_w \in \mathcal{F}_G$ that perfectly fits the data (for all $i \in [m]: f_w(x_i) = y_i$), we have*

$$\text{err}_P(f_w) \leq \frac{(\rho(w)+1)}{m}\left(1 + \sqrt{2L\log(2\deg(G))}\right) \cdot \sqrt{\max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2} + 3\sqrt{\frac{\log(2(\rho(w)+2)^2/\delta)}{2m}},$$

where the maximum is taken over $j_1, \dots, j_L$, such that, $j_{L-l+1} \in \text{pred}(l, j_{L-l})$ for all $l \in [L]$.

The theorem above provides a generalization bound for neural networks of a given architecture G. To understand this bound, we first analyze the term $\max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \cdot \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2$. We consider a setting where $d_0 = 2^L$, $c_l = 1$ and each neuron takes two neurons as input, $k_l := |\text{pred}(l, j)| = 2$ for all $l \in [L]$ and $j \in [d_l]$. In particular, $\prod_{l=1}^{L} k_l = 2^L$ and $z^0_j(x_i) = x_{ij}$ is the jth pixel of x_i. By assuming that the norms of the pixels are β-balanced, i.e., $\forall i \in [m]: \max_{j \in [d_0]} \|x_{ij}\|^2 \leq \beta \operatorname{Avg}_{j \in [d_0]}[\|x_{ij}\|^2]$ (for some constant $\beta > 0$), we obtain that $\prod_{l=1}^{L} k_l \cdot \max_j \sum_{i=1}^{m} \|z^0_j(x_i)\|^2 \leq \beta \sum_{i=1}^{m} \|x_i\|^2$. In addition, we note that the second term in the bound is typically smaller than the first term as it scales with $\sqrt{\log(\rho(w))}$ instead of $\rho(w)$ and has no dependence on the size of the network. Therefore, our bound can be simplified to

$$\mathcal{O}\left(\frac{\rho(w)}{\sqrt{m}}\sqrt{L\beta\log(\deg(G))\operatorname{Avg}_{i=1}^{m}[\|x_i\|^2]}\right). \tag{5}$$

Similar to the bound in (Golowich et al., 2017), our bound scales with $\mathcal{O}(\sqrt{L})$, where L is the depth of the network.

Convolutional neural networks. As previously stated in section 2, convolutional neural networks utilize weight sharing neurons in each layer, with each neuron in the lth layer having an input dimension of k_l. The norm of the network is calculated as $\rho(w) = \prod_{l=1}^{L} \|w^l\|_F$, and the degree of the network is determined by the maximum input dimension across all layers, $\deg(G) = \max_{l \in [L]} k_l$. This results in a simplified version of the theorem.

Corollary 3.3 (Rademacher Complexity of ConvNets)**.** *Let G be a neural network architecture of depth L and let $\rho > 0$. Let $X = \{x_i\}_{i=1}^m$ be a set of samples. Then,*

$$\mathcal{R}_S(\mathcal{F}^{\text{sh}}_{G,\rho}) \leq \frac{\rho}{m}\left(1 + \sqrt{2L\log(2\deg(G))}\right) \cdot \sqrt{\prod_{l=1}^{L} k_l \cdot \max_{j \in [d_0]} \sum_{i=1}^{m} \|z^0_j(x_i)\|^2},$$

where k_l denotes the kernel size in the l'th layer.

Comparison with the bound of (Golowich et al., 2017). The result in Corollary 3.3 is a refined version of the analysis in (Golowich et al., 2017) for the specific case of convolutional neural networks. Theorem 1 in (Golowich et al., 2017) can of course be applied to convolutional networks by treating their convolutional layers as fully-connected layers. However, this approach yields a substantially worse bound compared to the one proposed in Corollary 3.3.

Consider a convolutional neural network architecture G. The lth convolutional layer takes the concatenation of $(\sigma(z^l_1), \dots, \sigma(z^l_{d_l}))$ as input and returns $(z^{l+1}_1, \dots, z^{l+1}_{d_{l+1}})$ as its output. Each z^{l+1}_j is computed as follows $z^{l+1}_j = w^{l+1}\sigma(v^l_j(x))$. Therefore, the matrix W^{l+1} associated with the convolutional layer contains d_{l+1} copies of w^{l+1} and its Frobenius norm is therefore $\sqrt{d_{l+1}} \cdot \|w^{l+1}\|_F$. In particular, by applying Theorem 1 in (Golowich et al., 2017), we obtain a bound that scales as $\mathcal{O}\left(\frac{\rho}{m}\sqrt{L\prod_{l=1}^{L} d_l \cdot \sum_{i=1}^{m}\|x_i\|^2}\right)$. In particular, if each convolutional layer has $k_l = 2$ with no overlaps and $d_0 = 2^L$, then, $d_l = 2^{L-l}$ and the bound therefore scales as $\mathcal{O}\left(\frac{\rho}{\sqrt{m}}\sqrt{L 2^{0.5L(L-1)} \cdot \operatorname{Avg}_{i=1}^{m}[\|x_i\|^2]}\right)$. On the other hand, as we discussed earlier, if the norms of the pixels of each sample x are β-balanced (for some constant $\beta > 0$), our bound scales as $\mathcal{O}\left(\frac{\rho}{\sqrt{m}}\sqrt{L\,\mathrm{Avg}_{i=1}^{m}[\|x_i\|^2]}\right)$ which is smaller by a factor of $2^{0.25L(L-1)}$ than the previous bound.

Comparison with the bound of (Long & Sedghi, 2020). A recent paper (Long & Sedghi, 2020) introduced generalization bounds for convolutional networks based on parameter counting. This bound roughly scales like

$$\mathcal{O}\left(\sqrt{\frac{N(\sum_{l=1}^{L}\|w^l\|_2+\log(1/\gamma))+\log(1/\delta)}{m}}\right),$$

where γ is a margin (typically smaller than 1), and N is the number of trainable parameters (taking weight sharing into account by counting each parameter of convolutional filters only once). While these bounds provide improved generalization guarantees when reusing parameters, it scales as $\Omega(\sqrt{N/m})$ which is very large in practice. For example, the standard ResNet-50 architecture has approximately $N = 23M$ trainable parameters while the MNIST dataset has only $m = 50000$ training samples.

Comparison with the bound of (Ledent et al., 2021). A recent paper (Ledent et al., 2021) introduces a generalization bound for convolutional networks that is similar to the analysis presented in (Bartlett et al., 2017). Specifically, the bounds in Theorem 17 of (Ledent et al., 2021) roughly scale as

$$\mathcal{O}\left(\frac{\prod_{l=1}^{L}\|W^l\|_2}{\sqrt{m}}\left(\sum_{l=1}^{L-1}\frac{k_l^{\frac{\alpha}{2}}\|(w^l-u^l)^\top\|_{2,1}^{\alpha}}{\|w^l\|_2^{\alpha}}+\frac{\|w^L\|_2^{\alpha}}{\max_i\|w_{i,:}^L\|_2^{\alpha}}\right)^{\frac{1}{\alpha}} I_\alpha\right),$$

where k_l is the kernel size of the lth layer and W^l is the matrix corresponding to the linear operator associated with the lth convolutional layer, $w_{i,:}$ is the ith row of a matrix w, α is either 2 or 2/3, $I_\alpha = L$ if $\alpha = 2$ and $I_\alpha = 1$ otherwise and u^l are predefined "reference" matrices of the same dimensions as w^l.

In general, our bounds and the ones in (Ledent et al., 2021) cannot be directly compared, with each being better in different cases. However, our bound has a significantly better explicit dependence on the depth L than the bound in (Ledent et al., 2021). To see this, consider the simple case where each convolutional layer operates on non-overlapping patches and we choose $u^l = 0$ for all $l \in [L-1]$ (which is a standard choice of reference matrices). We notice that $\|W^l\|_2 = \|w^l\|_2$ and that for any matrix $A \in \mathbb{R}^{n\times m}$, the following inequalities hold: $\mathrm{rank}(A) \geq \frac{\|A^\top\|_{2,1}}{\|A\|_2} \geq \frac{\|A\|_F}{\|A\|_2} \geq 1$ and $\mathrm{rank}(A) \geq \frac{\|A\|_2}{\max_i\|A_{i,:}\|_2} \geq 1$. Therefore, the bound in (Ledent et al., 2021) is at least $\frac{\prod_{l=1}^{L}\|w^l\|_2}{\sqrt{m}} \cdot L^{3/2}$, which scales at least as $\Omega(L^{3/2})$ with respect to L, while our bound scales as $\mathcal{O}(\sqrt{L})$ (when $\rho(w)$ is independent of L), meaning that the dependence on the depth is significantly better than that of the bound in (Ledent et al., 2021).

3.1. Proof Sketch

We propose an extension to a well-established method for bounding the Rademacher complexity of norm-bounded deep neural networks. This approach, originally developed by (Neyshabur et al., 2015) and later improved by (Golowich et al., 2017), utilizes a "peeling" argument, where the complexity bound for a depth L network is reduced to a complexity bound for a depth $L-1$ network and applied repeatedly. Specifically, the lth step bounds the complexity bound for depth l by using the product of the complexity bound for depth $l-1$ and the norm of the lth layer. By the end of this process, we obtain a bound that depends on the term $\mathbb{E}_\xi g(|\sum_{i=1}^{m}\xi_i x_i|)$ ($g(x) = x$ in (Neyshabur et al., 2015) and $g = \exp$ in (Golowich et al., 2017)), which can be further bounded using $\max_{x\in X}\|x\|^2$. The final bound scales with $\tilde{\rho}(w)\cdot\max_{x\in X}\|x\|$. Our extension aims to further improve the tightness of these bounds by incorporating additional information about the network's degrees of sparsity.

To bound $\mathcal{R}_X(\mathcal{F}_{G,\rho})$ using $\rho(w)\cdot\max_{x\in X}\|x\|$, we notice that each neuron operates on a small subset of the neurons from the previous layer. Therefore, we can bound the contribution of a certain constituent function $z_j^l(x) = w_j^l v_j^{l-1}(x)$ in the network using the norm $\|w_j^l\|_F$ and the complexity of $v_j^{l-1}(x)$ instead of the full layer $v^{l-1}(x)$.

To explain this process, we provide a proof sketch of Proposition 3.1 for convolutional networks $G = (V, E)$ with non-overlapping patches. For simplicity, we assume that $d_0 = 2^L$, $c_l = 1$, and the strides and kernel sizes at each layer are $k = 2$. In particular, the network f_w can be represented as a binary tree, where the output neuron is computed as $f_w(x) = z_{j_0}^L(x) = w^L\cdot\sigma(z_1^{L-1}(x), z_2^{L-1}(x))$, $z_1^{L-1}(x) = w^{L-1}\cdot\sigma(z_1^{L-2}(x), z_2^{L-2}(x))$ and $z_2^{L-1}(x) = w^{L-1}\cdot\sigma(z_3^{L-2}(x), z_4^{L-2}(x))$ and so on. Similar to (Golowich et al., 2017), we first bound the Rademacher complexity using Jensen's inequality,

$$\begin{aligned} m\mathcal{R}_X(\mathcal{F}_{G,\rho}) &= \tfrac{1}{\lambda}\log\exp\left(\lambda\mathbb{E}_\xi\sup_{f_w}\sum_{i=1}^{m}\xi_i f_w(x_i)\right)\\ &\leq \tfrac{1}{\lambda}\log\left(\mathbb{E}_\xi\sup_{f_w}\exp\left(\lambda\sum_{i=1}^{m}\xi_i f_w(x_i)\right)\right), \quad (6)\end{aligned}$$

where $\lambda > 0$ is an arbitrary parameter. As a next step, we rewrite the Rademacher complexity in the following manner:

$$\begin{aligned} &\mathbb{E}_\xi\sup_{f_w}\exp\left|\sum_{i=1}^{m}\xi_i\cdot f_w(x_i)\right|\\ &= \mathbb{E}_\xi\sup_{f_w}\exp\sqrt{\left|\sum_{i=1}^{m}\xi_i\cdot w^L\cdot\sigma(z_1^{L-1}(x_i), z_2^{L-1}(x_i))\right|^2}\\ &\leq \mathbb{E}_\xi\sup_{f_w}\exp\sqrt{\|w^L\|_2^2\cdot\sum_{j=1}^{2}\left\|\sum_{i=1}^{m}\xi_i\cdot\sigma(z_j^{L-1}(x_i))\right\|^2}. \quad (7)\end{aligned}$$

We notice that each $z_j^{L-1}(x)$ is itself a depth $L-1$ binary-tree neural network. Therefore, intuitively we would like to apply the same argument $L-1$ more times. However, in contrast to the above, the networks $\sigma(z_1^{L-1}(x)) = \sigma(w^{L-1}(z_1^{L-2}(x), z_2^{L-2}(x)))$ and $\sigma(z_2^{L-1}(x)) = \sigma(w^{L-1}(z_3^{L-2}(x), z_4^{L-2}(x)))$ end with a ReLU activation. To address this issue, (Neyshabur et al., 2015; Golowich et al., 2017) proposed a "peeling process" based on Equation 4.20 in (Ledoux & Talagrand, 1991) that can be used to bound terms of the form $\mathbb{E}_\xi \sup_{f'\in\mathcal{F}', W:\ \|W\|_F\le R} \exp[\sqrt{\alpha\,\|\sum_{i=1}^m \xi_i\cdot\sigma(Wf'(x_i))\|^2}]$. However, this bound is not directly applicable when there is a sum inside the square root, as in equation 7 which includes a sum over $j\in\{1,2\}$. Therefore, a modified peeling lemma is required to deal with this case.

Lemma 3.4 (Peeling Lemma). *Let σ be a 1-Lipschitz, positive-homogeneous activation function which is applied element-wise (such as the ReLU). Then for any class of vector-valued functions $\mathcal{F}\subset\{f=(f_1,\dots,f_q)\mid \forall j\in[q]:\ f_j:\mathbb{R}^d\to\mathbb{R}^p\}$, and any convex and monotonically increasing function $g:\mathbb{R}\to[0,\infty)$,*

$$\mathbb{E}_\xi \sup_{\substack{f\in\mathcal{F}\\ W_j:\ \|W_j\|\le R}} g\left(\sqrt{\sum_{j=1}^q \left\|\sum_{i=1}^m \xi_i\cdot\sigma(W_j f_j(x_i))\right\|^2}\right) \le 2\mathbb{E}_\xi \sup_{j\in[q],\ f\in\mathcal{F}} g\left(\sqrt{q}R\left\|\sum_{i=1}^m \xi_i\cdot f_j(x_i)\right\|\right).$$

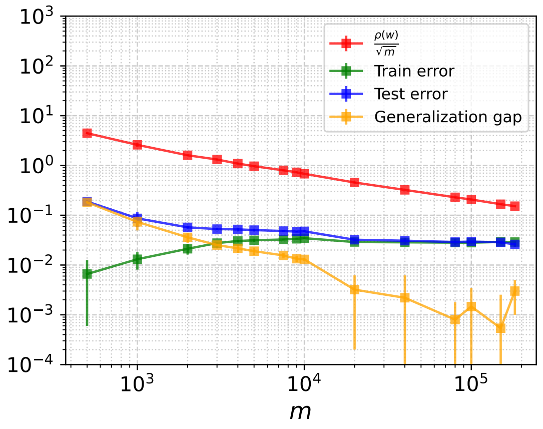
By applying this lemma $L-1$ times with $g=\exp$ and f representing the neurons preceding a certain neuron at a certain layer, we obtain the following inequality

$$\begin{aligned}
&\mathbb{E}_\xi \sup_{f_w} \exp\left|\sum_{i=1}^m \xi_i\cdot f_w(x_i)\right| \\
&\le 2^L\mathbb{E}_\xi \sup_{j,w}\exp\sqrt{\|w^L\|_2^2\prod_{l=1}^{L-1}\|w^l\|_F^2\cdot 2^L\left|\sum_{i=1}^m \xi_i x_{ij}\right|^2} \\
&\le 2^L\sum_{j=1}^d \mathbb{E}_\xi\exp\left(\lambda 2^{L/2}\rho\cdot\left|\sum_{i=1}^m \xi_i x_{ij}\right|\right) \\
&\le 4^L\sup_j\exp\left(\frac{\lambda^2 2^L\rho^2\cdot\sum_{i=1}^m x_{ij}^2}{2} + \lambda 2^{L/2}\rho\cdot\sqrt{\sum_{i=1}^m x_{ij}^2}\right),
\end{aligned}$$

where the last inequality follows from standard concentration bounds. Finally, by equation 6 and properly adjusting λ, we can finally bound $\mathcal{R}_X(\mathcal{F}_{G,\rho})$ as $\mathcal{O}(\frac{\sqrt{L}\rho}{\sqrt{m}})$.

4. Experiments

In section 3 we showed that the Rademacher complexity of compositionally sparse networks is bounded by $\mathcal{O}(\frac{\rho(w)}{\sqrt{m}})$ (when $\max_{i\in[m]}\|x_i\|$ and L are constants). In this section, we conduct an empirical evaluation of the performance, the term $\frac{\rho(w)}{\sqrt{m}}$ and $\rho(w)$ for neural networks trained with a varying number of training samples. Further, in Appendix A, we provide additional experiments that illustrate the evolution of these quantities throughout the training process.



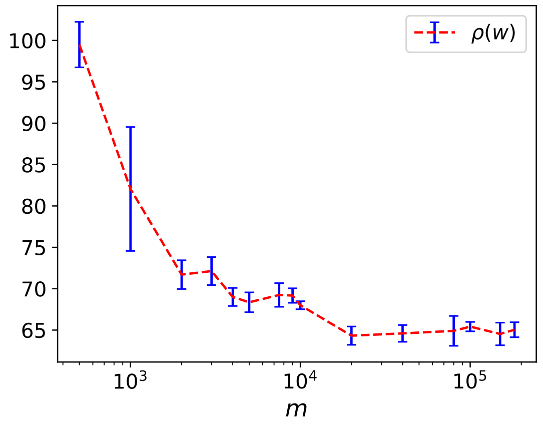


Figure 1. **Comparing our bound with the train and test errors and the generalization gap of a 5-layers network. (top)** We report $\frac{\rho(w)}{\sqrt{m}}$, the train error $\text{err}_S(f_w)$, the test error $\text{err}_P(f_w)$ and the generalization gap $|\text{err}_P(f_w)-\text{err}_S(f_w)|$ when varying the number of training samples (in logarithmic scales). **(bottom)** We display the value of $\rho(w)$ when varying the number of training samples. We used the following hyperparameters: $\rho^l = 0.1$, $\lambda = 1\text{e}{-3}$. More detailed results can be found in Table 1.

Network architecture. We use two types of deep neural network architectures. Both consist of four hidden convolutional layers, which use 3×3 convolutions, stride 2, and padding 0, and have output channel numbers of 32, 64, 128, and 128, respectively. The final fully connected layer maps the 3200-dimensional output of the final convolutional layer to 2 outputs, with ReLU activation applied to all layers except the last one. The second architecture is identical, but it replaces the last linear fully connected layer with two fully connected layers that project the 3200-dimensional

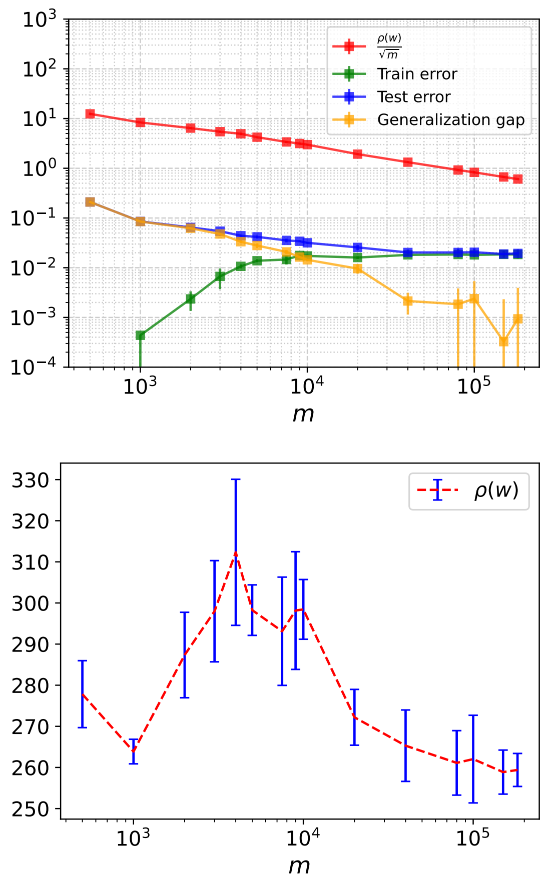


Figure 2. **Comparing our bound with the generalization gap and the test error of a 6-layers network.** See Figure 1 for details. More detailed numerical results can be found in Table 2. We used the following hyperparameters: $\rho^l = 0.01$, $\lambda = 5\mathrm{e}{-4}$.

output of the final convolutional layer to a 128-dimensional vector before mapping it to 2 outputs. The total number of parameters in the first model is 246886 and in the second is 650343 parameters.

Optimization process. In Theorem 3.2, a generalization bound is proposed that scales with $\rho(w) = \|w^L\|_2 \cdot \prod_{l=1}^{L-1} \|w^l\|_F$. To control $\rho(w)$, we regularize $\prod_{l=1}^{L} \|w^l\|_F \geq \rho(w)$ by applying weight normalization to all trainable layers, except for the last one, which is left un-normalized. Specifically, we fix the norm of the weights w^l in each layer by decomposing them into a direction and magnitude, such that $w^l = \rho^l v^l$ (where $\|v^l\|_F = 1$). To initialize w^l, we use the default PyTorch initialization and normalize v^l to have a norm of 1. We only update v^l using the method described in (Salimans & Kingma, 2016) while keeping $\rho^1, \dots, \rho^{L-1}$ constant. This way we can regularize $\prod_{l=1}^{L} \|w^l\|_F$, by applying weight decay of rate λ exclusively to the weights of the top layer.

Each model was trained using MSE-loss minimization between the logits of the network and the one-hot encodings of the training labels. To train the model we used the Stochastic Gradient Descent (SGD) optimizer with an initial learning rate $\mu = 0.03$, momentum of 0.9, batch size 128, and a cosine learning rate scheduler (Loshchilov & Hutter, 2016).

Varying the number of samples. In this experiment we trained the same model for binary classification between the first two classes of the CIFAR-5m dataset (Nakkiran et al., 2020) with a varying number of training samples. This dataset contains 6 million synthetic CIFAR-10-like images (including the CIFAR10 dataset). It was generated by sampling the DDPM generative model of (Ho et al., 2020), which was trained on the CIFAR-10 training set. For each number of samples m, we chose m random training samples from the dataset and trained the model on these samples for 5000 epochs over 5 different runs.

In Figures 1-2, we report the values of $\rho(w)$, $\frac{\rho(w)}{\sqrt{m}}$, the train and test errors, and the generalization gap for each model as a function of m. Each quantity is averaged over the last 100 training epochs (i.e., epochs 4900-5000). Since we do not have access to the complete population distribution P, we estimated the test error by using 1000 test samples per class. As seen in the figures, $\frac{\rho(w)}{\sqrt{m}}$ provides a relatively tight estimation of the generalization gap even though the network is overparameterized. For example, when m is greater than 10000, the quantity $\frac{\rho(w)}{\sqrt{m}}$ is smaller than 1 for the 5-layer model. Additionally, it is observed that $\rho(w)$ is bounded as a function of m, even though it could potentially increase with the size of the training dataset. Therefore, $\frac{\rho(w)}{\sqrt{m}}$ appears to decrease at a rate of $\mathcal{O}(1/\sqrt{m})$.

5. Conclusions

We studied the question of why certain deep learning architectures, such as CNNs and Transformers, perform better than others on real-world datasets. To tackle this question, we derived Rademacher complexity generalization bounds for sparse neural networks, which are orders of magnitude better than a naive application of standard norm-based generalization bounds for fully-connected networks. In contrast to previous papers (Long & Sedghi, 2020; Ledent et al., 2021), our results do not rely on parameter sharing between filters, suggesting that the sparsity of the neural networks is the critical component to their success. This sheds new light on the central question of why certain architectures perform so well and suggests that sparsity may be a key factor in their success. Even though our bounds are not practical in general, our experiments show that they are quite tight for simple classification problems, unlike other bounds based on parameter counting, suggesting that the underlying theory is sound and does not need a basic reformulation.

Acknowledgments

We thank Akshay Rangamani, Eran Malach and Antoine Ledent for illuminating discussions during the preparation of this manuscript. This material is based upon work supported by the Center for Minds, Brains and Machines (CBMM), funded by NSF STC award CCF-1231216.

References

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019.

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. Stronger generalization bounds for deep nets via a compression approach. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 254–263. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/arora18b.html.

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks, 2019. URL https://arxiv.org/abs/1901.08584.

Arulkumaran, K., Cully, A., and Togelius, J. Alphastar: An evolutionary computation perspective, 2019. URL http://arxiv.org/abs/1902.01724. cite arxiv:1902.01724.

Bartlett, P. L. and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. In *J. Mach. Learn. Res.*, 2001.

Bartlett, P. L. and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates Inc., 2017.

Belkin, M. Fit without fear: remarkable mathematical phenomena of deep learning through the prism of interpolation. *Acta Numerica*, 30:203 – 248, 2021.

Boucheron, S., Lugosi, G., and Massart, P. *Concentration Inequalities - A Nonasymptotic Theory of Independence*. Oxford University Press, 2013. ISBN 978-0-19-953525-5. doi: 10.1093/acprof:oso/9780199535255.001.0001. URL https://doi.org/10.1093/acprof:oso/9780199535255.001.0001.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., 2020.

Cao, Y. and Gu, Q. Generalization bounds of stochastic gradient descent for wide and deep neural networks, 2019. URL https://arxiv.org/abs/1905.13210.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code, 2021.

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

Daniely, A. and Granot, E. Generalization bounds for neural networks via approximate description length. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, jun 2019.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby,

N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Du, S. S., Wang, Y., Zhai, X., Balakrishnan, S., Salakhutdinov, R. R., and Singh, A. How many samples are needed to estimate a convolutional neural network? In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Golowich, N., Rakhlin, A., and Shamir, O. Size-Independent Sample Complexity of Neural Networks. *arXiv e-prints*, art. arXiv:1712.06541, Dec 2017.

Hanin, B. and Sellke, M. Approximating continuous functions by relu nets of minimal width, 2017. URL https://arxiv.org/abs/1710.11278.

Harvey, N., Liaw, C., and Mehrabian, A. Nearly-tight vc-dimension bounds for piecewise linear neural networks. *ArXiv*, abs/1703.02930, 2017.

He, F., Liu, T., and Tao, D. Why resnet works? residuals generalize, 2019. URL https://arxiv.org/abs/1904.01367.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '16, pp. 770–778. IEEE, June 2016. doi: 10.1109/CVPR.2016.90. URL http://ieeexplore.ieee.org/document/7780459.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4:251–257, 1991.

Jiang, Y., Neyshabur, B., Mobahi, H., Krishnan, D., and Bengio, S. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJgIPJBFvH.

LeCun, Y. and Cortes, C. MNIST handwritten digit database, 2010. URL http://yann.lecun.com/exdb/mnist/.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Ledent, A., Mustafa, W., Lei, Y., and Kloft, M. Norm-based generalisation bounds for deep multi-class convolutional neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35 (9):8279–8287, May 2021. doi: 10.1609/aaai.v35i9.17007. URL https://ojs.aaai.org/index.php/AAAI/article/view/17007.

Ledoux, M. and Talagrand, M. *Probability in Banach spaces*. Springer Berlin, Heidelberg, 1991.

Li, X., Lu, J., Wang, Z., Haupt, J., and Zhao, T. On tighter generalization bound for deep neural networks: Cnns, resnets, and beyond, 2018. URL https://arxiv.org/abs/1806.05159.

Long, P. M. and Sedghi, H. Generalization bounds for deep convolutional neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1e_FpNFDr.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts, 2016. URL https://arxiv.org/abs/1608.03983.

Lotfi, S., Finzi, M. A., Kapoor, S., Potapczynski, A., Goldblum, M., and Wilson, A. G. PAC-bayes compression bounds so tight that they can explain generalization. In *Advances in Neural Information Processing Systems*, volume 35. Curran Associates Inc., 2022.

Maiorov, V. On best approximation by ridge functions. *J. Approx. Theory*, 99(1), 1999.

Maiorov, V. and Pinkus, A. Lower bounds for approximation by mlp neural networks. *NEUROCOMPUTING*, 25:81–91, 1999.

Maiorov, V., Meir, R., and Ratsaby, J. On the approximation of functional classes equipped with a uniform measure using ridge functions. *J. Approx. Theory*, 99(1):95–111, 1999.

Mhaskar, H., Liao, Q., and Poggio, T. When and why are deep networks better than shallow ones? In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pp. 2343–2349. AAAI Press, 2017.

Mhaskar, H. N. Neural networks for optimal approximation of smooth and analytic functions. *Neural Comput.*, 8(1): 164–177, 1996.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of Machine Learning*. MIT Press, Cambridge, MA, 2 edition, 2018a. ISBN 978-0-262-03940-6.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2nd edition, 2018b.

Nagarajan, V. and Kolter, Z. Deterministic PAC-bayesian generalization bounds for deep networks via generalizing noise-resilience. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Hygn2o0qKX.

Nakkiran, P., Neyshabur, B., and Sedghi, H. The deep bootstrap framework: Good online learners are good offline generalizers, 2020. URL https://arxiv.org/abs/2010.08127.

Neyshabur, B., Tomioka, R., and Srebro, N. Norm-based capacity control in neural networks. In Grünwald, P., Hazan, E., and Kale, S. (eds.), *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pp. 1376–1401, Paris, France, 03–06 Jul 2015. PMLR. URL https://proceedings.mlr.press/v40/Neyshabur15.html.

Neyshabur, B., Bhojanapalli, S., Mcallester, D., and Srebro, N. Exploring generalization in deep learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Neyshabur, B., Bhojanapalli, S., McAllester, D. A., and Srebro, N. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *ArXiv*, abs/1707.09564, 2018.

Poggio, T. Foundations of deep learning: Compositional sparsity of computable functions. *CBMM memo 138*, 2022.

Poggio, T., Banburski, A., and Liao, Q. Theoretical issues in deep networks. *Proceedings of the National Academy of Sciences*, 117(48):30039–30045, 2020. doi: 10.1073/pnas.1907369117. URL https://www.pnas.org/doi/abs/10.1073/pnas.1907369117.

Richards, D. and Kuzborskij, I. Stability & generalisation of gradient descent for shallow neural networks without the neural tangent kernel. In *Advances in Neural Information Processing Systems*, volume 34. Curran Associates Inc., 2021.

Salimans, T. and Kingma, D. P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks, 2016. URL https://arxiv.org/abs/1602.07868.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge eBooks, 2014.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016. ISSN 0028-0836. doi: 10.1038/nature16961.

Vapnik, V. N. *Statistical Learning Theory*. Wiley-Interscience, 1998.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Wei, C. and Ma, T. Data-dependent sample complexity of deep neural networks via lipschitz augmentation, 2019. URL https://arxiv.org/abs/1905.03684.

Zhai, X., Kolesnikov, A., Houlsby, N., and Beyer, L. Scaling vision transformers, 2021.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx.

Zhou, W., Veitch, V., Austern, M., Adams, R. P., and Orbanz, P. Non-vacuous generalization bounds at the imagenet scale: a PAC-bayesian compression approach. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJgqqsAct7.

A. Additional Experiments

A.1. Additional Details for the Experiments in Figures 1-2

In Figures 1-2 we provided multiple plots demonstrating the behaviors of various quantities (e.g., $\rho(w)$, the train and test errors) when varying the number of training samples m. For completeness, in Tables 1-2 we explicitly report the values of each quantity reported in Figures 1-2.

m	$\rho(w)$	Train error	Test error	Train loss	Test loss	Generalization gap	$\frac{\rho(w)}{\sqrt{m}}$
500	99.499	0.007	0.188	0.032	0.161	0.182	4.450
1000	82.050	0.013	0.087	0.038	0.085	0.074	2.595
2000	71.701	0.021	0.057	0.042	0.062	0.036	1.603
3000	72.128	0.028	0.053	0.045	0.058	0.025	1.317
4000	69.004	0.030	0.052	0.047	0.056	0.022	1.091
5000	68.359	0.031	0.05	0.048	0.056	0.019	0.967
7500	69.241	0.033	0.048	0.048	0.052	0.016	0.800
9000	69.172	0.034	0.047	0.048	0.052	0.013	0.729
10000	68.003	0.035	0.048	0.049	0.052	0.013	0.68
20000	64.326	0.029	0.032	0.044	0.046	0.003	0.455
40000	64.598	0.029	0.031	0.044	0.045	0.003	0.323
80000	64.904	0.028	0.029	0.043	0.044	0.004	0.229
100000	65.418	0.028	0.030	0.043	0.043	0.001	0.207
150000	64.530	0.029	0.029	0.044	0.044	0.001	0.167
182394	65.040	0.029	0.026	0.044	0.043	0.003	0.152

Table 1. We report the averaged values of the norm $\rho(w)$, the train and test errors, the training and test losses, the generalization gap, and $\frac{\rho(w)}{\sqrt{m}}$ for the experiment in Figure 1.

m	$\rho(w)$	Train error	Test error	Train loss	Test loss	Generalization gap	$\frac{\rho(w)}{\sqrt{m}}$
500	277.829	0.000	0.210	0.005	0.177	0.210	12.425
1000	263.867	0.000	0.085	0.008	0.068	0.085	8.344
2000	287.343	0.002	0.065	0.012	0.052	0.062	6.425
3000	297.993	0.007	0.054	0.015	0.045	0.048	5.441
4000	312.316	0.011	0.044	0.017	0.037	0.033	4.938
5000	298.258	0.014	0.042	0.019	0.036	0.028	4.218
7500	293.125	0.015	0.035	0.021	0.032	0.021	3.385
9000	298.155	0.018	0.034	0.022	0.031	0.016	3.143
10000	298.442	0.017	0.032	0.022	0.029	0.014	2.984
20000	272.198	0.016	0.026	0.018	0.024	0.010	1.925
40000	265.294	0.018	0.020	0.019	0.021	0.003	1.326
80000	261.114	0.018	0.020	0.020	0.020	0.004	0.923
100000	262.034	0.018	0.021	0.019	0.021	0.004	0.829
150000	258.874	0.019	0.019	0.020	0.020	0.003	0.668
182394	259.392	0.018	0.019	0.020	0.020	0.002	0.607

Table 2. We report the averaged values of the norm $\rho(w)$, the train and test errors, the training and test losses, the generalization gap, and $\frac{\rho(w)}{\sqrt{m}}$ for the experiment in Figure 2.

A.2. Evaluating Networks During Training

In section 4, we examined the behavior of $\rho(w)$, the train and test errors, and our bound while varying the number of training samples. In this section, we conduct supplementary experiments that compare these quantities throughout the training process. Additionally, to add diversity to the study, we utilize a slightly different training method in these experiments.

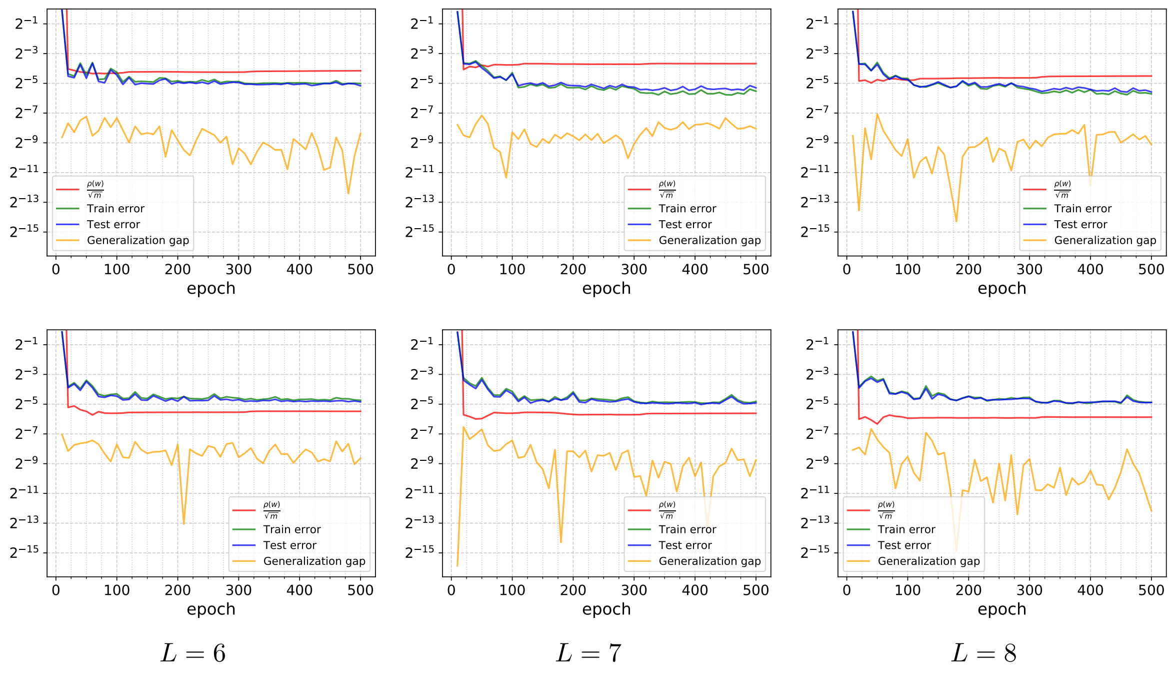


Figure 3. **Varying the number of layers.** We report $\frac{\rho(w)}{\sqrt{m}}$, the train error $\text{err}_S(f_w)$, the test error $\text{err}_P(f_w)$ and the generalization gap $|\text{err}_P(f_w) - \text{err}_S(f_w)|$ of CONV-L-1000 trained on MNIST with a varying number of layers. We trained the models with batch size 64 and learning rate $\mu = 1$. For the top plots we used $\lambda = 2\text{e}{-3}$ and for the bottom ones we used $\lambda = 3\text{e}{-3}$.

Network architecture. In this experiment, we employed a simple convolutional network architecture denoted by CONV-L-H. The network consists of a stack of two 2×2 convolutional layers with a stride of 2 and zero padding, utilizing ReLU activations. This is followed by $L - 2$ stacks of 3×3 convolutional layers with H channels, a stride of 1, and padding of 1, also followed by ReLU activations. The final layer is a fully-connected layer. No biases are used in any of the layers.

Optimization process. In the current experiment we trained each model with a standard weight normalization (Salimans & Kingma, 2016) for each parametric layer. Each model was trained using MSE-loss minimization between the logits of the network and the one-hot encodings of the training labels. To train the model we used the Stochastic Gradient Descent (SGD) optimizer with an initial learning rate μ that is decayed by a factor of 0.1 at epochs 60, 100, 300, momentum of 0.9 and weight decay with rate λ.

In Figure 3, we present the results of our experimentation where we trained models of varying depths on the MNIST dataset (LeCun & Cortes, 2010). One of the key observations from our experiment is that as we increase the depth of the model, the term $\frac{\rho(w)}{\sqrt{m}}$ empirically generally decreases, even though the overall number of training parameters grows with the number of layers. This is in correlation with the fact that the generalization gap is lower for deeper networks. suggests that deeper models have a better generalization ability despite having more parameters. Furthermore, we also observed that in all cases, the term $\frac{\rho(w)}{\sqrt{m}}$ is quite small, reflecting the tightness of our bound.

In Figures 4 and 5, we present the results of an experiment where we varied the number of channels H in models trained on MNIST and Fashion MNIST (respectively). As can be seen, the bound remains largely unchanged when increasing H despite the network's size scaling as $\Theta(H^2)$. We also observed that, after the network achieves good performance, the bound is highly correlated with the generalization gap. Specifically, for MNIST, the generalization gap and the bound are relatively stable, while for Fashion-MNIST, the bound seems to grow at the same rate as the generalization gap. Since the results are presented in log-scales, this suggests that the generalization gap is empirically proportional to our bound.

B. Proofs

Lemma B.1 (Peeling Lemma)**.** *Let σ be a 1-Lipschitz, positive-homogeneous activation function which is applied element-wise (such as the ReLU). Then for any class of vector-valued functions* $\mathcal{F} \subset \{f = (f_1, \dots, f_q) \mid \forall j \in [q] : \ f_j : \mathbb{R}^d \to \mathbb{R}^p\}$,

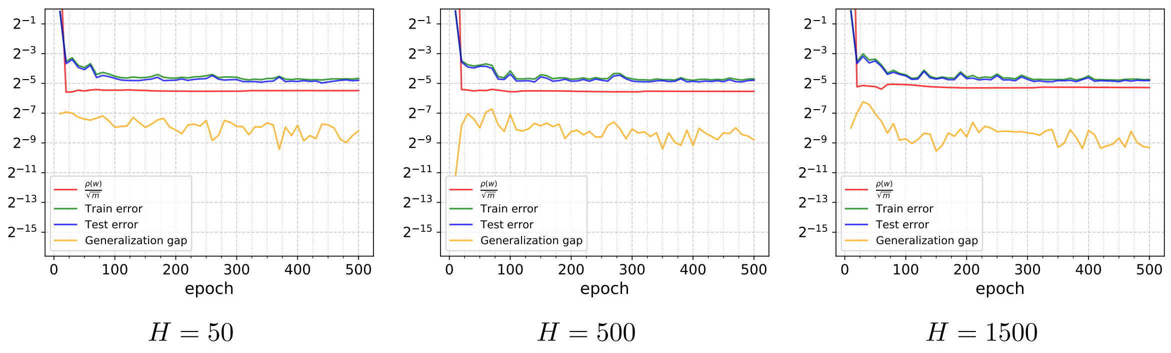


Figure 4. **Varying the number of channels.** We report $\frac{\rho(w)}{\sqrt{m}}$, the train error $\text{err}_S(f_w)$, the test error $\text{err}_P(f_w)$ and the generalization gap $|\text{err}_P(f_w) - \text{err}_S(f_w)|$ of CONV-6-H trained on MNIST with a varying number of channels. We trained the models with batch size 64, $\mu = 1$, and $\lambda = 3\text{e}{-3}$.

and any convex and monotonically increasing function $g : \mathbb{R} \to [0, \infty)$, we have

$$\mathbb{E}_\xi \sup_{\substack{f \in \mathcal{F} \\ W_j:\ \|W_j\| \leq R}} g\left(\sqrt{\sum_{j=1}^q \left\|\sum_{i=1}^m \xi_i \cdot \sigma(W_j f_j(x_i))\right\|^2}\right) \leq 2\mathbb{E}_\xi \sup_{j \in [q],\ f \in \mathcal{F}} g\left(\sqrt{q}R\left\|\sum_{i=1}^m \xi_i \cdot f_j(x_i)\right\|\right).$$

Proof. Let $W \in \mathbb{R}^{h \times p}$ be a matrix and let $w_1, \dots, w_h$ be the rows of the matrix W. Define a function $Q_j(w) := \left(\sum_{i=1}^m \xi_i \cdot \sigma(\frac{w_r^\top}{\|w_r\|} f_j(x_i))\right)^2$ for some fixed functions f_j. We notice that

$$\begin{aligned}\sum_{j=1}^q \left\|\sum_{i=1}^m \xi_i \cdot \sigma(W_j f_j(x_i))\right\|^2 &= \sum_{j=1}^q \sum_{r=1}^h \|w_{jr}\|^2 \left(\sum_{i=1}^m \xi_i \cdot \sigma(\frac{w_{jr}^\top}{\|w_{jr}\|} f_j(x_i))\right)^2 \\ &= \sum_{j=1}^q \sum_{r=1}^h \|w_{jr}\|^2 \cdot Q_j(\frac{w_{jr}}{\|w_{jr}\|}).\end{aligned}$$

For any $w_{j1}, \dots, w_{jh}$, we have

$$\sum_{r=1}^h \|w_{jr}\|^2 \cdot Q_j(\frac{w_{jr}}{\|w_{jr}\|}) \leq R \cdot \max_r Q_j(\frac{w_{jr}}{\|w_{jr}\|}), \tag{8}$$

which is obtained for $\hat{w}_{j1}, \dots, \hat{w}_{jh}$, where $\hat{w}_{ji} = 0$ for all $i \neq r^*$ and $\hat{w}_{jr^*}$ of norm R for some $r^* \in [h]$. Together with the fact that g is a monotonically increasing function, we obtain

$$\begin{aligned}\mathbb{E}_\xi \sup_{\substack{f \in \mathcal{F} \\ W_j:\ \|W_j\| \leq R}} g\left(\sqrt{\sum_{j=1}^q \left\|\sum_{i=1}^m \xi_i \cdot \sigma(W_j f_j(x_i))\right\|^2}\right) &\leq \mathbb{E}_\xi \sup_{\substack{f \in \mathcal{F} \\ w_1 \dots, w_q:\ \|w_j\| = R}} g\left(\sqrt{\sum_{j=1}^q \left|\sum_{i=1}^m \xi_i \cdot \sigma(w_j^\top f_j(x_i))\right|^2}\right) \\ &\leq \mathbb{E}_\xi \sup_{\substack{j \in [q],\ f \in \mathcal{F} \\ w_1 \dots, w_q:\ \|w_j\| = R}} g\left(\sqrt{q \cdot \left|\sum_{i=1}^m \xi_i \cdot \sigma(w_j^\top f_j(x_i))\right|^2}\right) \\ &= \mathbb{E}_\xi \sup_{\substack{j \in [q],\ f \in \mathcal{F} \\ w:\ \|w\| = R}} g\left(\sqrt{q} \cdot \left|\sum_{i=1}^m \xi_i \cdot \sigma(w^\top f_j(x_i))\right|\right).\end{aligned}$$

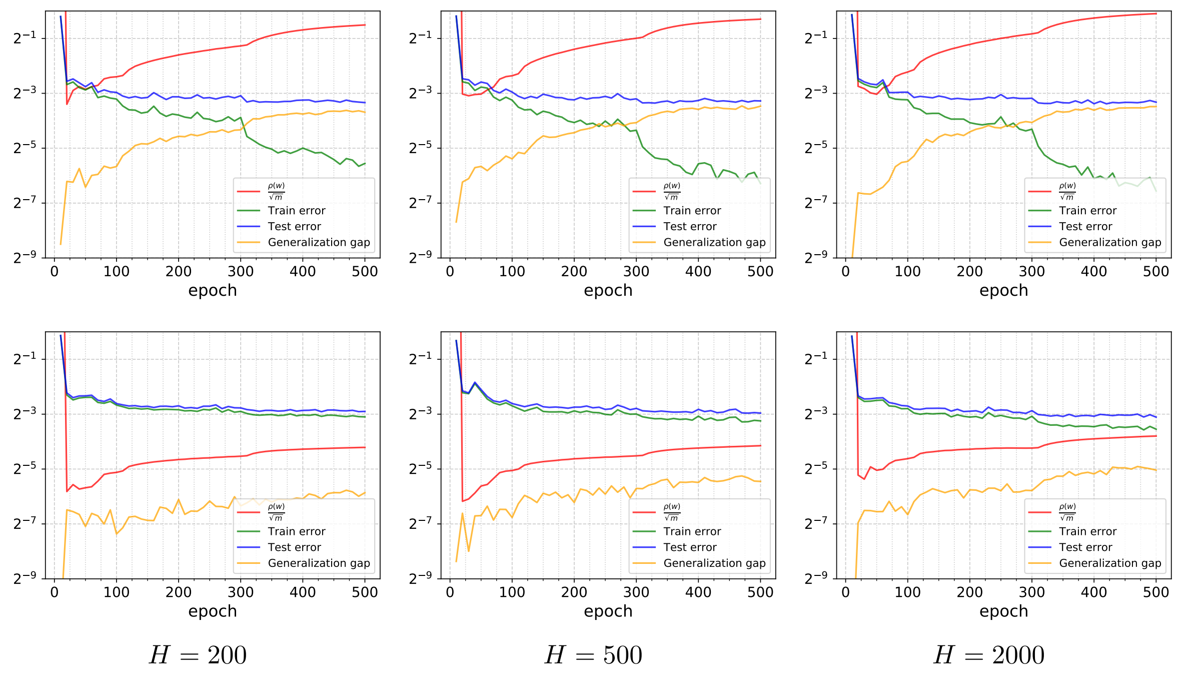


Figure 5. **Varying the number of channels.** We report $\frac{\rho(w)}{\sqrt{m}}$, the train error $\text{err}_S(f_w)$, the test error $\text{err}_P(f_w)$ and the generalization gap $|\text{err}_P(f_w) - \text{err}_S(f_w)|$ of CONV-8-H trained on Fashion-MNIST with a varying number of channels. We trained the models with batch size 128 and learning rate $\mu = 1$. For the top plots we used $\lambda = 1\text{e}{-3}$, and for the bottom ones we used $\lambda = 2\text{e}{-3}$.

Since $g(|z|) \leq g(z) + g(-z)$,

$$\begin{aligned}
\mathbb{E}_\xi \sup_{\substack{j\in[q],\ f\in\mathcal{F}\\ w:\ \|w\|=R}} g\left(\sqrt{q}\cdot\Big|\sum_{i=1}^{m}\xi_i\cdot\sigma(w^\top f_j(x_i))\Big|\right) \;&\leq\; \mathbb{E}_\xi \sup_{\substack{j\in[q],\ f\in\mathcal{F}\\ w:\ \|w\|=R}} g\left(\sqrt{q}\cdot\sum_{i=1}^{m}\xi_i\cdot\sigma(w^\top f_j(x_i))\right)\\
&\quad + \mathbb{E}_\xi \sup_{\substack{j\in[q],\ f\in\mathcal{F}\\ w:\ \|w\|=R}} g\left(-\sqrt{q}\cdot\sum_{i=1}^{m}\xi_i\cdot\sigma(w^\top f_j(x_i))\right)\\
&= 2\mathbb{E}_\xi \sup_{\substack{j\in[q],\ f\in\mathcal{F}\\ w:\ \|w\|=R}} g\left(\sqrt{q}\cdot\sum_{i=1}^{m}\xi_i\cdot\sigma(w^\top f_j(x_i))\right),
\end{aligned}$$

where the last equality follows from the symmetry in the distribution of the ξ_i random variables. By Equation 4.20 in (Ledoux & Talagrand, 1991), the right-hand side can be upper bounded as follows

$$\begin{aligned}
2\mathbb{E}_\xi \sup_{\substack{j\in[q],\ f\in\mathcal{F}\\ w:\ \|w\|=R}} g\left(\sqrt{q}\cdot\sum_{i=1}^{m}\xi_i\cdot\sigma(w^\top f_j(x_i))\right) \;&\leq\; 2\mathbb{E}_\xi \sup_{\substack{j\in[q],\ f\in\mathcal{F}\\ w:\ \|w\|=R}} g\left(\sqrt{q}\cdot\sum_{i=1}^{m}\xi_i\cdot w^\top f_j(x_i)\right)\\
&\leq\; 2\mathbb{E}_\xi \sup_{\substack{j\in[q],\ f\in\mathcal{F}\\ w:\ \|w\|=R}} g\left(\sqrt{q}\cdot\|w\|\left\|\sum_{i=1}^{m}\xi_i\cdot f_j(x_i)\right\|\right)\\
&\leq\; 2\mathbb{E}_\xi \sup_{j\in[q],\ f\in\mathcal{F}} g\left(\sqrt{q}R\left\|\sum_{i=1}^{m}\xi_i\cdot f_j(x_i)\right\|\right).
\end{aligned}$$

□

Proposition B.2. *Let G be a neural network architecture of depth L and let $\rho > 0$. Let $X = \{x_i\}_{i=1}^m$ be a set of samples. Then,*

$$\mathcal{R}_X(\mathcal{F}_{G,\rho}) \;\leq\; \frac{\rho}{m}\cdot\left(1+\sqrt{2L\log(2\deg(G))}\right)\cdot\sqrt{\max_{j_1,\dots,j_L}\prod_{l=1}^{L}|\text{pred}(l,j_{L-l})|\cdot\sum_{i=1}^{m}\|z^0_{j_L}(x_i)\|^2},$$

where the maximum is taken over $j_1, \dots, j_L$, such that, $j_{L-l+1} \in \text{pred}(l, j_{L-l})$ for all $l \in [L]$.

Proof. Due to the homogeneity of the ReLU function, each function $f_w \in \mathcal{F}_{G,\rho}$ can be rewritten as $f_{\hat{w}}$, where $\hat{w}^L_1 := \rho\frac{w^L_1}{\|w^L_1\|_2}$ and for all $l < L$ and $j_l \in [d_l]$, $\hat{w}^l_{j_l} := \frac{w^l_{j_l}}{\max_{j\in[d_l]}\|w^l_j\|_F}$. In particular, we have $\mathcal{F}_{G,\rho} \subset \hat{\mathcal{F}}_{G,\rho} := \{f_w \mid \|w^L_1\|_2 \leq \rho \text{ and } \forall i < L, j_l \in [d_l] : \|w^l_{j_l}\|_F \leq 1\}$ since the ReLU function is homogeneous. For simplicity, we denote by $f_{\tilde{w}}$ an arbitrary member of $\tilde{\mathcal{F}}_{G,\rho}$ and $\hat{w}^l_{j_l}$ the weights of the j_lth neuron of the lth layer. In addition, we denote $v^l_{j_1}(x_i) = (z^l_{j_2}(x_i))_{j_2\in\text{pred}(L,j_1)}$ and $z^l_j(x_i) = \sigma(\hat{w}^l_j v^{l-1}_j(x_i))$ and we denote $j_0 = 1$.

We apply Jensen's inequality,

$$m\mathcal{R} \;:=\; m\mathcal{R}_X(\hat{\mathcal{F}}_{G,\rho}) \;=\; \mathbb{E}_\xi\left[\sup_{\hat{w}}\sum_{i=1}^m \xi_i f_{\hat{w}}(x_i)\right] \;\leq\; \tfrac{1}{\lambda}\log\mathbb{E}_\xi\sup_{\hat{w}}\exp\left(\lambda\sum_{i=1}^m\xi_i f_{\hat{w}}(x_i)\right),$$

where the supremum is taken over the weights $\hat{w}^l_{j_l}$ ($l \in [L], j_l \in [d_l]$) that are described above. Since $\|\hat{w}^L_{j_0}\|_2 \leq \rho$, we have

$$\begin{aligned}
m\mathcal{R} \;&\leq\; \tfrac{1}{\lambda}\log\mathbb{E}_\xi\sup_{\hat{w}}\exp\left(\lambda\sum_{i=1}^m\xi_i f_{\hat{w}}(x_i)\right)\\
&=\; \tfrac{1}{\lambda}\log\mathbb{E}_\xi\sup_{\hat{w}}\exp\left(\lambda\left|\sum_{i=1}^m\xi_i\cdot\hat{w}^L_{j_0}\cdot v^{L-1}_{j_0}(x_i)\right|\right)\\
&\leq\; \tfrac{1}{\lambda}\log\left(\mathbb{E}_\xi\sup_{\hat{w}}\exp\left(\lambda\rho\cdot\sqrt{\left\|\sum_{i=1}^m\xi_i\cdot v^{L-1}_{j_0}(x_i)\right\|^2}\right)\right).
\end{aligned}$$

Next, we use Lemma 3.4,

$$\begin{aligned}
m\mathcal{R} \;&\leq\; \tfrac{1}{\lambda}\log\left(\mathbb{E}_\xi\sup_{\hat{w}}\exp\left(\lambda\rho\cdot\sqrt{\sum_{j_1\in\text{pred}(L,j_0)}\left\|\sum_{i=1}^m\xi_i\cdot z^{L-1}_{j_1}(x_i)\right\|^2}\right)\right)\\
&=\; \tfrac{1}{\lambda}\log\left(\mathbb{E}_\xi\sup_{\hat{w}}\exp\left(\lambda\rho\cdot\sqrt{\sum_{j_1\in\text{pred}(L,j_0)}\left\|\sum_{i=1}^m\xi_i\cdot\hat{w}^{L-1}_{j_1}\cdot\sigma(v^{L-2}_{j_1}(x_i))\right\|^2}\right)\right)\\
&\leq\; \tfrac{1}{\lambda}\log\left(2\mathbb{E}_\xi\sup_{j_1,\,\hat{w}}\exp\left(\lambda\rho\cdot\sqrt{|\text{pred}(L,j_0)|\cdot\left\|\sum_{i=1}^m\xi_i\cdot v^{L-2}_{j_1}(x_i)\right\|^2}\right)\right)\\
&=\; \tfrac{1}{\lambda}\log\left(2\mathbb{E}_\xi\sup_{j_1,\,\hat{w}}\exp\left(\lambda\rho\cdot\sqrt{|\text{pred}(L,j_0)|\sum_{j_2\in\text{pred}(L-1,j_1)}\cdot\left\|\sum_{i=1}^m\xi_i\cdot z^{L-2}_{j_2}(x_i)\right\|^2}\right)\right),
\end{aligned}$$

where the supremum is taken over the parameters of $f_{\hat{w}}$ and $j_1 \in \text{pred}(L, j_0)$. By applying this process recursively L times, we obtain the following inequality,

$$m\mathcal{R} \;=\; \mathbb{E}_\xi\left[\sup_{\hat{w}}\sum_{i=1}^m\xi_i f_{\hat{w}}(x_i)\right] \;\leq\; \tfrac{1}{\lambda}\log\left(2^L\mathbb{E}_\xi\sup_{j_1,\dots,j_L}\exp\left(\lambda\rho\cdot\sqrt{\prod_{l=1}^L|\text{pred}(l,j_{L-l})|}\cdot\left\|\sum_{i=1}^m\xi_i\cdot z^0_{j_L}(x_i)\right\|\right)\right), \qquad (9)$$

where the supremum is taken over $j_1,\dots,j_L$, such that, $j_{l+1} \in \text{pred}(l, j_{L-l})$. We notice that

$$\begin{aligned}
&\mathbb{E}_\xi \sup_{j_1,\dots,j_L} \exp\left(\lambda\rho \cdot \sqrt{\prod_{l=1}^{L} |\text{pred}(l, j_{L-l})|} \cdot \left\|\sum_{i=1}^{m} \xi_i \cdot z^0_{j_L}(x_i)\right\|\right) \\
&\leq \sum_{j_1,\dots,j_L} \mathbb{E}_\xi \exp\left(\lambda\rho \cdot \sqrt{\prod_{l=1}^{L} |\text{pred}(l, j_{L-l})|} \cdot \left\|\sum_{i=1}^{m} \xi_i \cdot z^0_{j_L}(x_i)\right\|\right) \\
&\leq \deg(G)^L \cdot \max_{j_1,\dots,j_L} \mathbb{E}_\xi \exp\left(\lambda\rho \cdot \sqrt{\prod_{l=1}^{L} |\text{pred}(l, j_{L-l})|} \cdot \left\|\sum_{i=1}^{m} \xi_i \cdot z^0_{j_L}(x_i)\right\|\right).
\end{aligned} \tag{10}$$

Following the proof of Theorem 1 in (Golowich et al., 2017), by applying Jensen's inequality and Theorem 6.2 in (Boucheron et al., 2013) we obtain that for any $\alpha > 0$,

$$\mathbb{E}_\xi \exp\left(\alpha \left\|\sum_{i=1}^{m} \xi_i \cdot z^0_{j_L}(x_i)\right\|\right) \;\leq\; \exp\left(\frac{\alpha^2 \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2}{2} + \alpha\sqrt{\sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2}\right). \tag{11}$$

Hence, by combining equations 9-11 with $\alpha = \lambda\rho \cdot \sqrt{\prod_{l=1}^{L} |\text{pred}(l, j_{L-l})|}$, we obtain that

$$\begin{aligned}
m\mathcal{R} &= \mathbb{E}_\xi\left[\sup_{f\in\mathcal{F}} \sum_{i=1}^{m} \xi_i f(x_i)\right] \\
&\leq \frac{1}{\lambda}\log\left((2\deg(G))^L \cdot \max_{j_1,\dots,j_L} \mathbb{E}_\xi \exp\left(\lambda\rho \cdot \sqrt{\prod_{l=1}^{L} |\text{pred}(l, j_{L-l})|} \cdot \left\|\sum_{i=1}^{m} \xi_i \cdot z^0_{j_L}(x_i)\right\|\right)\right) \\
&= \frac{1}{\lambda}\max_{j_1,\dots,j_L} \log\left((2\deg(G))^L \cdot \mathbb{E}_\xi \exp\left(\lambda\rho \cdot \sqrt{\prod_{l=1}^{L} |\text{pred}(l, j_{L-l})|} \cdot \left\|\sum_{i=1}^{m} \xi_i \cdot z^0_{j_L}(x_i)\right\|\right)\right) \\
&\leq \frac{\log(2\deg(G))L}{\lambda} + \frac{\lambda\rho^2 \max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \cdot \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2}{2} \\
&\quad + \rho\sqrt{\max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \cdot \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2}
\end{aligned}$$

The choice $\lambda = \sqrt{\frac{2\log(2\deg(G))L}{\rho^2 \max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \cdot \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2}}$, yields the desired inequality. □

Theorem B.3. *Let P be a distribution over $\mathbb{R}^{c_0 d_0} \times \{\pm 1\}$. Let $S = \{(x_i, y_i)\}_{i=1}^m$ be a dataset of i.i.d. samples selected from P. Then, with probability at least $1-\delta$ over the selection of S, for any $f_w \in \mathcal{F}_{G,\rho}$ that perfectly fits the data (for all $i \in [m]:\ f_w(x_i) = y_i$), we have*

$$\text{err}_P(f_w) \;\leq\; \frac{(\rho(w)+1)}{m}\left(1 + \sqrt{2L\log(2\deg(G))}\right) \cdot \sqrt{\max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2} + 3\sqrt{\frac{\log(2(\rho(w)+2)^2/\delta)}{2m}}$$

where the maximum is taken over $j_1,\dots,j_L$, such that, $j_{L-l+1} \in \text{pred}(l, j_{L-l})$ for all $l \in [L]$.

Proof. Let $t \in \mathbb{N} \cup \{0\}$ and $\mathcal{G}_t = \mathcal{F}_{G,\rho}$. By Lemma 2.2, with probability at least $1 - \frac{\delta}{t(t+1)}$, for any function $f_w \in \mathcal{G}_t$ that perfectly fits the training data, we have

$$\text{err}_P(f_w) \;\leq\; 2\mathcal{R}_X(\mathcal{G}_t) + 3\sqrt{\frac{\log(2(t+1)^2/\delta)}{2m}}. \tag{12}$$

By Proposition 3.1, we have

$$\mathcal{R}_X(\mathcal{G}_t) \;\leq\; t \cdot \left(1+\sqrt{2L\log(2\deg(G))}\right) \cdot \sqrt{\max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \cdot \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2} \tag{13}$$

because of the union bound over all $t \in \mathbb{N}$, equation 3 holds uniformly for all $t \in \mathbb{N}$ and $f_w \in \mathcal{G}_t$ with probability at least $1-\delta$. For each f_w with norm $\rho(w)$ we then apply the bound with $t = \lceil \rho(w) \rceil$ since $f_w \in \mathcal{G}_t$, and obtain,

$$\begin{aligned}
\text{err}_P(f_w) \;\leq\; & \frac{t\left(1+\sqrt{2L\log(2\deg(G))}\right)\sqrt{\max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2}}{m} \\
& + 3\sqrt{\frac{\log(2(t+1)^2/\delta)}{2m}} \\
\leq\; & \frac{(\rho(w)+1)\left(1+\sqrt{2L\log(2\deg(G))}\right)\sqrt{\max_{j_1,\dots,j_L} \prod_{l=1}^{L} |\text{pred}(l, j_{L-l})| \sum_{i=1}^{m} \|z^0_{j_L}(x_i)\|^2}}{m} \\
& + 3\sqrt{\frac{\log(2(\rho(w)+2)^2/\delta)}{2m}},
\end{aligned}$$

which proves the desired bound. □